



**UNIVERSIDAD NACIONAL AUTÓNOMA DE
MÉXICO**
FACULTAD DE CIENCIAS
PROCESAMIENTO DIGITAL DE
IMAGENES
(2024-1)
PRÁCTICA 04



• Vargas Bravo Paola 318074755

FECHA LÍMITE : 23/10/23

Contents

1	Reglas generales para el desarrollo de las Prácticas de Laboratorio	3
2	Objetivos	3
3	Introducción	3
4	Desarrollo	5
5	Código	34
6	Conclusiones	34
7	Referencias	34

1.

1 Reglas generales para el desarrollo de las Prácticas de Laboratorio

- Deberás respetar la estructura general de este documento, i.e, entregar tu práctica con las secciones: objetivos, introducción, desarrollo,código, conclusiones y referencias.
- El desarrollo de la práctica deberá ser auténtico. Aquellas personas que presenten los mismo cálculos, código fuente, etc , serán sancionados.
- El día de entrega establecido deberá ser respetado por todos. La hora límite de entrega será establecida en su momento y no se reciben trabajo posteriormente.
- Deberás entregar el documento impreso así como el código a Miguel Angel Veloz Lucas via Goolge Classroom <https://classroom.google.com/c/NjE4MjMxNzU1MDE2>

2 Objetivos

- Realizar diversas modificaciones al histograma con transformaciones básicas como son: negativo, exponencial, logarítmica y gamma.
- Ecualización el histograma de una imagen.
- Realizar operaciones de suavizado y de reducción de ruido en imágenes utilizando filtros espaciales.
- Realizar operaciones de detección de bordes en imágenes, tanto limpias como ruidosas, utilizando filtros basados en aproximaciones de gradientes y laplacianos.

3 Introducción

Los histogramas constituyen la base de varias técnicas de procesamiento en el dominio espacial. La manipulación de los histogramas es usada de manera eficiente en el realce o mejoramiento de la calidad de una imagen. La información estadística obtenida a partir de los histogramas se utiliza en diversas aplicaciones como compresión y segmentación de imágenes. La facilidad con la que se pueden calcular los histogramas usando software y su bajo consumo de recursos de hardware en su implementación, han hecho de esta herramienta una de las más usadas en el procesamiento en tiempo real.

El histograma de una imagen es la representación gráfica de la distribución que existe de las distintas tonalidades de grises con relación al número de píxeles o porcentaje de los mismos, es decir, un histograma representa la frecuencia relativa de ocurrencia de los niveles de gris. La representación de un histograma ideal sería la de una recta horizontal, ya que eso nos indicaría que todos los posibles valores de grises están distribuidos de manera uniforme en nuestra imagen.

La ecualización del histograma es una técnica bastante conocida y sirve para obtener un histograma uniforme de tal manera que los niveles de gris son distribuidos sobre la escala y un número igual de píxeles son colocados en cada nivel de gris. Para un observador, esta ecualización hace que las imágenes se vean más balanceadas y con mejor contraste.

Como consecuencia, una imagen ecualizada, permite que ciertos detalles sean visibles en regiones oscuras o brillantes.

Los filtros espaciales tienen como objetivo modificar la contribución de determinados rangos de frecuencias de una imagen. El término espacial se refiere a que el filtro se aplica directamente a la imagen y no a una transformada de la misma, es decir, el nivel de gris de un pixel se obtiene directamente en función del valor de sus vecinos. La convolución es la operación con la cual se hace filtrado espacial. Los filtros espaciales pueden clasificarse basándose en su linealidad en filtros lineales y en filtros no lineales. A su vez los filtros lineales pueden ser clasificados

según las frecuencias que dejen pasar: los filtros paso bajo atenúan o eliminan las componentes de alta frecuencia a la vez que dejan inalteradas las bajas frecuencias; los filtros paso altas atenúan o eliminan las componentes de baja frecuencia con lo que agudizan las componentes de alta frecuencia; los filtros paso banda eliminan regiones elegidas de frecuencias intermedias. A continuación se describe el uso de los diferentes filtros:

- Filtros paso bajas: son utilizados en la reducción de ruido; suavizan y aplanan un poco las imágenes y como consecuencia se reduce o se pierde la nitidez. En inglés son conocidos como Smoothing Spatial Filters.
- Filtros paso altas: estos filtros son utilizados para detectar cambios de luminosidad. Son utilizados en la detección de patrones como bordes o para resaltar detalles finos de una imagen. En inglés son conocidos como Sharpening Spatial Filters. Los filtros unsharp masking son filtros paso altas usados en el mejoramiento de la nitidez o de la calidad visual de una imagen.
- Filtros paso banda: son utilizados para detectar patrones de ruido. Ya que un filtro paso banda generalmente elimina demasiado contenido de una imagen casi no son usados, sin embargo, los filtros paso banda son útiles para aislar los efectos de ciertas bandas de frecuencias seleccionadas sobre una imagen. De esta manera, estos filtros ayudan a simplificar el análisis de ruido, razonablemente independiente del contenido de la imagen.

4 Desarrollo

- Aplicar a una imagen las diferentes transformaciones : negativa, logarítmica, y gama.
 - (a) Desplegar la imagen original y su histograma (Nota: si trabajas con MATLAB utilizas la función subplot y usa títulos en sus gráficas con title).
 - (b) Tu programa deberá ser capáz de tener como entrada diferentes imágenes. Cada transformación resultado deberá ser desplegada en una ventana a parte, de manera que tendrás 5 ventanas: una con la imagen original y otras 4 cada una con las transformaciones referidas.

Función : `image_histograma`

Parámetros : `image`

- Creamos un arreglo llamado histograma de ceros en 255 localidades.
- Recorremos la imagen, donde por cada coordenada (x, y) , obtenemos el nivel de gris de dicho píxel, ese nivel del gris será el índice dentro del histograma de ceros que creamos, donde en dicha localidad la aumentaremos en uno, dado que ese píxel tiene ese nivel de gris y por lo tanto aumentamos la frecuencia.
- Regresamos el histograma

- (c) Aplicar cada una de las transformaciones referidas a la imagen leída en (a) y desplegar en la misma ventana la imagen resultado.

Función : `transformacion_negativa`

Parámetros : `image_Abierta`

- Creamos nuestra imagen objetivo en escala de grises con dimensiones iguales a la imagen original.
- Recorremos dicha imagen de forma que obtenemos el píxel por cada coordenada (x, y) dentro de la imagen, donde a cada uno de los mismos aplicamos la formula :

$$255 - (x, y)$$

El resultado es el nuevo valor en (x, y) .

- Regresamos la imagen objetivo .

Función : `transformacion_logaritmica`

Parámetros : `image_Abierta`

- Creamos nuestra imagen objetivo en escala de grises, de acuerdo a las dimensiones de la imagen original.
- Recorremos dicha imagen, obtenemos el píxel por cada coordenada (x,y) dentro de la imagen, donde a cada uno de los mismo aplicamos la formula :

$$c \cdot \log(1 + (x, y))$$

Donde $c = 45$, dado que el realce de la imagen es subjetivo, probamos varios valores para c , hasta dar con el adecuado.

El resultado es el nuevo valor en (x,y).

- Regresamos la imagen objetivo.

Función : transformacion_gamma

Parámetros : image_Abierta

- Creamos nuestra imagen objetivo en escala de grises, de acuerdo a las dimensiones de la imagen original.
- Recorremos dicha imagen, obtenemos el píxel por cada coordenada (x,y) dentro de la imagen, donde a cada uno de los mismo aplicamos la formula :

$$c \cdot (x, y)^{\gamma}$$

Donde $c = 8.5$ y $\gamma = 0.6$, dado que el realce de la imagen es subjetivo, probamos varios valores para c , hasta dar con el adecuado.

El resultado es el nuevo valor en (x,y).

- Regresamos la imagen objetivo.

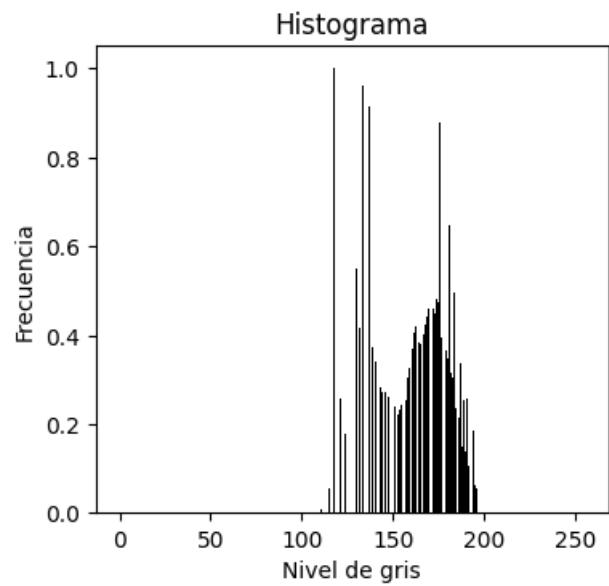
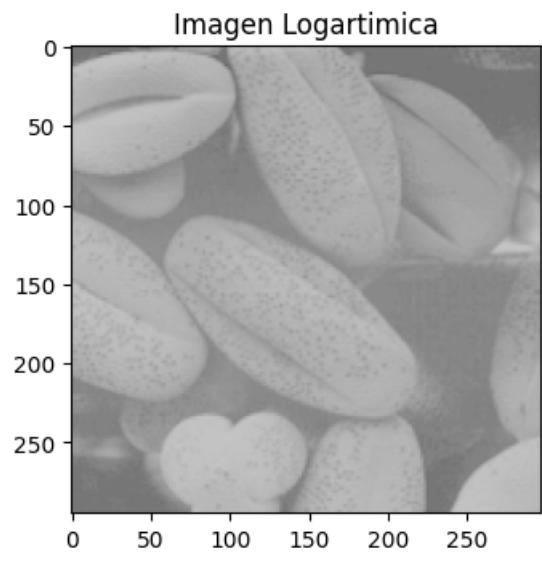
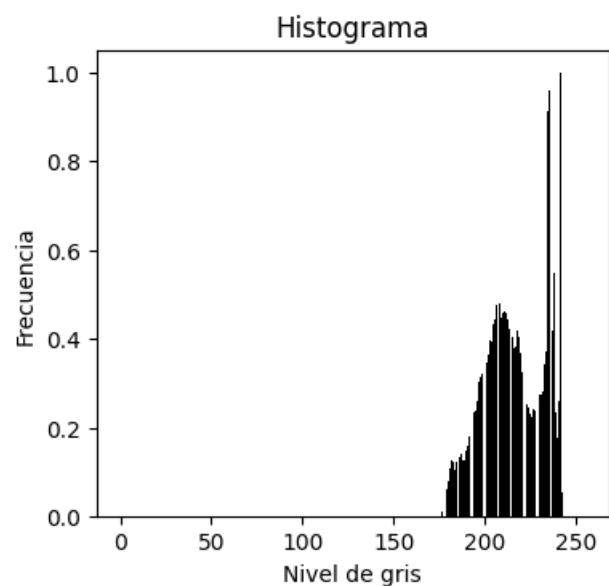
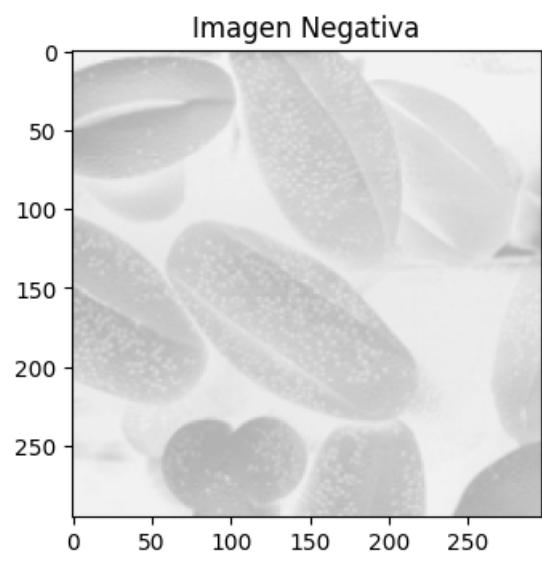
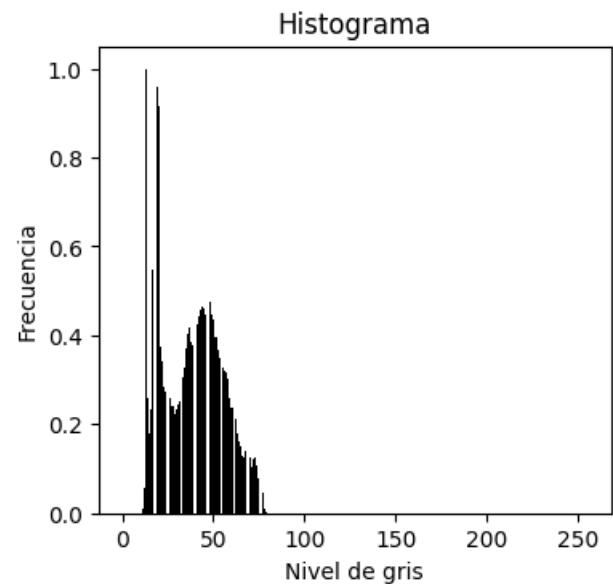
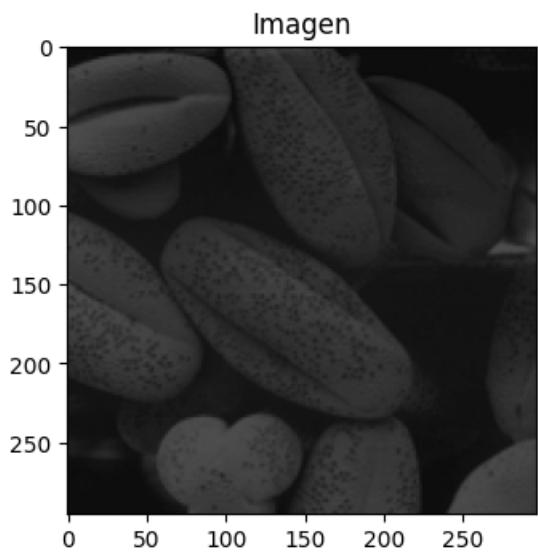
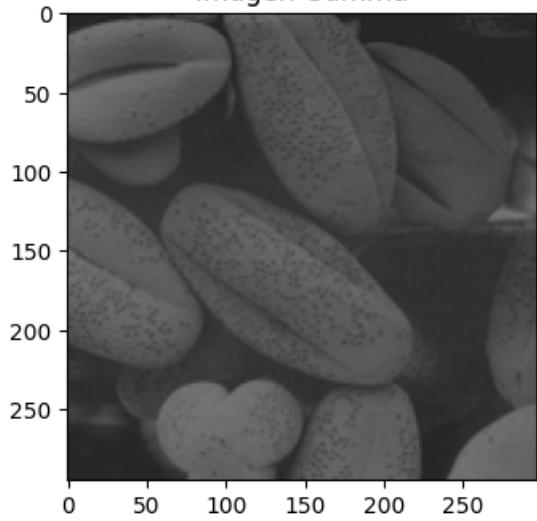
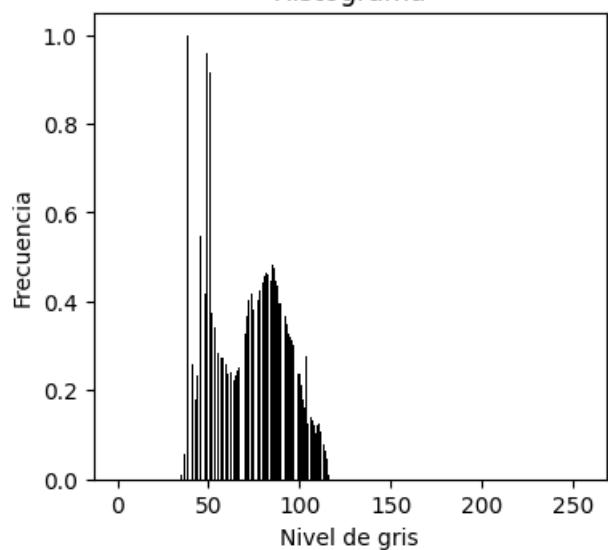


Imagen Gamma



Histograma



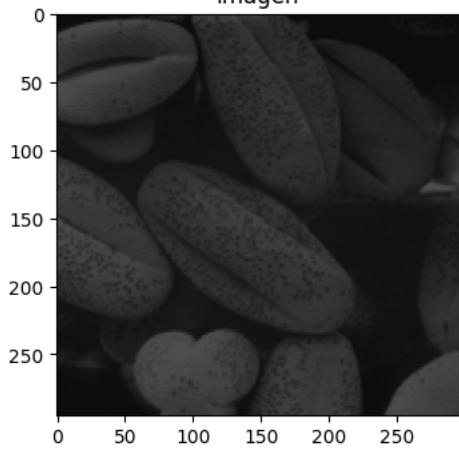
- (d) Ecualizar el histograma una imagen. Desplazar en la misma ventana: imagen original, su histograma, imagen ecualizada y su histograma.
- (e) Nota: Si trabajas en MATLAB está prohibido utilizar las funciones: imhist, imadjust, imcontrast, histeq.

Función : ecualizacion

Parámetros : imagen

- Creamos la imagen objetivo de acuerdo a las dimensiones de nuestra imagen original, en escala de grises.
Además de un arreglo llamado histograma ecualizado de ceros de tamaño 255.
- Sacamos el histograma de nuestra imagen y lo pasamos a la función de **pdf_cdf**, donde calculamos la pdf al dividir entre el numero total de píxeles y luego cdf la cual es la suma acumulada.
- Posteriormente tenemos un arreglo de los nuevos valores, en donde de nuevo recorremos la imagen original, donde por cada píxel en posición (x,y) , obtenemos su nivel de gris y será el índice dentro de los nuevos valores que nos dará su nuevo valor, el mismo nuevo valor será el índice dentro del arreglo ecualizado que aumentaremos en uno, y el nuevo valor literalmente será el nuevo valor del píxel en (x, y) .
De esta forma vamos creando el histograma ecualizado y la imagen ecualizada al mismo tiempo.

Imagen



Histograma

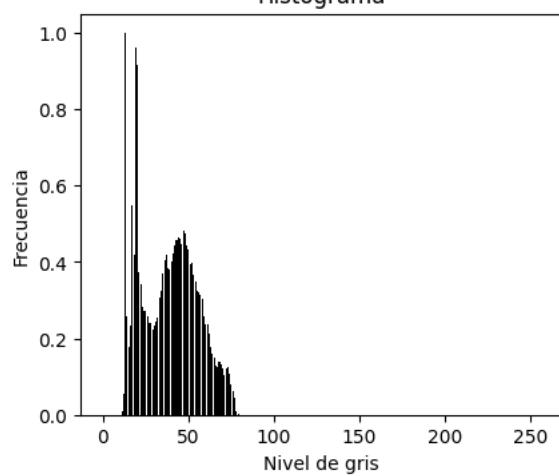
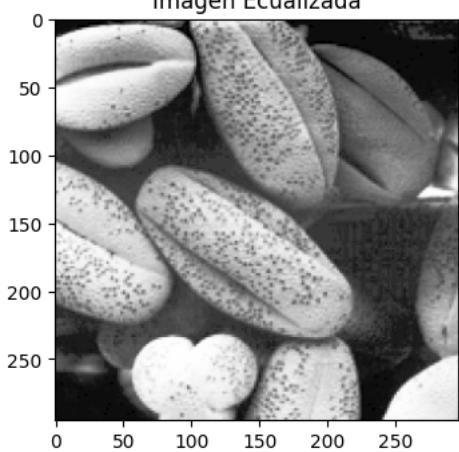
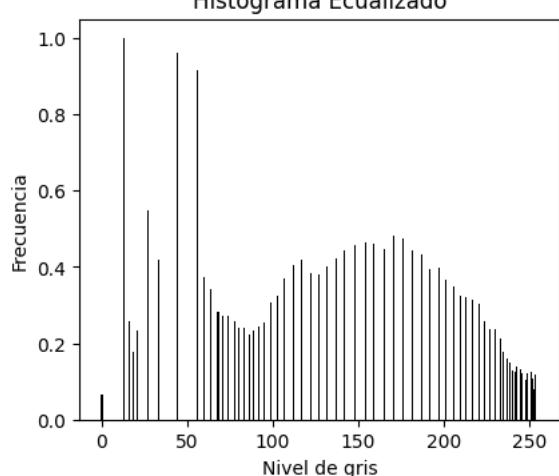


Imagen Ecualizada



Histograma Ecualizado



- Aplicar a una imagen sin ruido y la misma imagen con ruido filtros de suavizamiento y realce. La imagen con ruido se puede generar a partir de la imagen sin ruido usando el siguiente comando de MATLAB: $J = \text{imnoise}(I, \text{TIPO}, \dots)$, donde TIPO es una cadena que puede tomar valores 'gaussian', 'salt & pepper', etc. Ver comando help imnoise.
 - Aplicar los filtros paso bajas promedio estándar a la imagen sin ruido y a la imagen con ruido usando filtros de orden 3x3, 5x5, 7x7 y 11x11.

Función : filtro_pbp_estandar

Parámetros : orden,image

- * La técnica que usamos para el tratamiento de bordes es la técnica de **reducción de la imagen**.

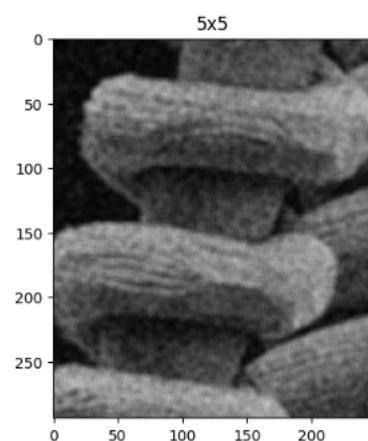
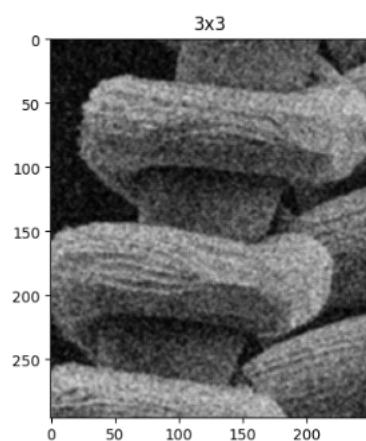
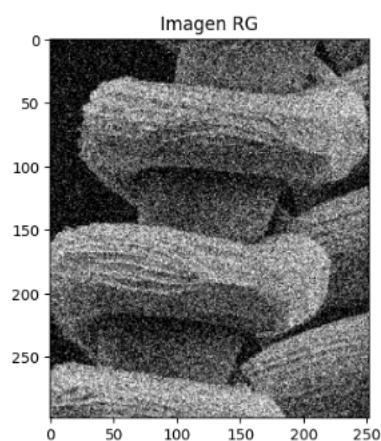
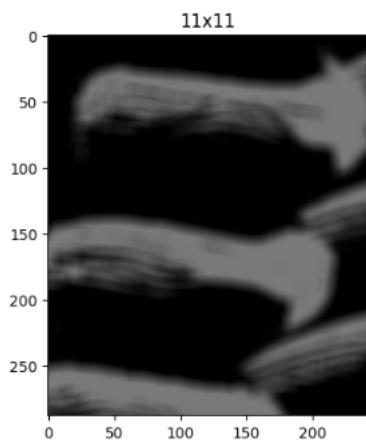
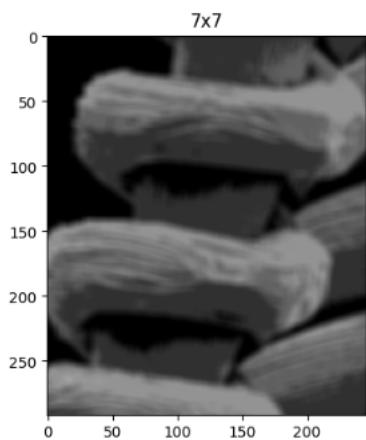
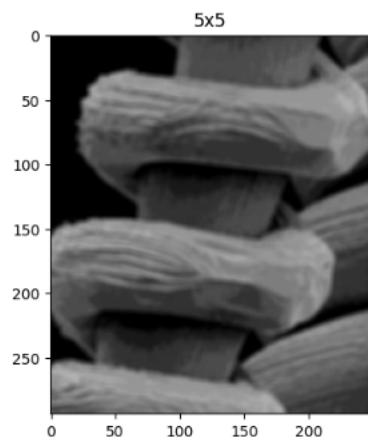
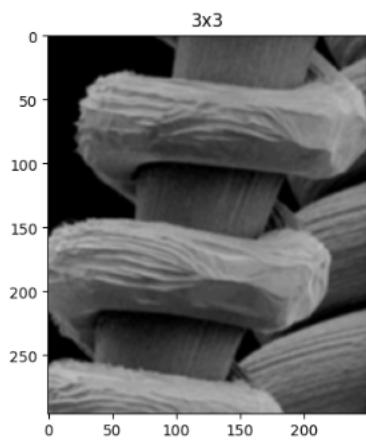
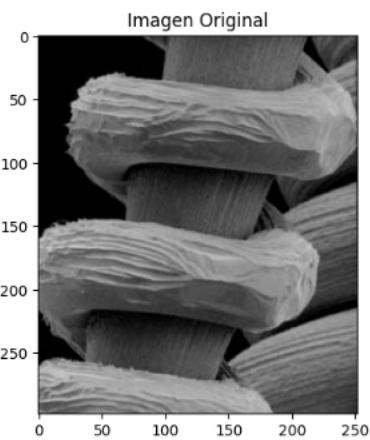
La explicaremos en esta función un poco más detallado pero a lo largo de este reporte se asumirá esto.

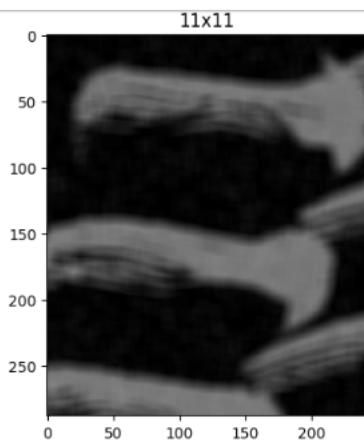
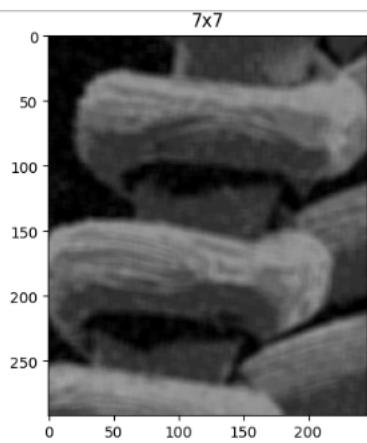
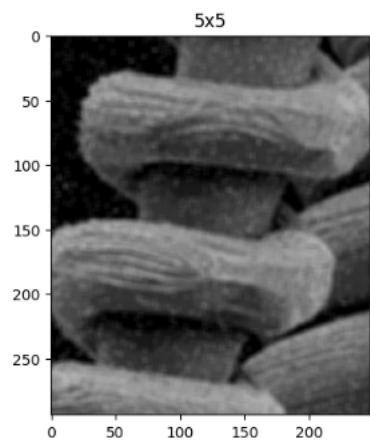
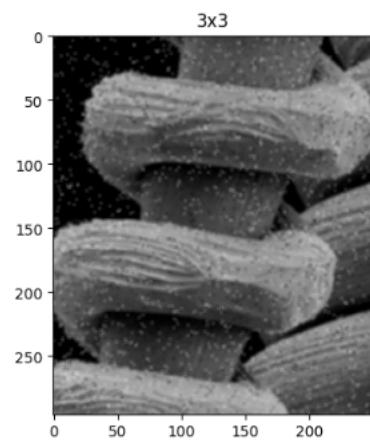
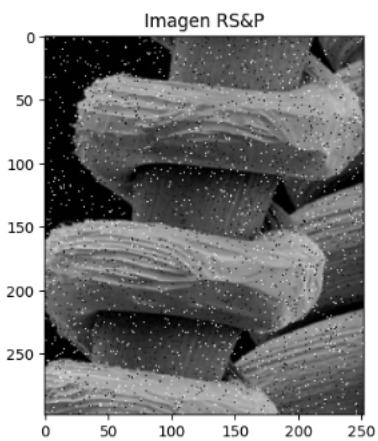
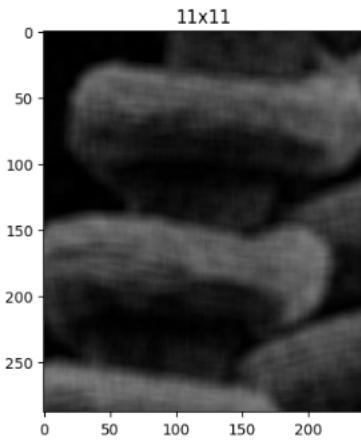
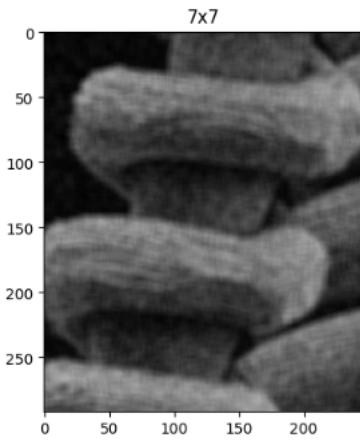
La aplicación se divide en dos partes :

- Resta de orden : La resta en *ancho x largo* de la imagen original para la imagen objetivo.
- Suma orden : La suma en las coordenadas dentro de la imagen objetivo para obtener su valor correspondiente en la imagen original.

La suma de orden y resta depende del orden nxn del filtro .

- * Creamos la imagen objetivo de acuerdo a la resta de orden, en escala de grises .
- * De acuerdo al orden llamamos a nuestra función **vecinos_nxn**, la cual nos da el área de vecinos de un píxel de acuerdo al orden del filtro que queremos.
- * Pasamos estos vecinos, la imagen original, un valor $n = \text{orden} * \text{orden}$, a la función auxiliar de convolución , donde cada píxel dentro de los vecinos de ese punto obtenidos de la imagen original se multiplican por 1, se dividen entre n , donde el resultado de ello será el nuevo valor de la coordenada (x, y) .
- * Regresamos la imagen objetivo.



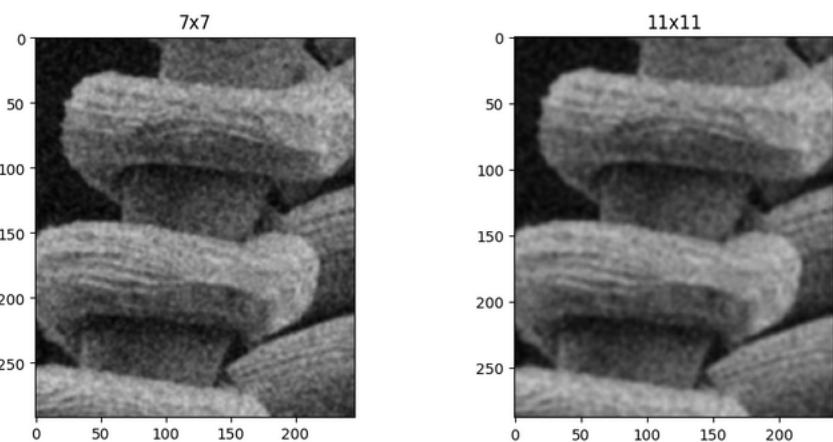
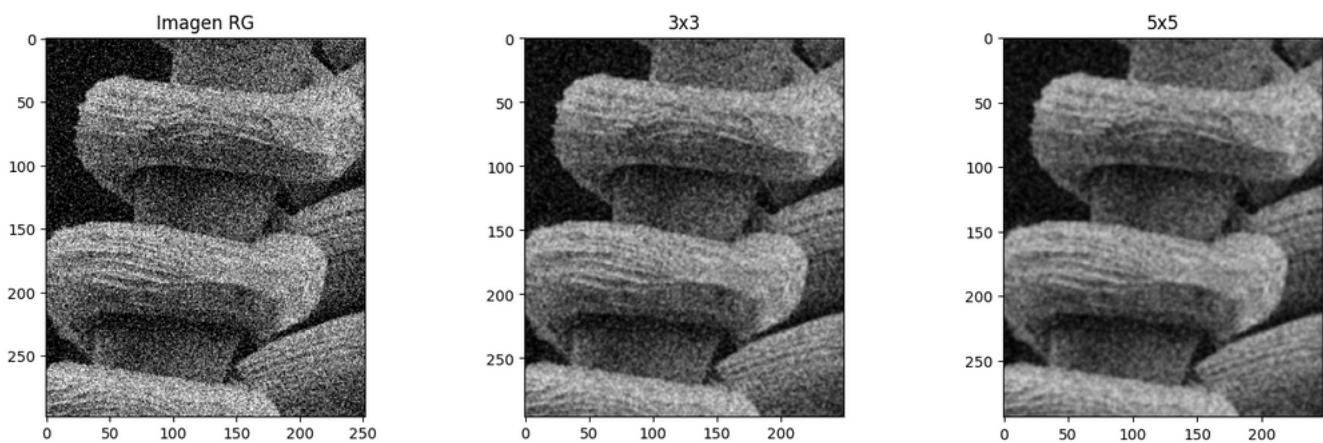
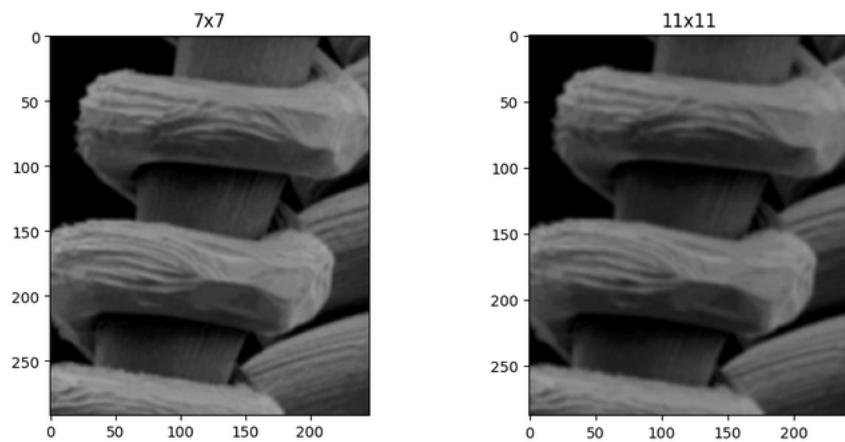
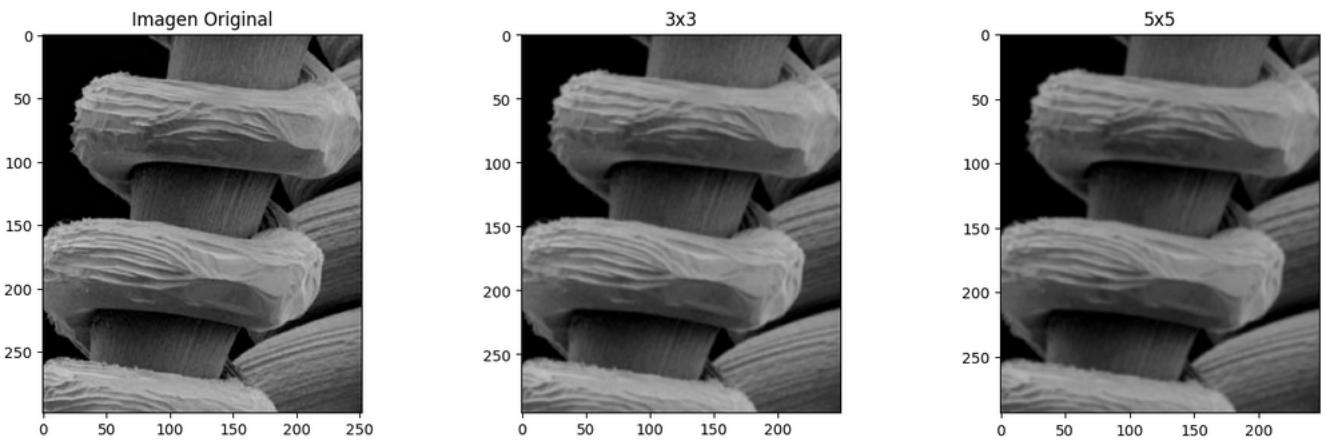


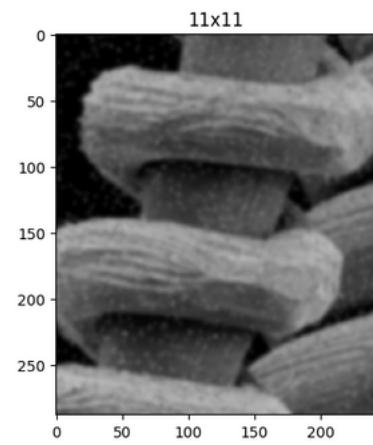
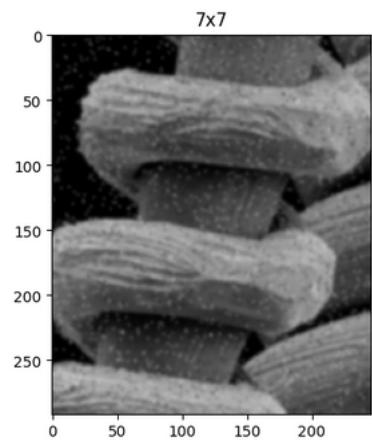
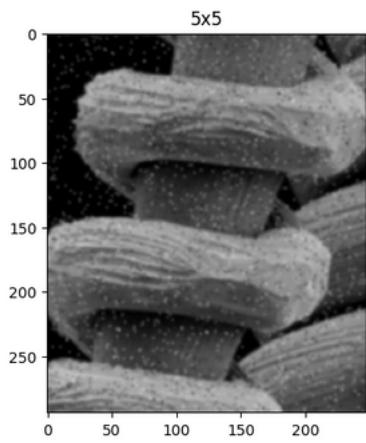
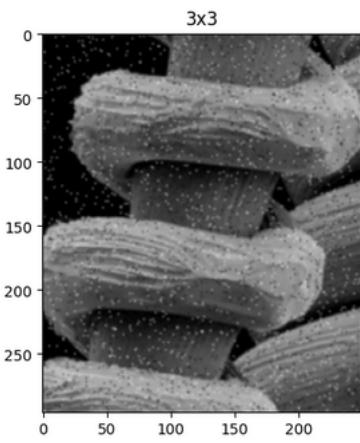
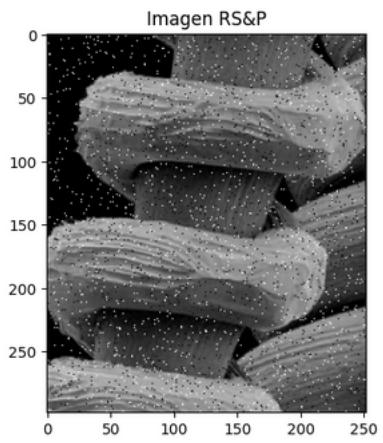
- Aplicar los filtros paso bajas promedio ponderado a la imagen sin ruido y a la imagen con ruido usando filtros de orden 3x3, 5x5, 7x7 y 11x11.

Función : filtro_pbp_ponderado

Parámetros : orden,image

- * Creamos la imagen objetivo de acuerdo a la resta de orden en escala de grises.
- * De acuerdo al orden llamamos a nuestra función **vecinos_nxn**, la cual nos da el área de vecinos de un píxel de acuerdo al orden del filtro que queremos.
- * Pasamos estos vecinos a la función auxiliar de convolución , donde cada píxel dentro de los vecinos de ese punto obtenidos de la imagen original se multiplican por el kernel gaussiano de acuerdo al orden el mismo será diferente para 3,5,7,11.
Por lo que para calcularlo nos apoyamos del triangulo de pascal , creamos una función que nos cree el triangulo de pascal, donde de acuerdo al orden obtiene la fila orden-1 digamos m, la misma se multiplicara así misma de forma vertical y horizontal, luego ese kernel (A_{nxn}) es multiplicado por la suma de los elementos de la fila m multiplicado por si mismo, ie $m*m$ entre 1 esto es $\frac{1}{m*m} \cdot A_{nxn}$.
El resultado será el nuevo valor del píxel en la coordenada (x, y) .
- * Regresamos la imagen objetivo.



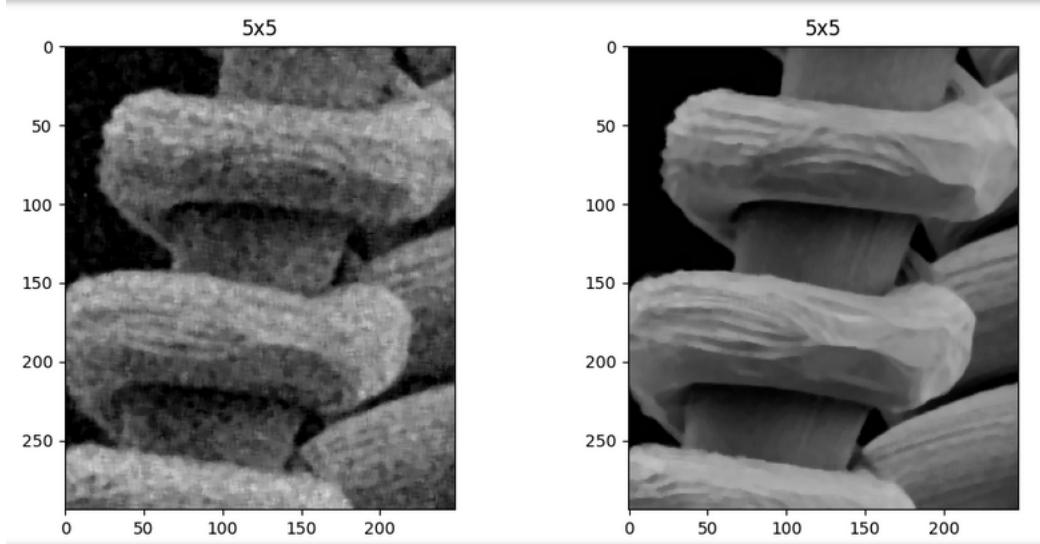
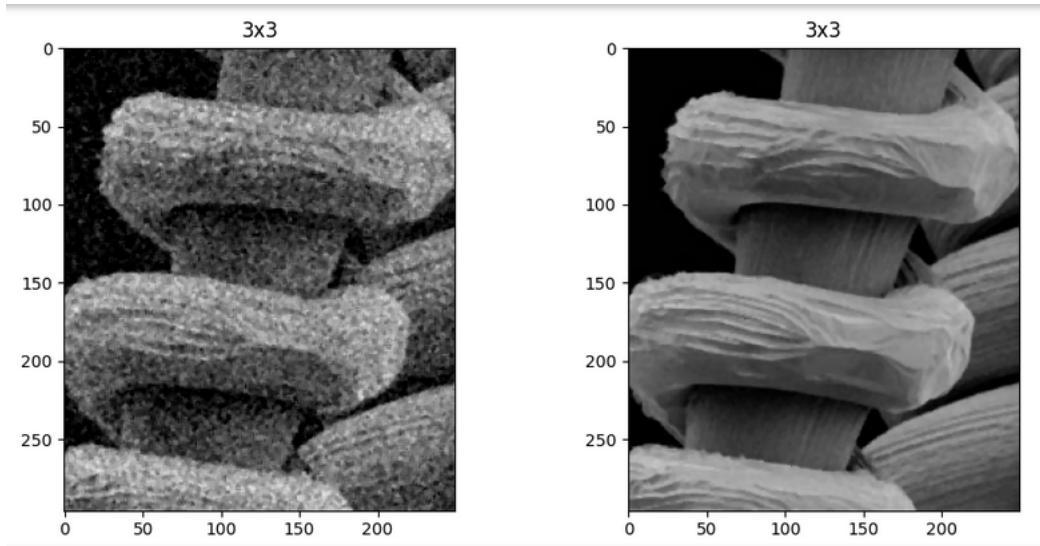
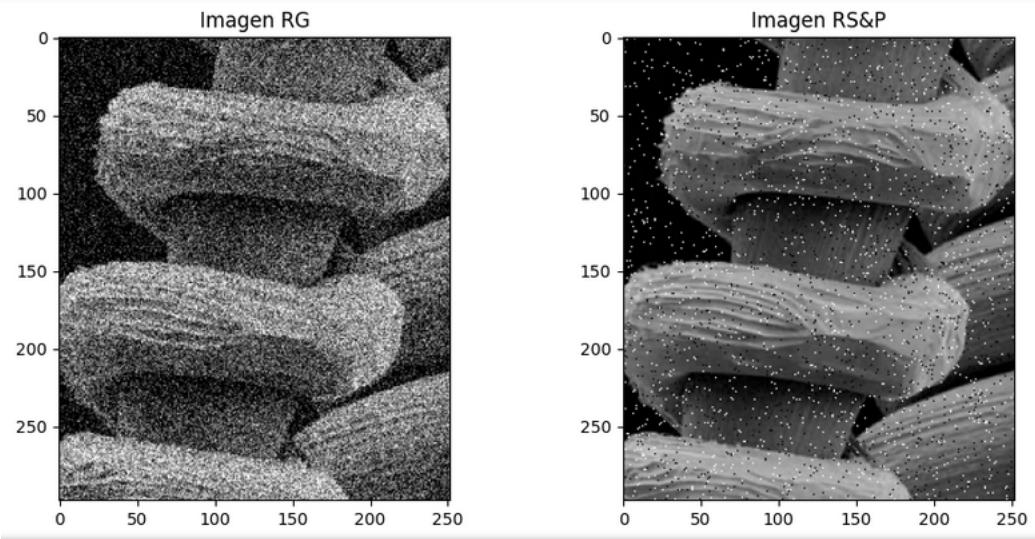


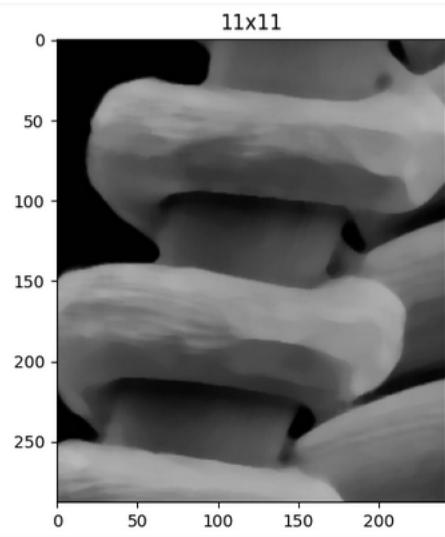
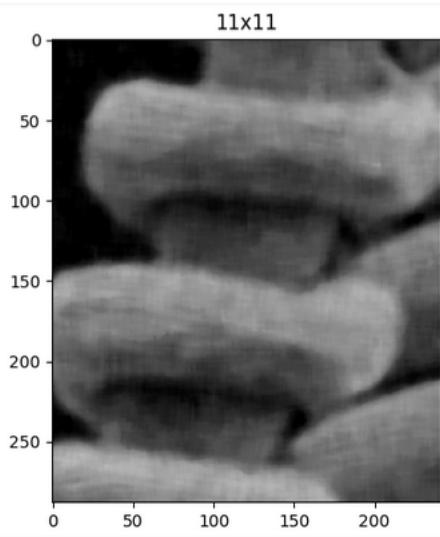
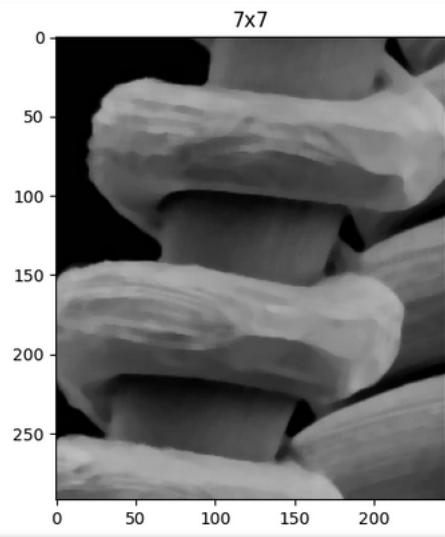
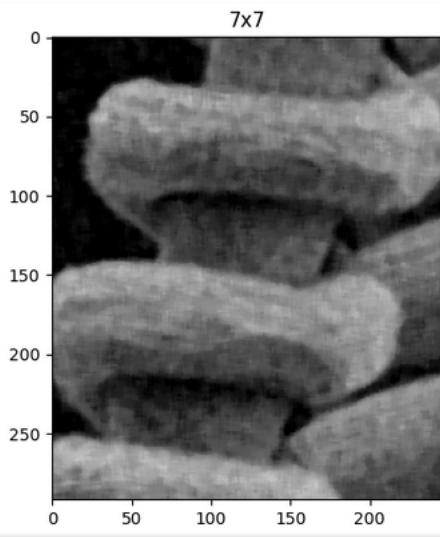
- Aplicar a la imagen con ruido el filtro mediana de 3x3, 5x5, 7x7 y 11x11. Utilizar ruido 'sal y pimienta' y 'gaussiano'. Compararlos.

Función : filtro_pbp_mediana

Parámetros : image,orden

- * Creamos la imagen objetivo de acuerdo a la resta de orden en escala de grises.
- * De acuerdo al orden llamamos a nuestra función `vecinos_nxn`, la cual nos da el área de vecinos de un píxel de acuerdo al orden del filtro que queremos.
- * Pasamos estos vecinos a la función auxiliar de convolución , donde cada píxel dentro de los vecinos de ese punto obtenidos de la imagen original se multiplica por uno, para posteriormente sacar la mediana de todos ellos. Donde esta mediana será el nuevo valor del píxel en la coordenada (x, y) .



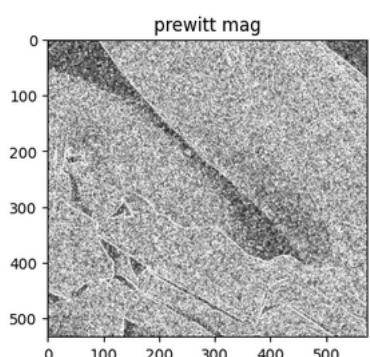
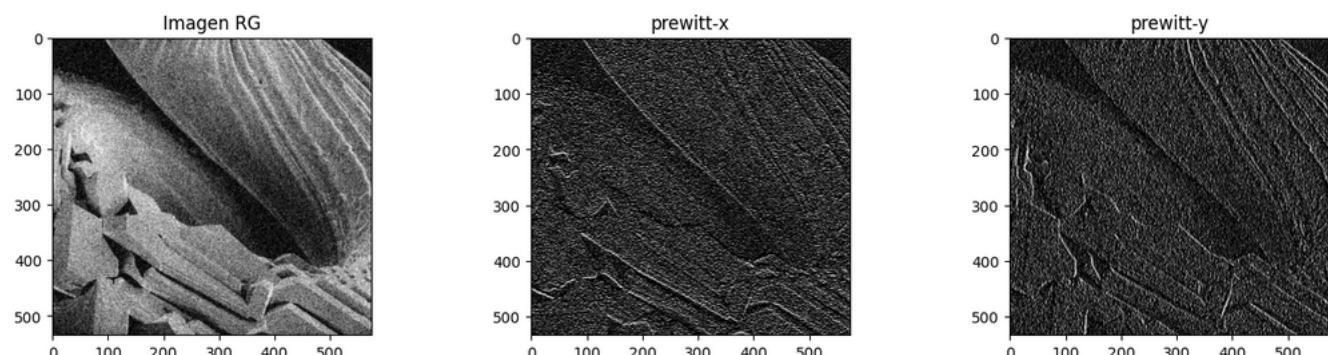
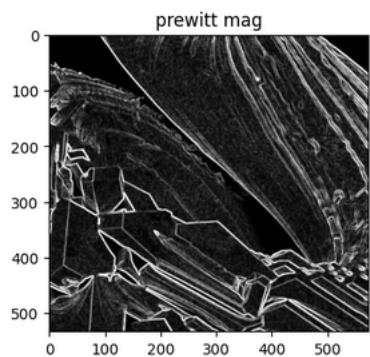
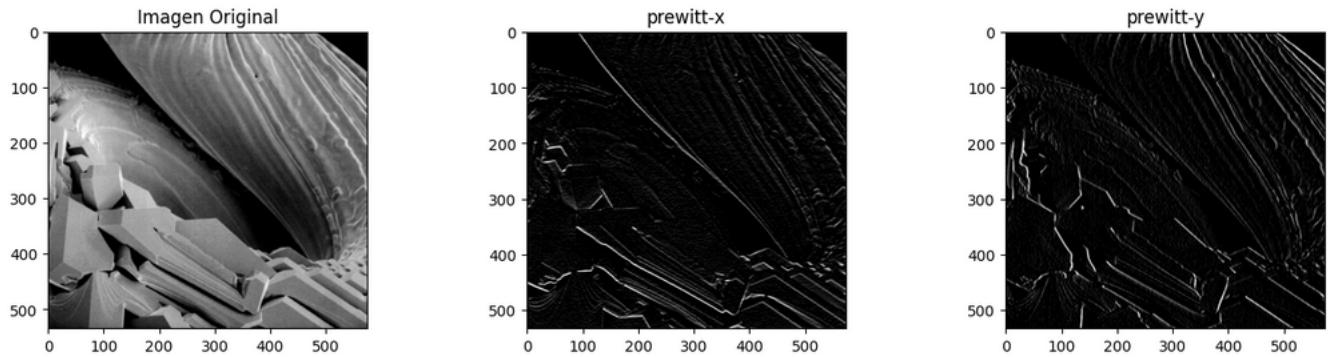


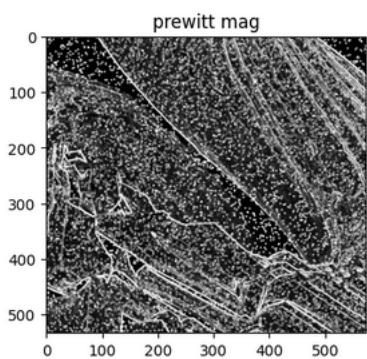
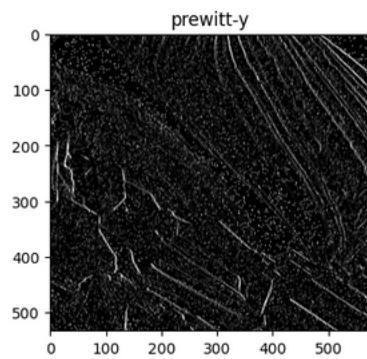
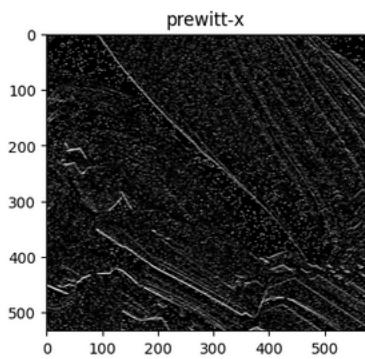
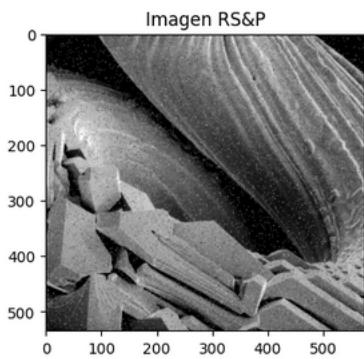
- Aplicar a la imagen sin ruido y con ruido los filtros basados en la primera derivada o detectores de borde siguientes :
 - * Prewitt en la dirección X y en la dirección Y , así como su magnitud.

Función : filtro_pbp_Prewitt

Parámetros : image

- Creamos la imagen objetivo de acuerdo a la resta de orden en escala de grises.
- De acuerdo al orden llamamos a nuestra función `vecinos_3xn 3`, la cual nos da el área de vecinos de 3x3 de un píxel.
- Pasamos estos vecinos a la función auxiliar de convolución en dirección de y y x , donde cada píxel dentro de los vecinos de ese punto obtenidos de la imagen original se multiplica por los valores correspondientes de los filtros en dirección de x y y , para posteriormente la suma del valor con valor absoluto de ambas, la cual representa la magnitud.
La misma magnitud el nuevo valor del píxel con coordenadas (x,y).
- Regresamos la imagen objetivo .



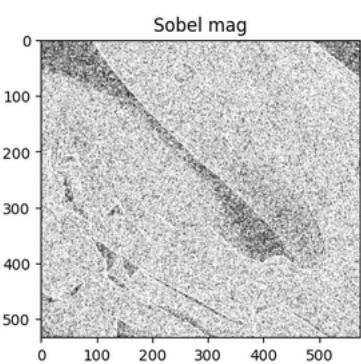
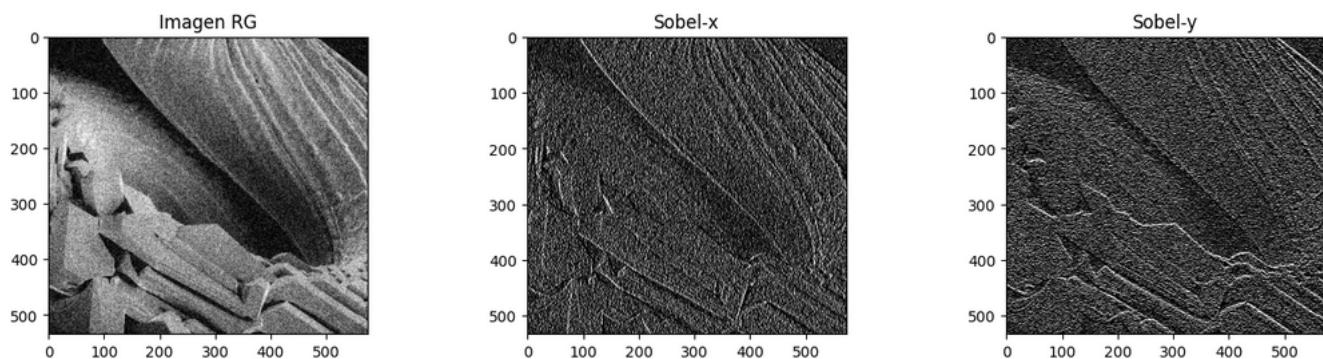
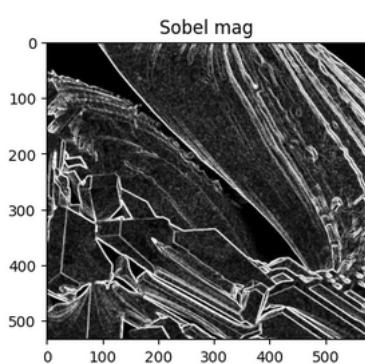
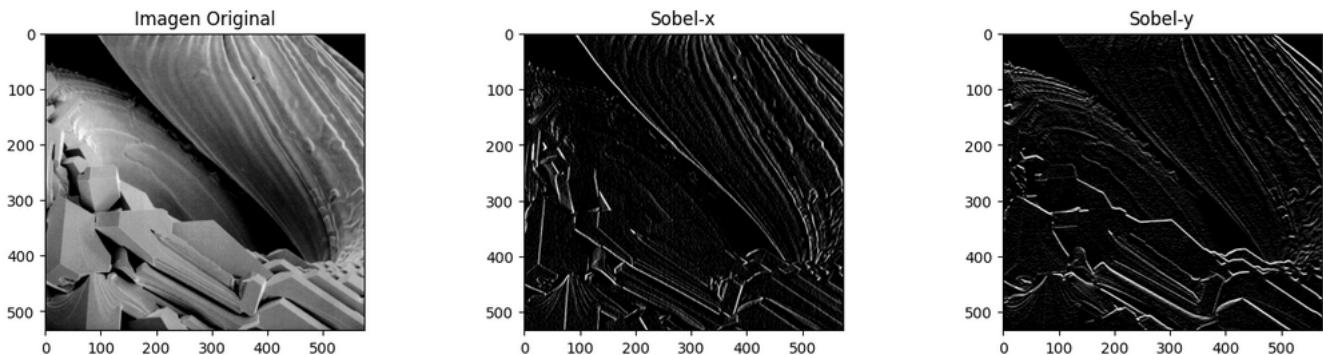


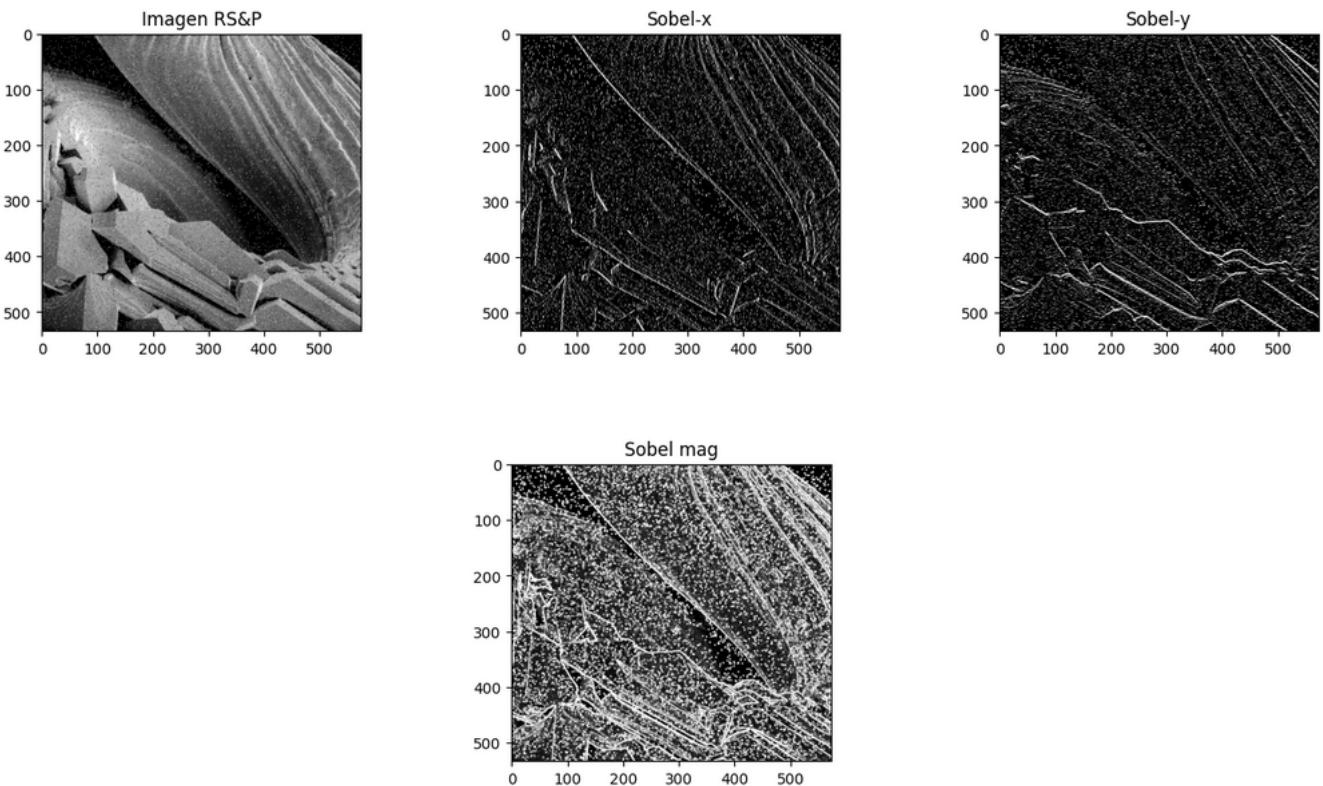
- * Sobel en la dirección X y en la dirección Y , así como su magnitud.

Función : filtro_pbp_sobel

Parámetros : image

- Creamos la imagen objetivo de acuerdo a la resta de orden en escala de grises.
- De acuerdo al orden llamamos a nuestra función `vecinos_3xn 3`, la cual nos da el área de vecinos de 3x3 de un píxel.
- Pasamos estos vecinos a la función auxiliar de convolución en dirección de y y x , donde cada píxel dentro de los vecinos de ese punto obtenidos de la imagen original se multiplica por los valores correspondientes de los filtros en dirección de x y y , para posteriormente la suma del valor con valor absoluto de ambas, la cual representa la magnitud.
- La misma magnitud el nuevo valor del píxel con coordenadas (x,y).
- Regresamos la imagen objetivo.



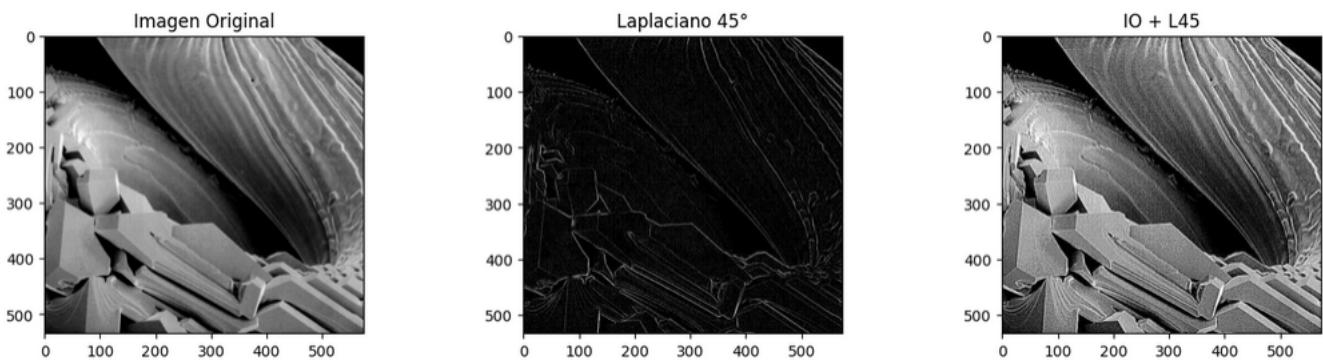


- Aplicar a una imagen el realce basado en las segundas derivadas (Laplaciano). Isotrópicos a 45 grados así como a 90 grados.

Función : laplaciano_45

Parámetros : image, metodo

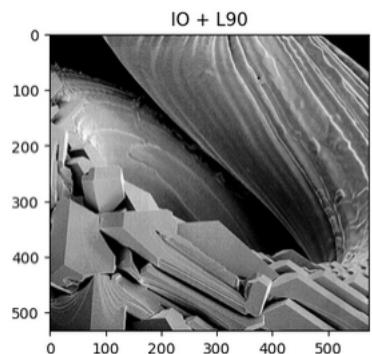
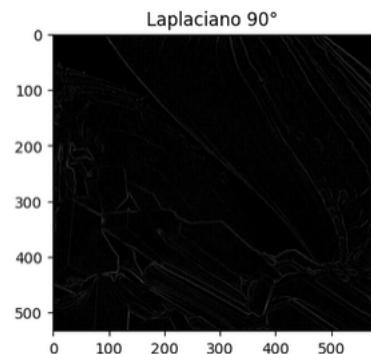
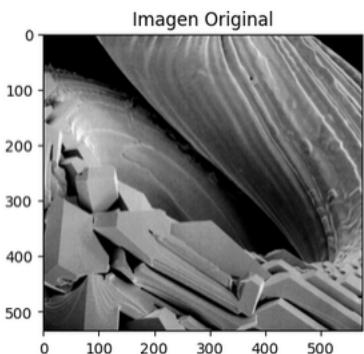
- * Creamos la imagen objetivo de acuerdo a la resta de orden en escala de grises.
- * De acuerdo al orden llamamos a nuestra función **vecinos_3xn 3**, la cual nos da el área de vecinos de 3x3 de un píxel.
- * Tenemos los siguientes casos :
 - Si el método es 'i', es un laplaciano individual, por lo que el filtro que aplicamos es un filtro negativo de 45 grados, a cada uno de los vecinos del píxel (x, y) y a su vez los vamos sumando, de manera que esa suma es el nuevo valor de nuestro píxel (x, y) (convolución).
 - Si el método es 'c', es un laplaciano compuesto, por lo que el filtro que aplicamos es un filtro negativo de 45 grados, con la diferencia de que el centro del filtro se le suma uno, dado que consideramos como si se sumará la imagen original y mascará del laplaciano de 45 grados.
- Entonces este filtro se aplica a cada uno de los vecinos del píxel (x, y) y a su vez los vamos sumando, de manera que esa suma es el nuevo valor de nuestro píxel (x, y) (convolución)
- Pasamos estos vecinos a la función auxiliar de convolución en dirección de y y x , donde cada píxel dentro de los vecinos de ese punto obtenidos de la imagen original se multiplica por los valores correspondientes de los filtros en dirección de x y y , para posteriormente la suma del valor con valor absoluto de ambas, la cual representa la magnitud.
- La misma magnitud el nuevo valor del píxel con coordenadas (x, y) .
- * Regresamos la imagen objetivo.



Función : laplaciano_90

Parámetros : image

- * Creamos la imagen objetivo de acuerdo a la resta de orden en escala de grises.
- * De acuerdo al orden llamamos a nuestra función `vecinos_3xn 3`, la cual nos da el área de vecinos de 3x3 de un píxel.
- * Tenemos los siguientes casos :
 - Si el método es 'i', es un laplaciano individual, por lo que el filtro que aplicamos es un filtro negativo de 45 grados, a cada uno de los vecinos del píxel (x,y) y a su vez los vamos sumando, de manera que esa suma es el nuevo valor de nuestro píxel (x,y), (convolución).
 - Si el método es 'c', es un laplaciano compuesto, por lo que el filtro que aplicamos es un filtro negativo de 90 grados, con la diferencia de que el centro del filtro se le suma uno, dado que consideramos como si se sumará la imagen original y mascará del laplaciano de 90 grados.
Entonces este filtro se aplica a cada uno de los vecinos del píxel (x,y) y a su vez los vamos sumando, de manera que esa suma es el nuevo valor de nuestro píxel (x,y) (convolución).
- Pasamos estos vecinos a la función auxiliar de convolución en dirección de y y x , donde cada píxel dentro de los vecinos de ese punto obtenidos de la imagen original se multiplica por los valores correspondientes de los filtros en dirección de x y y , para posteriormente la suma del valor con valor absoluto de ambas, la cual representa la magnitud.
- La misma magnitud el nuevo valor del píxel con coordenadas (x, y) .
- * Regresamos la imagen objetivo.

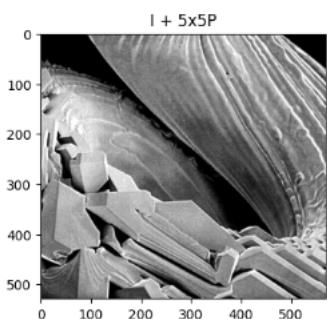
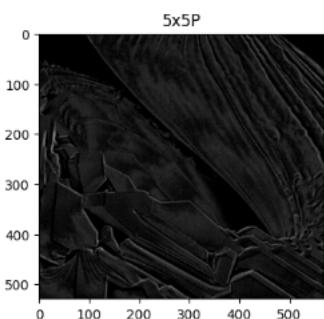
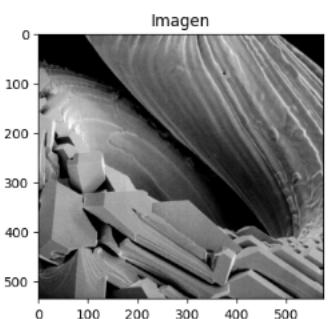
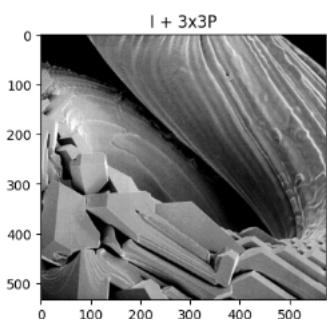
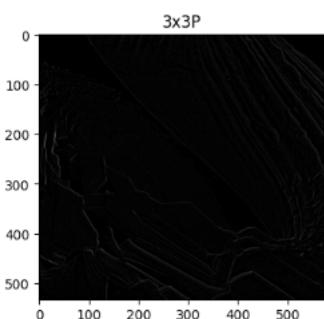
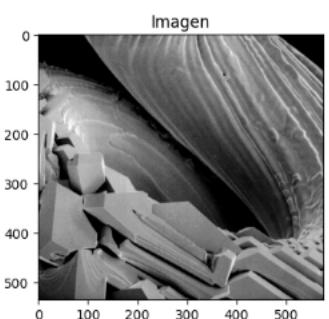
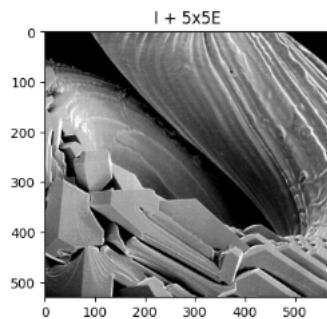
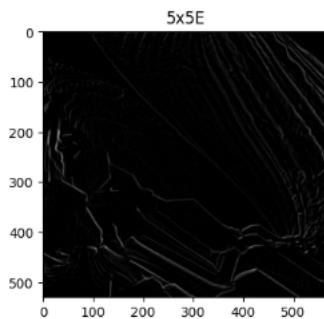
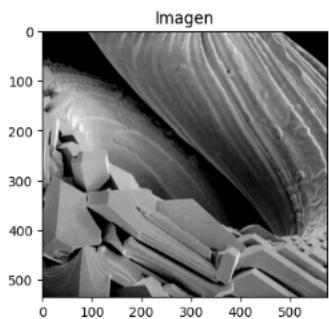
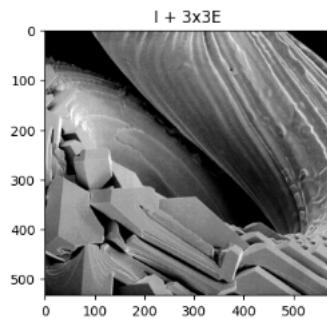
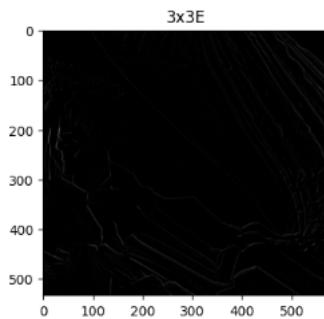
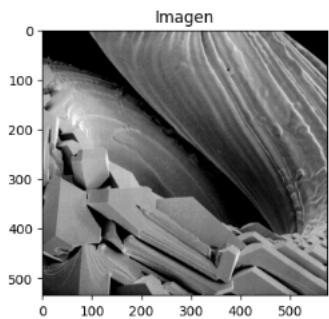


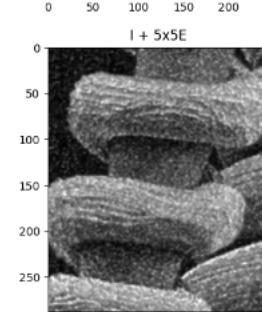
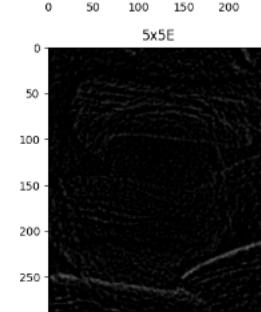
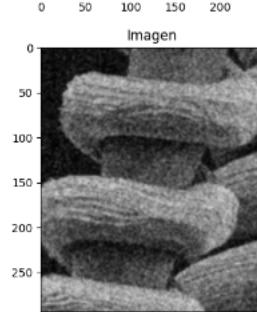
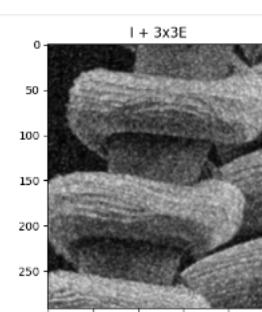
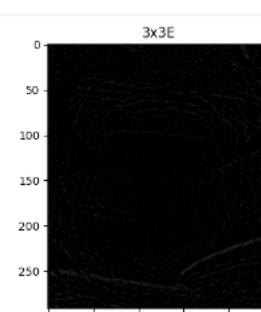
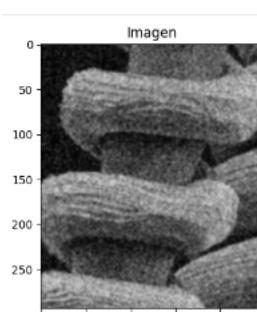
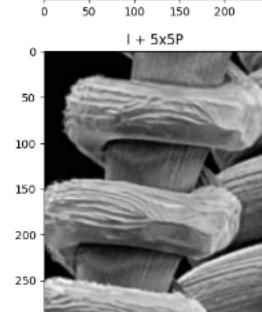
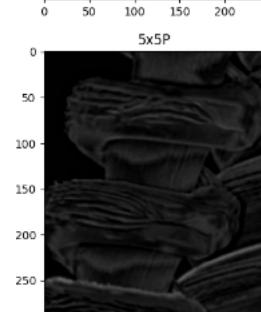
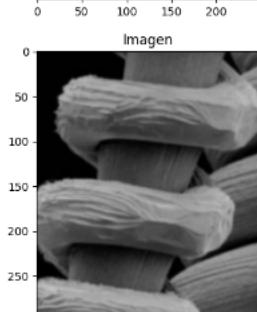
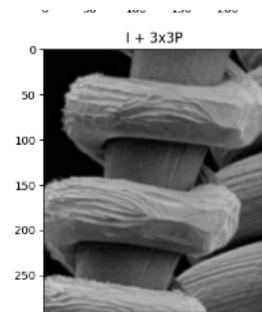
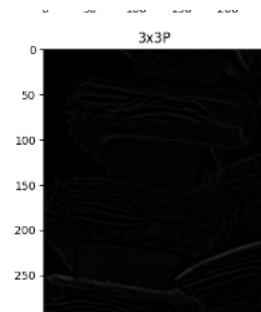
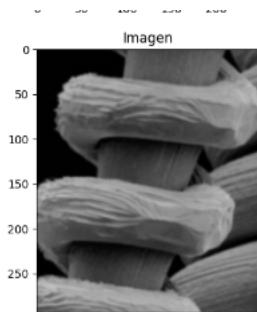
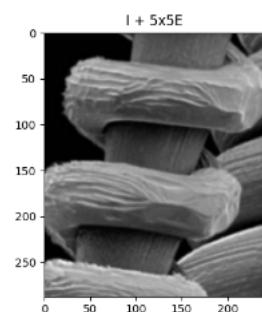
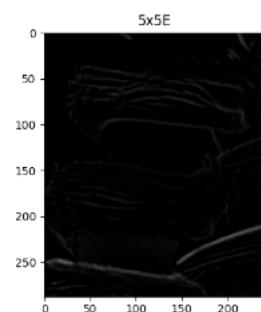
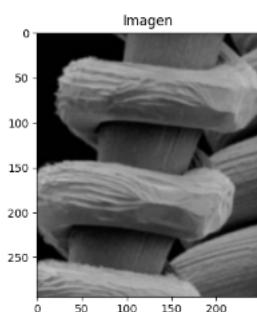
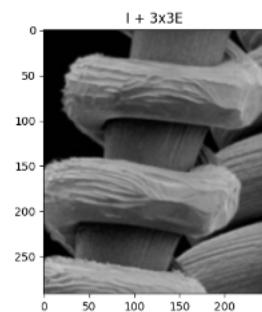
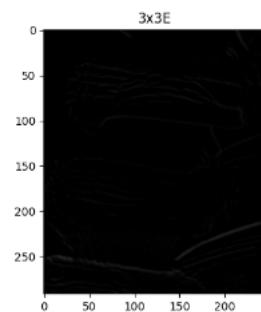
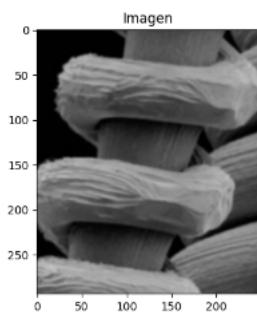
- Difuminar las imágenes sin ruido y con ruido usando un filtro paso bajas de orden 5x5, de tal manera que se obtenga una imagen sin ruido y con pérdida de nitidez y otra imagen con ruido y perdida de nitidez. Para cada uno de los siguientes incisos, filtrar las imágenes utilizando el filtro unsharp masking encontrado con los siguientes tipos de filtro paso bajas:
 - * Filtro paso bajas promedio estándar de orden 3x3 y 7x7.
 - * Filtro paso bajas promedio ponderado de orden 3x3 y 7x7.

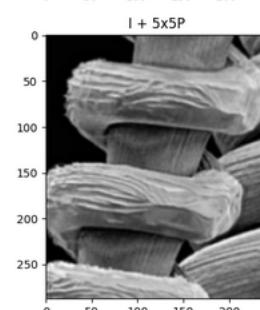
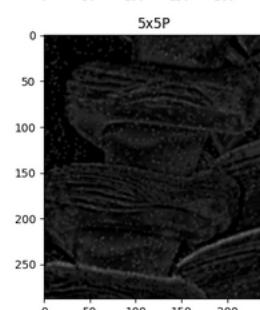
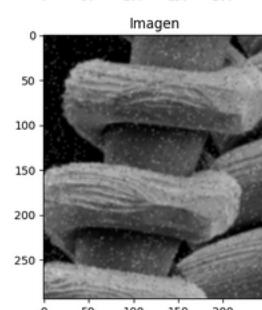
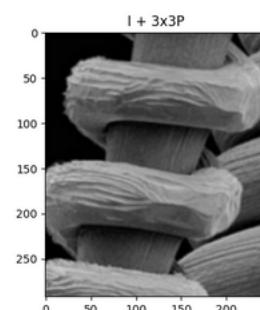
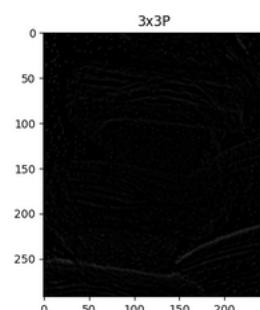
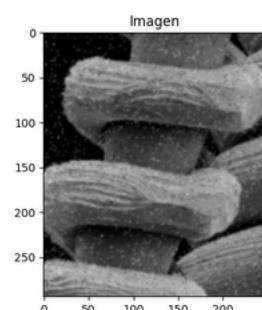
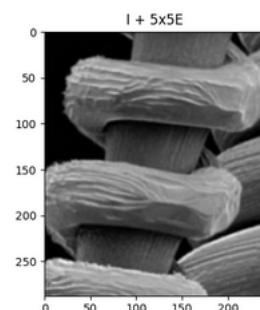
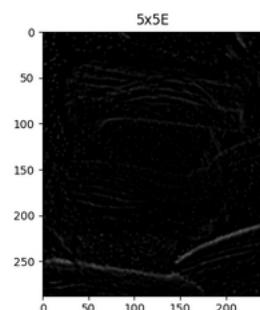
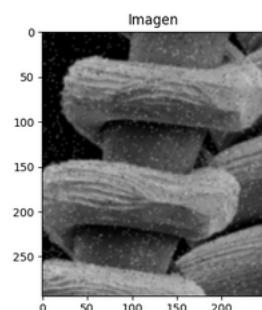
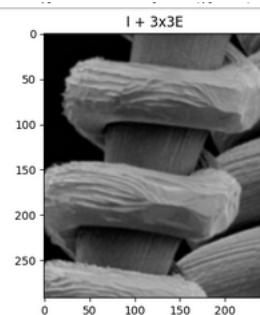
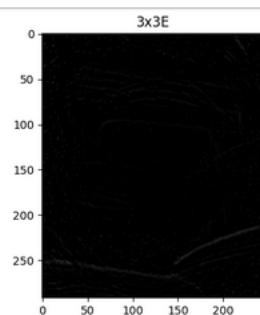
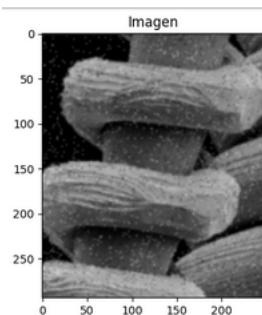
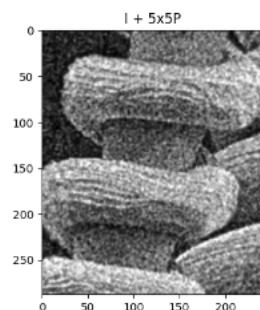
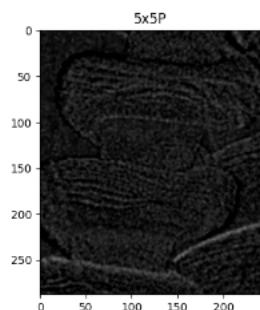
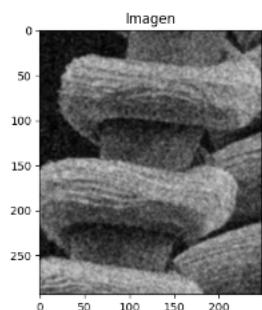
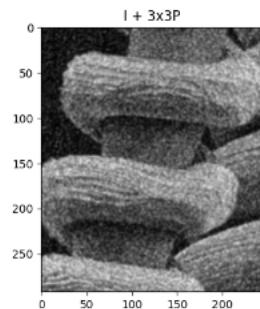
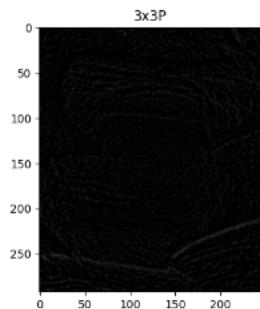
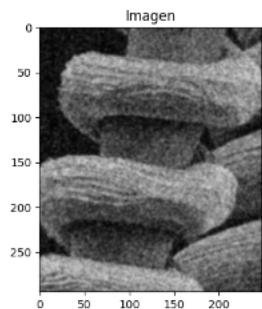
Función : mask_unsharpening

Parámetros : img_original,img_emborroneada

- Restamos a la imagen original, su versión emborroneada, de forma que creamos una nueva imagen en la que la diferencia de píxel a píxel entre ambas imágenes será el nuevo valor del píxel en la coordenada (x, y) .
Donde para asegurar que este dentro de un rango de (x, y) , los valores menores a 0 se multiplican por el mínimo, mientras que los mayores a 255 se bajan a 255.
- Regresamos la imagen objetivo.







5 Código

Observación : en esta práctica se usaron las imágenes de la carpeta de imágenes que proporciona la profesora.

- **NOTA**

En caso de requerir ver o correr de nuevo el código de cada una de las funciones mencionadas en este reporte, el mismo se proporcionará con el nombre **Practica03_PDI.ipynb**, junto con las imágenes usadas en la misma y mencionadas en este reporte.

6 Conclusiones

En general puedo decir que esta práctica me sirvió bastante para poder entender los temas del capítulo 3, dado que en las clases me hacía una idea de los filtros y en general de las convoluciones, pero el programarlo hizo que todo quedaría demasiado claro.

En general pude notar más de acuerdo a las clases es lo siguiente :

- En efecto al hacer el histograma ecualizado, se hace una distribución de la frecuencia y cambia la imagen para que se vea más uniforme.
- Para el promedio estandar pude ver más emborronamiento que en el promedio estándar, pero ambas hacen el emborronamiento.
- Pude ver más claro como en efecto el filtro mediana quito por completo el ruido sal y pimienta, pero en el caso del ruido gaussiano no quito casi nada, solo se suavizo la imagen, pero entre más aumente el orden se emborrona cada vez más, aunque sea el ruido sal y pimienta, por lo que conviene solo para un orden pequeño, para quitar el ruido sal y pimienta.
- En el caso de los filtro de realce, en imágenes que tienen ruido no sirven de mucho por que realzan el ruido junto con bordes, pero en general los filtro que se usen dependerá de lo que se quiera ver, si solo se quiere ver los bordes realizados y demás Sobel y Prewitt son la opción, pero si se quiere aplicar a la imagen los laplacianos son la opción, aunque de igual forma unsharp es útil en esta parte como los laplacianos.

Por lo que de igual forma sirve mucho para esta parte subjetiva de el realce en cierto punto; Me hizo comprender mucho esta parte en comparación con la restauración que estamos viendo actualmente el cual es más objetivo.

De igual forma de nuevo pude comprender más técnicas de implementación en python, que no sabía y pude saber más, que en particular me sirven demasiado para otros proyectos que puedo implementar.

En mi caso esta práctica fue más de mi agrado que la anterior, ambas fueron útiles pero esta además fue más entretenida.

7 Referencias

- 1. Gonzalez, R., Woods, R., Digital Image Processing, Prentice Hall, 2008.