CENTRO DE ENSEÑANZA TECNICA INDUSTRIAL
PRACTICAS DE LABORATORIO

INGENIERÍA CARRERA	PLAN DE ESTUDIO	CLAVE ASIGNATURA	NOMBRE DE LA ASIGNATURA
INGENIERÍA DE DESARROLLO DE SOFTWARE	2025-B	19SDS32	Procesamiento de Imágenes

ALUMNO		FECHA		EVALUACIÓN	

PRÁCTICA No.	LABORATORIO DE COMPUTACIÓN No	NOMBRE DE LA PRÁCTICA	DURACIÓN (HORAS)
5	LCS	FILTROS DE LA MEDIANA	2

INTRODUCCIÓN

El procesamiento digital de imágenes es una disciplina fundamental en campos como la visión por computadora, el análisis de imágenes médicas y la fotografía digital. Una de las problemáticas más comunes en este ámbito es la presencia de ruido en las imágenes, que puede distorsionar la calidad visual y afectar la precisión de los análisis posteriores. Entre los distintos tipos de ruido que pueden afectar una imagen, el ruido Sal y Pimienta es uno de los más frecuentes. Este ruido se manifiesta como píxeles aleatorios que toman valores extremos, es decir, 0 (negro) o 255 (blanco), lo que genera un patrón visual de puntos dispersos sobre la imagen.

La presencia de este tipo de ruido es especialmente problemática en imágenes capturadas por cámaras digitales, dispositivos de escaneo o sistemas de transmisión que experimentan interferencias. La eliminación de este ruido es crucial para mejorar la calidad visual de la imagen y facilitar su posterior análisis. Los filtros de suavizado, como los de media y Gaussiano, son herramientas útiles para reducir el ruido en general, pero pueden no ser efectivos para el ruido Sal y Pimienta debido a su naturaleza extrema. En estos casos, el filtro de mediana ha demostrado ser uno de los métodos más eficaces.

OBJETIVO (COMPETENCIA)

El objetivo principal de esta práctica es comparar la eficacia de varios filtros de imagen en la eliminación del ruido Sal y Pimienta en una imagen. En particular, se aplicarán los filtros de Box, Gaussiano, Suavizado 3x3, Laplaciano y de Mediana, y se evaluará su desempeño en la restauración de la calidad visual de la imagen. Al final, se determinará cuál de estos filtros es el más adecuado para mitigar el ruido sin perder detalles importantes en la imagen, como los bordes.

FUNDAMENTO

El ruido Sal y Pimienta es un tipo de distorsión impulsiva que afecta a las imágenes mediante la inserción de píxeles de intensidad máxima (255, blanco) y mínima (0, negro) en posiciones aleatorias. Este tipo de ruido puede surgir debido a fallos en el sistema de captura de la imagen, errores de transmisión o problemas en los sensores de la cámara. El efecto visual es el de pequeñas manchas negras y blancas que distorsionan la imagen original, lo que puede dificultar su análisis y procesamiento posterior.

Para eliminar este ruido, uno de los métodos más efectivos es el filtro de mediana. A diferencia de otros filtros, como el de media o el Gaussiano, que suavizan la imagen calculando el promedio ponderado o simple de los píxeles circundantes, el filtro de mediana funciona seleccionando el valor mediano de una vecindad de píxeles. Al ordenar los valores de intensidad dentro de una ventana 3x3 (o de otro tamaño) alrededor de cada píxel, el filtro de mediana reemplaza el valor del píxel central con el valor mediano de esa vecindad.

La ventaja principal de este enfoque es que la mediana es menos sensible a valores extremos como los píxeles de 0 o 255 que caracterizan el ruido Sal y Pimienta. De este modo, el filtro de mediana puede eliminar estos valores extremos sin difuminar los bordes de la imagen, lo que lo convierte en la opción más adecuada para restaurar imágenes afectadas por este tipo de ruido.

METODOLOGÍA (DESARROLLO DE LA PRACTICA)

El código realiza el siguiente proceso:

Lectura de la imagen: Se carga una imagen en formato RGB y luego se convierte a escala de grises para su procesamiento.
Adición de ruido Sal y Pimienta: La función salPimienta inserta píxeles con valores aleatorios de 0 (pimienta) y 255 (sal) en la imagen para simular el ruido.
Aplicación de filtros: Se prueban cinco filtros diferentes para comparar su efectividad en la reducción del ruido:
Visualización: Las imágenes resultantes, tanto la original con ruido como las imágenes procesadas por los filtros, se muestran en una cuadrícula de 2x3 para facilitar la comparación.

Código:

```
Im = imread('carro.jpg');
In = rgb2gray(Im);
% Mostrar la imagen con sal y pimienta
figure;
In = salPimienta(In);
```

```
subplot(2, 3, 1); % Posición 2 de la cuadrícula 3x2
imshow(In);
title('Imagen con Sal y Pimienta');
% Mostrar la imagen suavizada (Media 3x3)
subplot(2, 3, 2); % Posición 3 de la cuadrícula 3x2
imagenSuvizada = suavizado(In);
imshow(imagenSuvizada);
title('Imagen Suavizada (Media 3x3)');
% Mostrar la imagen con el filtro Box
subplot(2, 3, 3); % Posición 2 de la cuadrícula 3x2
imagenBox = box(In);
imshow(imagenBox);
title('Filtro Box con Bordes Negros');
% Mostrar la imagen con el filtro Gaussiano
subplot(2, 3, 4); % Posición 3 de la cuadrícula 3x2
imagenGauss = gauss(In);
imshow(imagenGauss);
title('Filtro Gaussiano 5x5 (sigma=1)');
% Mostrar la imagen con el filtro Laplaciano
subplot(2, 3, 5); % Posición 4 de la cuadrícula 3x2
imagenLaplace = laplace(In);
imshow(imagenLaplace);
title('Filtro Laplaciano 5x5');
% Mostrar la imagen con el filtro de la Mediana aplicado
subplot(2, 3, 6); % Posición 6 de la cuadrícula 3x2
imagenMediana = mediana(In);
imshow(imagenMediana);
title('Filtro de la Mediana 5x5');
%%%%%%%%%%%%%%%%%% Funciones %%%%%%%%%%%%%%%%%
function OutSuavizado = suavizado(grayIm)
   [FIL, COL] = size(grayIm);
   OutSuavizado = zeros(FIL, COL);
   filtro_suavizado = ones(3, 3) / 9;
   for rxp = 2:FIL-1
       for ryp = 2:COL-1
           % Vecindad 3x3 del píxel actual (rxp, ryp)
           vecindad = double(grayIm(rxp-1:rxp+1, ryp-1:ryp+1));

           % Aplicar la máscara de suavizado
           nuevo_valor = sum(sum(vecindad .* filtro_suavizado));

           % Asignar el valor calculado al píxel de salida
           OutSuavizado(rxp, ryp) = nuevo_valor;
       end
   end
   OutSuavizado = uint8(OutSuavizado);
end
function OutBox = box(grayIm)
   [FIL, COL] = size(grayIm);
   OutBox = zeros(FIL, COL);
   filtro_box = ones(3, 3) / 9;
```

```
    for rxp = 2:FIL-1

        for ryp = 2:COL-1
            % Vecindad 3x3 del píxel actual (rxp, ryp)
            vecindad = double(grayIm(rxp-1:rxp+1, ryp-1:ryp+1));

            % Aplicar la máscara del filtro Box
            nuevo_valor = sum(sum(vecindad .* filtro_box));

            % Asignar el valor calculado al píxel de salida
            OutBox(rxp, ryp) = nuevo_valor;
        end
    end
    OutBox(1, :) = 0;    % Primera fila
    OutBox(FIL, :) = 0; % Última fila
    OutBox(:, 1) = 0;    % Primera columna
    OutBox(:, COL) = 0; % Última columna
    OutBox = uint8(OutBox);
end
function OutGauss = gauss(grayIm)
    [FIL, COL] = size(grayIm);
    OutGauss = zeros(FIL, COL);
    sigma = 1;
    [x, y] = meshgrid(-2:2, -2:2); % Mascara de 5x5
    filtro_gaussiano = exp(-(x.^2 + y.^2) / (2 * sigma^2)); % Formula gaussiana
    filtro_gaussiano = filtro_gaussiano / sum(filtro_gaussiano(:)); % Normalizar
la mascara
    for rxp = 3:FIL-2
        for ryp = 3:COL-2
            % Vecindad 5x5 del píxel actual (rxp, ryp)
            vecindad = double(grayIm(rxp-2:rxp+2, ryp-2:ryp+2));

            % Aplicar la máscara gaussiana
            nuevo_valor = sum(sum(vecindad .* filtro_gaussiano));

            % Asignar el valor calculado al píxel correspondiente en la imagen de
salida
            OutGauss(rxp, ryp) = nuevo_valor;
        end
    end
    OutGauss = uint8(OutGauss);
end
function OutLaplace = laplace(grayIm)
    [FIL, COL] = size(grayIm);
    OutLaplace = zeros(FIL, COL);
    filtro_laplaciano = [ 0   0  -1   0   0;
                          0  -1  -2  -1   0;
                         -1  -2  16  -2  -1;
                          0  -1  -2  -1   0;
                          0   0  -1   0   0];
```

```
for rxp = 3:FIL-2
       for ryp = 3:COL-2
           % Vecindad 5x5 del píxel actual (rxp, ryp)
           vecindad = double(grayIm(rxp-2:rxp+2, ryp-2:ryp+2));

           % Aplicar la máscara Laplaciana
           nuevo_valor = sum(sum(vecindad .* filtro_laplaciano));

           % Asignar el valor calculado al píxel de salida
           OutLaplace(rxp, ryp) = nuevo_valor;
       end
   end
   OutLaplace = uint8(OutLaplace);
end
function OutMediana = mediana(grayIm)
   [FIL, COL] = size(grayIm);
   OutMediana = zeros(FIL, COL, 'uint8');
   for rxp = 3:FIL-2
       for ryp = 3:COL-2
           % Extraer la vecindad de 3x3 alrededor del píxel actual
           vecindad = grayIm(rxp-2:rxp+2, ryp-2:ryp+2);
           vector = vecindad(:);  % Convierte la matriz 3x3 en un vector
           vector_ordenado = sort(vector);  % Ordena los valores
           % La mediana es el valor central del vector ordenado
           valor_mediana = vector_ordenado(5);  % El 5to valor es la mediana
           % Asignar la mediana al píxel actual en la imagen filtrada
           OutMediana(rxp, ryp) = valor_mediana;
       end
   end
end
function ImSalPimienta = salPimienta(grayIm)
   [re, co] = size(grayIm);
   ImSalPimienta = grayIm;
   for v = 1:1000
       x = round(rand * (re - 1)) + 1;
       y = round(rand * (co - 1)) + 1;
       % Protege las fronteras para evitar que se salga del tamaño de la imagen
       if x >= re - 2
           x = re - 2;
       end
       if y >= co - 2
           y = co - 2;
       end
       % Inserta la estructura con valores de intensidad 255 (sal)
       ImSalPimienta(x:x+2, y:y+1) = 255;
   end
   % Inserción de 1000 puntos de ruido tipo 'pimienta' (valor 0)
   for v = 1:1000
       % Se calculan las posiciones de cada punto
       x = round(rand * (re - 1)) + 1;
       y = round(rand * (co - 1)) + 1;
       % Protege las fronteras
```

```
        if x >= re - 2
            x = re - 2;
        end
        if y >= co - 2
            y = co - 2;

        end
        % Inserta la estructura con valores de intensidad 0 (pimienta)
        Ir1(x:x+2, y:y+1) = 0;
    end
end
```

RESULTADOS Y CONCLUSIONES

REFERENCIAS

1. Ramírez, A. (2021). **Filtros para el procesamiento de imágenes: Filtro de mediana**. Ciencia y Tecnología Visual. Recuperado de https://www.cienciaytecnologiavisual.com/filtro-mediana
2. Pérez, L. (2020). **Procesamiento de imágenes: Técnicas de filtrado espacial para la eliminación de ruido**. Imágenes Digitales. Recuperado de https://www.imagenesdigitales.com/procesamiento-filtrado-espacial
3. Fernández, R. (2018). **Reducción de ruido en imágenes digitales: Uso del filtro de mediana**. Ingeniería en Sistemas. Recuperado de https://www.ingenieriaensistemas.com/reduccion-ruido-filtro-mediana
4. Castillo, J. (2019). **Filtros no lineales y su aplicación en la eliminación de ruido de tipo sal y pimienta**. Procesamiento de Imágenes Avanzado. Recuperado de https://www.procesamientoimagenesavanzado.com/filtros-no-lineales

Elavoro	Observaciones	Evaluacion
DR. Gerardo García Gil		