Nombre: Edgar Javier Fregoso Cuarenta

Registro: 22310285

Practica COLORS

Materia : visión Artificial

El objetivo de esta práctica es aplicar técnicas de **suavizado (smoothing)** y **desenfoque (blurring)** a imágenes utilizando la librería OpenCV en Python. Estas técnicas son útiles para reducir ruido, suavizar bordes y preparar imágenes para tareas de visión por computadora como detección de objetos o segmentación.

2. Herramientas utilizadas

- **Lenguaje de programación:** Python 3
- **Librería:** OpenCV (cv2)
- **Entorno de desarrollo:** [VS Code, Jupyter Notebook, etc.]

3. Desarrollo de la práctica

Paso 1: Importar librerías y cargar imagen

Se cargó una imagen para realizar los procesos de suavizado:

python

Copiar código

```
import cv2
import numpy as np

img = cv2.imread('imagen.jpg')
```

Paso 2: Aplicar diferentes métodos de suavizado

Se probaron distintas técnicas de desenfoque que ofrece OpenCV:

- **a) Promedio (Average Blurring)**

python

Copiar código

```
blur = cv2.blur(img, (5,5))
```

- Usa un kernel de tamaño 5x5 para promediar los valores de píxeles cercanos.
- Reduce el ruido de forma general.

🔹 **b) Desenfoque Gaussiano (Gaussian Blur)**

python

Copiar código

gaussian = cv2.GaussianBlur(img, (5,5), 0)

- Similar al anterior, pero aplica una **distribución gaussiana** para darle más peso al píxel central.
- Mejora resultados al preservar bordes mejor que el promedio.

🔹 **c) Desenfoque Mediano (Median Blur)**

python

Copiar código

median = cv2.medianBlur(img, 5)

- Usa la **mediana** en lugar del promedio.
- Muy útil para eliminar ruido de tipo sal y pimienta (salt-and-pepper noise).

🔹 **d) Filtro Bilateral (Bilateral Filter)**

python

Copiar código

bilateral = cv2.bilateralFilter(img, 9, 75, 75)

- Técnica avanzada que suaviza la imagen mientras **preserva los bordes**.
- Parámetros:
    - 9: diámetro del área de los píxeles vecinos

- 75: sigmaColor y sigmaSpace, que determinan la influencia del color y la distancia espacial

Paso 3: Mostrar los resultados

Se mostraron todas las imágenes procesadas para comparar los efectos de cada tipo de desenfoque:

python

Copiar código

```
cv2.imshow('Original', img)

cv2.imshow('Promedio', blur)

cv2.imshow('Gaussiano', gaussian)

cv2.imshow('Mediano', median)

cv2.imshow('Bilateral', bilateral)


cv2.waitKey(0)

cv2.destroyAllWindows()
```

4. Resultados

Cada técnica de suavizado mostró efectos distintos:

- **Promedio:** Desenfoque general, pero puede difuminar detalles.
- **Gaussiano:** Más natural y efectivo en preservación de bordes suaves.
- **Mediano:** Excelente para eliminar ruido tipo sal y pimienta.
- **Bilateral:** Mantiene los bordes nítidos mientras desenfoca áreas planas.

5. Conclusiones

- El suavizado de imágenes es una herramienta fundamental para el preprocesamiento en visión por computadora.

- OpenCV proporciona funciones versátiles para aplicar distintos tipos de desenfoque, cada uno con ventajas y usos específicos.
- Entender cuándo usar cada técnica mejora significativamente los resultados en tareas posteriores como detección de bordes o reconocimiento de patrones.