Nombre: Edgar Javier Fregoso Cuarenta

Registro: 22310285

Practica

Materia : visión Artificial

1. Objetivo de la práctica

El objetivo de esta práctica es comprender y aplicar el proceso de **umbralización** en imágenes, utilizando la biblioteca OpenCV en Python. La umbralización permite convertir imágenes en escala de grises a imágenes binarias, facilitando el análisis de formas, bordes y objetos.

2. Herramientas utilizadas

- **Lenguaje de programación:** Python 3
- **Librería:** OpenCV (cv2)
- **Entorno de desarrollo:** [Especificar IDE usado, como VS Code o Jupyter Notebook]
- **Imagen de prueba:** imagen en escala de grises o imagen cargada y convertida

3. Desarrollo de la práctica

Paso 1: Cargar y convertir una imagen a escala de grises

Se inicia cargando una imagen y convirtiéndola a escala de grises, que es requisito para aplicar thresholding:

python

Copiar código

```
import cv2

import numpy as np


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

grayscale = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

Paso 2: Aplicar umbralización simple (Thresholding básico)

Se establece un valor límite. Todos los píxeles por encima del umbral se convierten en blanco (255), y los demás en negro (0):

python

Copiar código

retval, threshold = cv2.threshold(grayscale, 127, 255, cv2.THRESH_BINARY)

- 127: valor del umbral
- 255: valor máximo
- cv2.THRESH_BINARY: tipo de umbralización (binaria simple)

Paso 3: Aplicar otros tipos de umbralización

Umbral binario invertido:

python

Copiar código

thresh_inv = cv2.threshold(grayscale, 127, 255, cv2.THRESH_BINARY_INV)[1]

Umbral truncado (los valores por encima del umbral se reemplazan por el umbral):

python

Copiar código

thresh_trunc = cv2.threshold(grayscale, 127, 255, cv2.THRESH_TRUNC)[1]

Umbral a cero (los valores debajo del umbral se convierten en 0):

python

Copiar código

thresh_tozero = cv2.threshold(grayscale, 127, 255, cv2.THRESH_TOZERO)[1]

Umbral a cero invertido (los valores encima del umbral se convierten en 0):

python

Copiar código

thresh_tozero_inv = cv2.threshold(grayscale, 127, 255, cv2.THRESH_TOZERO_INV)[1]

Paso 4: Mostrar las imágenes

Se muestran la imagen original y los diferentes resultados de umbralización:

python

Copiar código

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

cv2.imshow('Binary', threshold)

cv2.imshow('Binary Inv', thresh_inv)

cv2.imshow('Trunc', thresh_trunc)

cv2.imshow('Tozero', thresh_tozero)

cv2.imshow('Tozero Inv', thresh_tozero_inv)


cv2.waitKey(0)

cv2.destroyAllWindows()
```

4. Resultados

Se generaron múltiples versiones de la misma imagen con diferentes métodos de umbralización. Cada técnica resalta distintas características de la imagen original. Por ejemplo:

- La **umbralización binaria** es útil para segmentar objetos.
- La **umbralización truncada o a cero** conserva detalles en regiones específicas.

5. Conclusiones

- La umbralización es una herramienta fundamental en el análisis de imágenes, ya que permite simplificar la imagen y facilitar la detección de bordes o formas.

- OpenCV proporciona varias funciones de thresholding adaptables a diferentes tipos de imágenes y necesidades.
- Conocer los distintos tipos de umbralización ayuda a seleccionar el adecuado para cada aplicación, por ejemplo en visión artificial, análisis biomédico o control de calidad.