**Maestro:** Mauricio Alejandro Cabrera Arellano **Alumno:** Nene Miranda José Said 22110319 **Materia:** Visión Artificial

**Tarea:** Practica 2; +,-,\*,/, negación, transpuesta, aumento de tamaño o reducción, rotación, traslación.

**Fecha:** 24-03-2025

Practica 2; +, -, \*, /, negación, transpuesta, aumento de tamaño o reducción, rotación, traslación.

**Objetivo:**

Aplicar transformaciones cromáticas y clasificación de píxeles en una imagen digital para identificar objetos o regiones de interés bajo diferentes condiciones de iluminación.

**Código:**

# -\*- coding: utf-8 -\*-

"""

Created on Thu March 23 10:58:31 2025

@author: nene

"""

import cv2

import numpy as np

imagen = cv2.imread("pelayon.jpg")

imageno1 = imagen\*0.8

imageno2 = imagen\*0.4

def cromatico(imagen):

m,n,c=imagen.shape

imagenc = imagen.copy()

imagenc = imagenc.astype(np.float32)

imagen = imagen.astype(np.float32)

for x in range(m):

for y in range(n):

suma = imagen[x,y,0]+imagen[x,y,1]+imagen[x,y,2]

if suma != 0:

imagenc[x,y,0]=imagen[x,y,0]/suma

imagenc[x,y,1]=imagen[x,y,1]/suma

imagenc[x,y,2]=imagen[x,y,2]/suma

result = cv2.normalize(imagenc, dst=None, alpha=0, beta=255,norm\_type=cv2.NORM\_MINMAX, dtype=cv2.CV\_8U)

cv2.imwrite("imagen\_Cro.jpg",result)

return result

def clasificador(imagen):

m,n,c=imagen.shape

imagenb=np.zeros((m,n))

for x in range(m):

for y in range(n):

if 78<imagen[x,y,2]<145 and 62<imagen[x,y,1]<92 and 55<imagen[x,y,0]<115:

imagenb[x,y]=255

cv2.imwrite("imagen\_cla.jpg",imagenb)

return imagenb

imagenc1 = cromatico(imagen)

imagenc2 = cromatico(imageno1)

imagenc3 = cromatico(imageno2)

imagend1 = clasificador(imagenc1)

imagend2 = clasificador(imagenc2)

imagend3 = clasificador(imagenc3)

im\_final1 = np.hstack((imagen, imageno1, imageno2))

result1 = cv2.normalize(im\_final1, dst=None, alpha=0, beta=255,norm\_type=cv2.NORM\_MINMAX, dtype=cv2.CV\_8U)

im\_final2 = np.hstack((imagenc1, imagenc2, imagenc3))

result2 = cv2.normalize(im\_final2, dst=None, alpha=0, beta=255,norm\_type=cv2.NORM\_MINMAX, dtype=cv2.CV\_8U)

im\_final3 = np.hstack((imagend1, imagend2, imagend3))

result3 = cv2.normalize(im\_final3, dst=None, alpha=0, beta=255,norm\_type=cv2.NORM\_MINMAX, dtype=cv2.CV\_8U)

cv2.imshow("imagen original",result1)

cv2.imshow("imagen cromatica",result2)

cv2.imshow("imagen clasificada",result3)

cv2.waitKey(0)

cv2.destroyAllWindows()

**Demostración**:

