ING. Mecatrónica

visión Artificial

Saul Isaac Limon Bautista

22310278

Explicación paso a paso del código: Filtros de color HSV para imagen 'watch.jpg'

Este código tiene como objetivo aplicar filtros para detectar los colores rojo, verde y azul en una imagen estática, usando el espacio de color HSV.

```
import cv2

import numpy as np

from matplotlib import pyplot as plt
```

Importación de librerías:

- cv2: Librería OpenCV para procesamiento de imágenes.
- numpy: Para manejar arreglos y operaciones con matrices.
- matplotlib.pyplot: Para mostrar las imágenes en una sola ventana.

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

hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
```

Carga de imagen y conversión a HSV:

- cv2.imread: Carga la imagen 'watch.jpg' en formato BGR.
- cv2.cvtColor: Convierte la imagen de BGR a HSV, lo que facilita la segmentación por color.

```
lower_red1 = np.array([0, 120, 70])

upper_red1 = np.array([10, 255, 255])

mask_red1 = cv2.inRange(hsv, lower_red1, upper_red1)


lower_red2 = np.array([170, 120, 70])

upper_red2 = np.array([180, 255, 255])

mask_red2 = cv2.inRange(hsv, lower_red2, upper_red2)
```

mask_red = cv2.bitwise_or(mask_red1, mask_red2)

res_red = cv2.bitwise_and(img, img, mask=mask_red)

Detección del color rojo:

- El rojo está al inicio y al final del círculo de tono en HSV, por eso se usan dos rangos.
- cv2.inRange: Crea máscaras binarias para esos rangos.
- cv2.bitwise_or: Une ambas máscaras.
- cv2.bitwise_and: Aplica la máscara para mostrar solo los pixeles rojos.

---

lower_green = np.array([36, 50, 70])

upper_green = np.array([89, 255, 255])

mask_green = cv2.inRange(hsv, lower_green, upper_green)

res_green = cv2.bitwise_and(img, img, mask=mask_green)

Detección del color verde:

- Se define un rango en HSV para el color verde.
- Se genera la máscara y se aplica sobre la imagen.

---

lower_blue = np.array([90, 60, 70])

upper_blue = np.array([128, 255, 255])

mask_blue = cv2.inRange(hsv, lower_blue, upper_blue)

res_blue = cv2.bitwise_and(img, img, mask=mask_blue)

Detección del color azul:

- Igual que en los casos anteriores, se define un rango y se aplica la máscara.

---

titles = ['Original', 'Rojo', 'Verde', 'Azul']

images = [img, res_red, res_green, res_blue]

```
plt.figure(figsize=(12, 6))
for i in range(4):
    plt.subplot(1, 4, i+1)
    plt.imshow(cv2.cvtColor(images[i], cv2.COLOR_BGR2RGB))
    plt.title(titles[i])
    plt.axis('off')

plt.tight_layout()
plt.show()
```

Visualización de resultados:

- Se preparan títulos e imágenes para mostrarlas.
- cv2.cvtColor(..., cv2.COLOR_BGR2RGB): Convierte la imagen a RGB para que matplotlib la muestre correctamente.
- plt.subplot: Organiza las imágenes en una sola fila.
- plt.show: Muestra todas las imágenes juntas para comparar.

Resultado:

Este código permite identificar y visualizar las regiones de la imagen que contienen los colores rojo, verde y azul, lo cual es útil en aplicaciones como detección de objetos, seguimiento de color o efectos de pantalla verde.