Maestro: Mauricio Alejandro Cabrera Arellano

Alumno: Nene Miranda José Said 22110319

Materia: Visión Artificial

Tarea: Practica 11

Fecha: 01-06-2025

Practica 11:

Objetivo 1:

Objetivo: De la imagen deseada encontrar las similitudes en otra imagen.

Objetivo 2: En VIDEO poder extraer el fondo de la imagen mediante la detección de movimiento.

Codigo:

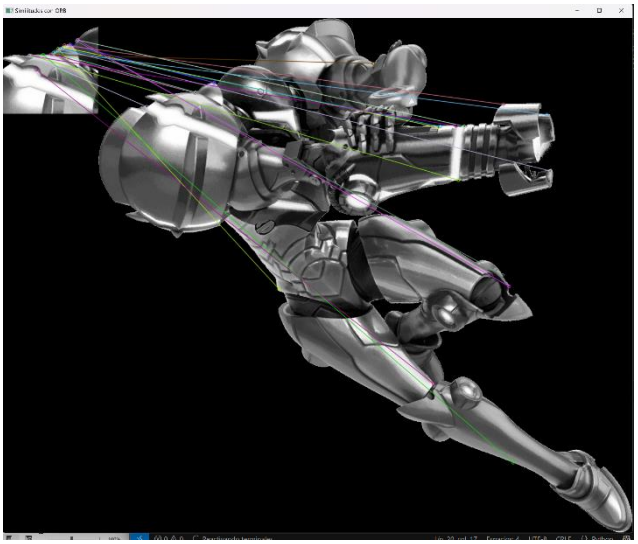
```
import cv2  # Librería OpenCV para procesamiento de imágenes


# --------- Cargar imágenes ---------

img1 = cv2.imread('template.png', 0)  # Imagen plantilla (recorte que se desea encontrar)

img2 = cv2.imread('samus.png', 0)     # Imagen donde se realizará la búsqueda

# Ambas se cargan en escala de grises (0) para simplificar el análisis


# --------- Inicializar el detector ORB ---------

orb = cv2.ORB_create()

# ORB (Oriented FAST and Rotated BRIEF) detecta puntos clave robustos a rotación y escala


# --------- Detectar puntos clave (keypoints) y descriptores ---------

kp1, des1 = orb.detectAndCompute(img1, None)

kp2, des2 = orb.detectAndCompute(img2, None)

# kp1, kp2 = puntos clave en cada imagen

# des1, des2 = vectores que describen las características locales en torno a cada keypoint


# --------- Comparar descriptores con Brute-Force matcher ---------

bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)

# NORM_HAMMING es la distancia utilizada para descriptores binarios como los de ORB

# crossCheck=True asegura que la coincidencia sea recíproca (más precisa)
```

```
matches = bf.match(des1, des2)

# Obtenemos una lista de coincidencias entre descriptores


# --------- Ordenar coincidencias por distancia (mejor a peor) ---------

matches = sorted(matches, key=lambda x: x.distance)

# Entre menor sea la distancia, mejor es la coincidencia entre características


# --------- Dibujar las mejores 20 coincidencias ---------

resultado = cv2.drawMatches(img1, kp1, img2, kp2, matches[:20], None, flags=2)

# Se dibujan las líneas que conectan los puntos clave coincidentes entre ambas imágenes


# --------- Mostrar el resultado ---------

cv2.imshow('Similitudes con ORB', resultado)

cv2.waitKey(0)  # Esperar hasta que el usuario presione una tecla

cv2.destroyAllWindows()  # Cerrar la ventana mostrada
```

Objetivo 2:

```
import cv2


cap = cv2.VideoCapture(0)  # o usa 'video.mp4'


# Leer primer frame como fondo inicial

ret, fondo = cap.read()

fondo_gray = cv2.cvtColor(fondo, cv2.COLOR_BGR2GRAY)


while True:

    ret, frame = cap.read()

    if not ret:

        break
```

```
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Diferencia absoluta entre fondo y frame actual
    diff = cv2.absdiff(fondo_gray, gray)

    # Umbral para mostrar sólo movimiento
    _, thresh = cv2.threshold(diff, 30, 255, cv2.THRESH_BINARY)

    cv2.imshow('Video Original', frame)
    cv2.imshow('Movimiento detectado', thresh)

    if cv2.waitKey(1) & 0xFF == 27:  # Salir con ESC
        break

cap.release()
cv2.destroyAllWindows()
```



Demostración: