Centro de Enseñanza Técnica Industrial

REPORTE DE PRÁCTICA: Detección de coincidencias y sustracción de fondo en video usando OpenCV

Nombre del alumno: Eduardo Correa Flores

Matrícula: 22310172

Materia: Visión Artificial

Semestre: 6to

Fecha: Junio 2025

1. Introducción

En esta práctica se trabajaron dos enfoques comunes en visión artificial utilizando la librería OpenCV. Primero, se realizó la detección de coincidencias entre dos imágenes utilizando el algoritmo ORB (Oriented FAST and Rotated BRIEF). Después, se implementó un sistema de sustracción de fondo para detectar movimiento en un video mediante el método MOG2.

2. Objetivo

Aplicar técnicas de procesamiento de imágenes para detectar similitudes entre imágenes estáticas usando descriptores ORB, y detectar objetos en movimiento mediante la sustracción de fondo en video, aplicando morfología para reducir el ruido.

3. Material y Método

• Lenguaje de programación: Python

• Librerías: OpenCV, NumPy, Matplotlib

• Recursos: Dos imágenes similares ('Cutter.jpg' y 'Imgcutter.jpg') y un video ('people-walking.mp4')

4. Desarrollo

4.1 Detección de coincidencias con ORB

Se cargaron dos imágenes en escala de grises y se aplicó el detector ORB para encontrar puntos clave y descriptores. Luego, se utilizó el emparejador BruteForce con la norma Hamming para obtener coincidencias entre los descriptores. Finalmente, se dibujaron las 10 mejores coincidencias.

Código fuente:

```
import numpy as np
import cv2
import matplotlib.pyplot as plt

img1 = cv2.imread('Cutter.jpg', 0)
img2 = cv2.imread('Imgcutter.jpg', 0)
orb = cv2.ORB_create()
kp1, des1 = orb.detectAndCompute(img1, None)
kp2, des2 = orb.detectAndCompute(img2, None)
bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)
matches = bf.match(des1, des2)
```

```
matches = sorted(matches, key=lambda x: x.distance)
img3 = cv2.drawMatches(img1, kp1, img2, kp2, matches[:10], None, flags=2)
plt.imshow(img3)
plt.title('Coincidencias ORB')
plt.axis('off')
plt.show()
```

4.2 Detección de movimiento con sustracción de fondo (MOG2)

Se utilizó el método de sustracción de fondo MOG2 con parámetros ajustados (history=500, varThreshold=50, sin sombras) para detectar objetos en movimiento en un video. Para mejorar la calidad de la máscara, se aplicaron operaciones morfológicas: opening para eliminar ruido y closing para cerrar huecos. Se mostraron tanto la imagen original como la máscara procesada.

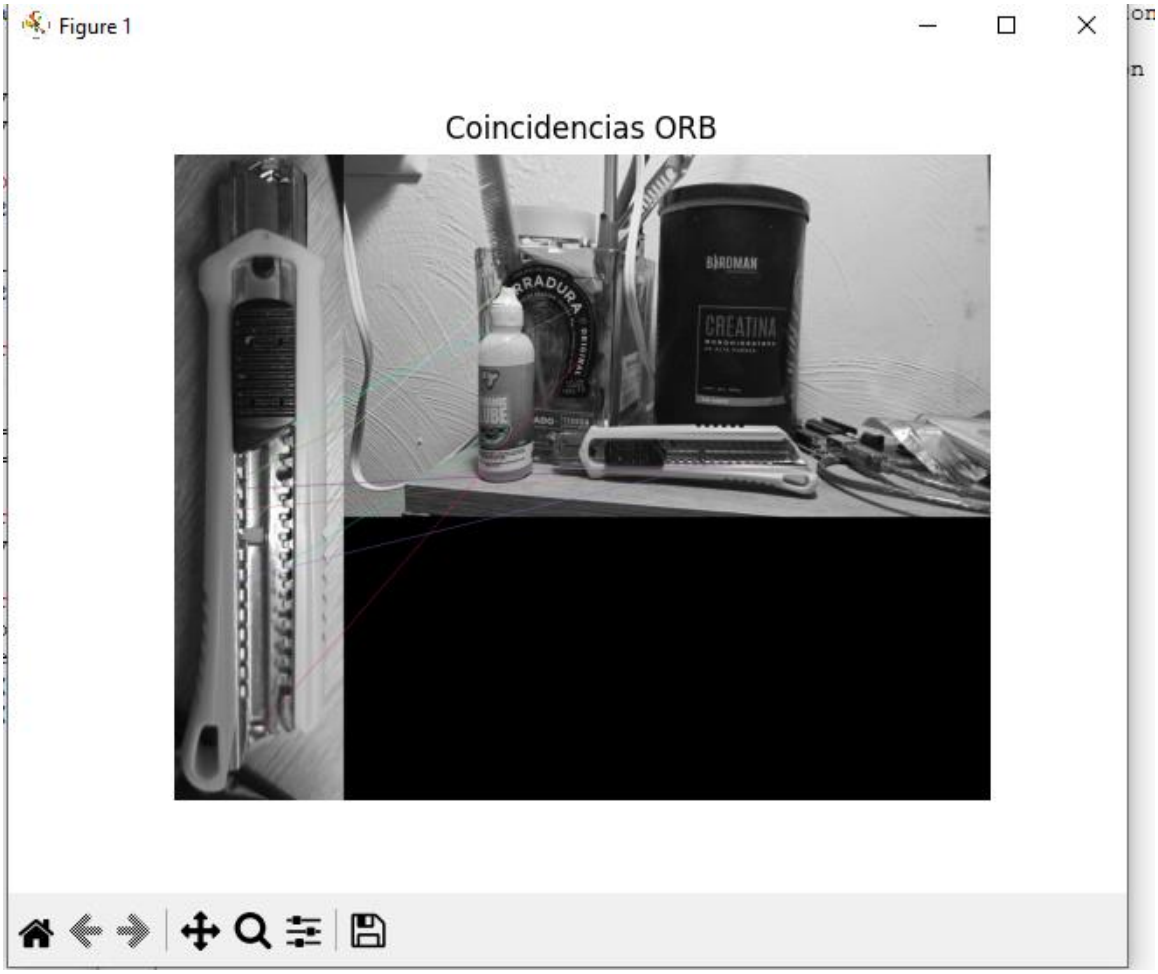
Código fuente:

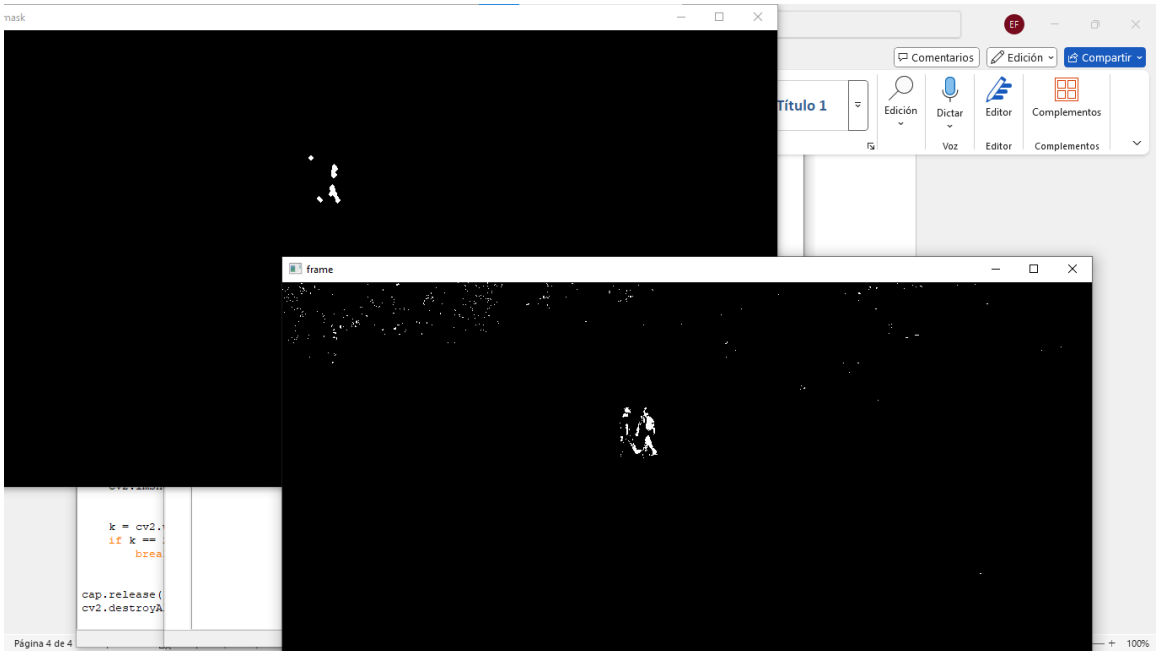
```
import numpy as np
import cv2

cap = cv2.VideoCapture('people-walking.mp4')
fgbg = cv2.createBackgroundSubtractorMOG2(history=500, varThreshold=50,
detectShadows=False)
kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (3, 3))

while(1):
   ret, frame = cap.read()
   fgmask = fgbg.apply(frame)
   _, fgmask = cv2.threshold(fgmask, 200, 255, cv2.THRESH_BINARY)
   fgmask = cv2.morphologyEx(fgmask, cv2.MORPH_OPEN, kernel, iterations=2)
   fgmask = cv2.morphologyEx(fgmask, cv2.MORPH_CLOSE, kernel, iterations=2)
   cv2.imshow('frame', frame)
   cv2.imshow('fgmask', fgmask)
   if cv2.waitKey(30) & 0xff == 27:
      break
cap.release()
cv2.destroyAllWindows()
```

5. Resultados

En la primera parte se observaron las coincidencias detectadas entre dos imágenes similares usando ORB, lo que permitió identificar similitudes estructurales. En la segunda

parte, se logró aislar el movimiento de personas en un video, reduciendo el ruido en la máscara binaria mediante operaciones morfológicas.





6. Conclusiones

Se comprendieron e implementaron dos técnicas fundamentales en visión artificial: la detección de coincidencias entre imágenes estáticas y la sustracción de fondo en secuencias de video. Ambas herramientas son ampliamente utilizadas en sistemas de reconocimiento, monitoreo y análisis de movimiento.