Universidad Autónoma de San Luis Potosí
Facultad de ingeniería
Tratamiento de Imágenes
Practica 8
Nombre Práctica: Detección de Esquinas y Flujo Óptico
Nombre del Alumno: Manuel Ramírez Galván

Fecha: 25/03/2025

Procedimiento

8.1. Diseñar un programa en Python en el cual se cumplan los siguientes requisitos:

- Cargar un video desde el disco duro.
- Encontrar las esquinas en una secuencia de video. Dibujar un círculo alrededor de cada esquina encontrada.
- Aplicar el flujo óptico de Lucas-Kanade sobre los puntos encontrados.
- Mostrar el movimiento mediante el uso de líneas de color sobre la secuencia original.

Resultados

```python
import cv2
import numpy as np

vid_capture = cv2.VideoCapture('Recursos/slow_traffic_small.mp4')

feature_params = dict(maxCorners=100, qualityLevel=0.3, minDistance=7, blockSize=7)

lk_params = dict(winSize=(15, 15), maxLevel=2, criteria=(cv2.TERM_CRITERIA_EPS | cv2.TERM_CRITERIA_COUNT,
10, 0.03))

ret, old_frame = vid_capture.read()

old_gray = cv2.cvtColor(old_frame, cv2.COLOR_BGR2GRAY)

gray_float = np.float32(old_gray)
harris_corners = cv2.cornerHarris(gray_float, blockSize=2, ksize=3, k=0.04)

harris_corners = cv2.dilate(harris_corners, None)

old_frame[harris_corners > 0.01 * harris_corners.max()] = [0, 100, 200]

p0 = cv2.goodFeaturesToTrack(old_gray, mask=None, **feature_params)

mask = np.zeros_like(old_frame)

while vid_capture.isOpened():
    ret, frame = vid_capture.read()
    if not ret:
        break

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

    p1, st, err = cv2.calcOpticalFlowPyrLK(old_gray, gray, p0, None, **lk_params)

    if p1 is not None:
        good_new = p1[st == 1]
        goodo_ld = p0[st == 1]

        for i, (new, old) in enumerate(zip(good_new, good_old)):
            a, b = new.ravel()
            c, d = old.ravel()
            a, b, c, d = int(a), int(b), int(c), int(d)
            mask = cv2.line(mask, (a, b), (c, d), (200, 100, 0), 2)
            frame = cv2.circle(frame, (a, b), 5, (0, 100, 200), -1)

        img = cv2.add(frame, mask)

        cv2.imshow('Ejercicio 1', img)

        old_gray = gray.copy()
        p0 = good_new.reshape(-1, 1, 2)

    key = cv2.waitKey(10)
    if key == ord('q'):
        break

vid_capture.release()
cv2.destroyAllWindows()
```

Imagen 1.- Código 1 Ejercicio 1

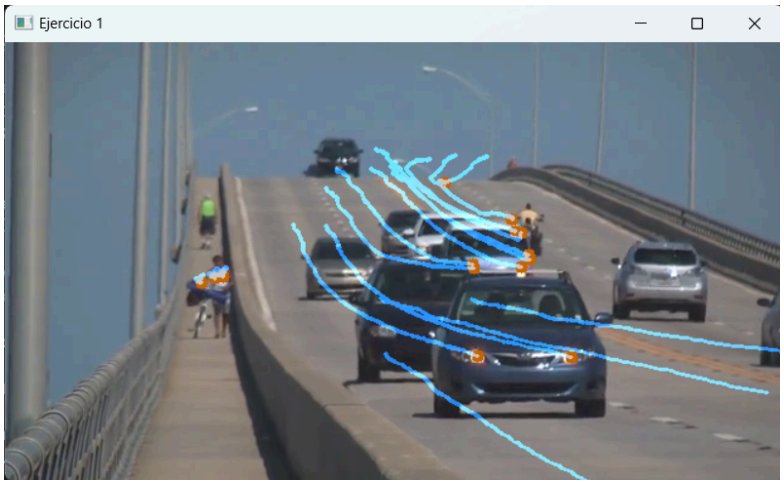

Imagen 2.- Flujo Óptico de Lucas-Kanade

Procedimiento

7.2. Diseñe un programa en Python en el cual se cumplan los siguientes requisitos:

- Acceder a la cámara web.
- Aplicar el flujo óptico denso de Gunnar Farneback.
- Mostrar el movimiento mediante el uso de color en una ventana.

Resultados

```python
import cv2
import numpy as np

cam = cv2.VideoCapture(0)

ret, prev_frame = cam.read()

prev_gray = cv2.cvtColor(prev_frame, cv2.COLOR_BGR2GRAY)

while cam.isOpened():
    ret, frame = cam.read()
    if not ret:
        break

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

    fl_op = cv2.calcOpticalFlowFarneback(prev_gray, gray, None, 0.5, 5, 15, 3, 5, 1.2, 0)

    mag, ang = cv2.cartToPolar(fl_op[..., 0], fl_op[..., 1])

    hsv = np.zeros((frame.shape[0], frame.shape[1], 3), dtype=np.uint8)
    hsv[..., 1] = 255
    hsv[..., 0] = ang * 180 / np.pi / 2
    hsv[..., 2] = cv2.normalize(mag, None, 0, 255, cv2.NORM_MINMAX)
    flow_rgb = cv2.cvtColor(hsv, cv2.COLOR_HSV2BGR)

    cv2.imshow('Ejercicio 2', flow_rgb)

    prev_gray = gray

    key = cv2.waitKey(1)
    if key == ord('q'):
        break

cam.release()
cv2.destroyAllWindows()
```

Imagen 3.- Código Ejercicio 2

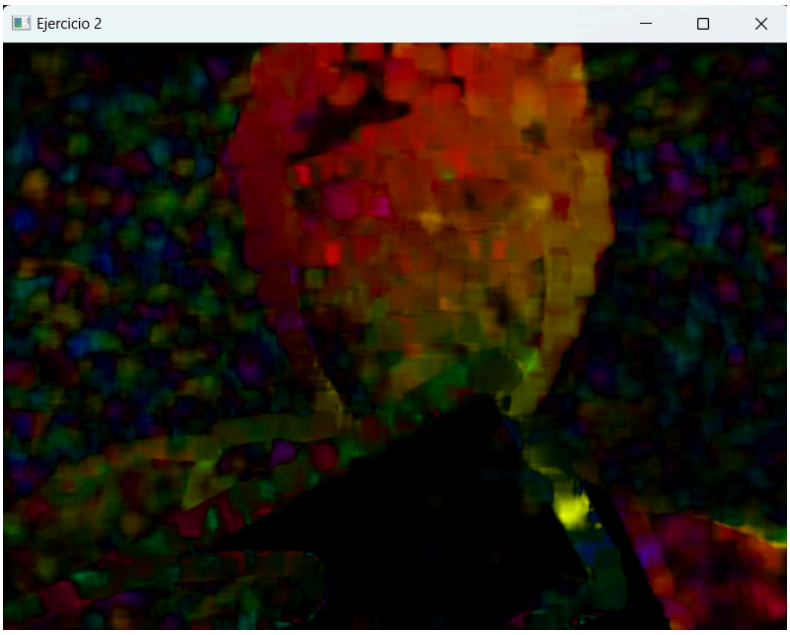

Imagen 4.- Flujo Óptico denso de Gunnar Farneback

Comprensión

1. **¿Qué es un punto de interés?**

Es una región en una imagen que contiene información significativa, como bordes, esquinas y que puede seguirse fácilmente entre diferentes fotogramas de un video.

2. **¿Para qué se utiliza la detección de esquinas?**

- Son fáciles de rastrear entre fotogramas porque tienen mucha información local.
- Se usan para comparar y emparejar imágenes.
- Permiten calcular correspondencias entre imágenes de diferentes ángulos para estimar la geometría 3D de una escena.
- Ayudan a distinguir formas específicas como señales, letras, figuras geométricas.

3. **¿Qué diferencia hay entre el flujo óptico disperso y el flujo óptico denso?**

El flujo óptico disperso analiza sólo ciertos puntos clave o de interés.

- Utiliza pocos puntos, como los detectados por Shi-Tomasi.
- Más rápido y eficiente computacionalmente.
- Ideal para seguimiento de objetos o puntos clave.

El flujo óptico denso calcula el movimiento en todos los píxeles de la imagen.

- Genera un mapa completo del movimiento.
- Más pesado computacionalmente.
- Proporciona más detalle sobre el flujo de la escena.

Conclusiones

El flujo óptico permite estimar el movimiento entre dos imágenes o fotogramas.

El flujo disperso analiza solo puntos clave y es ideal para aplicaciones rápidas, como la detección de movimiento de automóviles como en el ejercicio agregando que tiene aplicaciones en videos vigilancia o control vehicular.

El flujo denso, al calcular el movimiento de todos los píxeles, es útil en análisis globales como el seguimiento de movimiento general en una cámara, en este caso puede ser en una cámara web al detectar el movimiento en toda la escena.