231501008

EXP NO: 08
DATE: 19-09-2025

optical flow computation algorithm.

Aim: Implement optical flow computation algorithm.

Algorithm:

1. Read two consecutive video frames.
2. Convert both to grayscale.
3. Compute optical flow using cv2.calcOpticalFlowFarneback().
4. Visualize motion vectors using color coding.
5. Overlay flow on original image.
6. Display motion visualization.

Code:

```
import cv2

import numpy as np

from google.colab.patches import cv2_imshow


cap = cv2.VideoCapture('video.mp4')

ret, old = cap.read()

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


# Initial feature points

p0 = cv2.goodFeaturesToTrack(old_gray, maxCorners=50, qualityLevel=0.3, minDistance=7)

mask = np.zeros_like(old)


# Parameters for Lucas-Kanade optical flow

lk = dict(winSize=(15, 15), maxLevel=2,

        criteria=(cv2.TERM_CRITERIA_EPS | cv2.TERM_CRITERIA_COUNT, 10, 0.03))
```

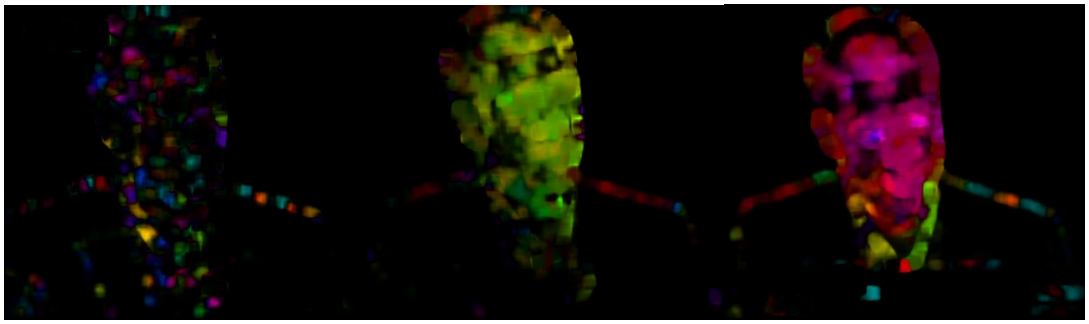
231501008

```
frame_count = 0
while True:
    ret, frame = cap.read()
    if not ret:
        break
    frame_gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Calculate optical flow
    p1, st, _ = cv2.calcOpticalFlowPyrLK(old_gray, frame_gray, p0, None, **lk)

    # Select good points
    good_new = p1[st == 1]
    good_old = p0[st == 1]

    # Draw motion vectors every 5th frame only
    if frame_count % 5 == 0:
        mask[:] = 0  # clear mask for clean output
        for (new, old_pt) in zip(good_new, good_old):
            a, b = new.ravel()
            c, d = old_pt.ravel()
            mask = cv2.line(mask, (int(a), int(b)), (int(c), int(d)), (0, 255, 0), 2)
            frame = cv2.circle(frame, (int(a), int(b)), 3, (0, 0, 255), -1)
        motion = np.mean(np.linalg.norm(good_new - good_old, axis=1))
        print(f"Frame {frame_count}: Avg motion = {motion:.2f}")
        cv2_imshow(cv2.add(frame, mask))

    # Update previous frame and points
    old_gray = frame_gray.copy()
```

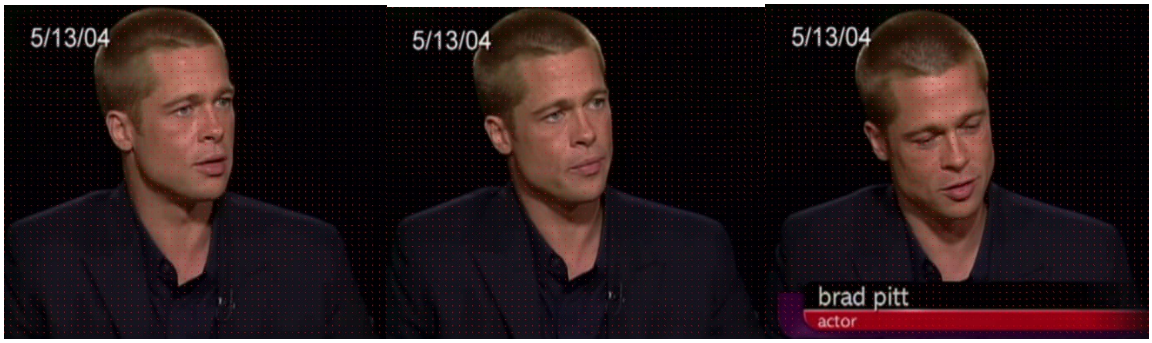
231501008

```
    p0 = good_new.reshape(-1, 1, 2)
    frame_count += 1

    if cv2.waitKey(30) & 0xFF == 27:  # ESC to quit
        break

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

Output:



231501008



Result: Thus, optical flow computation algorithm was implemented successfully.