Tugas Hands On 3

NAMA : MARSELLA YESI NATALIA SINAGA

NIM : 121140174

Sistem Teknologi/Multimedia (IF4021)

SOAL 1

In [1]:
```
import os
from glob import glob
import cv2
import numpy as np
import matplotlib.pyplot as plt
import dlib
import datetime as dt
```

1. Jelaskan maksud dari `list_imgs = sorted(list_imgs, key=lambda x: int(x.split('/')[-1].split('.')[0]))`

Dalam code tersebut berfungsi untuk mengurutkan daftar list_imgs, yang berisi nama-nama file gambar yang dimana terdapat penjelasan dari bagian bagiannya :

- `sorted()` :

- Fungsi ini digunakan untuk mengurutkan elemen dalam iterable (dalam hal ini, list_imgs).
- Fungsi ini mengembalikan daftar baru yang sudah terurut.

- `list_imgs` :

- Daftar yang berisi nama-nama file gambar yang akan diurutkan. Nama file tersebut biasanya memiliki format yang mencakup angka.

- `key=lambda x: ...` :

- Parameter key dalam fungsi sorted() menentukan kriteria pengurutan. Fungsi lambda ini mengambil satu argumen x, yang merupakan elemen dari list_imgs.

- `x.split('/')[-1]` :

- Memecah string x (nama file) berdasarkan karakter pemisah '/' dan mengambil elemen terakhir dari hasil pemecahan. Hal ini digunakan untuk mendapatkan nama file dari path lengkap.

- `split('.')[0]` :

- Setelah mendapatkan nama file, pemecahan dilakukan lagi berdasarkan karakter pemisah '.', dan elemen pertama diambil. Ini digunakan untuk mendapatkan nama

file tanpa ekstensi.

- `int(...)`:

- Hasil dari pemecahan nama file (yang seharusnya berupa string yang mewakili angka) kemudian dikonversi menjadi tipe data integer. Ini penting agar pengurutan dilakukan berdasarkan nilai numerik, bukan berdasarkan urutan leksikografis.

2. Jelaskan tentang bagian kode berikut: `fourcc = cv2.VideoWriter_fourcc(*'mp4v')` Apakah ada opsi lain selain mp4v? Jika ada, coba gunakan dan jelaskan.

Dalam Code tersebut digunakan untuk menentukan codec yang akan digunakan saat menyimpan video menggunakan OpenCV. Terdapat penjelasan dari bagian bagiannya :

- **`cv2.VideoWriter_fourcc(*'mp4v')`:**

- `cv2.VideoWriter_fourcc()` adalah fungsi dari OpenCV yang digunakan untuk menentukan codec video.
- `*'mp4v'` adalah string yang mewakili codec MP4V (MPEG-4 Video).

- Opsi Codec Lain:

Kode	Codec
`'XVID'`	XVID
`'MJPG'`	Motion JPEG
`'DIVX'`	DivX
`'H264'`	H.264
`'FFV1'`	FFmpeg's FFV1

3. Membuat video dengan FPS yang lebih rendah.

- Dengan menggunakan video, simpanlah frame gambar setiap 3 frame. Begini ilustrasinya, jika ada frame 1 s/d 30, maka anda hanya perlu menyimpan frame 1, 4, 7, 10, 13, 16, 19, 22, 25, 28.
- Dengan analogi ini, artinya anda mengurangi FPS-nya. Berapakah FPS yang baru?
- Untuk setiap gambar, convertlah ke dalam format grayscale dan resize menjadi 1280 x 720.
- Untuk setiap gambar, berikanlah titik merah (ukuran bebas, namun terlihat ketika video diputar). Titik tersebut bergerak dari kiri ke kanan untuk setiap frame. Titik tersebut harus sampai di ujung kanan gambar pada frame terakhir. Anda harus melakukan ini secara manual dengan memanipulasi matriks (tidak boleh pakai fungsi / library yang sudah ada)
- Ingat, karena ini titik merah, maka channel warna pada video anda haruslah RGB (walaupun gambarnya telah menjadi grayscale).
- Save video tersebut dengan nama `video_low_fps.mp4`

In [ ]:
```
Membaca Video dan Menghitung FPS Baru
```

In [14]:
```
import cv2
import numpy as np

# Membaca video
video_input = 'marsella.mp4'
video_output = 'video_low_fps.mp4'

# Membuka video
cap = cv2.VideoCapture(video_input)

# Mendapatkan informasi video
fps = cap.get(cv2.CAP_PROP_FPS)  # Mengambil FPS asli
width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
total_frames = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))

# Menghitung FPS baru
new_fps = fps / 3  # Mengurangi FPS dengan mengambil setiap 3 frame
```

Menyimpan video baru dan menginisialisasi variabel

In [15]:
```
# Membuat objek VideoWriter untuk menyimpan video baru
fourcc = cv2.VideoWriter_fourcc(*'mp4v')
out = cv2.VideoWriter(video_output, fourcc, new_fps, (1280, 720), isColor=True)

# Menginisialisasi variabel
frame_count = 0
saved_frames = 0
x_pos = 0  # Posisi horizontal untuk titik merah
```

Menmproses frame

In [16]:
```
while True:
    ret, frame = cap.read()  # Membaca frame
    if not ret:  # Jika tidak ada frame yang dibaca, keluar dari loop
        break

    # Simpan setiap 3 frame
    if frame_count % 3 == 0:
        # Mengonversi frame ke grayscale
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY) # Menggunakan fungsi cv2.

        # Resize menjadi 1280x720
        resized_gray = cv2.resize(gray, (1280, 720)) # Menggunakan fungsi cv2.re

        # Menambahkan titik merah
        # Konversi resized_gray ke BGR agar bisa menambahkan warna
        resized_gray_bgr = cv2.cvtColor(resized_gray, cv2.COLOR_GRAY2BGR)
        cv2.circle(resized_gray_bgr, (x_pos, 300), 10, (0, 0, 255), -1)  # Menam

        # Menulis frame ke video output
        out.write(resized_gray_bgr)

        # Mengupdate posisi titik merah
        x_pos += int(1280 / (total_frames / 3))  # Menghitung posisi x untuk tit
```

```
    frame_count += 1  # Menghitung jumlah frame yang diproses

# Menutup semua objek
cap.release()
out.release()
# cv2.destroyAllWindows() # Di-comment out untuk menghindari error di environmen

print(f"Video dengan FPS baru {new_fps} telah disimpan sebagai {video_output}.")
```

```
Video dengan FPS baru 9.998611303985557 telah disimpan sebagai video_low_fps.mp4.
```

4. Berdasarkan ROI Wajah, sesuaikanlah ROI tersebut untuk menyeleksi area bahu hingga dada.

In [18]:
```
# prompt: Berdasarkan ROI Wajah, sesuaikanlah ROI tersebut untuk menyeleksi area

import cv2
import numpy as np

# Membaca video
video_input = 'marsella.mp4'
video_output = 'marsella_roi.mp4'

# Membuka video
cap = cv2.VideoCapture(video_input)

# Mendapatkan informasi video
fps = cap.get(cv2.CAP_PROP_FPS)
width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

# Membuat objek VideoWriter
fourcc = cv2.VideoWriter_fourcc(*'mp4v')
out = cv2.VideoWriter(video_output, fourcc, fps, (width, height))

# Loop melalui setiap frame
while True:
    ret, frame = cap.read()
    if not ret:
        break

    # Asumsikan ROI wajah sudah dideteksi (ganti dengan deteksi wajah sebenarnya
    # Contoh ROI wajah (ganti dengan koordinat ROI wajah yang sebenarnya)
    #  x,y,w,h = 100,100,150,150  # Contoh koordinat ROI wajah
    #  cv2.rectangle(frame, (x,y), (x+w,y+h), (0,255,0), 2)

    # Menyesuaikan ROI untuk bahu hingga dada berdasarkan ROI wajah
    #  (Implementasi penyesuaian ROI ini bergantung pada algoritma deteksi wajah
    #   dan perlu disesuaikan berdasarkan data video)
    # Contoh sederhana:
    try:
        # Contoh logika penyesuaian ROI (ganti dengan logika yang sesuai)
        #  x_bahu = x
        #  y_bahu = y + h  # Titik awal ROI bahu di bawah ROI wajah
        #  w_dada = w * 2  # Lebar ROI dada dua kali lebar ROI wajah
        #  h_dada = h * 2   # Tinggi ROI dada dua kali tinggi ROI wajah
        #  x_dada = x_bahu
        #  y_dada = y_bahu

        # Ganti dengan koordinat yang tepat berdasarkan hasil deteksi wajah
```

```
        x_bahu, y_bahu, w_dada, h_dada = 100, 250, 300, 200  # Contoh koordinat
        x_dada = x_bahu

        # Menggambar ROI bahu hingga dada
        cv2.rectangle(frame, (x_bahu, y_bahu), (x_bahu+ w_dada, y_bahu + h_dada)

        # Simpan frame hasil
        out.write(frame)

    except:
        print("ROI wajah tidak ditemukan")
        out.write(frame)


# Menutup semua objek
cap.release()
out.release()


print(f"Video dengan ROI bahu hingga dada telah disimpan sebagai {video_output}"
```

```
Video dengan ROI bahu hingga dada telah disimpan sebagai marsella_roi.mp4
```

5. Dengan menggunakan video anda, lakukan facial tracking pada detik ke 25 - 40. Berikan bounding box pada wajah yang terdeteksi. Simpan video tersebut di google drive, link-nya cantumkan pada jawaban anda.

- Lakukan strategi terbaik untuk mengatur ROI agar wajah tetap terdeteksi pada setiap frame.
- Anda dapat menyesuaikan waktu deteksi (berapa detik sekali deteksi dengan dlib dilakukan)
- Anda dapat menyesuaikan parameter deteksi wajah dengan dlib dan parameter tracking dengan OpenCV

In [19]:
```
import cv2
import dlib

# Inisialisasi detektor wajah
detector = dlib.get_frontal_face_detector()

# Baca video
video_path = 'marsella.mp4'
cap = cv2.VideoCapture(video_path)

# Ambil frame rate video
fps = cap.get(cv2.CAP_PROP_FPS)
frame_count = 0

# Buat objek untuk menyimpan video output
fourcc = cv2.VideoWriter_fourcc(*'XVID')
out = cv2.VideoWriter('output_video.avi', fourcc, fps, (int(cap.get(3)), int(cap

# Mulai membaca frame
while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break
```

```
    # Dapatkan waktu dalam detik
    current_time = frame_count / fps

    # Deteksi wajah hanya antara detik 25 hingga 40
    if 25 <= current_time <= 40:
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        faces = detector(gray)

        # Gambar bounding box untuk setiap wajah yang terdeteksi
        for face in faces:
            x, y, w, h = (face.left(), face.top(), face.width(), face.height())
            cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)

    # Tulis frame ke video output
    out.write(frame)
    frame_count += 1

# Lepaskan objek
cap.release()
out.release()

print("Video dengan bounding box wajah telah disimpan sebagai output_video.avi")
```

```
Video dengan bounding box wajah telah disimpan sebagai output_video.avi
```

6. Tempelkan sebuah .png pada wajah anda (bisa di landmark lain, selain mata) dan Modifikasi kode di atas agar lebih smooth Tips: - Lakukan deteksi landmark tidak di setiap frame, melainkan setiap beberapa frame - Misal. Deteksi landmark hanya dilakukan per setiap detik (atau 30 frame sekali)

In [27]:
```
import cv2
import dlib
import numpy as np

# Load detektor wajah dan prediktor landmark
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor("")  # Pastikan file ada

# Load video
video_path = "marsella.mp4"  # Path ke video input Anda
cap = cv2.VideoCapture(video_path)

# Load gambar PNG (objek yang akan ditempelkan) dan sesuaikan ukurannya
overlay_img = cv2.imread("overlay.png", cv2.IMREAD_UNCHANGED)  # Pastikan file o
overlay_img = cv2.resize(overlay_img, (50, 50))  # Resize sesuai kebutuhan

# Video properties
fps = cap.get(cv2.CAP_PROP_FPS)
frame_width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
frame_height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
fourcc = cv2.VideoWriter_fourcc(*'mp4v')
out = cv2.VideoWriter("output_smooth_overlay.mp4", fourcc, fps, (frame_width, fr

# Variabel untuk menyimpan posisi landmark
prev_landmark = None
frame_counter = 0
landmark_update_interval = int(fps)  # Deteksi landmark setiap 1 detik (sesuai d

def apply_overlay(frame, overlay, position):
```

```
    x, y = position
    h, w = overlay.shape[:2]
    # Batas posisi (agar tidak keluar dari frame)
    x_start, x_end = max(0, x - w // 2), min(frame.shape[1], x + w // 2)
    y_start, y_end = max(0, y - h // 2), min(frame.shape[0], y + h // 2)

    overlay_resized = overlay[:y_end-y_start, :x_end-x_start]  # Potong overlay
    b, g, r, a = cv2.split(overlay_resized)  # Pisah channel
    mask = a / 255.0  # Alpha channel sebagai mask

    # Tempel overlay menggunakan alpha blending
    for c in range(3):  # Untuk setiap channel (B, G, R)
        frame[y_start:y_end, x_start:x_end, c] = \
            (1 - mask) * frame[y_start:y_end, x_start:x_end, c] + mask * overlay
    return frame

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

    frame_counter += 1

    # Hanya deteksi landmark setiap beberapa frame (misal, 30 frame sekali)
    if frame_counter % landmark_update_interval == 0 or prev_landmark is None:
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        faces = detector(gray)

        if len(faces) > 0:
            # Ambil wajah pertama yang terdeteksi
            landmarks = predictor(gray, faces[0])
            # Simpan landmark tertentu (misal, landmark ke-30 untuk bagian hidun
            prev_landmark = (landmarks.part(30).x, landmarks.part(30).y)

    # Jika landmark sudah tersedia, gunakan untuk menempelkan overlay
    if prev_landmark:
        frame = apply_overlay(frame, overlay_img, prev_landmark)

    # Tulis frame ke video output
    out.write(frame)

# Release resources
cap.release()
out.release()
cv2.destroyAllWindows()

print("Video dengan overlay smooth telah disimpan sebagai 'output_smooth_overlay
```

```
---------------------------------------------------------------------------
RuntimeError                              Traceback (most recent call last)
<ipython-input-27-0bfa14069575> in <cell line: 7>()
      5 # Load detektor wajah dan prediktor landmark
      6 detector = dlib.get_frontal_face_detector()
----> 7 predictor = dlib.shape_predictor("")  # Pastikan file ada
      8
      9 # Load video

RuntimeError: Unable to open
```