Hands-on 3

by Arsyadana Estu Aziz (121140068)

This Hands-on consist of several steps, it consist of 6 question and each question is different. The sample being used in here is on the `attachment` folder, on file `sample-renamed.mp4`

Task 1

Dari bagian kode ini:

```
list_imgs = sorted(list_imgs, key=lambda x: int(x.split('/')
[-1].split('.')[0]))
print(f"5 path pertama setelah diurutkan: {list_imgs[:5]}")
print(f"Total jumlah gambar: {len(list_imgs)}")
```

Kode ini bertujuan untuk mengurutkan (sorting) pada daftar gambar berdsarkan *key* yang ditentukan dalam *lambda function* seperti berikut.

- `x.split('/')[-1]`, berfungsi untuk memisahkan string menjadi beberapa item dalam list, dan akan di ambil item terakhir.
- `.split('.')[0]`, berfungsi untuk memisahkan item terakhir (berupa filename gambar) dan memisahkan filename dan extensi, lalu kita bisa mengambil filename sebagai key dalam proses konversi.
- `int()`, berfungsi untuk mengkonvensi string menjadi nilai integer.

Liat pada kasus ini, misalkan `list_imgs` berisi:

```
list_imgs = ['path/to/image/1.jpg', 'path/to/image/10.jpg',
'path/to/image/2.jpg']
```

Proses untuk setiap item akan seperti berikut:

1. Split `/`: `x.split('/')` akan memberikan `['path', 'to', 'image', '1.jpg']`
2. Split `.`: `'1.jpg'.split('.')` akan menghasilkan daftar: ['1', 'jpg'] Mengambil elemen pertama dengan [0]: `1`
3. Konversi ke Integer: `int('1')` akan menghasilkan: 1

Task 2

Jelaskan tentang bagian kode berikut:

```
fourcc = cv2.VideoWriter_fourcc(*'mp4v')
```

Dalam `OpenCV`, `FOURCC` (Four Character Code) adalah 4-byte code yang digunakan untuk encoding / decoding dari sebuah video. Normalnya, setiap video containers memiliki format codec-nya sendiri seperti berikut:

- XVID: MPEG-4 codec (often used for AVI files).

- MJPG: Motion JPEG codec.
- DIVX: DivX MPEG-4 codec.
- H264: H.264 codec (requires additional software support).
- MP4V: MPEG-4 codec for MP4 files.
- I420: Uncompressed YUV format.

Apa yang terjadi jika codec tidak sesuai dengan video container? Tidak ada kesalahan fatal, karena media player saat ini bisa mendukung berbagai macam codec, terlepas dari container format itu sendiri.

Import Dependencies

In [4]:
```
import os
from glob import glob
import cv2
import numpy as np
import matplotlib.pyplot as plt
import dlib
import datetime as dt
```

Playing the video input

Memasukan file sebagai sampel processing

In [5]:
```
video_path = os.path.join(os.getcwd(), 'attachment', 'sample-renamed.mp4')

if not os.path.exists(video_path):
    print("Video file not found")
    exit(1)

cap = cv2.VideoCapture(video_path)

vid_np = []
i = 0

while i < 150:
    # membaca frame
    ret, frame = cap.read()

    # jika frame sudah habis, maka break (keluar dari loop)
    if not ret:
        break

    # pindahin frame ke `vid_np`
    vid_np.append(frame)

    # nampilin frame
    # cv2.imshow('Video', frame) # saya comment karena error di MacOS baru

    # jika tombol `q` ditekan, maka video akan berhenti
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

    i += 1
```

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

vid_np = np.array(vid_np)

print(f"Shape dari `vid_np` adalah {vid_np.shape}")
```

```
Shape dari `vid_np` adalah (150, 1080, 1920, 3)
```

Showing some of the frames video

In [6]:
```
img_frame0 = vid_np[0]
img_frame1 = vid_np[50]

plt.figure(figsize=(10, 10))
plt.subplot(1, 2, 1)
plt.imshow(cv2.cvtColor(img_frame0, cv2.COLOR_BGR2RGB))
plt.title('Frame 0')
plt.axis('off')

plt.subplot(1, 2, 2)
plt.imshow(cv2.cvtColor(img_frame1, cv2.COLOR_BGR2RGB))
plt.title('Frame 50')
plt.axis('off')

plt.show()
```

Task 3

Converting a video into lowest third FPS of the orignal (since the every frame is taken every third), convert into grayscale and create a dot effect manually (not with the `cv2.cirlce()`)

In [7]:
```
import cv2
import numpy as np
import os

# Paths video
original_video_path = os.path.join(os.getcwd(), 'attachment', 'sample-renamed.mp
low_fps_video_path = os.path.join(os.getcwd(), 'attachment', 'video_low_fps.mp4'

# Initialize video capture
videoCapture = cv2.VideoCapture(original_video_path)
```

```
# Check if video opened successfully
if not videoCapture.isOpened():
    print("Error: Could not open video.")
    exit()

# Get original FPS and frame count
fps = videoCapture.get(cv2.CAP_PROP_FPS)
frame_count = int(videoCapture.get(cv2.CAP_PROP_FRAME_COUNT))
print(f'Original FPS: {fps}')
print(f'Total Frames: {frame_count}')

# Calculate new FPS (reduce by taking every 3rd frame)
new_fps = fps / 3
print(f'New FPS: {new_fps}')

# Define frame size
frame_size = (1280, 720)  # (width, height)

# Initialize video writer
fourcc = cv2.VideoWriter_fourcc(*'mp4v')  # Codec
output = cv2.VideoWriter(low_fps_video_path, fourcc, new_fps, frame_size, isColo

if not output.isOpened():
    print("Error: Could not open video writer.")
    videoCapture.release()
    exit()

frame_index = 0
processed_frame_num = 0  # To track the number of frames being written

# Calculate step for moving the dot across frames
# The dot should move from left (0) to right (frame_size[0]-1) over all processe
if frame_count // 3 > 1:
    step = (frame_size[0] - 1) / (frame_count // 3 - 1)
else:
    step = 0  # Avoid division by zero if only one frame is processed

while True:
    ret, frame = videoCapture.read()
    if not ret:
        break

    # Process every 3rd frame
    if frame_index % 3 == 0:
        # Resize to 1280x720
        resized_frame = cv2.resize(frame, frame_size)

        # Convert to grayscale
        gray_frame = cv2.cvtColor(resized_frame, cv2.COLOR_BGR2GRAY)

        # Convert grayscale back to BGR (to maintain 3 channels)
        bgr_frame = cv2.cvtColor(gray_frame, cv2.COLOR_GRAY2BGR)

        # Calculate the position of the red dot
        x = int(processed_frame_num * step)
        y = frame_size[1] // 2  # Center row (height)

        # Ensure x is within bounds
        x = min(x, frame_size[0] - 1)
```

```
        # Add the red dot manually
        # OpenCV uses BGR, so Red is [0, 0, 255]
        bgr_frame[y, x] = [0, 0, 255]

        # Or Easier one for displaying the dot
        # cv2.circle(bgr_frame, (x, y), 100, (0, 0, 255), -1)  # Increased radiu

        # Make the dot larger for visibility
        # Define the size of the dot
        dot_radius = 10
        for dy in range(-dot_radius, dot_radius + 1):
            for dx in range(-dot_radius, dot_radius + 1):
                if dx**2 + dy**2 <= dot_radius**2:
                    nx, ny = x + dx, y + dy
                    if 0 <= nx < frame_size[0] and 0 <= ny < frame_size[1]:
                        bgr_frame[ny, nx] = [0, 0, 255]

        # Debug: Print the RGB value of the dot
        print(f'Processed Frame {processed_frame_num}: Red dot at ({x}, {y}) wit

        # Write the frame to the output video
        output.write(bgr_frame)

        processed_frame_num += 1

    frame_index += 1

print(f'Processed {processed_frame_num} frames.')

# Release resources
videoCapture.release()
output.release()
cv2.destroyAllWindows()
```

```
Original FPS: 30.0
Total Frames: 1800
New FPS: 10.0
Processed Frame 0: Red dot at (0, 360) with BGR value [  0   0 255]
Processed Frame 1: Red dot at (2, 360) with BGR value [  0   0 255]
Processed Frame 2: Red dot at (4, 360) with BGR value [  0   0 255]
Processed Frame 3: Red dot at (6, 360) with BGR value [  0   0 255]
Processed Frame 4: Red dot at (8, 360) with BGR value [  0   0 255]
Processed Frame 5: Red dot at (10, 360) with BGR value [  0   0 255]
Processed Frame 6: Red dot at (12, 360) with BGR value [  0   0 255]
Processed Frame 7: Red dot at (14, 360) with BGR value [  0   0 255]
Processed Frame 8: Red dot at (17, 360) with BGR value [  0   0 255]
Processed Frame 9: Red dot at (19, 360) with BGR value [  0   0 255]
Processed Frame 10: Red dot at (21, 360) with BGR value [  0   0 255]
Processed Frame 11: Red dot at (23, 360) with BGR value [  0   0 255]
Processed Frame 12: Red dot at (25, 360) with BGR value [  0   0 255]
Processed Frame 13: Red dot at (27, 360) with BGR value [  0   0 255]
Processed Frame 14: Red dot at (29, 360) with BGR value [  0   0 255]
Processed Frame 15: Red dot at (32, 360) with BGR value [  0   0 255]
Processed Frame 16: Red dot at (34, 360) with BGR value [  0   0 255]
Processed Frame 17: Red dot at (36, 360) with BGR value [  0   0 255]
Processed Frame 18: Red dot at (38, 360) with BGR value [  0   0 255]
Processed Frame 19: Red dot at (40, 360) with BGR value [  0   0 255]
Processed Frame 20: Red dot at (42, 360) with BGR value [  0   0 255]
Processed Frame 21: Red dot at (44, 360) with BGR value [  0   0 255]
Processed Frame 22: Red dot at (46, 360) with BGR value [  0   0 255]
Processed Frame 23: Red dot at (49, 360) with BGR value [  0   0 255]
Processed Frame 24: Red dot at (51, 360) with BGR value [  0   0 255]
Processed Frame 25: Red dot at (53, 360) with BGR value [  0   0 255]
Processed Frame 26: Red dot at (55, 360) with BGR value [  0   0 255]
Processed Frame 27: Red dot at (57, 360) with BGR value [  0   0 255]
Processed Frame 28: Red dot at (59, 360) with BGR value [  0   0 255]
Processed Frame 29: Red dot at (61, 360) with BGR value [  0   0 255]
Processed Frame 30: Red dot at (64, 360) with BGR value [  0   0 255]
Processed Frame 31: Red dot at (66, 360) with BGR value [  0   0 255]
Processed Frame 32: Red dot at (68, 360) with BGR value [  0   0 255]
Processed Frame 33: Red dot at (70, 360) with BGR value [  0   0 255]
Processed Frame 34: Red dot at (72, 360) with BGR value [  0   0 255]
Processed Frame 35: Red dot at (74, 360) with BGR value [  0   0 255]
Processed Frame 36: Red dot at (76, 360) with BGR value [  0   0 255]
Processed Frame 37: Red dot at (79, 360) with BGR value [  0   0 255]
Processed Frame 38: Red dot at (81, 360) with BGR value [  0   0 255]
Processed Frame 39: Red dot at (83, 360) with BGR value [  0   0 255]
Processed Frame 40: Red dot at (85, 360) with BGR value [  0   0 255]
Processed Frame 41: Red dot at (87, 360) with BGR value [  0   0 255]
Processed Frame 42: Red dot at (89, 360) with BGR value [  0   0 255]
Processed Frame 43: Red dot at (91, 360) with BGR value [  0   0 255]
Processed Frame 44: Red dot at (93, 360) with BGR value [  0   0 255]
Processed Frame 45: Red dot at (96, 360) with BGR value [  0   0 255]
Processed Frame 46: Red dot at (98, 360) with BGR value [  0   0 255]
Processed Frame 47: Red dot at (100, 360) with BGR value [  0   0 255]
Processed Frame 48: Red dot at (102, 360) with BGR value [  0   0 255]
Processed Frame 49: Red dot at (104, 360) with BGR value [  0   0 255]
Processed Frame 50: Red dot at (106, 360) with BGR value [  0   0 255]
Processed Frame 51: Red dot at (108, 360) with BGR value [  0   0 255]
Processed Frame 52: Red dot at (111, 360) with BGR value [  0   0 255]
Processed Frame 53: Red dot at (113, 360) with BGR value [  0   0 255]
Processed Frame 54: Red dot at (115, 360) with BGR value [  0   0 255]
Processed Frame 55: Red dot at (117, 360) with BGR value [  0   0 255]
Processed Frame 56: Red dot at (119, 360) with BGR value [  0   0 255]
```

```
Processed Frame 57: Red dot at (121, 360) with BGR value [  0   0 255]
Processed Frame 58: Red dot at (123, 360) with BGR value [  0   0 255]
Processed Frame 59: Red dot at (125, 360) with BGR value [  0   0 255]
Processed Frame 60: Red dot at (128, 360) with BGR value [  0   0 255]
Processed Frame 61: Red dot at (130, 360) with BGR value [  0   0 255]
Processed Frame 62: Red dot at (132, 360) with BGR value [  0   0 255]
Processed Frame 63: Red dot at (134, 360) with BGR value [  0   0 255]
Processed Frame 64: Red dot at (136, 360) with BGR value [  0   0 255]
Processed Frame 65: Red dot at (138, 360) with BGR value [  0   0 255]
Processed Frame 66: Red dot at (140, 360) with BGR value [  0   0 255]
Processed Frame 67: Red dot at (143, 360) with BGR value [  0   0 255]
Processed Frame 68: Red dot at (145, 360) with BGR value [  0   0 255]
Processed Frame 69: Red dot at (147, 360) with BGR value [  0   0 255]
Processed Frame 70: Red dot at (149, 360) with BGR value [  0   0 255]
Processed Frame 71: Red dot at (151, 360) with BGR value [  0   0 255]
Processed Frame 72: Red dot at (153, 360) with BGR value [  0   0 255]
Processed Frame 73: Red dot at (155, 360) with BGR value [  0   0 255]
Processed Frame 74: Red dot at (158, 360) with BGR value [  0   0 255]
Processed Frame 75: Red dot at (160, 360) with BGR value [  0   0 255]
Processed Frame 76: Red dot at (162, 360) with BGR value [  0   0 255]
Processed Frame 77: Red dot at (164, 360) with BGR value [  0   0 255]
Processed Frame 78: Red dot at (166, 360) with BGR value [  0   0 255]
Processed Frame 79: Red dot at (168, 360) with BGR value [  0   0 255]
Processed Frame 80: Red dot at (170, 360) with BGR value [  0   0 255]
Processed Frame 81: Red dot at (172, 360) with BGR value [  0   0 255]
Processed Frame 82: Red dot at (175, 360) with BGR value [  0   0 255]
Processed Frame 83: Red dot at (177, 360) with BGR value [  0   0 255]
Processed Frame 84: Red dot at (179, 360) with BGR value [  0   0 255]
Processed Frame 85: Red dot at (181, 360) with BGR value [  0   0 255]
Processed Frame 86: Red dot at (183, 360) with BGR value [  0   0 255]
Processed Frame 87: Red dot at (185, 360) with BGR value [  0   0 255]
Processed Frame 88: Red dot at (187, 360) with BGR value [  0   0 255]
Processed Frame 89: Red dot at (190, 360) with BGR value [  0   0 255]
Processed Frame 90: Red dot at (192, 360) with BGR value [  0   0 255]
Processed Frame 91: Red dot at (194, 360) with BGR value [  0   0 255]
Processed Frame 92: Red dot at (196, 360) with BGR value [  0   0 255]
Processed Frame 93: Red dot at (198, 360) with BGR value [  0   0 255]
Processed Frame 94: Red dot at (200, 360) with BGR value [  0   0 255]
Processed Frame 95: Red dot at (202, 360) with BGR value [  0   0 255]
Processed Frame 96: Red dot at (204, 360) with BGR value [  0   0 255]
Processed Frame 97: Red dot at (207, 360) with BGR value [  0   0 255]
Processed Frame 98: Red dot at (209, 360) with BGR value [  0   0 255]
Processed Frame 99: Red dot at (211, 360) with BGR value [  0   0 255]
Processed Frame 100: Red dot at (213, 360) with BGR value [  0   0 255]
Processed Frame 101: Red dot at (215, 360) with BGR value [  0   0 255]
Processed Frame 102: Red dot at (217, 360) with BGR value [  0   0 255]
Processed Frame 103: Red dot at (219, 360) with BGR value [  0   0 255]
Processed Frame 104: Red dot at (222, 360) with BGR value [  0   0 255]
Processed Frame 105: Red dot at (224, 360) with BGR value [  0   0 255]
Processed Frame 106: Red dot at (226, 360) with BGR value [  0   0 255]
Processed Frame 107: Red dot at (228, 360) with BGR value [  0   0 255]
Processed Frame 108: Red dot at (230, 360) with BGR value [  0   0 255]
Processed Frame 109: Red dot at (232, 360) with BGR value [  0   0 255]
Processed Frame 110: Red dot at (234, 360) with BGR value [  0   0 255]
Processed Frame 111: Red dot at (237, 360) with BGR value [  0   0 255]
Processed Frame 112: Red dot at (239, 360) with BGR value [  0   0 255]
Processed Frame 113: Red dot at (241, 360) with BGR value [  0   0 255]
Processed Frame 114: Red dot at (243, 360) with BGR value [  0   0 255]
Processed Frame 115: Red dot at (245, 360) with BGR value [  0   0 255]
Processed Frame 116: Red dot at (247, 360) with BGR value [  0   0 255]
```

```
Processed Frame 117: Red dot at (249, 360) with BGR value [  0   0 255]
Processed Frame 118: Red dot at (251, 360) with BGR value [  0   0 255]
Processed Frame 119: Red dot at (254, 360) with BGR value [  0   0 255]
Processed Frame 120: Red dot at (256, 360) with BGR value [  0   0 255]
Processed Frame 121: Red dot at (258, 360) with BGR value [  0   0 255]
Processed Frame 122: Red dot at (260, 360) with BGR value [  0   0 255]
Processed Frame 123: Red dot at (262, 360) with BGR value [  0   0 255]
Processed Frame 124: Red dot at (264, 360) with BGR value [  0   0 255]
Processed Frame 125: Red dot at (266, 360) with BGR value [  0   0 255]
Processed Frame 126: Red dot at (269, 360) with BGR value [  0   0 255]
Processed Frame 127: Red dot at (271, 360) with BGR value [  0   0 255]
Processed Frame 128: Red dot at (273, 360) with BGR value [  0   0 255]
Processed Frame 129: Red dot at (275, 360) with BGR value [  0   0 255]
Processed Frame 130: Red dot at (277, 360) with BGR value [  0   0 255]
Processed Frame 131: Red dot at (279, 360) with BGR value [  0   0 255]
Processed Frame 132: Red dot at (281, 360) with BGR value [  0   0 255]
Processed Frame 133: Red dot at (283, 360) with BGR value [  0   0 255]
Processed Frame 134: Red dot at (286, 360) with BGR value [  0   0 255]
Processed Frame 135: Red dot at (288, 360) with BGR value [  0   0 255]
Processed Frame 136: Red dot at (290, 360) with BGR value [  0   0 255]
Processed Frame 137: Red dot at (292, 360) with BGR value [  0   0 255]
Processed Frame 138: Red dot at (294, 360) with BGR value [  0   0 255]
Processed Frame 139: Red dot at (296, 360) with BGR value [  0   0 255]
Processed Frame 140: Red dot at (298, 360) with BGR value [  0   0 255]
Processed Frame 141: Red dot at (301, 360) with BGR value [  0   0 255]
Processed Frame 142: Red dot at (303, 360) with BGR value [  0   0 255]
Processed Frame 143: Red dot at (305, 360) with BGR value [  0   0 255]
Processed Frame 144: Red dot at (307, 360) with BGR value [  0   0 255]
Processed Frame 145: Red dot at (309, 360) with BGR value [  0   0 255]
Processed Frame 146: Red dot at (311, 360) with BGR value [  0   0 255]
Processed Frame 147: Red dot at (313, 360) with BGR value [  0   0 255]
Processed Frame 148: Red dot at (316, 360) with BGR value [  0   0 255]
Processed Frame 149: Red dot at (318, 360) with BGR value [  0   0 255]
Processed Frame 150: Red dot at (320, 360) with BGR value [  0   0 255]
Processed Frame 151: Red dot at (322, 360) with BGR value [  0   0 255]
Processed Frame 152: Red dot at (324, 360) with BGR value [  0   0 255]
Processed Frame 153: Red dot at (326, 360) with BGR value [  0   0 255]
Processed Frame 154: Red dot at (328, 360) with BGR value [  0   0 255]
Processed Frame 155: Red dot at (330, 360) with BGR value [  0   0 255]
Processed Frame 156: Red dot at (333, 360) with BGR value [  0   0 255]
Processed Frame 157: Red dot at (335, 360) with BGR value [  0   0 255]
Processed Frame 158: Red dot at (337, 360) with BGR value [  0   0 255]
Processed Frame 159: Red dot at (339, 360) with BGR value [  0   0 255]
Processed Frame 160: Red dot at (341, 360) with BGR value [  0   0 255]
Processed Frame 161: Red dot at (343, 360) with BGR value [  0   0 255]
Processed Frame 162: Red dot at (345, 360) with BGR value [  0   0 255]
Processed Frame 163: Red dot at (348, 360) with BGR value [  0   0 255]
Processed Frame 164: Red dot at (350, 360) with BGR value [  0   0 255]
Processed Frame 165: Red dot at (352, 360) with BGR value [  0   0 255]
Processed Frame 166: Red dot at (354, 360) with BGR value [  0   0 255]
Processed Frame 167: Red dot at (356, 360) with BGR value [  0   0 255]
Processed Frame 168: Red dot at (358, 360) with BGR value [  0   0 255]
Processed Frame 169: Red dot at (360, 360) with BGR value [  0   0 255]
Processed Frame 170: Red dot at (362, 360) with BGR value [  0   0 255]
Processed Frame 171: Red dot at (365, 360) with BGR value [  0   0 255]
Processed Frame 172: Red dot at (367, 360) with BGR value [  0   0 255]
Processed Frame 173: Red dot at (369, 360) with BGR value [  0   0 255]
Processed Frame 174: Red dot at (371, 360) with BGR value [  0   0 255]
Processed Frame 175: Red dot at (373, 360) with BGR value [  0   0 255]
Processed Frame 176: Red dot at (375, 360) with BGR value [  0   0 255]
```

```
Processed Frame 177: Red dot at (377, 360) with BGR value [  0   0 255]
Processed Frame 178: Red dot at (380, 360) with BGR value [  0   0 255]
Processed Frame 179: Red dot at (382, 360) with BGR value [  0   0 255]
Processed Frame 180: Red dot at (384, 360) with BGR value [  0   0 255]
Processed Frame 181: Red dot at (386, 360) with BGR value [  0   0 255]
Processed Frame 182: Red dot at (388, 360) with BGR value [  0   0 255]
Processed Frame 183: Red dot at (390, 360) with BGR value [  0   0 255]
Processed Frame 184: Red dot at (392, 360) with BGR value [  0   0 255]
Processed Frame 185: Red dot at (395, 360) with BGR value [  0   0 255]
Processed Frame 186: Red dot at (397, 360) with BGR value [  0   0 255]
Processed Frame 187: Red dot at (399, 360) with BGR value [  0   0 255]
Processed Frame 188: Red dot at (401, 360) with BGR value [  0   0 255]
Processed Frame 189: Red dot at (403, 360) with BGR value [  0   0 255]
Processed Frame 190: Red dot at (405, 360) with BGR value [  0   0 255]
Processed Frame 191: Red dot at (407, 360) with BGR value [  0   0 255]
Processed Frame 192: Red dot at (409, 360) with BGR value [  0   0 255]
Processed Frame 193: Red dot at (412, 360) with BGR value [  0   0 255]
Processed Frame 194: Red dot at (414, 360) with BGR value [  0   0 255]
Processed Frame 195: Red dot at (416, 360) with BGR value [  0   0 255]
Processed Frame 196: Red dot at (418, 360) with BGR value [  0   0 255]
Processed Frame 197: Red dot at (420, 360) with BGR value [  0   0 255]
Processed Frame 198: Red dot at (422, 360) with BGR value [  0   0 255]
Processed Frame 199: Red dot at (424, 360) with BGR value [  0   0 255]
Processed Frame 200: Red dot at (427, 360) with BGR value [  0   0 255]
Processed Frame 201: Red dot at (429, 360) with BGR value [  0   0 255]
Processed Frame 202: Red dot at (431, 360) with BGR value [  0   0 255]
Processed Frame 203: Red dot at (433, 360) with BGR value [  0   0 255]
Processed Frame 204: Red dot at (435, 360) with BGR value [  0   0 255]
Processed Frame 205: Red dot at (437, 360) with BGR value [  0   0 255]
Processed Frame 206: Red dot at (439, 360) with BGR value [  0   0 255]
Processed Frame 207: Red dot at (441, 360) with BGR value [  0   0 255]
Processed Frame 208: Red dot at (444, 360) with BGR value [  0   0 255]
Processed Frame 209: Red dot at (446, 360) with BGR value [  0   0 255]
Processed Frame 210: Red dot at (448, 360) with BGR value [  0   0 255]
Processed Frame 211: Red dot at (450, 360) with BGR value [  0   0 255]
Processed Frame 212: Red dot at (452, 360) with BGR value [  0   0 255]
Processed Frame 213: Red dot at (454, 360) with BGR value [  0   0 255]
Processed Frame 214: Red dot at (456, 360) with BGR value [  0   0 255]
Processed Frame 215: Red dot at (459, 360) with BGR value [  0   0 255]
Processed Frame 216: Red dot at (461, 360) with BGR value [  0   0 255]
Processed Frame 217: Red dot at (463, 360) with BGR value [  0   0 255]
Processed Frame 218: Red dot at (465, 360) with BGR value [  0   0 255]
Processed Frame 219: Red dot at (467, 360) with BGR value [  0   0 255]
Processed Frame 220: Red dot at (469, 360) with BGR value [  0   0 255]
Processed Frame 221: Red dot at (471, 360) with BGR value [  0   0 255]
Processed Frame 222: Red dot at (474, 360) with BGR value [  0   0 255]
Processed Frame 223: Red dot at (476, 360) with BGR value [  0   0 255]
Processed Frame 224: Red dot at (478, 360) with BGR value [  0   0 255]
Processed Frame 225: Red dot at (480, 360) with BGR value [  0   0 255]
Processed Frame 226: Red dot at (482, 360) with BGR value [  0   0 255]
Processed Frame 227: Red dot at (484, 360) with BGR value [  0   0 255]
Processed Frame 228: Red dot at (486, 360) with BGR value [  0   0 255]
Processed Frame 229: Red dot at (488, 360) with BGR value [  0   0 255]
Processed Frame 230: Red dot at (491, 360) with BGR value [  0   0 255]
Processed Frame 231: Red dot at (493, 360) with BGR value [  0   0 255]
Processed Frame 232: Red dot at (495, 360) with BGR value [  0   0 255]
Processed Frame 233: Red dot at (497, 360) with BGR value [  0   0 255]
Processed Frame 234: Red dot at (499, 360) with BGR value [  0   0 255]
Processed Frame 235: Red dot at (501, 360) with BGR value [  0   0 255]
Processed Frame 236: Red dot at (503, 360) with BGR value [  0   0 255]
```

```
Processed Frame 237: Red dot at (506, 360) with BGR value [  0   0 255]
Processed Frame 238: Red dot at (508, 360) with BGR value [  0   0 255]
Processed Frame 239: Red dot at (510, 360) with BGR value [  0   0 255]
Processed Frame 240: Red dot at (512, 360) with BGR value [  0   0 255]
Processed Frame 241: Red dot at (514, 360) with BGR value [  0   0 255]
Processed Frame 242: Red dot at (516, 360) with BGR value [  0   0 255]
Processed Frame 243: Red dot at (518, 360) with BGR value [  0   0 255]
Processed Frame 244: Red dot at (520, 360) with BGR value [  0   0 255]
Processed Frame 245: Red dot at (523, 360) with BGR value [  0   0 255]
Processed Frame 246: Red dot at (525, 360) with BGR value [  0   0 255]
Processed Frame 247: Red dot at (527, 360) with BGR value [  0   0 255]
Processed Frame 248: Red dot at (529, 360) with BGR value [  0   0 255]
Processed Frame 249: Red dot at (531, 360) with BGR value [  0   0 255]
Processed Frame 250: Red dot at (533, 360) with BGR value [  0   0 255]
Processed Frame 251: Red dot at (535, 360) with BGR value [  0   0 255]
Processed Frame 252: Red dot at (538, 360) with BGR value [  0   0 255]
Processed Frame 253: Red dot at (540, 360) with BGR value [  0   0 255]
Processed Frame 254: Red dot at (542, 360) with BGR value [  0   0 255]
Processed Frame 255: Red dot at (544, 360) with BGR value [  0   0 255]
Processed Frame 256: Red dot at (546, 360) with BGR value [  0   0 255]
Processed Frame 257: Red dot at (548, 360) with BGR value [  0   0 255]
Processed Frame 258: Red dot at (550, 360) with BGR value [  0   0 255]
Processed Frame 259: Red dot at (553, 360) with BGR value [  0   0 255]
Processed Frame 260: Red dot at (555, 360) with BGR value [  0   0 255]
Processed Frame 261: Red dot at (557, 360) with BGR value [  0   0 255]
Processed Frame 262: Red dot at (559, 360) with BGR value [  0   0 255]
Processed Frame 263: Red dot at (561, 360) with BGR value [  0   0 255]
Processed Frame 264: Red dot at (563, 360) with BGR value [  0   0 255]
Processed Frame 265: Red dot at (565, 360) with BGR value [  0   0 255]
Processed Frame 266: Red dot at (567, 360) with BGR value [  0   0 255]
Processed Frame 267: Red dot at (570, 360) with BGR value [  0   0 255]
Processed Frame 268: Red dot at (572, 360) with BGR value [  0   0 255]
Processed Frame 269: Red dot at (574, 360) with BGR value [  0   0 255]
Processed Frame 270: Red dot at (576, 360) with BGR value [  0   0 255]
Processed Frame 271: Red dot at (578, 360) with BGR value [  0   0 255]
Processed Frame 272: Red dot at (580, 360) with BGR value [  0   0 255]
Processed Frame 273: Red dot at (582, 360) with BGR value [  0   0 255]
Processed Frame 274: Red dot at (585, 360) with BGR value [  0   0 255]
Processed Frame 275: Red dot at (587, 360) with BGR value [  0   0 255]
Processed Frame 276: Red dot at (589, 360) with BGR value [  0   0 255]
Processed Frame 277: Red dot at (591, 360) with BGR value [  0   0 255]
Processed Frame 278: Red dot at (593, 360) with BGR value [  0   0 255]
Processed Frame 279: Red dot at (595, 360) with BGR value [  0   0 255]
Processed Frame 280: Red dot at (597, 360) with BGR value [  0   0 255]
Processed Frame 281: Red dot at (599, 360) with BGR value [  0   0 255]
Processed Frame 282: Red dot at (602, 360) with BGR value [  0   0 255]
Processed Frame 283: Red dot at (604, 360) with BGR value [  0   0 255]
Processed Frame 284: Red dot at (606, 360) with BGR value [  0   0 255]
Processed Frame 285: Red dot at (608, 360) with BGR value [  0   0 255]
Processed Frame 286: Red dot at (610, 360) with BGR value [  0   0 255]
Processed Frame 287: Red dot at (612, 360) with BGR value [  0   0 255]
Processed Frame 288: Red dot at (614, 360) with BGR value [  0   0 255]
Processed Frame 289: Red dot at (617, 360) with BGR value [  0   0 255]
Processed Frame 290: Red dot at (619, 360) with BGR value [  0   0 255]
Processed Frame 291: Red dot at (621, 360) with BGR value [  0   0 255]
Processed Frame 292: Red dot at (623, 360) with BGR value [  0   0 255]
Processed Frame 293: Red dot at (625, 360) with BGR value [  0   0 255]
Processed Frame 294: Red dot at (627, 360) with BGR value [  0   0 255]
Processed Frame 295: Red dot at (629, 360) with BGR value [  0   0 255]
Processed Frame 296: Red dot at (632, 360) with BGR value [  0   0 255]
```

```
Processed Frame 297: Red dot at (634, 360) with BGR value [  0   0 255]
Processed Frame 298: Red dot at (636, 360) with BGR value [  0   0 255]
Processed Frame 299: Red dot at (638, 360) with BGR value [  0   0 255]
Processed Frame 300: Red dot at (640, 360) with BGR value [  0   0 255]
Processed Frame 301: Red dot at (642, 360) with BGR value [  0   0 255]
Processed Frame 302: Red dot at (644, 360) with BGR value [  0   0 255]
Processed Frame 303: Red dot at (646, 360) with BGR value [  0   0 255]
Processed Frame 304: Red dot at (649, 360) with BGR value [  0   0 255]
Processed Frame 305: Red dot at (651, 360) with BGR value [  0   0 255]
Processed Frame 306: Red dot at (653, 360) with BGR value [  0   0 255]
Processed Frame 307: Red dot at (655, 360) with BGR value [  0   0 255]
Processed Frame 308: Red dot at (657, 360) with BGR value [  0   0 255]
Processed Frame 309: Red dot at (659, 360) with BGR value [  0   0 255]
Processed Frame 310: Red dot at (661, 360) with BGR value [  0   0 255]
Processed Frame 311: Red dot at (664, 360) with BGR value [  0   0 255]
Processed Frame 312: Red dot at (666, 360) with BGR value [  0   0 255]
Processed Frame 313: Red dot at (668, 360) with BGR value [  0   0 255]
Processed Frame 314: Red dot at (670, 360) with BGR value [  0   0 255]
Processed Frame 315: Red dot at (672, 360) with BGR value [  0   0 255]
Processed Frame 316: Red dot at (674, 360) with BGR value [  0   0 255]
Processed Frame 317: Red dot at (676, 360) with BGR value [  0   0 255]
Processed Frame 318: Red dot at (679, 360) with BGR value [  0   0 255]
Processed Frame 319: Red dot at (681, 360) with BGR value [  0   0 255]
Processed Frame 320: Red dot at (683, 360) with BGR value [  0   0 255]
Processed Frame 321: Red dot at (685, 360) with BGR value [  0   0 255]
Processed Frame 322: Red dot at (687, 360) with BGR value [  0   0 255]
Processed Frame 323: Red dot at (689, 360) with BGR value [  0   0 255]
Processed Frame 324: Red dot at (691, 360) with BGR value [  0   0 255]
Processed Frame 325: Red dot at (693, 360) with BGR value [  0   0 255]
Processed Frame 326: Red dot at (696, 360) with BGR value [  0   0 255]
Processed Frame 327: Red dot at (698, 360) with BGR value [  0   0 255]
Processed Frame 328: Red dot at (700, 360) with BGR value [  0   0 255]
Processed Frame 329: Red dot at (702, 360) with BGR value [  0   0 255]
Processed Frame 330: Red dot at (704, 360) with BGR value [  0   0 255]
Processed Frame 331: Red dot at (706, 360) with BGR value [  0   0 255]
Processed Frame 332: Red dot at (708, 360) with BGR value [  0   0 255]
Processed Frame 333: Red dot at (711, 360) with BGR value [  0   0 255]
Processed Frame 334: Red dot at (713, 360) with BGR value [  0   0 255]
Processed Frame 335: Red dot at (715, 360) with BGR value [  0   0 255]
Processed Frame 336: Red dot at (717, 360) with BGR value [  0   0 255]
Processed Frame 337: Red dot at (719, 360) with BGR value [  0   0 255]
Processed Frame 338: Red dot at (721, 360) with BGR value [  0   0 255]
Processed Frame 339: Red dot at (723, 360) with BGR value [  0   0 255]
Processed Frame 340: Red dot at (725, 360) with BGR value [  0   0 255]
Processed Frame 341: Red dot at (728, 360) with BGR value [  0   0 255]
Processed Frame 342: Red dot at (730, 360) with BGR value [  0   0 255]
Processed Frame 343: Red dot at (732, 360) with BGR value [  0   0 255]
Processed Frame 344: Red dot at (734, 360) with BGR value [  0   0 255]
Processed Frame 345: Red dot at (736, 360) with BGR value [  0   0 255]
Processed Frame 346: Red dot at (738, 360) with BGR value [  0   0 255]
Processed Frame 347: Red dot at (740, 360) with BGR value [  0   0 255]
Processed Frame 348: Red dot at (743, 360) with BGR value [  0   0 255]
Processed Frame 349: Red dot at (745, 360) with BGR value [  0   0 255]
Processed Frame 350: Red dot at (747, 360) with BGR value [  0   0 255]
Processed Frame 351: Red dot at (749, 360) with BGR value [  0   0 255]
Processed Frame 352: Red dot at (751, 360) with BGR value [  0   0 255]
Processed Frame 353: Red dot at (753, 360) with BGR value [  0   0 255]
Processed Frame 354: Red dot at (755, 360) with BGR value [  0   0 255]
Processed Frame 355: Red dot at (758, 360) with BGR value [  0   0 255]
Processed Frame 356: Red dot at (760, 360) with BGR value [  0   0 255]
```

```
Processed Frame 357: Red dot at (762, 360) with BGR value [  0   0 255]
Processed Frame 358: Red dot at (764, 360) with BGR value [  0   0 255]
Processed Frame 359: Red dot at (766, 360) with BGR value [  0   0 255]
Processed Frame 360: Red dot at (768, 360) with BGR value [  0   0 255]
Processed Frame 361: Red dot at (770, 360) with BGR value [  0   0 255]
Processed Frame 362: Red dot at (772, 360) with BGR value [  0   0 255]
Processed Frame 363: Red dot at (775, 360) with BGR value [  0   0 255]
Processed Frame 364: Red dot at (777, 360) with BGR value [  0   0 255]
Processed Frame 365: Red dot at (779, 360) with BGR value [  0   0 255]
Processed Frame 366: Red dot at (781, 360) with BGR value [  0   0 255]
Processed Frame 367: Red dot at (783, 360) with BGR value [  0   0 255]
Processed Frame 368: Red dot at (785, 360) with BGR value [  0   0 255]
Processed Frame 369: Red dot at (787, 360) with BGR value [  0   0 255]
Processed Frame 370: Red dot at (790, 360) with BGR value [  0   0 255]
Processed Frame 371: Red dot at (792, 360) with BGR value [  0   0 255]
Processed Frame 372: Red dot at (794, 360) with BGR value [  0   0 255]
Processed Frame 373: Red dot at (796, 360) with BGR value [  0   0 255]
Processed Frame 374: Red dot at (798, 360) with BGR value [  0   0 255]
Processed Frame 375: Red dot at (800, 360) with BGR value [  0   0 255]
Processed Frame 376: Red dot at (802, 360) with BGR value [  0   0 255]
Processed Frame 377: Red dot at (804, 360) with BGR value [  0   0 255]
Processed Frame 378: Red dot at (807, 360) with BGR value [  0   0 255]
Processed Frame 379: Red dot at (809, 360) with BGR value [  0   0 255]
Processed Frame 380: Red dot at (811, 360) with BGR value [  0   0 255]
Processed Frame 381: Red dot at (813, 360) with BGR value [  0   0 255]
Processed Frame 382: Red dot at (815, 360) with BGR value [  0   0 255]
Processed Frame 383: Red dot at (817, 360) with BGR value [  0   0 255]
Processed Frame 384: Red dot at (819, 360) with BGR value [  0   0 255]
Processed Frame 385: Red dot at (822, 360) with BGR value [  0   0 255]
Processed Frame 386: Red dot at (824, 360) with BGR value [  0   0 255]
Processed Frame 387: Red dot at (826, 360) with BGR value [  0   0 255]
Processed Frame 388: Red dot at (828, 360) with BGR value [  0   0 255]
Processed Frame 389: Red dot at (830, 360) with BGR value [  0   0 255]
Processed Frame 390: Red dot at (832, 360) with BGR value [  0   0 255]
Processed Frame 391: Red dot at (834, 360) with BGR value [  0   0 255]
Processed Frame 392: Red dot at (837, 360) with BGR value [  0   0 255]
Processed Frame 393: Red dot at (839, 360) with BGR value [  0   0 255]
Processed Frame 394: Red dot at (841, 360) with BGR value [  0   0 255]
Processed Frame 395: Red dot at (843, 360) with BGR value [  0   0 255]
Processed Frame 396: Red dot at (845, 360) with BGR value [  0   0 255]
Processed Frame 397: Red dot at (847, 360) with BGR value [  0   0 255]
Processed Frame 398: Red dot at (849, 360) with BGR value [  0   0 255]
Processed Frame 399: Red dot at (851, 360) with BGR value [  0   0 255]
Processed Frame 400: Red dot at (854, 360) with BGR value [  0   0 255]
Processed Frame 401: Red dot at (856, 360) with BGR value [  0   0 255]
Processed Frame 402: Red dot at (858, 360) with BGR value [  0   0 255]
Processed Frame 403: Red dot at (860, 360) with BGR value [  0   0 255]
Processed Frame 404: Red dot at (862, 360) with BGR value [  0   0 255]
Processed Frame 405: Red dot at (864, 360) with BGR value [  0   0 255]
Processed Frame 406: Red dot at (866, 360) with BGR value [  0   0 255]
Processed Frame 407: Red dot at (869, 360) with BGR value [  0   0 255]
Processed Frame 408: Red dot at (871, 360) with BGR value [  0   0 255]
Processed Frame 409: Red dot at (873, 360) with BGR value [  0   0 255]
Processed Frame 410: Red dot at (875, 360) with BGR value [  0   0 255]
Processed Frame 411: Red dot at (877, 360) with BGR value [  0   0 255]
Processed Frame 412: Red dot at (879, 360) with BGR value [  0   0 255]
Processed Frame 413: Red dot at (881, 360) with BGR value [  0   0 255]
Processed Frame 414: Red dot at (883, 360) with BGR value [  0   0 255]
Processed Frame 415: Red dot at (886, 360) with BGR value [  0   0 255]
Processed Frame 416: Red dot at (888, 360) with BGR value [  0   0 255]
```

```
Processed Frame 417: Red dot at (890, 360) with BGR value [  0   0 255]
Processed Frame 418: Red dot at (892, 360) with BGR value [  0   0 255]
Processed Frame 419: Red dot at (894, 360) with BGR value [  0   0 255]
Processed Frame 420: Red dot at (896, 360) with BGR value [  0   0 255]
Processed Frame 421: Red dot at (898, 360) with BGR value [  0   0 255]
Processed Frame 422: Red dot at (901, 360) with BGR value [  0   0 255]
Processed Frame 423: Red dot at (903, 360) with BGR value [  0   0 255]
Processed Frame 424: Red dot at (905, 360) with BGR value [  0   0 255]
Processed Frame 425: Red dot at (907, 360) with BGR value [  0   0 255]
Processed Frame 426: Red dot at (909, 360) with BGR value [  0   0 255]
Processed Frame 427: Red dot at (911, 360) with BGR value [  0   0 255]
Processed Frame 428: Red dot at (913, 360) with BGR value [  0   0 255]
Processed Frame 429: Red dot at (916, 360) with BGR value [  0   0 255]
Processed Frame 430: Red dot at (918, 360) with BGR value [  0   0 255]
Processed Frame 431: Red dot at (920, 360) with BGR value [  0   0 255]
Processed Frame 432: Red dot at (922, 360) with BGR value [  0   0 255]
Processed Frame 433: Red dot at (924, 360) with BGR value [  0   0 255]
Processed Frame 434: Red dot at (926, 360) with BGR value [  0   0 255]
Processed Frame 435: Red dot at (928, 360) with BGR value [  0   0 255]
Processed Frame 436: Red dot at (930, 360) with BGR value [  0   0 255]
Processed Frame 437: Red dot at (933, 360) with BGR value [  0   0 255]
Processed Frame 438: Red dot at (935, 360) with BGR value [  0   0 255]
Processed Frame 439: Red dot at (937, 360) with BGR value [  0   0 255]
Processed Frame 440: Red dot at (939, 360) with BGR value [  0   0 255]
Processed Frame 441: Red dot at (941, 360) with BGR value [  0   0 255]
Processed Frame 442: Red dot at (943, 360) with BGR value [  0   0 255]
Processed Frame 443: Red dot at (945, 360) with BGR value [  0   0 255]
Processed Frame 444: Red dot at (948, 360) with BGR value [  0   0 255]
Processed Frame 445: Red dot at (950, 360) with BGR value [  0   0 255]
Processed Frame 446: Red dot at (952, 360) with BGR value [  0   0 255]
Processed Frame 447: Red dot at (954, 360) with BGR value [  0   0 255]
Processed Frame 448: Red dot at (956, 360) with BGR value [  0   0 255]
Processed Frame 449: Red dot at (958, 360) with BGR value [  0   0 255]
Processed Frame 450: Red dot at (960, 360) with BGR value [  0   0 255]
Processed Frame 451: Red dot at (962, 360) with BGR value [  0   0 255]
Processed Frame 452: Red dot at (965, 360) with BGR value [  0   0 255]
Processed Frame 453: Red dot at (967, 360) with BGR value [  0   0 255]
Processed Frame 454: Red dot at (969, 360) with BGR value [  0   0 255]
Processed Frame 455: Red dot at (971, 360) with BGR value [  0   0 255]
Processed Frame 456: Red dot at (973, 360) with BGR value [  0   0 255]
Processed Frame 457: Red dot at (975, 360) with BGR value [  0   0 255]
Processed Frame 458: Red dot at (977, 360) with BGR value [  0   0 255]
Processed Frame 459: Red dot at (980, 360) with BGR value [  0   0 255]
Processed Frame 460: Red dot at (982, 360) with BGR value [  0   0 255]
Processed Frame 461: Red dot at (984, 360) with BGR value [  0   0 255]
Processed Frame 462: Red dot at (986, 360) with BGR value [  0   0 255]
Processed Frame 463: Red dot at (988, 360) with BGR value [  0   0 255]
Processed Frame 464: Red dot at (990, 360) with BGR value [  0   0 255]
Processed Frame 465: Red dot at (992, 360) with BGR value [  0   0 255]
Processed Frame 466: Red dot at (995, 360) with BGR value [  0   0 255]
Processed Frame 467: Red dot at (997, 360) with BGR value [  0   0 255]
Processed Frame 468: Red dot at (999, 360) with BGR value [  0   0 255]
Processed Frame 469: Red dot at (1001, 360) with BGR value [  0   0 255]
Processed Frame 470: Red dot at (1003, 360) with BGR value [  0   0 255]
Processed Frame 471: Red dot at (1005, 360) with BGR value [  0   0 255]
Processed Frame 472: Red dot at (1007, 360) with BGR value [  0   0 255]
Processed Frame 473: Red dot at (1009, 360) with BGR value [  0   0 255]
Processed Frame 474: Red dot at (1012, 360) with BGR value [  0   0 255]
Processed Frame 475: Red dot at (1014, 360) with BGR value [  0   0 255]
Processed Frame 476: Red dot at (1016, 360) with BGR value [  0   0 255]
```

```
Processed Frame 477: Red dot at (1018, 360) with BGR value [  0   0 255]
Processed Frame 478: Red dot at (1020, 360) with BGR value [  0   0 255]
Processed Frame 479: Red dot at (1022, 360) with BGR value [  0   0 255]
Processed Frame 480: Red dot at (1024, 360) with BGR value [  0   0 255]
Processed Frame 481: Red dot at (1027, 360) with BGR value [  0   0 255]
Processed Frame 482: Red dot at (1029, 360) with BGR value [  0   0 255]
Processed Frame 483: Red dot at (1031, 360) with BGR value [  0   0 255]
Processed Frame 484: Red dot at (1033, 360) with BGR value [  0   0 255]
Processed Frame 485: Red dot at (1035, 360) with BGR value [  0   0 255]
Processed Frame 486: Red dot at (1037, 360) with BGR value [  0   0 255]
Processed Frame 487: Red dot at (1039, 360) with BGR value [  0   0 255]
Processed Frame 488: Red dot at (1041, 360) with BGR value [  0   0 255]
Processed Frame 489: Red dot at (1044, 360) with BGR value [  0   0 255]
Processed Frame 490: Red dot at (1046, 360) with BGR value [  0   0 255]
Processed Frame 491: Red dot at (1048, 360) with BGR value [  0   0 255]
Processed Frame 492: Red dot at (1050, 360) with BGR value [  0   0 255]
Processed Frame 493: Red dot at (1052, 360) with BGR value [  0   0 255]
Processed Frame 494: Red dot at (1054, 360) with BGR value [  0   0 255]
Processed Frame 495: Red dot at (1056, 360) with BGR value [  0   0 255]
Processed Frame 496: Red dot at (1059, 360) with BGR value [  0   0 255]
Processed Frame 497: Red dot at (1061, 360) with BGR value [  0   0 255]
Processed Frame 498: Red dot at (1063, 360) with BGR value [  0   0 255]
Processed Frame 499: Red dot at (1065, 360) with BGR value [  0   0 255]
Processed Frame 500: Red dot at (1067, 360) with BGR value [  0   0 255]
Processed Frame 501: Red dot at (1069, 360) with BGR value [  0   0 255]
Processed Frame 502: Red dot at (1071, 360) with BGR value [  0   0 255]
Processed Frame 503: Red dot at (1074, 360) with BGR value [  0   0 255]
Processed Frame 504: Red dot at (1076, 360) with BGR value [  0   0 255]
Processed Frame 505: Red dot at (1078, 360) with BGR value [  0   0 255]
Processed Frame 506: Red dot at (1080, 360) with BGR value [  0   0 255]
Processed Frame 507: Red dot at (1082, 360) with BGR value [  0   0 255]
Processed Frame 508: Red dot at (1084, 360) with BGR value [  0   0 255]
Processed Frame 509: Red dot at (1086, 360) with BGR value [  0   0 255]
Processed Frame 510: Red dot at (1088, 360) with BGR value [  0   0 255]
Processed Frame 511: Red dot at (1091, 360) with BGR value [  0   0 255]
Processed Frame 512: Red dot at (1093, 360) with BGR value [  0   0 255]
Processed Frame 513: Red dot at (1095, 360) with BGR value [  0   0 255]
Processed Frame 514: Red dot at (1097, 360) with BGR value [  0   0 255]
Processed Frame 515: Red dot at (1099, 360) with BGR value [  0   0 255]
Processed Frame 516: Red dot at (1101, 360) with BGR value [  0   0 255]
Processed Frame 517: Red dot at (1103, 360) with BGR value [  0   0 255]
Processed Frame 518: Red dot at (1106, 360) with BGR value [  0   0 255]
Processed Frame 519: Red dot at (1108, 360) with BGR value [  0   0 255]
Processed Frame 520: Red dot at (1110, 360) with BGR value [  0   0 255]
Processed Frame 521: Red dot at (1112, 360) with BGR value [  0   0 255]
Processed Frame 522: Red dot at (1114, 360) with BGR value [  0   0 255]
Processed Frame 523: Red dot at (1116, 360) with BGR value [  0   0 255]
Processed Frame 524: Red dot at (1118, 360) with BGR value [  0   0 255]
Processed Frame 525: Red dot at (1120, 360) with BGR value [  0   0 255]
Processed Frame 526: Red dot at (1123, 360) with BGR value [  0   0 255]
Processed Frame 527: Red dot at (1125, 360) with BGR value [  0   0 255]
Processed Frame 528: Red dot at (1127, 360) with BGR value [  0   0 255]
Processed Frame 529: Red dot at (1129, 360) with BGR value [  0   0 255]
Processed Frame 530: Red dot at (1131, 360) with BGR value [  0   0 255]
Processed Frame 531: Red dot at (1133, 360) with BGR value [  0   0 255]
Processed Frame 532: Red dot at (1135, 360) with BGR value [  0   0 255]
Processed Frame 533: Red dot at (1138, 360) with BGR value [  0   0 255]
Processed Frame 534: Red dot at (1140, 360) with BGR value [  0   0 255]
Processed Frame 535: Red dot at (1142, 360) with BGR value [  0   0 255]
Processed Frame 536: Red dot at (1144, 360) with BGR value [  0   0 255]
```

```
Processed Frame 537: Red dot at (1146, 360) with BGR value [  0   0 255]
Processed Frame 538: Red dot at (1148, 360) with BGR value [  0   0 255]
Processed Frame 539: Red dot at (1150, 360) with BGR value [  0   0 255]
Processed Frame 540: Red dot at (1153, 360) with BGR value [  0   0 255]
Processed Frame 541: Red dot at (1155, 360) with BGR value [  0   0 255]
Processed Frame 542: Red dot at (1157, 360) with BGR value [  0   0 255]
Processed Frame 543: Red dot at (1159, 360) with BGR value [  0   0 255]
Processed Frame 544: Red dot at (1161, 360) with BGR value [  0   0 255]
Processed Frame 545: Red dot at (1163, 360) with BGR value [  0   0 255]
Processed Frame 546: Red dot at (1165, 360) with BGR value [  0   0 255]
Processed Frame 547: Red dot at (1167, 360) with BGR value [  0   0 255]
Processed Frame 548: Red dot at (1170, 360) with BGR value [  0   0 255]
Processed Frame 549: Red dot at (1172, 360) with BGR value [  0   0 255]
Processed Frame 550: Red dot at (1174, 360) with BGR value [  0   0 255]
Processed Frame 551: Red dot at (1176, 360) with BGR value [  0   0 255]
Processed Frame 552: Red dot at (1178, 360) with BGR value [  0   0 255]
Processed Frame 553: Red dot at (1180, 360) with BGR value [  0   0 255]
Processed Frame 554: Red dot at (1182, 360) with BGR value [  0   0 255]
Processed Frame 555: Red dot at (1185, 360) with BGR value [  0   0 255]
Processed Frame 556: Red dot at (1187, 360) with BGR value [  0   0 255]
Processed Frame 557: Red dot at (1189, 360) with BGR value [  0   0 255]
Processed Frame 558: Red dot at (1191, 360) with BGR value [  0   0 255]
Processed Frame 559: Red dot at (1193, 360) with BGR value [  0   0 255]
Processed Frame 560: Red dot at (1195, 360) with BGR value [  0   0 255]
Processed Frame 561: Red dot at (1197, 360) with BGR value [  0   0 255]
Processed Frame 562: Red dot at (1199, 360) with BGR value [  0   0 255]
Processed Frame 563: Red dot at (1202, 360) with BGR value [  0   0 255]
Processed Frame 564: Red dot at (1204, 360) with BGR value [  0   0 255]
Processed Frame 565: Red dot at (1206, 360) with BGR value [  0   0 255]
Processed Frame 566: Red dot at (1208, 360) with BGR value [  0   0 255]
Processed Frame 567: Red dot at (1210, 360) with BGR value [  0   0 255]
Processed Frame 568: Red dot at (1212, 360) with BGR value [  0   0 255]
Processed Frame 569: Red dot at (1214, 360) with BGR value [  0   0 255]
Processed Frame 570: Red dot at (1217, 360) with BGR value [  0   0 255]
Processed Frame 571: Red dot at (1219, 360) with BGR value [  0   0 255]
Processed Frame 572: Red dot at (1221, 360) with BGR value [  0   0 255]
Processed Frame 573: Red dot at (1223, 360) with BGR value [  0   0 255]
Processed Frame 574: Red dot at (1225, 360) with BGR value [  0   0 255]
Processed Frame 575: Red dot at (1227, 360) with BGR value [  0   0 255]
Processed Frame 576: Red dot at (1229, 360) with BGR value [  0   0 255]
Processed Frame 577: Red dot at (1232, 360) with BGR value [  0   0 255]
Processed Frame 578: Red dot at (1234, 360) with BGR value [  0   0 255]
Processed Frame 579: Red dot at (1236, 360) with BGR value [  0   0 255]
Processed Frame 580: Red dot at (1238, 360) with BGR value [  0   0 255]
Processed Frame 581: Red dot at (1240, 360) with BGR value [  0   0 255]
Processed Frame 582: Red dot at (1242, 360) with BGR value [  0   0 255]
Processed Frame 583: Red dot at (1244, 360) with BGR value [  0   0 255]
Processed Frame 584: Red dot at (1246, 360) with BGR value [  0   0 255]
Processed Frame 585: Red dot at (1249, 360) with BGR value [  0   0 255]
Processed Frame 586: Red dot at (1251, 360) with BGR value [  0   0 255]
Processed Frame 587: Red dot at (1253, 360) with BGR value [  0   0 255]
Processed Frame 588: Red dot at (1255, 360) with BGR value [  0   0 255]
Processed Frame 589: Red dot at (1257, 360) with BGR value [  0   0 255]
Processed Frame 590: Red dot at (1259, 360) with BGR value [  0   0 255]
Processed Frame 591: Red dot at (1261, 360) with BGR value [  0   0 255]
Processed Frame 592: Red dot at (1264, 360) with BGR value [  0   0 255]
Processed Frame 593: Red dot at (1266, 360) with BGR value [  0   0 255]
Processed Frame 594: Red dot at (1268, 360) with BGR value [  0   0 255]
Processed Frame 595: Red dot at (1270, 360) with BGR value [  0   0 255]
Processed Frame 596: Red dot at (1272, 360) with BGR value [  0   0 255]
```

```
Processed Frame 597: Red dot at (1274, 360) with BGR value [  0   0 255]
Processed Frame 598: Red dot at (1276, 360) with BGR value [  0   0 255]
Processed Frame 599: Red dot at (1279, 360) with BGR value [  0   0 255]
Processed 600 frames.
```

Display the Low Fps Video and show some of the frames

In [10]:
```python
low_fps_video_path = os.path.join(os.getcwd(), 'attachment', 'video_low_fps.mp4'

cap = cv2.VideoCapture(low_fps_video_path)
frames= []
while True:
    ret, frame = cap.read()
    if not ret:
        break
    frames.append(frame)

frames_array = np.array(frames)
cap.release()

print(f"Shape of frames_array: {frames_array.shape}")

# Extract R, G, B channels
r_channel, g_channel, b_channel = frames_array[:, :, 0], frames_array[:, :, 1],

## Print length of a  frame
print(f"Length of a frame: {frames_array[0].shape}")

## Print rgb value of this frame
print(f"RGB value of the frame: {frames_array[0][3]}")

single_img = frames_array[200].copy()
single_img = cv2.cvtColor(single_img, cv2.COLOR_BGR2RGB)

plt.figure(figsize=(5, 5))
plt.imshow(single_img)
plt.axis('off')
plt.show()
```

```
Shape of frames_array: (600, 720, 1280, 3)
Length of a frame: (720, 1280, 3)
RGB value of the frame: [[200 203 201]
 [183 186 184]
 [143 146 144]
 ...
 [167 170 168]
 [167 170 168]
 [167 170 168]]
```

Explanation

Video yang digunakan mempunyai durasi 60 detik dengan 30 fps. Karena akan di ambil setiap frame setiap 3 gambar, maka idealnya proses fps akan berkurang menjadi $\frac{30}{3} = 10$ fps. Dan memang hasil akhir video menunjukan 10 fps.

Untuk grayscale sendiri saya baru tahu, bahwasannya proses konversi dari RGB / BGR ke Grayscale dan sebaliknya hanya untuk menambah / menghapus 3 chanel warna dari sebuah frame video.

Informasi mengenai nilai RGB frame sebelumnya sudah hilang ketika dilakukan konversi dari RGB ke Grayscale, nilai grayscale tersebut akan tetap sama meskipun di konversi balik menjadi format RGB / BGR, hanya saja terdapat 3 chanel warna yang ditambahkan.

Setelah menambahkan 3 chanel warna, kita dapat melakukan proses penambahan titik pada pixel dan amplifikasi frame tersebut sehingga dapat terlihat dengan mudah di layar.

```
# Calculate the position of the red dot
    x = int(processed_frame_num * step)
    y = frame_size[1] // 2  # Center row (height)

    # Ensure x is within bounds
    x = min(x, frame_size[0] - 1)

    # Add the red dot manually
    # OpenCV uses BGR, so Red is [0, 0, 255]
    bgr_frame[y, x] = [0, 0, 255]
```

Proses ini akan menghitung posisi dari titik merah berdasarkan frame (karena kita membagi frame menjadi setiap 3 frame, maka red dot akan di gambar pada setiap 3 frame tersbut, lalu bergerak ke kanan).

Di sisi lain, dot merah ini masih terlalu kecil, maka akan dilakukan amplifikasi untuk menambah ukuran dari dot tersebut.

```
dot_radius = 5
for dy in range(-dot_radius, dot_radius + 1):
    for dx in range(-dot_radius, dot_radius + 1):
        if dx**2 + dy**2 <= dot_radius**2:
```

```
            nx, ny = x + dx, y + dy
            if 0 <= nx < frame_size[0] and 0 <= ny < frame_size[1]:
                bgr_frame[ny, nx] = [0, 0, 255]
```

Proses disini cukup sederhana, dot tersebut akan di amplifikasikan untuk setiap panjang dan lebar dari dot tersebut lalu di masukan ke frame.

Detection with DLIB

Task 4

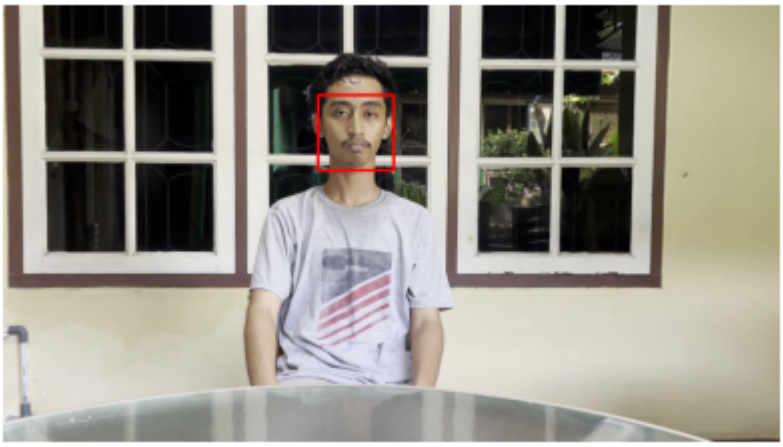
Using DLIB, based on the Face ROI (Region of Interest) and expand for shoulder and chest

In [13]:
```
## Moving the frame video into nuympy array
video_path = os.path.join(os.getcwd(), 'attachment', 'sample-renamed.mp4')

videoCapture = cv2.VideoCapture(video_path)
video_frames = []

## Adding the frames to the video_frames list
while True:
    ret, frame = videoCapture.read()
    if not ret:
        break
    ## Converting into RGB value
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    video_frames.append(frame)

frames_array = np.array(video_frames)
videoCapture.release()

## Print the shape of the frames_array
print(f"Shape of frames_array: {frames_array.shape}")
```

```
Shape of frames_array: (1800, 1080, 1920, 3)
```

In [14]:
```
## Showing a frame of the original image
single_img = frames_array[200].copy()
plt.figure(figsize=(5, 5))
plt.imshow(single_img)
plt.axis('off')
plt.show()
```

In [15]:
```
detector = dlib.get_frontal_face_detector()

faces = detector(single_img, 1)
for i, face in enumerate(faces): # untuk setiap wajah yang terdeteksi (bisa saja
    x, y, w, h = face.left(), face.top(), face.width(), face.height()
    x2 = x + w
    y2 = y + h
    cv2.rectangle(single_img, (x, y), (x2, y2), (255, 0, 0), 5)

plt.figure(figsize=(5, 5))
plt.imshow(single_img)
plt.axis('off')
plt.show()
```

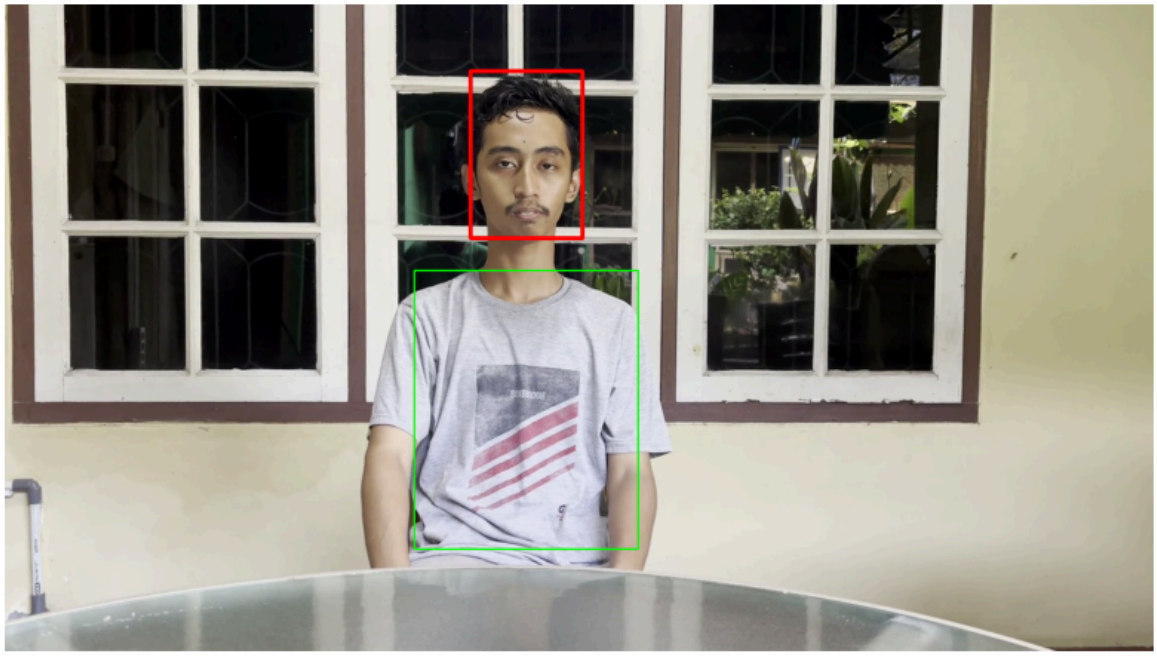


Task 4

Based on the Face ROI (Expand for shoulder and chest)

In [16]:
```
## Setup the Dlib
detector = dlib.get_frontal_face_detector()

## Obtain a image and faces detection frame
single_img  = frames_array[200].copy()
faces = detector(single_img, 1)

## Loop through the faces and draw the bounding box
for face in faces:
    x, y, w, h = (face.left(), face.top(), face.width(), face.height())

    ## Adding the bounding box to the hair
    y_hair = int(y - (0.5 * y))
    h_hair = int(h + (0.5 * h))
    cv2.rectangle(single_img, (x, y_hair), (x2, h_hair + y_hair), (255, 0, 0), 5


    # Adjust ROI to include shoulders and chest
    roi_x = int(x - (0.5 * w))
    roi_y = int(y * 2)
    roi_w = int(w * 2)
    roi_h = int(h * 2.5)  # Adjust the height to include shoulders and chest

    # Ensure the new ROI does not go out of frame bounds
    roi_h = min(roi_h, single_img.shape[0] - roi_y)
```

```
    # Draw rectangle around the adjusted ROI
    cv2.rectangle(single_img, (roi_x, roi_y), (roi_x+roi_w, roi_y+roi_h), (0, 25

# Display the image with the adjusted bounding box
plt.figure(figsize=(10, 10))
plt.imshow(single_img)
plt.axis('off')
plt.show()
```



Explanation

```
## Iteration
for i, face in enumerate(faces):
```

- **`for i, face in enumerate(faces)`** : Ini adalah loop yang akan berjalan sebanyak jumlah wajah yang terdeteksi.
  - **`enumerate(faces)`** : `enumerate` memberikan dua nilai untuk setiap iterasi loop:
    - `i` : Indeks dari wajah yang terdeteksi (dimulai dari 0).
    - `face` : Objek bounding box dari setiap wajah yang terdeteksi. Objek ini memiliki koordinat posisi wajah dalam gambar.

```
## Set domain
x, y, w, h = face.left(), face.top(), face.width(), face.height()
```

- **`face.left()`** : Mengambil koordinat x dari tepi kiri kotak pembatas (bounding box) wajah.
- **`face.top()`** : Mengambil koordinat y dari tepi atas kotak pembatas wajah.
- **`face.width()`** : Mengambil lebar bounding box wajah.
- **`face.height()`** : Mengambil tinggi bounding box wajah.
- **`x, y, w, h`** : Variabel ini menyimpan posisi dan ukuran dari bounding box wajah yang terdeteksi, di mana:
  - `x` : Koordinat x dari sudut kiri atas wajah.

- `y` : Koordinat y dari sudut kiri atas wajah.
- `w` : Lebar bounding box wajah.
- `h` : Tinggi bounding box wajah.

```
## Set bouding for width
x2 = x + w
```

- **`x2 = x + w`** : Menghitung koordinat x dari sudut kanan bawah bounding box. Ini didapat dengan menjumlahkan nilai `x` (koordinat kiri atas) dengan `w` (lebar bounding box).

```
## Set bouding for height
y2 = y + h
```

- **`y2 = y + h`** : Menghitung koordinat y dari sudut kanan bawah bounding box. Ini didapat dengan menjumlahkan nilai `y` (koordinat kiri atas) dengan `h` (tinggi bounding box).

```
## Set bounding box for the face
cv2.rectangle(single_img, (x, y), (x2, y2), (255, 0, 0), 5)
```

- **`cv2.rectangle(single_img, (x, y), (x2, y2), (255, 0, 0), 5)`** :
  - Ini menggunakan OpenCV (`cv2`) untuk menggambar kotak persegi panjang (bounding box) di sekitar wajah yang terdeteksi.
  - **`single_img`** : Gambar tempat kotak akan digambar.
  - **`(x, y)`** : Titik sudut kiri atas dari bounding box (posisi awal persegi).
  - **`(x2, y2)`** : Titik sudut kanan bawah dari bounding box (posisi akhir persegi).
  - **`(255, 0, 0)`** : Warna persegi panjang dalam format BGR (Biru, Hijau, Merah), di mana (255, 0, 0) berarti biru.
  - **`5`** : Ketebalan garis persegi panjang.

Bagian bahu

Untuk bagian dada dan bahu, kita dapat melakukan seleksi manual untuk membuat bounding box dengan mendaptkan nilai `x,y,w,h` dari wajah dan kita lakukan proses secara manual untuk membuat bounding box tersebut.

Facial Tracking dengan Optical Flow

Menggunakan dlib untuk melakukan re-deteksi setiap frame, sangat tidak efisien dan akan memakan proses komputasi yang lama.

Proses facial tracking memungkinkan untuk prediksi identifikasi wajah untuk frame selanjutnya tanpa harus melakukan re-detetksi wajah

Task 5

In seconds 25 - 40, using facial tracking for the face that got detected. Save the video in drive.

Here's the link of the video: Link

Explanation

Proses facial tracking yang dilakukan pada file `video_processing` melakukan re-deteksi setiap 3 frame, bukanlah proses yang cukup efisien.

Lucas Canade Optical Flow cukup bagus ketika melakukan prediksi wajah pada objek yang stabil (tidak ada gerakan tiba-tiba). Namun untuk sampel video pada detik 25-40, terdapat gerakan kepala ke kiri dan kanan sehinggga bounding box ketika terjadi gerakan teresbut akan semakin meleset dari wajah.

Salah satu cara yang dilakukan dengan optical flow adalah dengan membuat sebuah `tracking failure`, jadi ketika nilai dari optical flow (Lucas Canade Optical Flow) status itu valid (1) atau tidak (0) sebelum melakukan proses re-deteksi wajah

Implementasi

Pertama, akan dilakukan proses deteksi wajah dengan dlib untuk mendapatkan bounding box awal.

Selanjutnya akan digunkan proses `Lucas Canade Optical Flow` untuk melakukan cek pada frame selanjutnya apakah wajah masih terprediksi.

```
# Use optical flow to track the bounding box corners
new_points, status, _ = cv2.calcOpticalFlowPyrLK(old_gray, gray_frame,
bbox_points, None, **lk_params)
```

Metode ini menghasilkan point baru dan nilai dari status tersebut, jika nilai status 1, maka prediksi tracking suksses dan sebaliknya. Nilai dari status ini bisa di gunakan sebagai kondisi untuk melakukan re-deteksi.

```
# Check if tracking fails
if np.sum(status) < len(bbox_points) * 0.5:  # Re-detect if less than
50% points are tracked
    faces = detector(gray_frame)
    if len(faces) > 0:
        face = faces[0]
        bbox_points = np.array([[face.left(), face.top()],
                                [face.right(), face.top()],
                                [face.right(), face.bottom()],
                                [face.left(), face.bottom()]],
dtype=np.float32).reshape(-1, 1, 2)
else:
    bbox_points = new_points
```

Terakhir, akan dilakukkan proses penggambaran bounding box pada wajah.

In [17]:
```
import dlib
import cv2
import numpy as np
import os
```

```
# Initialize dlib's face detector
detector = dlib.get_frontal_face_detector()

# Initialize video capture
VID_PATH = os.path.join(os.getcwd(), 'attachment', 'sample-renamed.mp4')

cap = cv2.VideoCapture(VID_PATH)

# Get video dimensions
frame_width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
frame_height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

# Define the codec and create VideoWriter object
out = cv2.VideoWriter('output.avi', cv2.VideoWriter_fourcc(*'XVID'), 30.0, (fram

# Parameters for Lucas-Kanade Optical Flow
lk_params = dict(winSize=(15, 15), maxLevel=2,
                 criteria=(cv2.TERM_CRITERIA_EPS | cv2.TERM_CRITERIA_COUNT, 10,

# Process the first frame
ret, frame = cap.read()
gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

# Initial detection
faces = detector(gray_frame)

## If face is detected, make a bounding box, else exit
if len(faces) > 0:
    face = faces[0]
    bbox_points = np.array([[face.left(), face.top()],
                            [face.right(), face.top()],
                            [face.right(), face.bottom()],
                            [face.left(), face.bottom()]], dtype=np.float32).res
else:
    print("No faces detected!")
    cap.release()
    out.release()
    exit()

old_gray = gray_frame

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

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

    # Use optical flow to track the bounding box corners
    new_points, status, _ = cv2.calcOpticalFlowPyrLK(old_gray, gray_frame, bbox_

    # Check if tracking fails
    if np.sum(status) < len(bbox_points) * 0.5:  # Re-detect if less than 50% po
        faces = detector(gray_frame)
        if len(faces) > 0:
            face = faces[0]
            bbox_points = np.array([[face.left(), face.top()],
                                    [face.right(), face.top()],
                                    [face.right(), face.bottom()],
```

```
                                        [face.left(), face.bottom()]], dtype=np.floa
    else:
        bbox_points = new_points

    # Get the minimum enclosing rectangle for the bounding box points
    x, y, w, h = cv2.boundingRect(bbox_points.astype(np.int32))

    # Draw the bounding box
    cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)

    # Write the frame to the video file
    out.write(frame)

    # Display the video
    cv2.imshow("Tracking", frame)
    old_gray = gray_frame

    # Exit on 'q'
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# Cleanup
cap.release()
out.release()
cv2.destroyAllWindows()
```

Alternative Method for Tracking

`Dlib` menyediakan metode `dlib.correlation_tracker()`, sebuah metode untuk tracking objek (bisa untuk wajah), yang mudah dan lebih ringan daripada melakukan optical flow secara manual.

In [18]:
```
import cv2
import dlib

# Initialize video capture and dlib components
VID_PATH = os.path.join(os.getcwd(), 'attachment', 'sample-renamed.mp4')

cap = cv2.VideoCapture(VID_PATH)
detector = dlib.get_frontal_face_detector()
tracker = dlib.correlation_tracker()

initialized = False  # To check if tracking has started

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

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

    if not initialized:
        # Detect the face
        faces = detector(gray)
        if len(faces) > 0:
            # Initialize the tracker with the first detected face
            tracker.start_track(frame, dlib.rectangle(
                faces[0].left(), faces[0].top(),
```

```
                faces[0].right(), faces[0].bottom()
            ))
            initialized = True
    else:
        # Update the tracker
        tracker.update(frame)
        pos = tracker.get_position()

        # Get the coordinates of the tracked region
        x1, y1, x2, y2 = int(pos.left()), int(pos.top()), int(pos.right()), int(
        cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 2)

    cv2.imshow("Tracking", frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

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

Task 6

- Put other object (png) on the face (other than eye)
- Modified the code to be more smoother (Way to detect the face)

Mediapipe dan Face Landmarking

Mediapipe adalah sebuah framework yang dikembangkan oleh Google untuk melakukan proses inferensi model *Machine Learning* .

Face Landmarking adalah sebuah solusi dari Mediapipe untuk mendeteksi dan melakukan tracking pada wajah manusia. Terdapat beberapa key point pada face landmarking

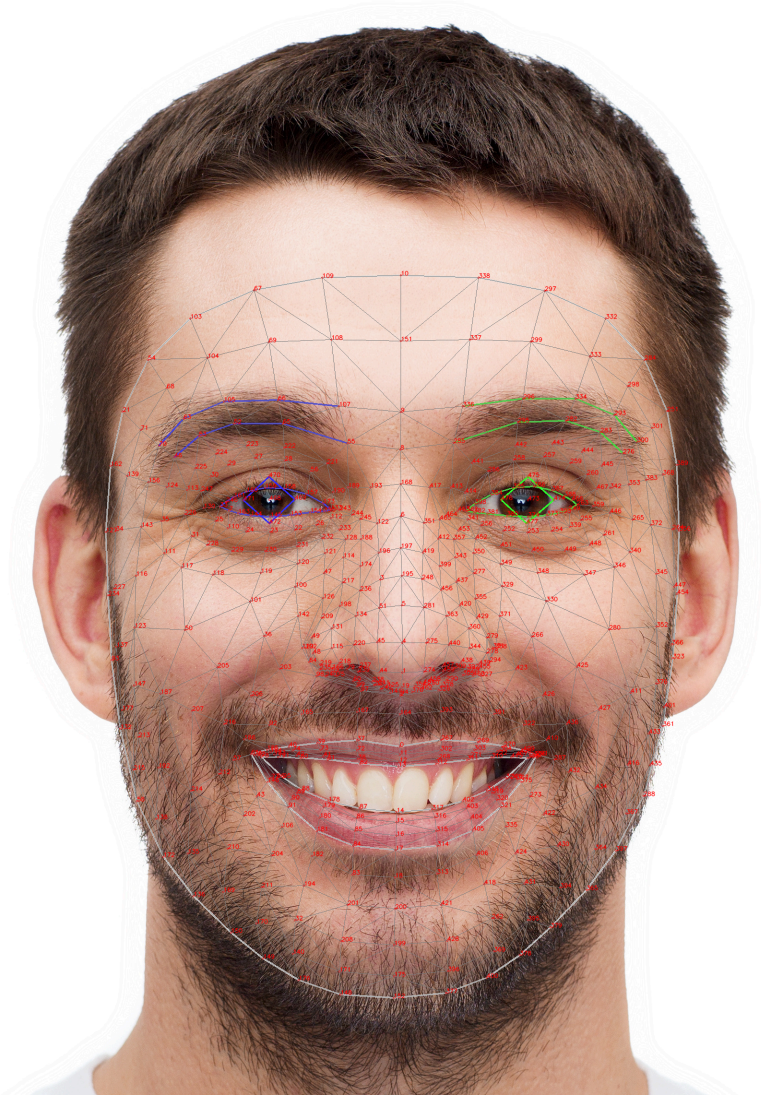
Key Capabilities:

- Real-time Face Detection: Detects faces and provides bounding boxes.
- 468 Facial Landmarks: Offers a dense map of 468 3D coordinates on the face, covering the eyes, nose, mouth, cheeks, and jawline.
- High Precision: Tracks landmarks even under varying lighting, occlusion, and head poses.
- 3D Modeling: Outputs 3D coordinates (x, y, z), where z denotes depth.

In [19]:
```
## Import Depdenencies
import cv2
import mediapipe as mp
import os
from glob import glob
import numpy as np
import matplotlib.pyplot as plt

## Setup video and image path
vid_path = os.path.join(os.getcwd(), 'attachment', 'sample-renamed.mp4')
img_glasses = os.path.join(os.getcwd(), 'attachment', 'sdg.png')
```

Detecting Landmark

Mediapipe sudah menyedikan banyak point landmark untuk muka yang ada di gambar berikut docs



Karena kita akan menggunakan gambar kacamata dan rokok, maka akan dipilih point landmark disekitaran wajah dan mulut

In [20]:
```
# Lokasi gambar kacamata
IMG_KCMT = os.path.join(os.getcwd(), 'attachment', 'sdg.png')
IMG_CIGAR = os.path.join(os.getcwd(), 'attachment', 'cigar.png')
```

```
# Load gambar kacamata
kcmt = cv2.imread(IMG_KCMT, cv2.IMREAD_UNCHANGED)
cigar = cv2.imread(IMG_CIGAR, cv2.IMREAD_UNCHANGED)

# Check alpha channel
if kcmt.shape[2] != 4:
    raise ValueError('Gambar kacamata tidak memiliki alpha channel')
if cigar.shape[2] != 4:
    raise ValueError('Gambar cerutu tidak memiliki alpha channel')
```

In [21]:
```
# # Initialize video capture
VID_PATH = os.path.join(os.getcwd(), 'attachment', 'sample-renamed.mp4')

cap = cv2.VideoCapture(VID_PATH)

# Get video dimensions
frame_width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
frame_height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

# Define the codec and create VideoWriter object
out = cv2.VideoWriter('attachment/output.avi', cv2.VideoWriter_fourcc(*'XVID'),
```

In [22]:
```
# Inisialisasi Face Mesh
mp_face_mesh = mp.solutions.face_mesh
face_mesh = mp_face_mesh.FaceMesh(
    static_image_mode=False,        # False untuk video, True untuk gambar/foto
    max_num_faces=1,                # Jumlah maksimal wajah yang dideteksi
    min_detection_confidence=0.7,   # Tingkat kepercayaan deteksi wajah
    min_tracking_confidence=0.7     # Tingkat kepercayaan pelacakan wajah
)
```

In [23]:
```
# Helper Functions for Calculating Angle for the Image png
def calculate_angle(landmark1, landmark2):
    """
    Calculate the angle between two landmarks in degrees.
    """
    delta_y = landmark2[1] - landmark1[1]
    delta_x = landmark2[0] - landmark1[0]
    angle = np.arctan2(delta_y, delta_x) * 180.0 / np.pi
    return -angle
```

In [24]:
```
## Helper function to Roate the image based on the angle of the face
def rotate_image(image, angle):
    """
    Rotate an image by a given angle around its center.
    """
    h, w = image.shape[:2]
    center = (w // 2, h // 2)
    rotation_matrix = cv2.getRotationMatrix2D(center, angle, 1.0)
    rotated_image = cv2.warpAffine(
        image, rotation_matrix, (w, h),
        flags=cv2.INTER_LINEAR, borderMode=cv2.BORDER_CONSTANT, borderValue=(0,
    )
    return rotated_image
```

In [25]:
```
## Core function to overlay the image on the face
def overlay_image(background, overlay, x, y):
    """
```

```
    Overlay an RGBA image onto a background image at position (x, y)
    """
    h, w = overlay.shape[:2]

    if x >= background.shape[1] or y >= background.shape[0]:
        return background

    if x + w > background.shape[1]:
        w = background.shape[1] - x
    if y + h > background.shape[0]:
        h = background.shape[0] - y

    if w <= 0 or h <= 0:
        return background

    overlay_colors = overlay[:h, :w, :3]
    alpha = overlay[:h, :w, 3] / 255.0
    alpha = np.dstack((alpha, alpha, alpha))

    background_region = background[y:y+h, x:x+w]
    composite = background_region * (1 - alpha) + overlay_colors * alpha

    result = background.copy()
    result[y:y+h, x:x+w] = composite
    return result
```

In [26]:

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

    frame = cv2.flip(frame, 1)
    rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

    results = face_mesh.process(rgb_frame)

    tinggi, lebar, _ = frame.shape

    if results.multi_face_landmarks:
        for flm in results.multi_face_landmarks:
            ## Eye Landmark
            left_outer = flm.landmark[33]
            left_inner = flm.landmark[133]
            right_outer = flm.landmark[362]
            right_inner = flm.landmark[263]

            ## Mouth Landmark
            mouth_left = flm.landmark[61]
            mouth_right = flm.landmark[291]
            mouth_top = flm.landmark[13]
            mouth_bottom = flm.landmark[14]

            # Convert landmarks to pixel coordinates
            left_outer = (int(left_outer.x * lebar), int(left_outer.y * tinggi))
            left_inner = (int(left_inner.x * lebar), int(left_inner.y * tinggi))
            right_outer = (int(right_outer.x * lebar), int(right_outer.y * tingg
            right_inner = (int(right_inner.x * lebar), int(right_inner.y * tingg

            # Convert mouth landmarks to pixel coordinates
            mouth_left = (int(mouth_left.x * lebar), int(mouth_left.y * tinggi))
```

```
            mouth_right = (int(mouth_right.x * lebar), int(mouth_right.y * tingg
            mouth_top = (int(mouth_top.x * lebar), int(mouth_top.y * tinggi))
            mouth_bottom = (int(mouth_bottom.x * lebar), int(mouth_bottom.y * ti

            # Calculate rotation angle using left eye corner points
            eye_angle = calculate_angle(left_outer, left_inner)

            # Calculate Rotation Angle for Cigar based on the left and right
            cigar_angle = calculate_angle(mouth_left, mouth_right)

            # Calculate glasses dimensions
            eye_distance = abs(right_outer[0] - left_outer[0])
            glasses_width = int(eye_distance * 1.8)
            aspect_ratio = kcmt.shape[0] / kcmt.shape[1]
            glasses_height = int(glasses_width * aspect_ratio)

            ## Calculate cigar dimension
            mouth_distance = abs(mouth_right[0] - mouth_left[0])
            cigar_width = int(mouth_distance * 1.8)
            aspect_ratio_cigar = cigar.shape[0] / cigar.shape[1]
            cigar_height = int(cigar_width * aspect_ratio_cigar)

            ## Resize and Rotate the Cigar
            cigar_resized = cv2.resize(cigar, (cigar_width, cigar_height))
            cigar_rotated = rotate_image(cigar_resized, cigar_angle)

            ## Scale the glasses
            glasses_width = int(glasses_width * 1.8)
            glasses_height = int(glasses_height * 1.8)

            # Resize and rotate glasses
            kcmt_resized = cv2.resize(kcmt, (glasses_width, glasses_height))
            kcmt_rotated = rotate_image(kcmt_resized, eye_angle)

            # Calculate glasses position
            center_x = (left_outer[0] + right_outer[0]) // 2
            center_y = (left_outer[1] + right_outer[1]) // 2
            x = center_x - kcmt_rotated.shape[1] // 2 + int(glasses_width * 0.05
            y = center_y - kcmt_rotated.shape[0] // 2

            # Calculate cigar position
            center_x_cigar = (mouth_left[0] + mouth_right[0]) // 2
            center_y_cigar = (mouth_bottom[1] + mouth_top[1]) // 2
            x_cigar = center_x_cigar - cigar_rotated.shape[1] // 2 - int(cigar_w
            y_cigar = center_y_cigar - cigar_rotated.shape[0] // 2 + int(cigar_h

            # Overlay rotated cigar on the frame
            frame = overlay_image(frame, cigar_rotated, x_cigar, y_cigar)

            # Overlay rotated glasses on the frame
            frame = overlay_image(frame, kcmt_rotated, x, y)

    cv2.imshow('Facial Landmark Detection', frame)

    ## Write the frame to the output video

    out.write(frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
```

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

```
c:\Users\ACER\miniconda3\envs\py310_multimedia\lib\site-packages\google\protobuf
\symbol_database.py:55: UserWarning: SymbolDatabase.GetPrototype() is deprecated.
Please use message_factory.GetMessageClass() instead. SymbolDatabase.GetPrototype
() will be removed soon.
  warnings.warn('SymbolDatabase.GetPrototype() is deprecated. Please '
```

In [27]:
```
## Show some random frame of the output video
videoCapture = cv2.VideoCapture('attachment/output.avi')
video_frames = []
while True:
    ret, frame = videoCapture.read()
    if not ret:
        break
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    video_frames.append(frame)

frames_array = np.array(video_frames)
videoCapture.release()

## Show the example of images
single_img = frames_array[200].copy()
plt.figure(figsize=(5, 5))
plt.imshow(single_img)
plt.axis('off')
plt.show()
```

Optical Flow / Face Tracking with Mediapipe

Karena kode dari mediapipe itu sendiri akan melakukan tracking / deteksi wajah untuk setiap frame video. Kita bisa melakukan optimasi parameter ketika melakukan instantiasi Face Mesh berdasarkan dokumentasi mediapipe.

```
import mediapipe as mp

mp_face_mesh = mp.solutions.face_mesh
face_mesh = mp_face_mesh.FaceMesh(
    static_image_mode=False,
```

```
    max_num_faces=1,
    refine_landmarks=True,
    min_detection_confidence=0.5,
    min_tracking_confidence=0.5
)
```

- static_image_mode (bool):
  - True: Process images (no temporal smoothing).
  - False: Optimized for video streams with temporal smoothing.
- max_num_faces (int): Maximum number of faces to detect.
- refine_landmarks (bool): Enables refinement for specific landmarks like eyes and lips.
- min_detection_confidence (float): Minimum confidence value [0.0 - 1.0] for detecting a face.
- min_tracking_confidence (float): Minimum confidence value [0.0 - 1.0] for tracking landmarks.

Berdasarkan dokumentasi ini, kita dapat meningkatkan / menurunkan kualitas dari tracking / detecing dengan melakukan set pada nilai `min_detection_confidence` ataupun `min_tracking_confidence`

Di lain sisi, penggunaan tracking seperti optical flow atau re-deteksi setiap beberap frame, kurang menurut saya karena objek png kacamata dan rokok membutuhkan landmark point, ketika terjadi perubahan gerakan kecil pada wajah, maka gambar tersebut juga akan bergerak, dan akan terakumulasi menjadi semakin menjauh dari wajah ketika menggunakan optical flow

Referensi

- 1. Dlib in Python
- 2. Lucas Canade Optical Flow
- 3. Optical Flow
- 4. Mediapipe Overview