Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Звіт

Лабораторна робота № 2 з дисципліни
«Штучний інтелект в задачах обробки зображень»

«Виявлення об’єктів засобами OpenCV»

Виконав: *ІП-01 Черпак А. В.* _____
(шифр, прізвище, ім'я, по батькові)

Перевірив: *Нікітін В. А.* _____
(прізвище, ім'я, по батькові)

Київ 2022

Мета:

Навчитись виявляти обличчя та пішоходів в режимі реального часу за допомогою OpenCV

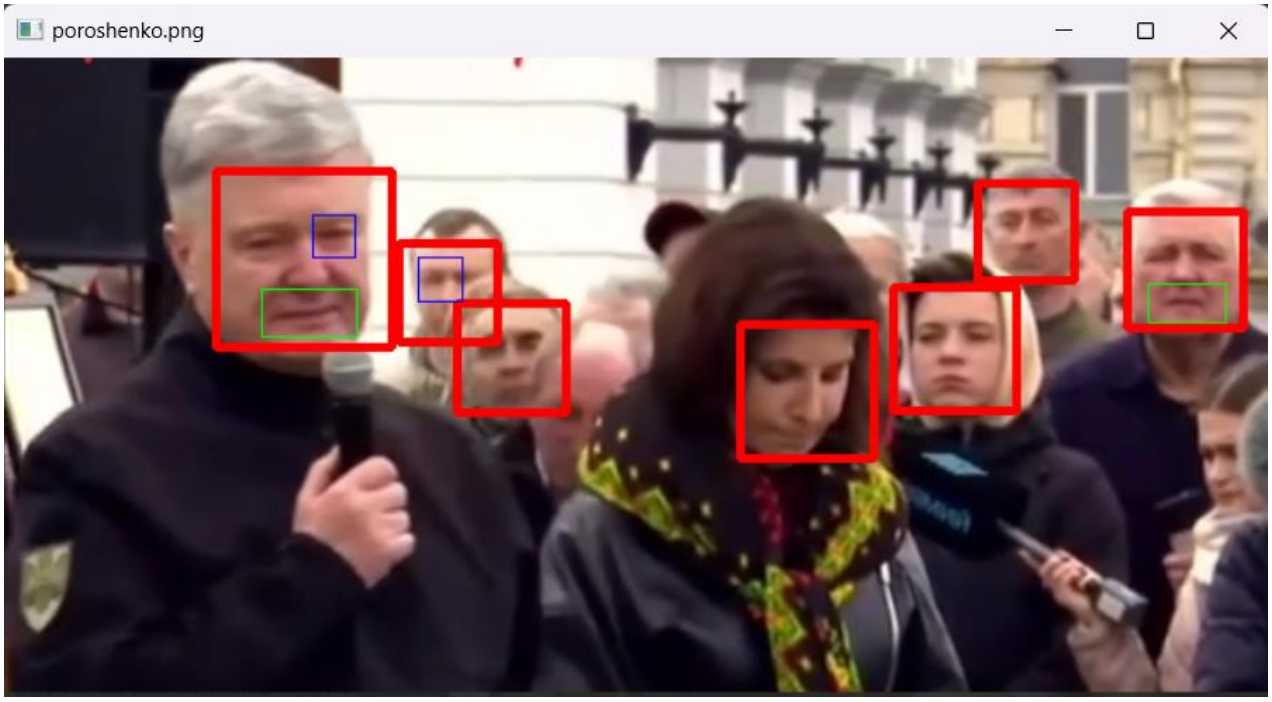
Завдання

1. Використовуючи будь-яку фотографію з декількома людьми, виявити на ньому обличчя, очі, усмішку. Порахувати кількість осіб на фото;
2. Зробити розпізнавання використовуючи будь-яке відео з обличчям людини, тривалістю не менше 30 секунд. Можно використати камеру ноутбука;
3. Обробити відеофал, так щоб він виділяв пішоходів і, по можливості, їхні обличчя. Файл можна взяти з youtube і вирізати ролик тривалістю не менше 30 секунд.

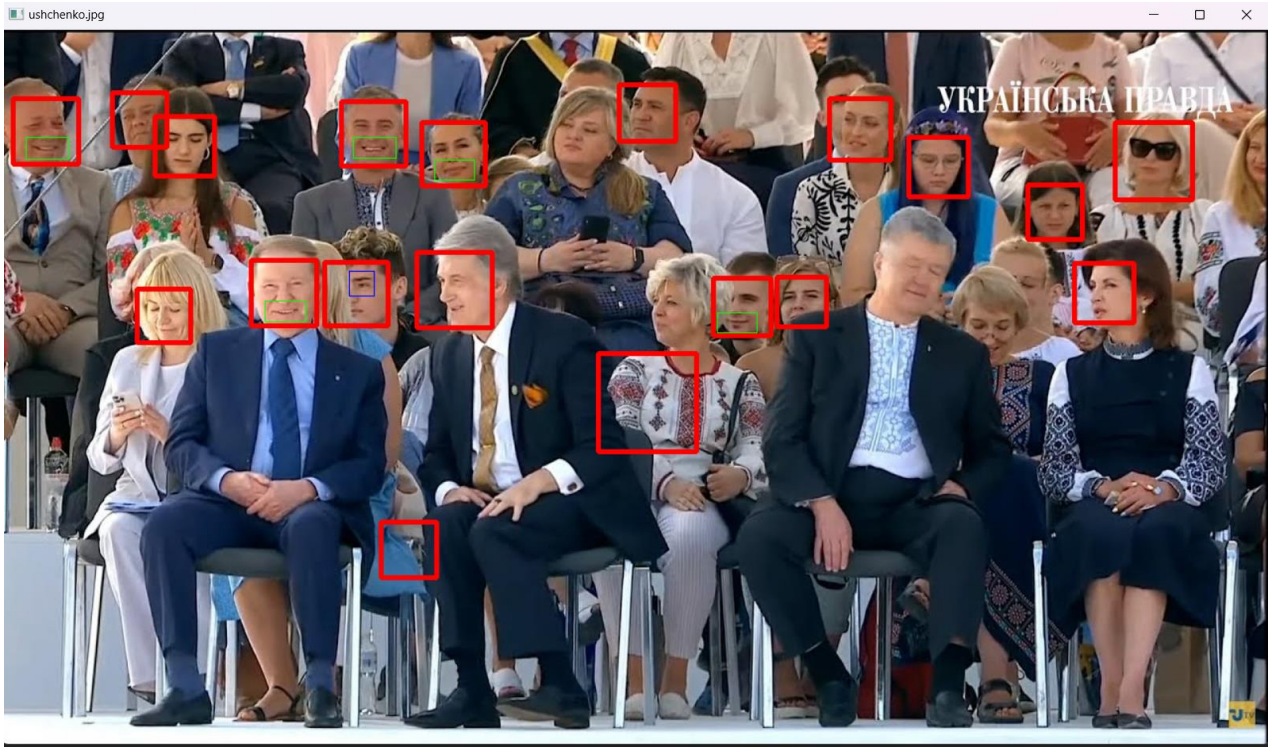
Хід роботи

1. Порахувати кількість людей на фото, знайти їх обличчя, усмішки та очі:

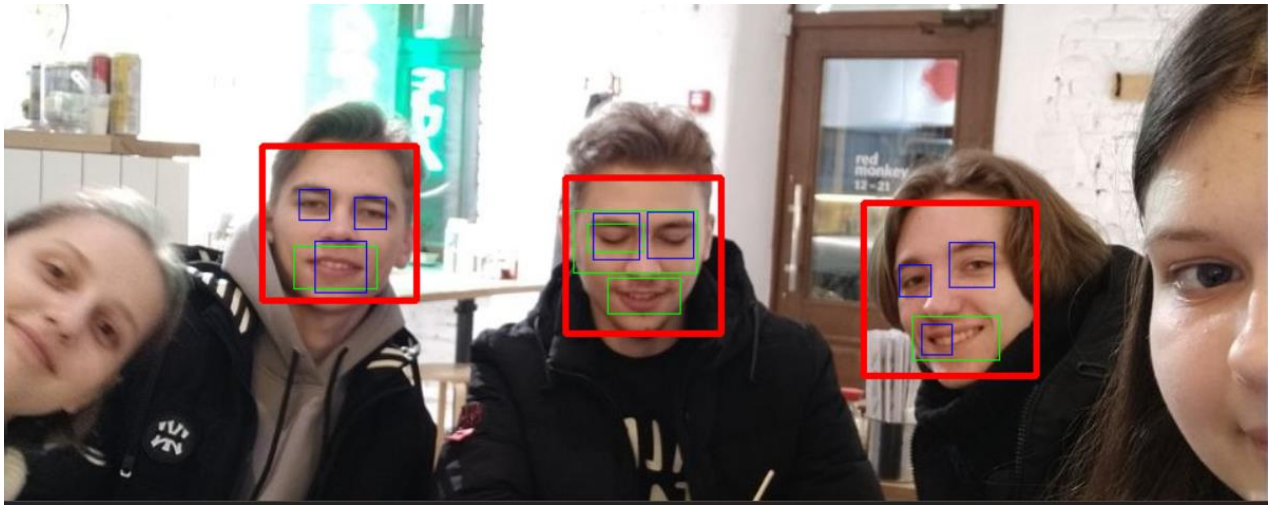
```
import cv2


if __name__ == '__main__':
    face_classifier = cv2.CascadeClassifier(cv2.data.haarcascades +
"haarcascade_frontalface_default.xml")
    smile_classifier = cv2.CascadeClassifier(cv2.data.haarcascades +
"haarcascade_smile.xml")
    eye_classifier = cv2.CascadeClassifier(cv2.data.haarcascades +
"haarcascade_eye.xml")

    for photo in ("poroshenko.png", "ushchenko.jpg", "homies.jpg"):
        frame = cv2.imread(photo)
        gray_filter = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        face_rects = face_classifier.detectMultiScale(gray_filter,
scaleFactor=1.1, minNeighbors=5)

        for (x, y, w, h) in face_rects:
            cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 3)
            roi_gray = gray_filter[y:y+h, x:x+w]
            roi_color = frame [y:y+h, x:x+w]
            smile = smile_classifier.detectMultiScale(roi_gray,
scaleFactor=1.1, minNeighbors=10)
            eye = eye_classifier.detectMultiScale(roi_gray, scaleFactor=1.15,
minNeighbors=5)
            for (sx, sy, sw, sh) in smile:
                cv2.rectangle(roi_color, (sx, sy), (sx + sw, sy + sh), (0,
255, 0), 1)
            for (ex, ey, ew, eh) in eye:
                cv2.rectangle(roi_color, (ex, ey), (ex + ew, ey + eh), (255,
0, 0), 1)
        print(f"Found {len(face_rects)} faces!")
        cv2.imshow(photo, frame)
        cv2.waitKey()
        cv2.destroyAllWindows()
```




```
Found 7 faces!
```



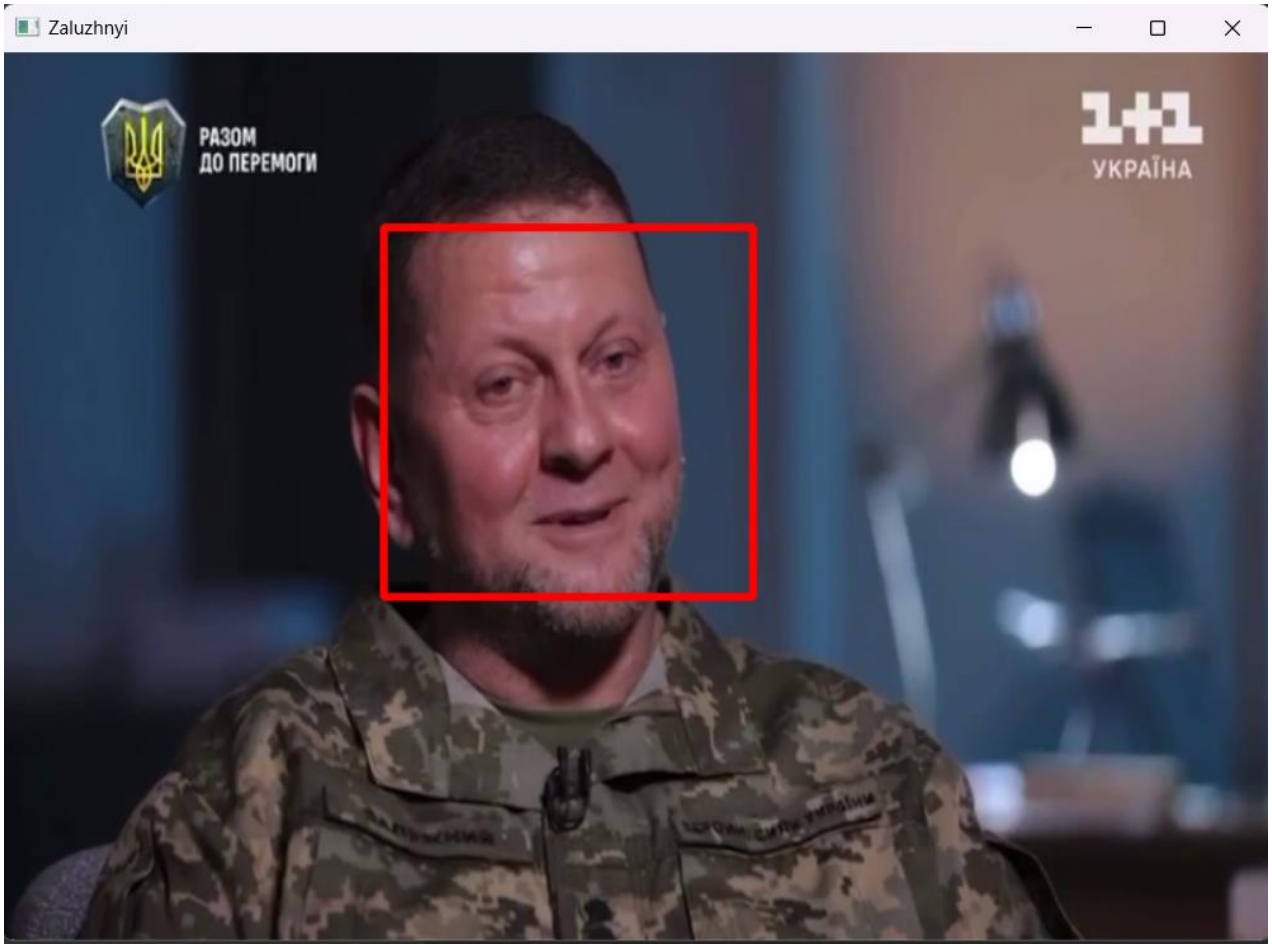

```
Found 19 faces!
```

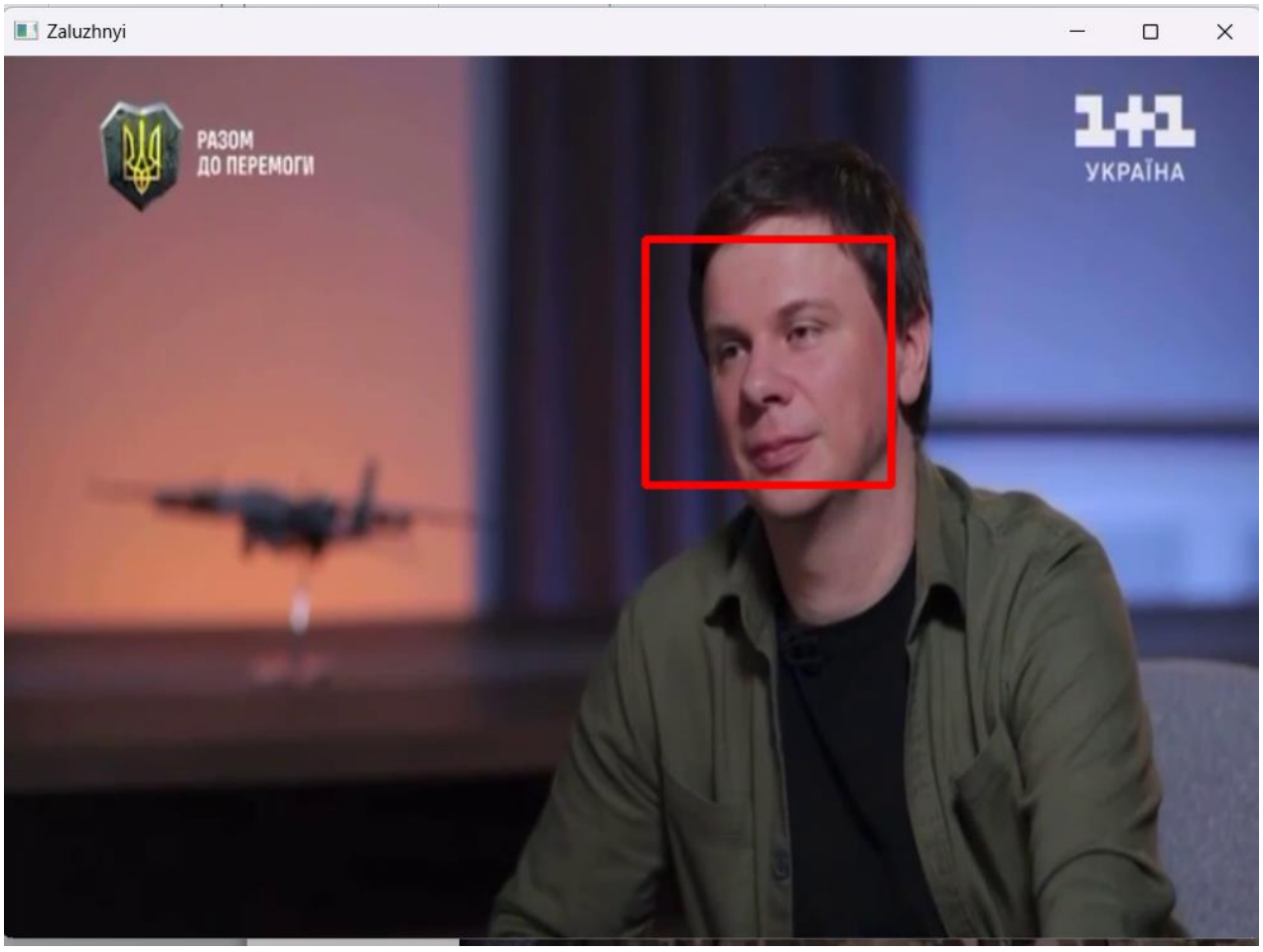


```
Found 3 faces!
```

2. Розпізнати обличчя людини на відео:

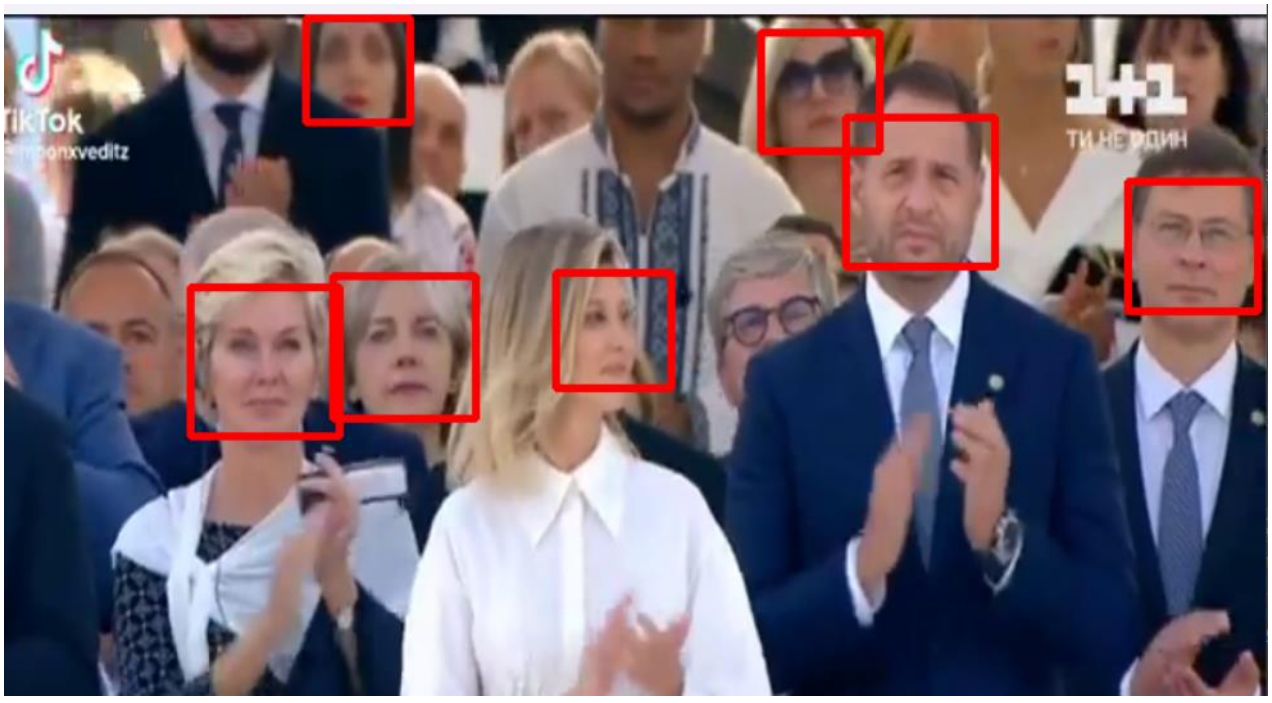
```
import cv2

if __name__ == '__main__':
    face_classifier = cv2.CascadeClassifier(cv2.data.haarcascades +
"haarcascade_frontalface_default.xml")
    cv2.startWindowThread()
    cap = cv2.VideoCapture("zaluzhnyi.MP4")
    while True:
        ret, frame = cap.read()
        if not ret or cv2.waitKey(1) & 0xFF == ord("q"):
            break
        frame = cv2.resize(frame, (800, 560))
        gray_filter = cv2.cvtColor(frame, cv2.COLOR_RGB2GRAY)
        face_rects = face_classifier.detectMultiScale(gray_filter,
scaleFactor=1.3, minNeighbors=5)
        for (x, y, w, h) in face_rects:
            cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 3)
        cv2.imshow("Zaluzhnyi", frame)
    cap.release()
    cv2.destroyAllWindows()
```

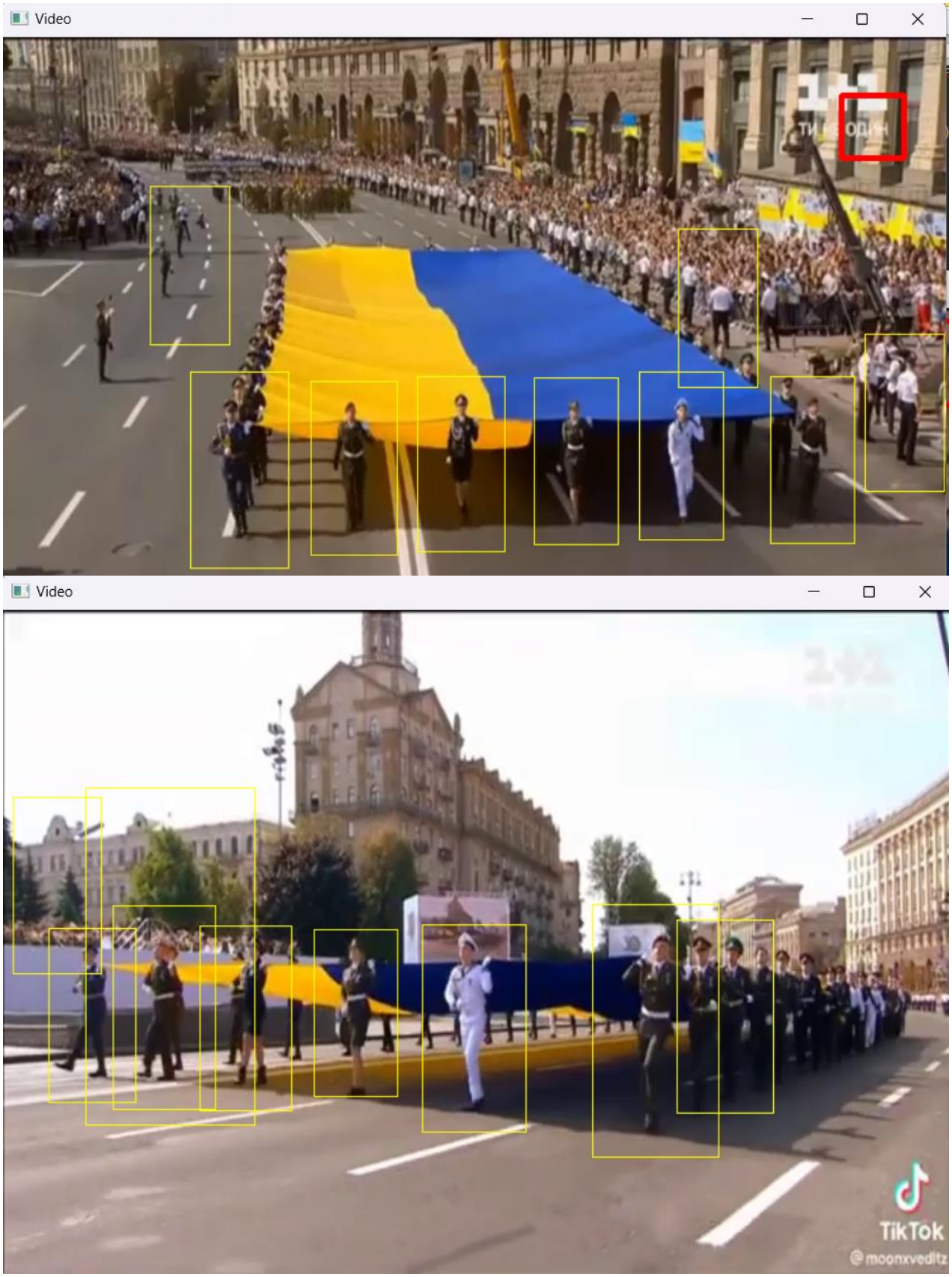


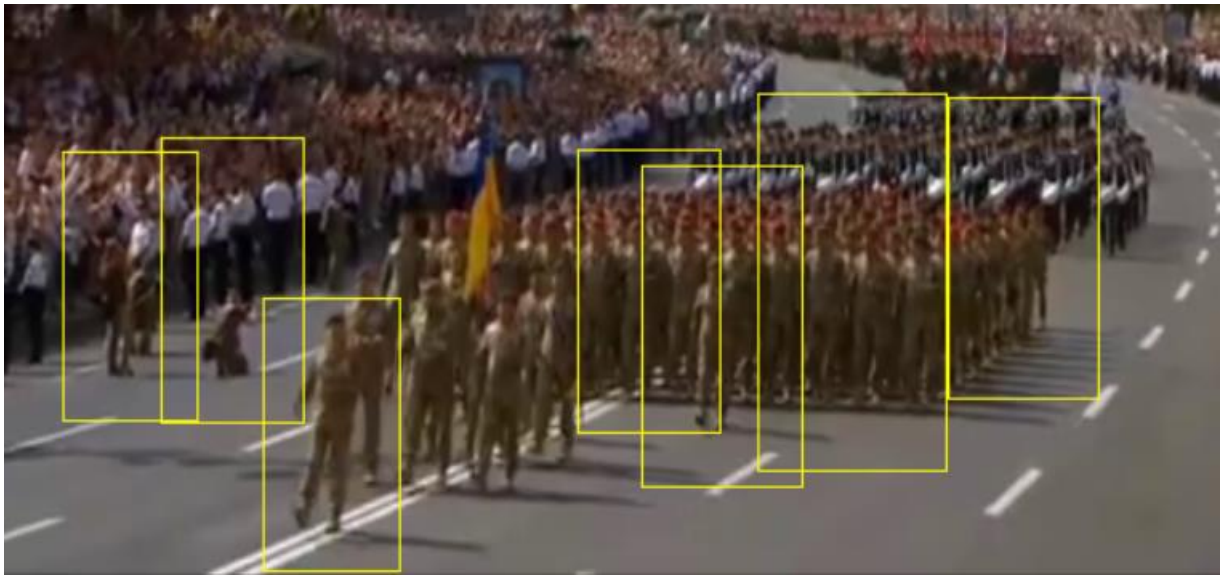


3. Виділити на відео пішоходів та їх обличчя:

```python
import cv2
import numpy as np
from get_video import download

if __name__ == '__main__':
    # download("https://www.youtube.com/watch?v=_mPelM7Yaq8")
    # download("https://www.youtube.com/watch?v=UWiinhrRcKA")
    face_classifier = cv2.CascadeClassifier(cv2.data.haarcascades +
"haarcascade_frontalface_default.xml")
    hog = cv2.HOGDescriptor()
    hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())
    cv2.startWindowThread()
    cap = cv2.VideoCapture("parad.mp4")
    while True:
        ret, frame = cap.read()
        if not ret or cv2.waitKey(1) & 0XFF == ord("q"):
            break
        frame = cv2.resize(frame, (800, 560))
        gray_filter = cv2.cvtColor(frame, cv2.COLOR_RGB2GRAY)
        face_rects = face_classifier.detectMultiScale(gray_filter,
scaleFactor=1.1, minNeighbors=5)
        boxes, weights = hog.detectMultiScale(frame, winStride=(8, 8))
        boxes = np.array([[x, y, x + w, y + h] for (x, y, w, h) in boxes])
        for (xa, ya, xb, yb) in boxes:
            cv2.rectangle(frame, (xa, ya), (xb, yb), (0, 255, 255), 1)
        for (x, y, w, h) in face_rects:
            cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 3)
        cv2.imshow("Video", frame)

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




Video
Video
TikTok
@moonxveditz




1+1
ТИ НЕ ОДИН

Контрольні запитання

1. Алгоритм Віоли-Джонсона – це алгоритм, що дозволяє розпізнавати об’єкти на зображеннях у режимі реального часу. Його створювали з метою розпізнавання облич, хоча він може розпізнавати будь-які інші об’єкти.
2. haarcascade - це метод виявлення об'єктів, що використовується для визначення розташування об'єктів на зображеннях чи відео. Цей алгоритм використовує набір функцій Хаара, які є простими прямокутними шаблонами чорних і білих пікселів.
3. HOG-дескриптор - це набір числових значень, які описують зміни яскравості та структуру переходів градієнту на зображенні. Шляхом порівняння дескрипторів подібних зображень нейронна мережа або лінійний класифікатор можуть визначити їх схожість. За допомогою дескриптора HOG можна знайти конкретний об'єкт на зображенні або перевірити, чи належить зображення певній категорії.
4. SVM-детектор - це алгоритм машинного навчання на основі методу опорних векторів (SVM), який використовується для виявлення об'єктів на зображеннях або відео. SVM-детектор використовує навчальну вибірку, щоб визначити границі рішень та розділити позитивні та негативні зразки.
5. “cv2.cvtColor()” метод використовується для перетворення зображення з одного колірного простору на інший. Ми використовуємо cvtColor, щоб змінити колірний простір на сірий. Сірі відтінки краще підходять для тих алгоритмів, які ми використовуємо в нашій лабораторній, оскільки нас більше цікавлять характеристики форм и зображення, а колірні канали можуть не давати багато інформації про форму.