Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота № 2

з дисципліни
«Штучний інтелект в задачах обробки зображень»

Виконав:

студент групи ІМ-11
Царик Микола Миколайович
варіант відповідно до списку: 22

Перевірив:

Нікітін Валерій Андрійович

Київ 2023

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 виконання:

Перше завдання:

1. **Імпорт бібліотек**:

```
import cv2
```

2. **Ініціалізація класифікаторів обличь, посмішок та очей**:

```
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +
'haarcascade_frontalface_default.xml')
smile_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +
'haarcascade_smile.xml')
eye_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +
'haarcascade_eye.xml')
```

3. **Налаштування масштабування та завантаження зображення**:

```
scaling_factor = 0.56
frame = cv2.imread("Lab2\data\image.jpg")
frame = cv2.resize(frame, None, fx=scaling_factor, fy=scaling_factor,
interpolation=cv2.INTER_AREA)
```

4. **Перетворення зображення в відтінки сірого**:

```
gray_filter = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

5. **Виявлення облич та їх обробка**:

```
face_rects = face_cascade.detectMultiScale(gray_filter, 1.1, 5)
for (x,y,w,h) in face_rects:
    cv2.rectangle(frame, (x,y), (x+w,y+h),(255,0,0),2)
    roi_gray = gray_filter[y:y+h,x:x+w]
    roi_color= frame[y:y+h,x:x+w]
    smile= smile_cascade.detectMultiScale(roi_gray)
    eye= eye_cascade.detectMultiScale(roi_gray)
```

6. **Обробка знайдених посмішок та очей**:

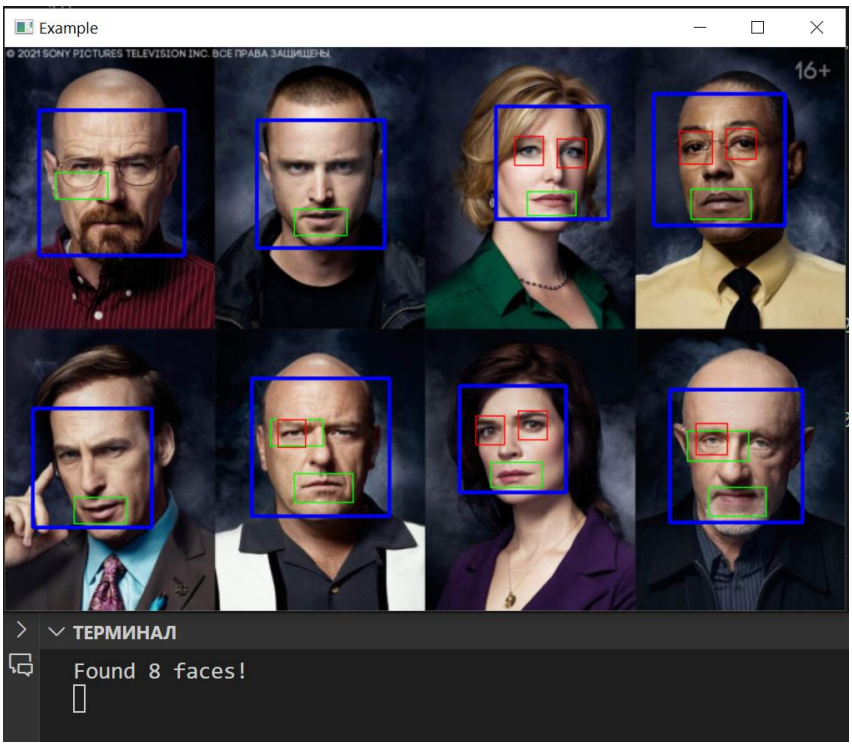
```
for(sx,sy,sw,sh) in smile:
        cv2.rectangle(roi_color,(sx,sy),(sx+sw,sy+sh),(0,255,0),1)

    for(ex,ey,ew,eh) in eye:
        cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,0,255),1)
```

7. **Відображення результатів**:

```
print("Found", len(face_rects), "faces!")
cv2.imshow('Example', frame)
cv2.waitKey(0)
```

Результат



Друге завдання:

Алгоритм пошуку обличь , посмішок та очей залишився той самий, але тепер додалась обробка відео.

1) Підключемо відео

```
cap = cv2.VideoCapture("Lab2\data\Video.mp4")
```

2) Тепер створимо нескінченний цикл (щоб швидко і послідовно обробляти кожен кадр), в ньому зчитуємо поточний кадр, змешуємо його для покращення плавності відео та робимо його чорно білим для полегшення пошуку об’єктів

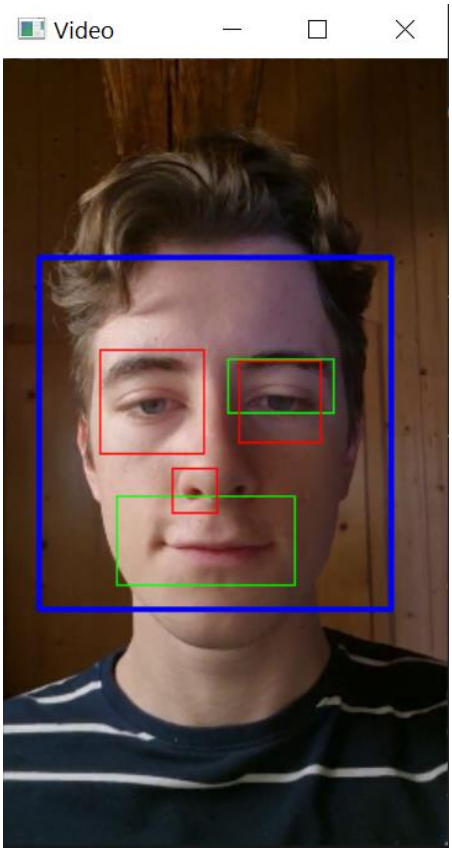
```
while True:
    ret, frame = cap.read()

    frame = cv2.resize(frame, (240, 425))
    gray_filter = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
...
```

3) Також додамо можливість зупинення відео при натисканні “q”

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

Результат



Третє завдання:

Знову працюємо з відео, але тепер окрім обличчя нам треба роспізнавати щей людей вцілому. Для цього ми використаємо HOG-класифікатор (Histogram of Oriented Gradients) разом із SVM-детектором (Support Vector Machine)

```
hog = cv2.HOGDescriptor()
hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())
```

Далі в циклі відокремлюємо всіх людей) Подібно до того як ми це робили з обличчями

```
boxes, weights = hog.detectMultiScale(frame, winStride=(8, 8))
boxes = numpy.array([[x, y, x + w, y + h] for (x,y,w,h) in boxes])
```

Після цього промальовуємо прямокутник навколо кожної особи.

```
for (xa,ya,xb,yb) in boxes:
     cv2.rectangle(frame,(xa,ya),(xb,yb),(0,255 ,255),1)
```

Результат

Висновок

В ході виконання лабораторної , окрім навичок роботи з бібліотекою cv2 для редагування зображень я зробив кілька висновків

1) OpenCV хоча і надає непогані інструменти для пошуку об'єктів на фото, вони все ж таки не є ідеальними. І хоча в зображеннях необхідні елементи знаходяться доволі точно, але в відео не рідко підкреслюються зайві об'єкти
2) Доволі просто можна одночасно шукати багато різних об'єктів (наприклад моніторити людей і в той же час підкреслювати їх обличчя)
3) Для меншої нагрузки компютера та плавнішої картинки, відео та зображення необхідно перед обробкою зменшити за допомогою resize()

Контрольні питання

1. **Що таке алгоритм Віоли-Джонсона?**
   Це алгоритм машинного навчання, використовуваний для виявлення об'єктів у зображеннях або відео в режимі реального часу. Спочатку створювався для пошуку обличь, але потім спектр шуканих елемнтів розширився.
2. **Що таке haarcascade?**
   Haar cascade - це XML-файл, який містить інформацію про навчену модель, яка визначає признаки, що розпізнають об'єкт, і використовується алгоритмом Віоли-Джонсона.
3. **Що таке HOG-класифікатор?**
   Це метод в комп'ютерному зорі для виявлення об'єктів на зображеннях. HOG-класифікатор використовує гістограми напрямів градієнтів пікселів для створення опису об'єктів на зображенні.
4. **Що таке SVM-детектор?**
   Це метод машинного навчання для класифікації або регресії даних. В контексті виявлення об'єктів на зображеннях SVM може бути використаний як детектор для виявлення об'єктів на основі описів, отриманих від HOG або інших алгоритмів. Основним завданням алгоритму є знайти найбільш правильну лінію, або гіперплощину, що розділяє дані на два класи. SVM це алгоритм, який отримує на вході дані, і повертає таку лінію, що розділяє.
5. **Що робить метод cvtColor та яка його мета використання у цій лабораторній?**
   Метод cvtColor в бібліотеці OpenCV призначений для конвертації зображення з одного кольорового простору в інший. У цій лабораторній він використовується для конвертації кольорового зображення frame з формату RGB в відтінки сірого (градації сірого), оскільки HOG-детектор працює з одноканальними зображеннями.