**Міністерство освіти і науки України**

**Національний технічний університет України «КПІ» імені Ігоря Сікорського**

**Кафедра обчислювальної техніки ФІОТ**

**ЗВІТ**

**з модульної контрольної роботи**

**з навчальної дисципліни «Computer Vision»**

**Виконав:**

Студент 3 курсу кафедри ІПІ ФІОТ,

Навчальної групи ІП-11

Трикош І. В.

**Перевірив:**

Професор кафедри ОТ ФІОТ

Писарчук О.О.

**Київ 2024**

## **І. Білет №10:**

## **ІІ. Завдання:**

1. Фрактальні зображення.
2. Дескриптор зображення.
3. Здійснити розробку програмного скрипта з виявлення об’єктів за алгоритмами каскадів Хаара.

## **ІІІ. Результати виконання модульної контрольної роботи.**

* 1. **Відповідь на теоретичне питання №1.**

Фрактали – це складні геометричні форми, які можна розділити на частини, кожна з яких є зменшеною копією цілого. Вони характеризуються самоподібністю (якщо збільшити якусь частину фрактала, то вона виглядатиме так само, як і сам фрактал) та нескінченною складністю.

Види фракталів: геометричні (трикутник Серпінського, крива Коха), алгебраїчні (множина Мандельброта), стохастичні.

Фрактальні зображення – це зображення, що створені на основі математичних фракталів та імітують оригінальні рішення живої та неживої природи – повторення та тиражування вдалих комбінацій в структурі об’єкту.

Переваги фрактальних зображень:

* Самоподібність і нескінченна деталізація
* Ефективність зберігання даних
* Масштабованість
* Генеративний дизайн

Недоліки фрактальних зображень:

* Обчислювальна складність
* Складність параметризації
* Відсутність універсальності

Застосування фрактальних зображень:

* Комп’ютерна графіка та анімація – створення реалістичних текстур, ефектів візуалізації та анімації
* Моделювання природних об’єктів – гір, річок, хмар тощо
* Фрактальне стиснення – використовується для стиснення графічних даних для забезпечення високої якості зображень при зменшенні розміру файлу
  1. **Відповідь на теоретичне питання №2.**

Дескриптор – набір числових характеристик або ознак, які описують особливу точку (точка зображення, околицю якої можна відрізнити від околиці будь-якої іншої точки зображення в деякій іншій околиці іншої особливої точки) або фрагмент зображення так, щоб вони могли бути ідентифіковані й розпізнані серед інших точок або фрагментів. Вони використовуються для задач виявлення і зіставлення особливих точок, відстеження руху, ідентифікація об’єктів тощо.

Основні аспекти дескриптора:

1. Виділення особливих точок (з допомогою алгоритмів SIFT, SURF, ORB)
2. Опис особливих точок (виявлення їх локальних ознак та їх представлення у відповідному форматі)
3. Масштабна та орієнтаційна інваріантність (адаптація дескриптора до різних розмірів та орієнтацій зображення)
4. Розрахунок градієнтів (зміни інтенсивності пікселів в області навколо особливої точки) і побудова по агрегованих значеннях вектора, який буде представляти дескриптор
5. Стійкість до освітлення (зменшення впливу змін освітлення)
6. Зіставлення дескрипторів (знаходження найближчих сусідів серед дескрипторів для ідентифікації відповідних точок на різних зображення)
   1. **Відповідь на практичне питання №3.**
      1. **Синтезована математична модель**

Відповідно до умов задачі синтезовано математичну модель, що реалізує виявлення облич людей на цифровому зображенні за алгоритмом каскадів Хаара.

Каскади Хаара – метод виявлення об’єктів на зображеннях, що базується на використанні Хаар-подібних особливостей (це прості прямокутники, що використовуються для виявлення контрасту між різними частинами зображеннями). Алгоритм використовує інтегральні зображення, що дозволяє швидко обчислити ознаки та суму прямокутних областей за формулою:

де – точки інтегрального зображення, – інтегральне зображення.

На кінцевому етапі алгоритм використовує каскадний класифікатор, у якому кожен класифікатор у каскаді відповідає за певну підмножину ознак. Якщо частина зображення проходить через усі рівні каскаду, то вона розпізнається як об’єкт.

Синтезована модель використовуватиме уже готові каскади для ідентифікації облич:

<https://github.com/opencv/opencv/blob/master/data/haarcascades/haarcascade_frontalface_default.xml>

* + 1. **Результати архітектурного проектування та їх опис**

На рис. 1 зображена блок-схема до моделі до програмного скрипту:

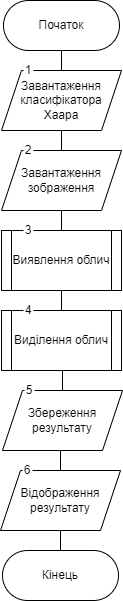


Рисунок 1 – Блок схема алгоритму

Робота алгоритму розпочинається із завантаження класифікатора каскадів Хаара з допомогою готових каскадів для ідентифікації облич. Ці дії виконуються у блоці 1 блок-схеми алгоритму рис.1.

У блоці 2 виконується завантаження зображення з файлу і перетворення його у відтінки сірого.

У блоці 3 виконується виявлення облич з допомогою класифікатора.

У блоці 4 виконується виділення виявлених облич з допомогою прямокутників.

У блоці 5 відбувається збереження результатів ідентифікації об’єктів у файл.

У блоці 6 виконується відображення отриманого результату. На цьому виконання алгоритму завершене.

* + 1. **Опис структури проекту програми**

Для реалізації розробленого алгоритму мовою програмування Python з використанням можливостей інтегрованого середовища PyCharm сформовано проєкт.

Проєкт базується на лінійній бізнес-логіці функціонального програмування та має таку структуру.

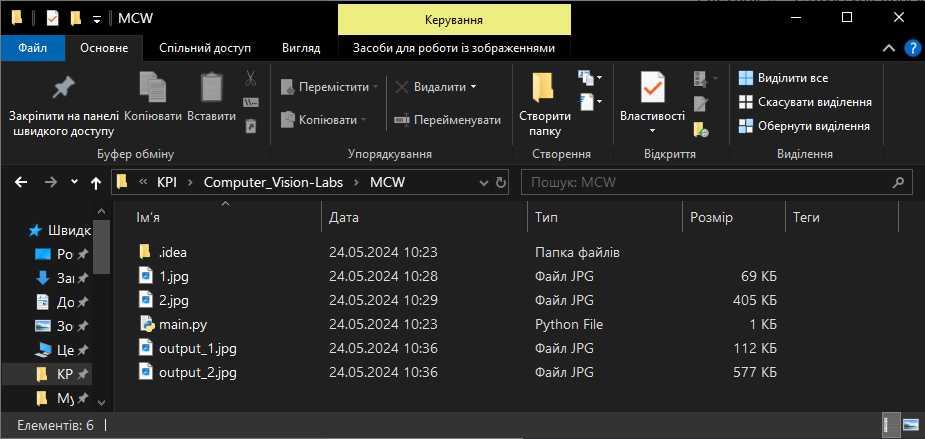


Рисунок 2 – Структура проєкту

MCW – головний каталог проєкту

main.py – файл програмного коду

1.jpg, 2.jpg – файли для перевірки роботи алгоритму

output\_1.jpg, output\_2.jpg – результати роботи алгоритму (ідентифіковані обличчя) для файлів 1.jpg та 2.jpg відповідно

* + 1. **Результати роботи програми відповідно до завдання (допускається у формі скриншотів)**

Результатом роботи програми є сукупність послідовності графічних вікон, що реалізують умови завдання лабораторної роботи.

1. Зображення 1.jpg:

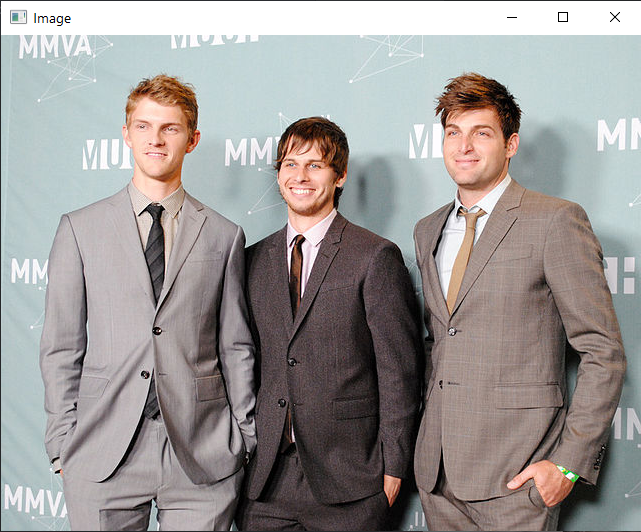


Рисунок 3 – Початкове зображення 1.jpg

1. Результат виявлення облич для 1.jpg:

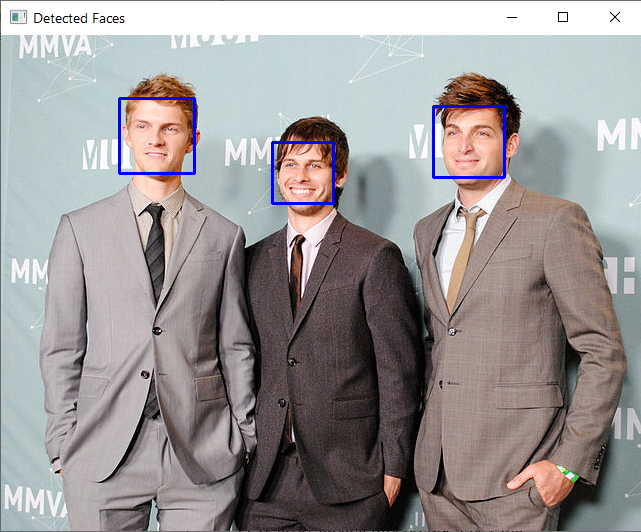


Рисунок 4 – Результат виявлення облич для першого зображення

1. Зображення 2.jpg:



Рисунок 5 – Початкове зображення 2.jpg

1. Результат виявлення облич для 2.jpg:



Рисунок 6 – Результат виявлення облич для другого зображення

Бачимо, що для першого зображення алгоритм відпрацював чудово, ідентифікував усі три обличчя. Для другого зображення алгоритм відпрацював непогано, ідентифікував 7 із 8 облич. Загалом, бачимо, що алгоритм добре виконує поставлену задачу.

* + 1. **Програмний код, що забезпечує отримання результату (допускається у формі скриншотів)**

Програмний код послідовно реалізує алгоритм на рис.1 та спрямований на отримання результатів, поданих на рис.3-6.

Для спрощення програмного коду і раціоналізації обчислень застосовано функціональні механізми створення підпрограм.

При цьому використано можливості Python бібліотек: OpenCV.

Контекстні коментарі пояснюють сутність окремих скриптів наведеного коду програми.

main.py

*"""  
Ідентифікація облич з допомогою алгоритму каскадів Хаара  
"""  
  
import* cv2  
  
*# Завантаження класифікатора каскадів Хаара для облич*face\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade\_frontalface\_default.xml')  
  
*# Завантаження зображення*image\_path = '2.jpg'  
image = cv2.imread(image\_path)  
result\_image = image.copy()  
gray\_image = cv2.cvtColor(result\_image, cv2.COLOR\_BGR2GRAY)  
  
*# Виявлення облич*faces = face\_cascade.detectMultiScale(gray\_image, scaleFactor=1.1, minNeighbors=10, minSize=(30, 30))  
  
*# Малювання прямокутників навколо виявлених облич  
for* (x, y, w, h) *in* faces:  
 cv2.rectangle(result\_image, (x, y), (x+w, y+h), (255, 0, 0), 2)  
  
*# Збереження результату*output\_path = f'output\_{image\_path[:image\_path.rfind(".")]}.jpg'  
cv2.imwrite(output\_path, result\_image)  
  
*# Відображення результату*cv2.imshow('Image', image)  
cv2.imshow('Detected Faces', result\_image)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

Виконав: студент Трикош І. В.