## **Державний університет інформаційно-комунікаційних технологій**

**Практична робота №9**

**з дисципліни: Технології штучного інтелекту**

**Тема: Обробка зображень та відео з використанням**

**бібліотеки OpenCV в Python**

Виконав:  
студент ДУІКТ

Тертишний В.Ю.

група: ШІДМ-51

м.Київ

**Мета:** ознайомитися з основами використання бібліотеки OpenCV для Python, яка є потужним інструментом для обробки та аналізу зображень; навчитися завантажувати, відображати, змінювати розміри зображень та застосовувати основні фільтри (розмиття, конвертацію в градації сірого тощо); відпрацювати базові методи виділення контурів, морфологічних операцій та фільтрації зображень; отримати навички роботи з відеопотоком, зокрема навчитись захоплювати відео з камери та обробляти його в режимі реального часу; розвинути навички у використанні основних функцій обробки зображень створення простих систем комп'ютерного зору.

**Код №1:**

import cv2

import numpy as np

def create\_collage(images, grid\_size, scale=1.0, border\_size=5, border\_color=(0, 0, 0)):

# Масштабування зображень та приведення до однакових розмірів

# Визначаємо мінімальну висоту і ширину серед усіх зображень

min\_width = min(img.shape[1] for img in images)

min\_height = min(img.shape[0] for img in images)

resized\_images = [cv2.resize(img, (min\_width, min\_height)) for img in images]

# Масштабування зображень, якщо заданий параметр scale

resized\_images = [cv2.resize(img, None, fx=scale, fy=scale) for img in resized\_images]

# Додавання рамок між зображеннями

bordered\_images = []

for img in resized\_images:

bordered\_img = cv2.copyMakeBorder(img, border\_size, border\_size, border\_size, border\_size,

cv2.BORDER\_CONSTANT, value=border\_color)

bordered\_images.append(bordered\_img)

# Створення сітки зображень

rows = []

for i in range(grid\_size[0]):

row\_images = bordered\_images[i \* grid\_size[1]: (i + 1) \* grid\_size[1]]

row = np.hstack(row\_images)

rows.append(row)

collage = np.vstack(rows)

return collage

# Завантаження зображень (три конкретних зображення)

images = [

cv2.imread("фото.jpg"), #своє фото СЮДИ

cv2.imread("фото.jpg"), #своє фото СЮДИ

cv2.imread("фото.jpg") #своє фото СЮДИ

]

# Перевірка, чи зображення завантажені правильно

if any(img is None for img in images):

print("Не вдалося завантажити одне з зображень.")

else:

# Створення колажу (наприклад, 1x3, так як у нас 3 зображення)

collage = create\_collage(images, (1, 3), scale=0.5, border\_size=10)

# Відображення колажу

cv2.imshow("Collage", collage)

cv2.waitKey(0)

cv2.destroyAllWindows()

**Робота коду №1:**  
(**>Скріни сюди<**)

**Код №2:**

import cv2

import numpy as np

def detect\_faces(image\_path):

# Завантаження зображення

img = cv2.imread(image\_path)

# Перетворення в сірий колір для спрощення обробки

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

# Завантаження попередньо тренованого каскадного класифікатора для обличчя

face\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade\_frontalface\_default.xml')

# Виявлення облич на зображенні

faces = face\_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5, minSize=(30, 30))

# Виведення кількості знайдених облич

print(f"Знайдено {len(faces)} облич.")

# Малювання рамок навколо облич

img\_with\_faces = img.copy()

for (x, y, w, h) in faces:

cv2.rectangle(img\_with\_faces, (x, y), (x + w, y + h), (0, 255, 0), 2)

# Відображення зображення з рамками навколо облич

cv2.imshow("Faces Detected", img\_with\_faces)

cv2.waitKey(0)

cv2.destroyAllWindows()

detect\_faces("фото з обличчями.jpg") #**>фото з кількома обличчями СЮДИ<**

**Робота коду №2:**  
(**>Скріни сюди<**)

**Висновки**

У процесі виконання поставленої мети було ознайомлено з основами використання бібліотеки OpenCV для Python, яка є потужним інструментом для обробки та аналізу зображень. Було освоєно базові операції, такі як завантаження, відображення та зміна розміру зображень, а також застосування основних фільтрів, таких як розмиття та перетворення в градації сірого. Вивчення цих технік допомогло зрозуміти важливість попередньої обробки зображень для подальших етапів їх аналізу. Крім того, було опрацьовано методи виділення контурів, морфологічні операції та фільтрацію зображень, що дозволяє підвищити ефективність аналізу структур на зображеннях. Окрему увагу було приділено роботі з відеопотоком, зокрема захопленню відео з камери та його обробці в режимі реального часу, що розширює можливості OpenCV для роботи з динамічними даними. У результаті було досягнуто значного прогресу у використанні основних функцій обробки зображень та створенні простих систем комп'ютерного зору, що є основою для подальших розробок у цій сфері.