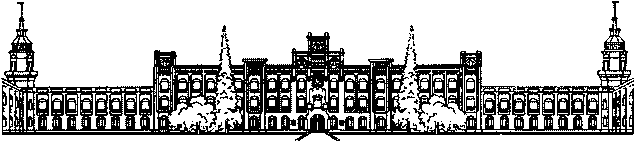
****

Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

Лабораторна робота №6

**З дисципліни «Технології Computer Vision»**

*ДОСЛІДЖЕННЯ ТЕХНОЛОГІЙ ПОРІВНЯННЯ ЦИФРОВИХ ЗОБРАЖЕНЬ ДЛЯ СТЕЖЕННЯ ЗА ОБ’ЄКТАМИ У ВІДЕОПОТОЦІ*

| Виконала  студент кафедри ІСТ ФІОТ, групи ІА-12: |  | Перевірив:  пос. Баран Д. Р. |
| --- | --- | --- |
| Яковенко Д. О. |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Київ 2024

**І. Мета:**

Дослідити принципи та особливості практичного застосування технологій порівняння цифрових зображень для стеження за об’єктами у відеопотоці з використанням спеціалізованих програмних бібліотек.

**ІІ. Завдання:**

*Реалізація проекту триває та спрямовано на збільшення функціональності*

*програмної компоненти. Лабораторія провідної ІТ-компанії реалізує масштабний проект розробки універсальної платформи з цифрової обробки зображень для задач Computer Vision. Платформа передбачає розташування back-end компоненти на власному хмарному сервері з наданням повноважень користувачам заздалегідь адаптованого front-end функціоналу універсальної платформи. Цим формується унікальна для потреб замовника ERP система з технологіями Computer Vision. Замовниками ресурсів платформи є: державні та комерційні компанії, що розробляють медичне обладнання з діагностування захворювань за візуальною інформацією; автоматизації аграрного бізнесу в аспекті обліку посівних територій за даними з БПЛА; візуального контролю безпекових заходів на об’єктах критичної інфраструктури: аеропорти, торгівельно-розважальні центри, житлові комплекси тощо.*

ГРУПА ВИМОГ 1.

Удосконалити скрипт із завдання Лр\_5 додавши можливість реалізації порівняння

об’єкта ідентифікації (оперативні – високоточні дані ДЗЗ, див. таблицю нижче) з

використанням дескриптора зображень.

Вимоги та обмеження:

Дескриптор зображення має стосуватись об’єкта ідентифікації.

Алгоритм та технологія визначення дескриптора зображень – за власним вибором.

Результат порівняння – кількість збігів особливих точок;

Кількість збігів особливих точок дескриптора зображень перевести в ймовірність

ідентифікації.

ГРУПА ВИМОГ 3.

Розробити програмний скрипт, що реалізує стеження за об’єктом у цифровому

відеопотоці. Зміст відео, об’єкт стеження – обрати самостійно. Метод та технологію

стеження обрати такою, що забезпечує стійкість процесу object-tracking для обраних

вихідними даними (відео, об’єкт стеження). Вибір обґрунтувати та довести його

ефективність.

**ІІ рівень складності – максимально 9 балів. Реалізувати дві з наданих груп вимог за власним вибором.**

| **Варіант (місяць народження)** | **Джерела даних ДЗЗ** | **Технічні умови** |
| --- | --- | --- |
| 3 | 1. Оперативні: <https://livingatlas2.arcgis.com/landsatexplorer/>  2. Високоточні: https://www.google.com.ua/maps | Район спостереження – земна поверхня. Об’єкти ідентифікації – лісові насадження довкола озер. Дата оперативних даних – 10.11.23.  Метод і технологія кластеризації / сегментації – повинні забезпечувати можливість розрізнення та ідентифікацію обраних об’єктів спостереження. |

**ІІІ. Результати виконання лабораторної роботи.**

**3.1. Синтезована математична модель перетворень графічних об’єктів відповідно до індивідуального завдання.**

**ГРУПА ВИМОГ 1:**

Відповідно до умов задачі синтезовано математичну модель операцій над цифровими растровими знімками земної поверхні. Задачею було ідентифікувати лісові насадження навколо озер та порівняти кількість співпадінь ключових точок на оперативному та високоточному зображенню. Довелося змінити джерело оперативних зображень, оскільки попереднє джерело, що використовувалося у лабораторній №5, мало дуже низьку якість даних, і через це програма працювала некоректно.

Модель реалізує порівняння двох зображень для знаходження спільних ознак за допомогою алгоритму SIFT (Scale-Invariant Feature Transform) і методу FLANN (Fast Library for Approximate Nearest Neighbors).

* **SIFT** є методом, розробленим для виявлення характерних точок (features) на зображеннях, які є стійкими до масштабних змін, обертання, а також частково стійкими до змін освітлення і шуму. Процес роботи SIFT складається з кількох основних етапів. По-перше, алгоритм виявляє ключові точки шляхом побудови масштабно-просторової репрезентації зображення, яка називається октавою. В кожній октаві зображення розмивається кілька разів і зменшується в розмірі. Ключові точки виявляються як максимуми і мінімуми в різних масштабах, що дозволяє знаходити цікаві точки на різних рівнях деталізації. Після виявлення ключових точок алгоритм обчислює дескриптори для кожної точки, які описують локальну структуру навколо неї за допомогою гістограм градієнтів. Ці дескриптори використовуються для порівняння ключових точок між зображеннями.
* **FLANN** є бібліотекою, яка забезпечує швидкий пошук найближчих сусідів для наборів високовимірних точок, що є критично важливим у задачах порівняння дескрипторів. FLANN використовує різні алгоритми для прискорення процесу пошуку, зокрема KD-дерево та інші методи, що дозволяють ефективно знаходити найближчих сусідів у великих наборах даних. Це значно зменшує час, необхідний для знаходження відповідностей між дескрипторами зображень.

Додатково підраховуємо кількість співпадінь контрольних точок, що відповідають лісовим насадженням. Результат виводимо в консолі програми.

**ГРУПА ВИМОГ 3:**

Відповідно до умов задачі синтезовано математичну модель для виявлення номерних знаків на транспортних засобах за допомогою каскадного класифікатора. Використання цієї моделі у реальному часі дозволяє аналізувати відеопотік та визначати місцезнаходження номерних знаків у кожному кадрі. Завдяки застосуванню багатопотокового підходу, обробка кадрів виконується паралельно з їх відображенням, що забезпечує високу продуктивність системи.

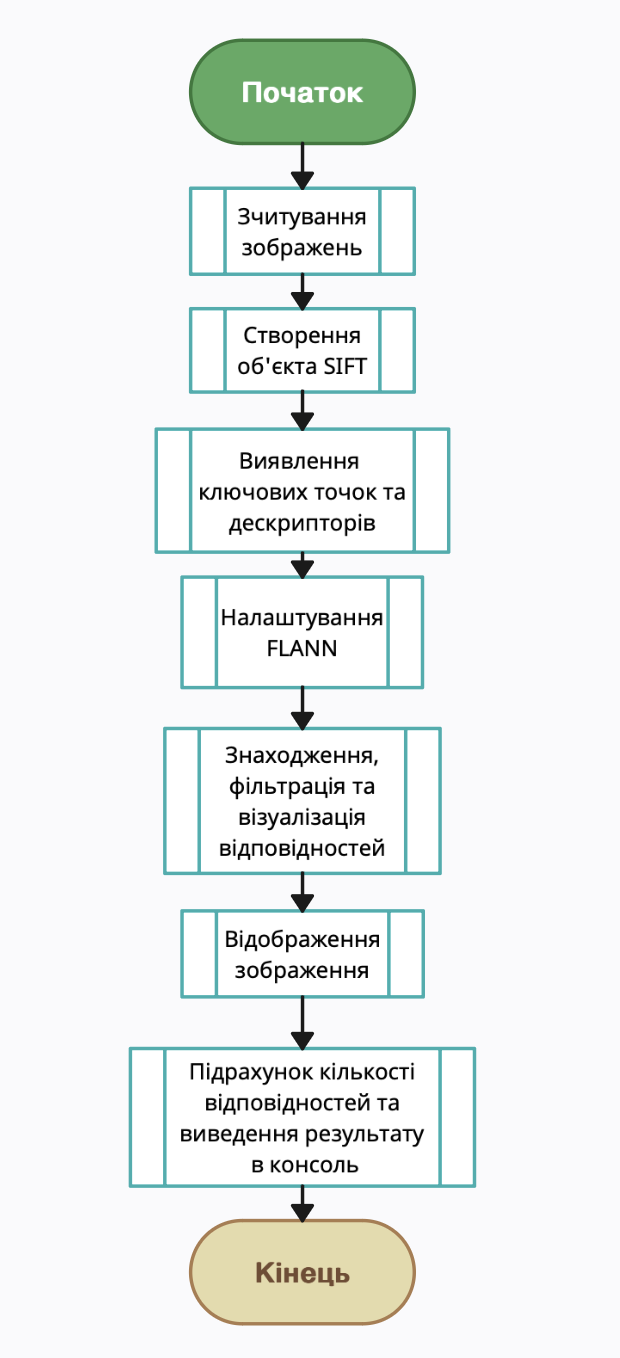
* При розробці цієї системи було використано **каскадний класифікатор**, побудований на основі **алгоритму Хаара**, який є одним з ефективних методів для виявлення об'єктів на зображеннях. Зокрема, класифікатор був налаштований для виявлення номерних знаків, що дозволяє йому успішно знаходити ці об'єкти в різних умовах освітлення та під різними кутами.
* Для забезпечення точності та ефективності алгоритму було налаштовано мінімальний та максимальний розміри виявлюваних об'єктів (minSize та maxSize). Це дозволяє обмежити пошук номерних знаків у відповідному діапазоні розмірів, що підвищує швидкість та точність обробки.

Висновки даної роботи можуть бути застосовані для різних задач, пов'язаних з розпізнаванням об'єктів у відеопотоці, таких як автоматичне розпізнавання номерних знаків для контролю доступу, моніторинг транспортного потоку та інші сфери, де необхідне ефективне виявлення і відстеження об'єктів на зображеннях.

**3.2. Блок схема алгоритму та її опис.**

**ГРУПА ВИМОГ 1:**

Застосування синтезованих моделей здійснюється у порядку, що відображає суть алгоритму реалізації завдань лабораторної роботи.



| Рис.1. Блок-схема алгоритму програми для групи вимог 1. |
| --- |
|  |

Розглянемо даний алгоритм. Тут використовується алгоритм SIFT (Scale-Invariant Feature Transform) для знаходження відповідностей між ключовими точками та їх описами на двох зображеннях. Спочатку зчитуються два зображення з файлів та перетворюються в градації сірого кольору. Потім за допомогою алгоритму SIFT визначаються ключові точки та їх описи на кожному зображенні.

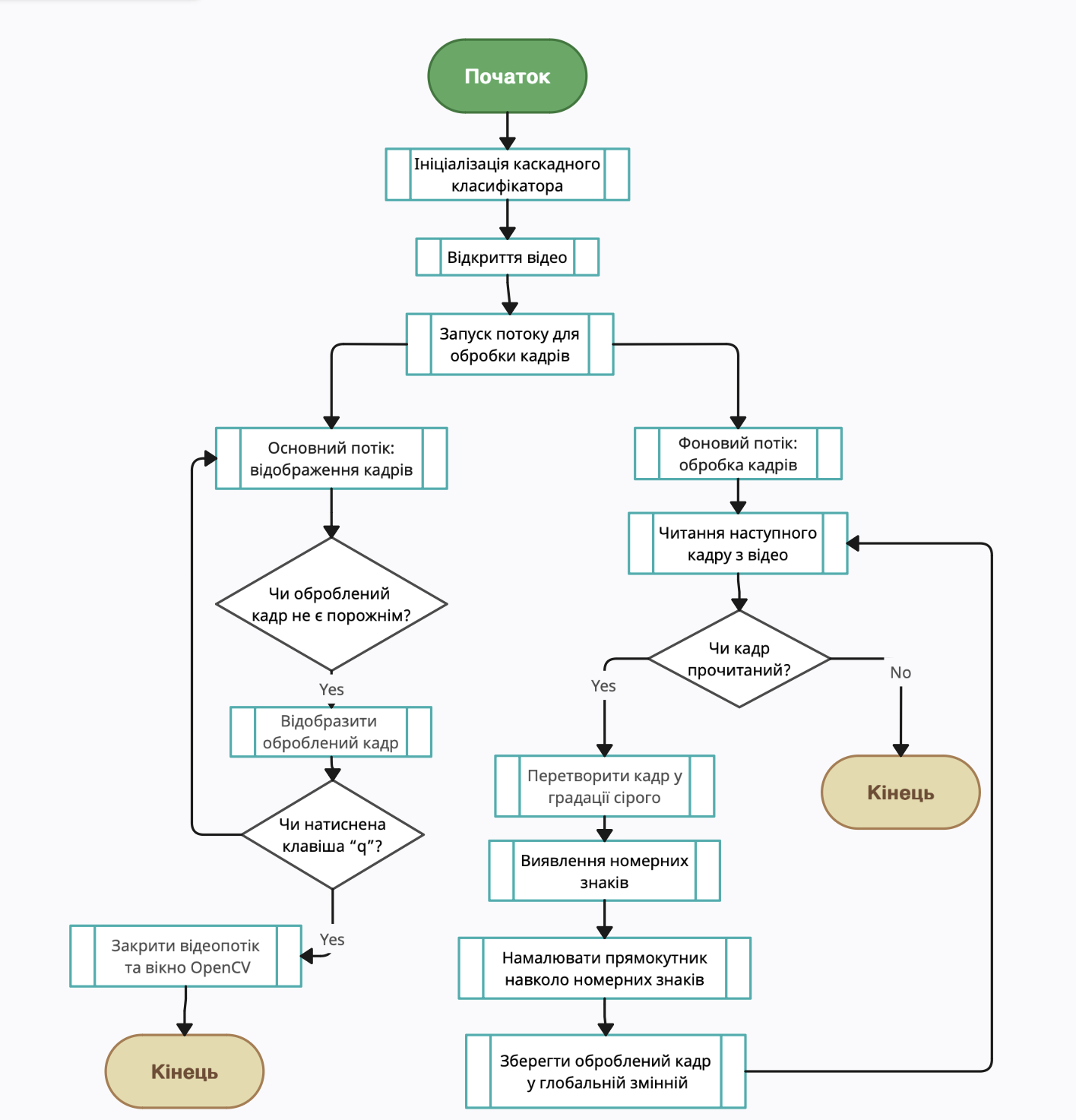
Після цього застосовується метод FLANN (Fast Library for Approximate Nearest Neighbors) для знаходження найближчих сусідів для кожного опису ключової точки на першому зображенні серед описів ключових точок на другому зображенні.

Далі використовується поріговий підхід, де лише ті збіги, для яких відстань до найближчого сусіда менше ніж 70% відстані до другого найближчого сусіда, вважаються правильними збігами.

Ці збіги відображаються на новому зображенні за допомогою функції cv.drawMatchesKnn(), а також виводяться на екрані. Програма чекає натискання клавіші 'q' для закриття вікна з результатами.

Нарешті, обчислюється та повертається кількість збігів для кожної пари зображень.

**ГРУПА ВИМОГ 3:**



| Рис.2. Блок-схема алгоритму програми для групи вимог 3. |
| --- |
|  |

Ця програма використовує каскадний класифікатор для виявлення номерних знаків на кадрах з відео. Спочатку завантажується каскадний класифікатор для номерних знаків, а також відкривається відео.

Потім створюється та запускається окремий потік для обробки кадрів. У цьому потоці кожен кадр зчитується з відео, конвертується у градації сірого кольору, та подається на вхід каскадному класифікатору для пошуку номерних знаків. Знайдені номерні знаки виділяються прямокутниками на кадрі. Якщо ж кадр не прочитаний – програма завершує роботу.

Одночасно у головному потоці перевіряється, чи був оброблений кадр. Якщо так, то він відображається на екрані. Потік очікує натискання клавіші 'q' для завершення відтворення відео. Якщо ж клавішу не натиснено – програма продовжує працювати та відтворювати відеопотік.

Після завершення відтворення відео, програма чекає завершення обробки кадрів у фоновому потоці. Потім відеопотік закривається, а всі вікна OpenCV очищаються.

**3.3. Опис структури проекту програми в середовищі PyCharm.**

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

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

|  |
| --- |
| Рис.3. Структура проекту. |

CVLab6 – головний каталог проекту

cascade – папка з каскадами

images – папка із зображеннями для дескриптора

video – папка з відео для програми розпізнавання номерниз знаків

carPlateDetection.py – файл програмного коду лабораторної роботи (група вимог 3)

imageDescriptor.py – файл програмного коду лабораторної роботи (група вимог 1);

**3.4. Результати роботи програми відповідно до завдання.**

Результатом роботи програми є програма, в якій при її запуску отримуємо такий результат:

**ГРУПА ВИМОГ 1:**

|  |
| --- |
| Рис.4. Результат обробки зображень та співпадінь контрольних точок.    Рис.5. Результат збігів. |

Обробка двох зображень, високоточного та оперативного алгоритмами SIFT та FLANN – це поєднання є основою для багатьох застосувань, таких як стічинг зображень, 3D реконструкція, відстеження об'єктів, та багато інших завдань у сфері Computer Vision. Ці два алгоритми дозволяють ефективно ідентифікувати та зіставляти характерні точки між різними зображеннями. Як бачимо з результатів – програма чудово ідентифікує кількість лісових насаджень на зображенях у різний період часу, та підраховує кількість співпадінь.

**ГРУПА ВИМОГ 3:**

|  |
| --- |
| Рис.6. Результат ідентифікації номерних знаків авто.  Продовжуючи завдання лабораторної роботи №4, у якій я здійснювала ідентифікацію автомобілів на дорозі, мені було цікаво поглибитись в цю тему, та розпізнавати знаходження номерних знаків авто у потоці.  Реалізована система є прикладом успішного використання Computer Vision для вирішення прикладних задач в реальному часі, демонструючи високу продуктивність та точність виявлення об'єктів на основі каскадного класифікатора Хаара. В подальшому дослідженні можливе удосконалення алгоритмів виявлення та використання додаткових методів машинного навчання для підвищення точності і швидкості обробки (наприклад розпізнавання тексту на номерах автомобілів). |

**3.5. Програмний код.**

Програмний код послідовно реалізує алгоритми для виконання завдання другого рівня складності.

При цьому використано можливості Python бібліотек: openCV, threading.

**ГРУПА ВИМОГ 1:**

import cv2 as cv

def sift\_feature\_matching(filename\_1, filename\_2):

img1 = cv.imread(filename\_1, cv.IMREAD\_GRAYSCALE)

img2 = cv.imread(filename\_2, cv.IMREAD\_GRAYSCALE)

sift = cv.SIFT\_create()

kp1, des1 = sift.detectAndCompute(img1, None)

kp2, des2 = sift.detectAndCompute(img2, None)

FLANN\_INDEX\_KDTREE = 1

index\_params = dict(algorithm=FLANN\_INDEX\_KDTREE, trees=5)

search\_params = dict(checks=50)

flann = cv.FlannBasedMatcher(index\_params, search\_params)

matches = flann.knnMatch(des1, des2, k=2)

matchesMask = [[0, 0] for i in range(len(matches))]

for i, (m, n) in enumerate(matches):

if m.distance < 0.7 \* n.distance:

matchesMask[i] = [1, 0]

draw\_params = dict(matchColor=(0, 255, 0),

singlePointColor=(255, 0, 0),

matchesMask=matchesMask,

flags=cv.DrawMatchesFlags\_DEFAULT)

img3 = cv.drawMatchesKnn(img1, kp1, img2, kp2, matches, None, \*\*draw\_params)

cv.imshow('sift\_feature\_matching', img3)

if cv.waitKey(0) & 0xff == 27:

cv.destroyAllWindows()

num\_matches = sum(match[0] for match in matchesMask)

return num\_matches

if \_\_name\_\_ == '\_\_main\_\_':

num\_matches\_1 = sift\_feature\_matching('images/3.png', 'images/4.png')

print(f"Кількість співпадінь: {num\_matches\_1}")

num\_matches\_2 = sift\_feature\_matching('images/5.png', 'images/6.png')

print(f"Кількість співпадінь: {num\_matches\_2}")

num\_matches\_3 = sift\_feature\_matching('images/1.png', 'images/2.png')

print(f"Кількість співпадінь: {num\_matches\_3}")

**ГРУПА ВИМОГ 3:**

import cv2

import threading

def process\_frames():

global plates\_cascade

global cap

global processed\_frame

while True:

ret, frame = cap.read()

if not ret:

break

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

plates = plates\_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5, minSize=(100, 50),

maxSize=(300, 250))

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

cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)

processed\_frame = frame.copy()

# Завантаження каскадного класифікатора для номерних знаків

plates\_cascade = cv2.CascadeClassifier('cascade/haarcascade\_plate\_number.xml')

# Відкриття відео

cap = cv2.VideoCapture('video/car\_flow.mp4')

# Глобальна змінна для зберігання обробленого кадру

processed\_frame = None

# Створення та запуск потоку для обробки кадрів

processing\_thread = threading.Thread(target=process\_frames)

processing\_thread.start()

while True:

if processed\_frame is not None:

cv2.imshow('License Plate Recognition', processed\_frame)

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

break

# Чекаємо, доки обробка кадрів не завершиться

processing\_thread.join()

# Закриття відеопотоку та вікна OpenCV

cap.release()

cv2.destroyAllWindows()

**3.6. Аналіз результатів відлагодження та верифікації результатів роботи програми.**

Результати відлагодження та тестування довели працездатність розробленого коду.

Верифікація функціоналу програмного коду, порівняння отриманих результатів з технічними умовами завдання на лабораторну роботу доводять, що завдання виконано у повному обсязі.

**IV. Висновки.**

У ході виконання лабораторної роботи був реалізований алгоритм використання SIFT (Scale-Invariant Feature Transform) для виявлення та порівняння ключових особливостей на двох зображеннях. Для цього було використано бібліотеку OpenCV у сполученні з методом FLANN (Fast Library for Approximate Nearest Neighbors). Також був розглянутий процес розпізнавання номерних знаків у відеопотоці за допомогою комп'ютерного зору. Для цього було використано бібліотеку OpenCV у сполученні з алгоритмом каскадних класифікаторів Хаара.

Ця лабораторна робота демонструє практичне використання Computer Vision у різних сферах застосування для вирішення різноманітних завдань, пов'язаних з обробкою та аналізом зображень, розпізнавання об'єктів на відеопотоці. Реалізовані алгоритми можуть бути використані у різних областях, включаючи дослідження територій лісових насаджень, відеоспостереження, автоматизоване розпізнавання автомобільних номерних знаків та інші сценарії, де потрібно виявляти об'єкти на відео та фото.

Виконала: студент Яковенко Д.О.