МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут автоматики і електротехніки

Кафедра комп'ютерних технологій та інформаційної безпеки

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних технологій та інформаційної безпеки, к.т.н., доцент
_____С.М. Нужний
«_18_» ____06_____ 2025 р.

Кваліфікаційна робота
на правах рукопису

КВАЛІФІКАЦІЙНА РОБОТА

ОБРОБКА ЗОБРАЖЕНЬ В ЗАДАЧАХ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ І КОНТРОЛЮ ДОСТУПУ

Ступінь вищої освіти: БАКАЛАВР
Галузь знань: 12 «Інформаційні технології»
Спеціальність: 125 «Кібербезпека та захист інформації»
Освітньо-професійна програма: Системи технічного захисту інформації, автоматизація її обробки

Виконав: студент 4 курсу групи 4387зст
Зосімов Олег Сергійович

Кваліфікаційна робота містить результати самостійного виконання навчального завдання. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Зосімов О.С.

Керівник: старший викладач кафедри комп'ютерних технологій та інформаційної безпеки **Касьянов Юрій Іванович** _____

Миколаїв – 2025

Миколаїв – 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут автоматики і електротехніки

ЗАТВЕРДЖУЮ
Гарант освітньої програми, к.т.н., доцент, завідувач кафедри КТІБ
_____ С.М. Нужний
«_18_» ____06____ 2025 р.

ЗАВДАННЯ

на кваліфікаційну роботу

Студент: Зосімов Олег Сергійович
Курс:4
Група: 4387зст
Ступінь вищої освіти: бакалавр
Галузь знань: 12 Інформаційні технології
Спеціальність: 125 «Кібербезпека та захист інформації»
Освітньо-професійна програма: Системи технічного захисту інформації, автоматизація її обробки

1. Тема роботи: «Обробка зображень в задачах розпізнавання об'єктів і контролю доступу»
затверджена наказом НУК від «_23__» _квітня_____2025 р. № УЧ- 343

2. Вихідні дані до роботи: види обробки – детектування облич, виявлення контурів, сегментація зображень, побудова ознак, порівняння з еталонами; кількість градацій яскравості – 256; розмір зображення – до 300х300 пікселів.

3. Перелік питань, які необхідно розробити: 1 Загальні положення обробки зображень. 2 Постановка задачі. 3 Аналіз алгоритмів обробки зображень. 4 Розробка програми. 5 Тестування і результати.

4. Терміни виконання завдання:
термін видачі завдання: «_03_» лютого 2025 р.
термін подання роботи: «18» червня 2025 р.

6. План-графік виконання кваліфікаційної роботи

№ п/п	Заходи	Дата		Готовність	Відмітка про виконання
		Навч. тиждень	Контрольна дата		
10.	Організаційні збори.	23	03.02.2025	Вибір теми, визначення мети, завдань, об'єкта та предмета дослідження	
11.	Організаційні збори	25	20.02.2025	Попередній варіант оглядових розділів, підбір і опрацювання списку використаних джерел	
12.	Рубіжний контроль	31	03.04.2025	Попередній варіант повної КР.	
13.	Рубіжний контроль	33	20.04.2025	Попередній варіант презентації.	
14.	ЄДКІ на рівень «бакалавр»	34	29.04.2025	Здавання ЄДКІ зі спеціальності на ступінь бакалавра	
15.	Перевірка на плагіат	42	14.06.2025	1. Електронний варіант кваліфікаційної роботи; 2. Подання КР на перевірку на плагіат.	
16.	Кафедральний захист	43	18.06.2025	1. Оформлена КР (в форматі pdf) (передається студентом на кафедрі для внутрішньої рецензії, висновок керівника). У разі отримання позитивної внутрішньої рецензії, КР подається на зовнішню рецензію. 2. Оформлена презентація (в форматі pdf).	
17.	Подання документів на захист	44	24.06.2025	1. Подання щодо захисту КР 2. Зовнішня рецензія (окремо від КР). Резюме на КР. 3. Висновок кафедри про КР, допуск до захисту; 4. Електронний варіант та роздрукована презентація в кількості 5 примірників. 5. Електронний варіант КР на диску	
18.	Захист КР	44	25.06.2025	1. Приєднання студента до засідання кваліфікаційної комісії (конференції) у встановлений час для проведення процедури дистанційного захисту 2. Процедура захисту КР.	

ПРИМІТКА: Завдання додається до виконаної роботи та подається до атестаційної комісії.

Керівник
Старший викладач кафедри комп'ютерних технологій
та інформаційної безпеки _____ Ю.І. Касьянов

Студент _____ О.С. Зосімов

АНОТАЦІЯ

Зосімов О.С. Обробка зображень в задачах розпізнавання об’єктів і контролю доступу.

Кваліфікаційна робота: 84 с.; 17 рис.; 2 табл.; 37 джерел, 1 додаток

Актуальність роботи обумовлена важливістю задачі контролю доступу та ідентифікації осіб на об’єктах інформаційної діяльності (ОІД) і використання для цієї задачі біометричних систем.

Метою роботи є створення функціонального прототипу системи контролю доступу, що базується на порівнянні геометричних ознак обличчя, отриманих за допомогою алгоритмів обробки зображень.

Об’єкт дослідження - процес ідентифікації об’єктів в системах відеоспостереження та контролю доступу.

В кваліфікаційній роботі було досліджено теоретичні основи розпізнавання образів, зокрема — попередньої обробки зображень, виділення ключових ознак та методів класифікації. Практична частина реалізована у вигляді графічного інтерфейсу, який дозволяє користувачу здійснити ідентифікацію на основі обраного зображення, отримати результат порівняння з базою еталонів та переглянути додаткову візуалізацію у вигляді ключових точок і теплових карт.

Результатом роботи є функціонуюча система з підтримкою логування, яка може бути використана як автономне рішення для обмежених середовищ.

ANNOTATION

Zosimov O.S. Image processing in the tasks of object recognition and access control.

Qualification work: 84 p.; 17 figs; 2 tables; 37 sources, 1 appendix

The relevance of the work is due to the importance of the task of access control and identification of persons at information activity objects (IAOs) and the use of biometric systems for this task.

The aim of the work is to create a functional prototype of an access control system based on the comparison of geometric features of the face obtained using image processing algorithms.

The object of research is the process of object identification in video surveillance and access control systems.

The qualification work investigated the theoretical foundations of pattern recognition, in particular, image preprocessing, key feature extraction, and classification methods. The practical part is implemented in the form of a graphical interface that allows the user to perform identification based on the selected image, get the result of comparison with the reference database, and view additional visualization in the form of key points and heat maps.

The result is a functioning, logging-enabled system that can be used as a standalone solution for confined environments.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ОІД – об’єкт інформаціної діяльності;
БСКД – біометрична система контролю доступу;
HOG – алгоритм Histogram of Oriented Gradients;
SIFT – Scale-Invariant Feature Transform;
SURF – Speeded Up Robust Features;
CNN – Convolutional Neural Networks;
R-CNN – Region-based CNN
YOLO – You Only Look Once;
SSD – Single Shot Detector;
LFW – Labeled Faces in the Wild;
ViT – модель Vision Transformer
ПЗ – програмне забезпечення;
ПК – персональний комп’ютер;

ВСТУП

За сучасних умов стрімкого розвитку інформаційних технологій та постійного зростання обсягів цифрової взаємодії все більшого значення набуває проблема ідентифікації особи. Особливо актуальною вона стає в контексті забезпечення контролю доступу до захищених систем, приміщень або інформаційних ресурсів. Традиційні методи автентифікації, такі як паролі або фізичні ключі, поступово втрачають ефективність через низьку надійність, ризики компрометації та людський фактор. У зв’язку з цим біометричні системи, зокрема технології розпізнавання облич, дедалі частіше розглядаються як більш надійна та зручна альтернатива.

Тема дипломної роботи є актуальною як з точки зору прикладної реалізації, так і в контексті дослідження ефективності сучасних алгоритмів комп’ютерного зору. В рамках цієї роботи здійснюється розробка програмного забезпечення, яке дозволяє здійснювати ідентифікацію особи за зображенням обличчя, що може бути використане як елемент системи локального біометричного доступу.

Метою роботи є створення функціонального прототипу системи контролю доступу, що базується на порівнянні геометричних ознак обличчя, отриманих за допомогою алгоритмів обробки зображень.

Досягнення поставленої мети передбачає вирішення таких основних задач:

– аналіз наявних методів розпізнавання та основних етапів;

– вибір інструментальних засобів розробки;

– розробка структури даних для збереження еталонів, механізму порівняння ознак та інтерфейсу користувача і створення програмного модуля, призначеного для розпізнавання обличчя з метою контролю доступу;

– тестування розробленої програми та аналіз результатів.

Об’єктом дослідження є процес ідентифікації об’єктів в системах відеоспостереження та контролю доступу.

Предмет дослідження —біометрична автентифікація з використанням технологій комп’ютерного зору, зокрема бібліотеки MediaPipe та мови програмування Python у середовищі настільного застосунку.

В рамках виконання дипломної роботи було досліджено теоретичні основи розпізнавання образів, зокрема — попередньої обробки зображень, виділення ключових ознак та методів класифікації. Практична частина реалізована у вигляді графічного інтерфейсу, який дозволяє користувачу здійснити ідентифікацію на основі обраного зображення, отримати результат порівняння з базою еталонів та переглянути додаткову візуалізацію у вигляді ключових точок і теплових карт.

Результатом роботи є функціонуюча система з підтримкою логування, яка може бути використана як автономне рішення для обмежених середовищ. У процесі тестування було перевірено її точність, стійкість до зовнішніх факторів та обмеження, що визначають можливості її подальшого застосування або масштабування.

Таким чином, дипломна робота поєднує теоретичну частину з аналізом предметної області та практичну реалізацію актуального програмного рішення у сфері біометричного контролю доступу.

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ ОБРОБКИ ЗОБРАЖЕНЬ

1.1 Основи ком’ютерного зору

Завдання комп'ютерного зору включають методи отримання, обробки, аналізу та розуміння цифрових зображень, а також вилучення високовимірних даних з реального світу з метою отримання числової або символьної інформації, наприклад, у формі рішень [1-4]. «Розуміння» в цьому контексті означає перетворення візуальних зображень (вхідних даних на сітківку) на описи, які мають сенс для розумових процесів і можуть спонукати до відповідних дій. Це розуміння зображень можна розглядати як розплутування символьної інформації з даних зображення за допомогою моделей, побудованих за допомогою геометрії, фізики, статистики та теорії навчання.

Наукова дисципліна комп'ютерного зору займається теорією штучних систем, які витягують інформацію із зображень. Дані зображення можуть мати різні форми, такі як відеопослідовності, зображення з кількох камер, багатовимірні дані з 3D-сканера, 3D-хмари точок з датчиків LiDaR або медичні скануючі пристрої (рис.1.1). Технологічна дисципліна комп'ютерного зору прагне застосувати свої теорії та моделі до побудови систем комп'ютерного зору.

Піддисципліни комп'ютерного зору включають реконструкцію сцени, виявлення об'єктів, виявлення подій, розпізнавання активності, відстеження відео, розпізнавання об'єктів, 3D-оцінку пози, навчання, індексування, оцінку руху, візуальне сервоуправління, 3D-моделювання сцени та відновлення зображень.

Комп'ютерний зір — це міждисциплінарна галузь, яка займається тим, як комп'ютери можуть отримувати високий рівень розуміння з цифрових зображень або відео. З точки зору інженерії, вона прагне автоматизувати завдання, які може виконувати людська зорова система [5-7].

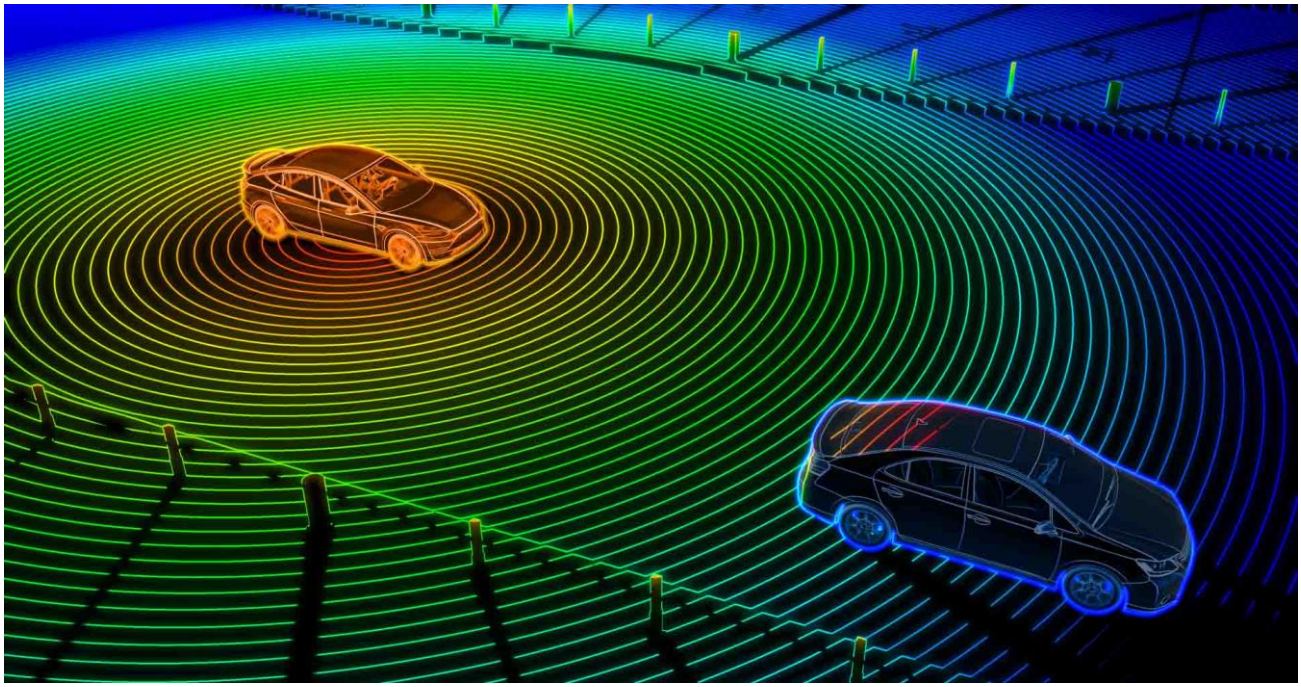

Рисунок 1.1 — 3D-хмара точок з датчиків LiDaR

Комп'ютерний зір займається автоматичним вилученням, аналізом та розумінням корисної інформації з одного зображення або послідовності зображень. Він передбачає розробку теоретичної та алгоритмічної основи для досягнення автоматичного візуального розуміння [8]. Як наукова дисципліна, комп'ютерний зір займається теорією штучних систем, які витягують інформацію із зображень. Дані зображення можуть мати різні форми, такі як відеопослідовності, зображення з кількох камер або багатовимірні дані з медичного сканера [9]. Як технологічна дисципліна, комп'ютерний зір прагне застосовувати свої теорії та моделі для побудови систем комп'ютерного зору. Машинний зір відноситься до дисципліни системної інженерії, особливо в контексті автоматизації виробництва. Останнім часом терміни «комп'ютерний зір» та «машинний зір» більшою мірою зблизилися [10].

Наприкінці 1960-х років комп'ютерний зір з'явився в університетах, які були піонерами штучного інтелекту. Він мав на меті імітувати зорову систему людини як крок до наділення роботів інтелектуальною поведінкою [11]. У 1966 році вважалося, що цього можна досягти за допомогою літнього

студентського проекту [12], підключивши камеру до комп'ютера та попросивши його «описати те, що він бачить» [13, 14].

Те, що відрізняло комп'ютерний зір від поширеної на той час галузі цифрової обробки зображень, було бажання витягти тривимірну структуру з зображень з метою досягнення повного розуміння сцени. Дослідження 1970-х років заклали ранні основи для багатьох алгоритмів комп'ютерного зору, які існують сьогодні, включаючи витяг ребер із зображень, маркування ліній, неполіедричне та поліедричне моделювання, представлення об'єктів як взаємозв'язків менших структур, оптичний потік та оцінку руху [11].

Наступне десятиліття ознаменувалося дослідженнями, заснованими на більш суворому математичному аналізі та кількісних аспектах комп'ютерного зору. До них належать концепція масштабного простору, висновок форми з різних ознак, таких як затінення, текстура та фокус, а також контурні моделі, відомі як «змії». Дослідники також усвідомили, що багато з цих математичних концепцій можна розглядати в рамках тієї ж оптимізації, що й регуляризація та марковські випадкові поля [15]. До 1990-х років деякі з попередніх дослідницьких тем стали більш активними, ніж інші. Дослідження проективних 3D-реконструкцій призвели до кращого розуміння калібрування камери. З появою методів оптимізації калібрування камери стало зрозуміло, що багато ідей вже були досліджені в теорії коригування зв'язок з галузі фотограмметрії. Це призвело до появи методів розріджених 3D-реконструкцій сцен з кількох зображень. Був досягнутий прогрес у проблемі щільної стереовідповідності та подальших методах багатовидового стерео. Водночас для вирішення сегментації зображень використовувалися варіації розрізу графіків. Це десятиліття також ознаменувало перший випадок, коли методи статистичного навчання були використані на практиці для розпізнавання облич на зображеннях (див. Власне обличчя). Ближче до кінця 1990-х років відбулися значні зміни зі збільшенням взаємодії між галузями комп'ютерної графіки та комп'ютерного зору. Це включало рендеринг на основі зображень,

морфінг зображень, інтерполяцію вигляду, панорамне зшивання зображень та ранній рендеринг світлового поля [11].

Нещодавні роботи показали відродження методів на основі ознак, що використовуються разом із методами машинного навчання та складними фреймворками оптимізації [16, 17] Розвиток методів глибокого навчання вдихнув нове життя в сферу комп'ютерного зору. Точність алгоритмів глибокого навчання на кількох еталонних наборах даних комп'ютерного зору для завдань, починаючи від класифікації,[18] сегментації та оптичного потоку, перевершила попередні методи [19].

1.2 Застосування в системах контролю доступу

У сучасному світі системи контролю доступу дедалі частіше інтегрують методи обробки зображень, адже необхідність у надійному, гнучкому та швидкому механізмі ідентифікації особи набуває особливої актуальності. Традиційні методи — такі як фізичні ключі, магнітні картки або паролі — поступово відходять на другий план через свою вразливість до втрати, крадіжки або копіювання. Натомість технології, що ґрунтуються на аналізі візуальних ознак, дозволяють підвищити рівень безпеки та зменшити залежність від людського чинника.

Особливої популярності набули біометричні методи, серед яких найбільш розповсюдженими є системи розпізнавання обличчя, райдужної оболонки ока, а також аналіз геометрії руки або долоні. З-поміж них розпізнавання обличчя вирізняється тим, що не потребує фізичного контакту з пристроєм, дозволяючи здійснювати ідентифікацію навіть на відстані або під час руху. Цей аспект виявився особливо корисним у контексті пандемії COVID-19, коли прямий контакт з поверхнями був небажаним, а обличчя залишалося чи не єдиним зручним способом безконтактного розпізнавання особистості.

У системах контролю доступу, що ґрунтуються на обробці зображень, вирішальну роль відіграють алгоритми комп’ютерного зору. Вони дають змогу в реальному часі виявляти об’єкти, аналізувати їх форму, положення та інші візуальні характеристики. Наприклад, система може миттєво зафіксувати появу людини у кадрі, виявити обличчя, визначити його положення, провести порівняння з наявними шаблонами та прийняти рішення про надання чи відмову в доступі. Ці кроки виконуються за частки секунди, що робить подібні рішення придатними для впровадження у місцях із великим потоком людей: на входах до офісів, у навчальних закладах, аеропортах або навіть у житлових будинках.

Важливим є також питання точності. У випадку з контролем доступу помилкове спрацьовування може мати серйозні наслідки: від незручностей до загроз безпеці. Саме тому сучасні системи все частіше комбінують кілька алгоритмів або методів перевірки — наприклад, одночасно розпізнають обличчя та перевіряють наявність картки чи введення PIN-коду. Це дозволяє значно знизити ймовірність помилкової авторизації або несанкціонованого проникнення.

Окремої уваги заслуговують адаптивні системи, що здатні навчатися на основі нових даних. Наприклад, система може підлаштовуватись під зміну зовнішності користувача: нову зачіску, наявність окулярів або зміну виразу обличчя. Такі гнучкі підходи базуються на застосуванні методів глибокого навчання, зокрема згорткових нейронних мереж, які продемонстрували високу ефективність у задачах розпізнавання образів.

У випадках, коли йдеться про контроль доступу до зон з підвищеним рівнем захисту, використовують ще складніші системи. Вони можуть не лише фіксувати та розпізнавати обличчя (рис.1.2), а й відстежувати поведінкові параметри — наприклад, особливості ходи, жести, міміку або навіть типові маршрути пересування в межах будівлі. Такі функції дають змогу не лише ідентифікувати особу, а й виявити підозрілу поведінку або потенційні загрози ще до моменту здійснення порушення.

Рисунок 1.2 — Охоронна система з розпізнаванням обличчя

Роль обробки зображень не обмежується лише фазою розпізнавання. Важливим етапом є також фільтрація, попередня обробка та нормалізація вхідних даних. У реальних умовах камера може фіксувати зображення за різного освітлення, під різними кутами або з частковим перекриттям об’єкта. Тому до системи інтегрують попередню обробку кадрів — видалення шуму, вирівнювання яскравості, корекцію кольорів тощо — що забезпечує стабільність і точність подальших операцій.

Застосування подібних систем виходить далеко за межі виключно безпекового сегмента. Вони дедалі активніше використовуються у фінансовому секторі, державних установах, медичних закладах, а також у побуті — наприклад, для розумного керування доступом до житла або

персональних пристроїв. Розпізнавання облич стало повсякденним елементом взаємодії з технікою, а рівень його інтеграції в цифрові системи лише зростає.

Попри безліч переваг, впровадження систем контролю доступу на основі зображень викликає й низку етичних, юридичних та соціальних питань. Зокрема, йдеться про дотримання конфіденційності, зберігання біометричних даних, прозорість алгоритмів прийняття рішень. Тому поряд із розвитком технологічних складових все більшого значення набуває необхідність створення чітких нормативів і регуляторних механізмів.

Таким чином, обробка зображень у системах контролю доступу є не лише технічно ефективною, а й багатоаспектною сферою, яка поєднує в собі інженерні, соціальні та етичні виклики. Її значення зростає в умовах глобальної цифровізації та підвищеного запиту на безпеку, а її розвиток формує нові стандарти взаємодії людини з цифровими системами.

1.3 Методи розпізнавання об’єктів

Сучасні інформаційні системи дедалі активніше покладаються на візуальні дані, які надходять із камер відеоспостереження, сенсорних пристроїв або відсканованих зображень. У межах завдань безпеки, логістики, автоматизації обліку та доступу ключову роль відіграє здатність програмного забезпечення не просто фіксувати наявність об’єкта у полі зору, а й коректно його розпізнавати. Саме тому методи розпізнавання об’єктів стали критично важливим інструментом у сфері розробки інтелектуальних систем, і з кожним роком їх точність, адаптивність та швидкодія невпинно зростають.

На початкових етапах розвитку комп’ютерного зору використовувалися здебільшого класичні підходи, зосереджені на аналізі контурів, геометричних ознак або кольорових характеристик. Серед найперших алгоритмів, що отримали широке поширення, можна згадати метод шаблонів, де кожен об'єкт представлявся заздалегідь збереженим зразком, з яким порівнювався фрагмент зображення в процесі аналізу. Такий підхід виявився надзвичайно обмеженим

у змінних умовах освітлення, масштабування, поворотів об'єкта та наявності шумів, тому його поступово витіснили більш гнучкі моделі.

Одним із перших серйозних проривів стала побудова ознак на основі градієнтної інформації. Зокрема, алгоритм Histogram of Oriented Gradients (HOG) (рис. 1.3) продемонстрував, що опис об'єкта за допомогою орієнтацій країв у локальних областях дозволяє з високою точністю виділяти, скажімо, людські постаті у відеопотоці [20]. Метод HOG спрацював особливо добре в задачах, де форму об’єкта можна визначити за допомогою меж і контурів. Він став основою для численних систем пішохідного виявлення, які й досі залишаються актуальними у транспортній та міській інфраструктурі.

Рисунок 1.3 — Розпізнавання обличчя методом HOG

Паралельно з цим розвивалися інші дескриптори, зокрема Scale-Invariant Feature Transform (SIFT) та Speeded Up Robust Features (SURF), які дозволяли виявляти характерні точки об'єкта незалежно від масштабу чи орієнтації [21]. Їх використання значно розширило можливості розпізнавання в нестабільних умовах, а також забезпечило порівняння об’єктів за складною структурою.

Проте справжній прорив у точності й масштабованості відбувся завдяки застосуванню методів глибокого навчання. Упродовж останнього десятиліття

згорткові нейронні мережі (Convolutional Neural Networks, CNN) стали золотим стандартом у сфері розпізнавання зображень. Їх архітектура, побудована на чергуванні згорток, пулінгів і повнозв’язних шарів, дозволяє моделі автоматично навчатися релевантних ознак із великого обсягу зображень, що неможливо досягнути ручними підходами [22].

Однією з найбільш знакових моделей, що зробила методи CNN доступними для широкого застосування, стала архітектура AlexNet, яка перемогла у конкурсі ImageNet 2012, продемонструвавши майже дворазове зниження помилки розпізнавання в порівнянні з традиційними підходами [23]. Надалі з’явилися VGGNet, ResNet, Inception, кожна з яких вдосконалювала глибину, швидкодію або адаптивність мереж.

Особливе місце займають так звані регіонні мережі (Region-based CNN, R-CNN), що уможливили локалізацію об'єктів на зображенні. Вони не лише класифікують наявність певного класу, а й точно визначають координати області, де об'єкт знаходиться. Це стало вкрай важливим для систем відеоспостереження, де потрібно в реальному часі виявляти ідентифіковані об’єкти: обличчя, транспорт, номерні знаки. У подальших ітераціях — Fast R-CNN, Faster R-CNN та Mask R-CNN — розв’язувалися задачі багатокласової локалізації та навіть семантичної сегментації [24].

Проте паралельно з регіонними підходами активно розвивались і одностадійні архітектури, які забезпечують більшу швидкодію. Зокрема, YOLO (You Only Look Once) та SSD (Single Shot Detector) дозволили здійснювати розпізнавання та локалізацію об’єктів буквально за одну операцію проходу зображення через мережу. Саме ці моделі найчастіше застосовуються у реальному часі, включно із задачами контролю доступу — наприклад, коли необхідно виявити на зображенні обличчя та миттєво звірити його з базою даних [25].

Що стосується безпосередньо розпізнавання облич, яке є найбільш поширеним у системах контролю доступу, тут застосовуються як класичні, так і глибокі підходи. З одного боку, популярним залишається метод Хаара —

cascade classifiers, що використовуються, зокрема, в OpenCV для швидкої детекції облич [26]. З іншого — у більш точних і стійких до викривлень системах використовують глибокі згорткові моделі, натреновані на великих наборах фотографій, як-от FaceNet або DeepFace [27].

Ці моделі не просто "бачать" обличчя, а й переводять їх у векторні представлення у високовимірному просторі, де кожне обличчя кодується певним набором координат. Саме це дозволяє швидко знаходити збіги між новим зображенням і вже збереженим шаблоном, навіть у випадках часткових перекриттів, змін виразу обличчя або освітлення. Такий векторний підхід відкрив можливості для реалізації ефективного розпізнавання у великих базах, де працювати з пікселями напряму було б надто повільно та неточно.

Ще одним напрямом є використання гібридних моделей, які комбінують класичні алгоритми попередньої обробки з глибоким навчанням. Наприклад, попереднє вирівнювання або нормалізація зображень дозволяє значно покращити результат навіть без зміни самої нейронної моделі. У таких випадках підвищення точності досягається не лише за рахунок архітектури, а й завдяки грамотній підготовці даних та фільтрації шумів.

Не менш важливим є питання адаптивності систем. Розпізнавання об’єктів у реальних умовах потребує, щоб алгоритми могли адаптуватися до змін середовища — наприклад, до нового освітлення, кута зйомки чи зовнішнього вигляду людини. Сучасні підходи вирішують цю проблему за допомогою механізмів донавчання або fine-tuning, що дозволяє переналаштовувати вже натреновану модель під специфіку конкретного об'єкта або середовища без потреби у повному перенавчанні [28].

Слід окремо відзначити роль розширених датасетів, які є критичними для тренування точних моделей. У контексті контролю доступу найчастіше використовуються набори даних типу LFW (Labeled Faces in the Wild), VGGFace2 або CASIA-WebFace. Вони містять тисячі зображень реальних осіб у різноманітних умовах, що робить тренування більш наближеним до реальних сценаріїв [29].

Таким чином, методи розпізнавання об’єктів на сьогодні є багатогранною і стрімко розвиваючоюся галуззю. Вони поєднують у собі як класичні аналітичні підходи, так і сучасні нейромережеві алгоритми, здатні до самонавчання і глибокого узагальнення. У сфері контролю доступу це відкриває можливості для створення систем нового покоління, які забезпечують не лише безпеку, а й комфорт користувача, швидкість реагування та високу адаптивність.

1.4 Використання алгоритмів машинного навчання

У сучасній парадигмі комп’ютерного зору важко уявити ефективну систему обробки зображень без залучення алгоритмів машинного навчання. Їх роль давно вже не обмежується лише підтримкою або підсиленням класичних методів — натомість вони стали фундаментом, на якому базується більшість високоточних систем розпізнавання. Особливо це стосується задач, де людське око помиляється або де об’єкти не мають чітко визначеної структури, яка дозволяла б застосовувати детерміновані підходи.

Машинне навчання у контексті обробки зображень дає змогу створювати системи, що не просто виконують заздалегідь визначені правила, а й здатні адаптуватися до нових умов, виявляти закономірності та вчитися на прикладах. Йдеться не лише про класичну задачу класифікації — наприклад, визначення, чи є об’єктом на зображенні автомобіль чи людина, — а про повноцінну побудову моделей, що дозволяють виявляти нові класи, прогнозувати майбутні стани системи, а іноді — навіть пояснювати причини тих чи інших рішень.

Ключовою перевагою машинного навчання є можливість моделювання складних залежностей між вхідними піксельними даними та бажаними вихідними мітками. У випадку з візуальною інформацією, де сигнал багатовимірний, неоднорідний і часто зашумлений, це дозволяє перейти від евристичних підходів до статистично обґрунтованих. Уже базові моделі на

зразок логістичної регресії або методів опорних векторів продемонстрували здатність до побудови ефективних класифікаторів при достатньому обсязі навчальної вибірки [30]. Проте саме з появою нейромереж ситуація принципово змінилась.

Глибокі нейронні мережі стали основою потужних моделей, здатних до складного представлення ознак. У контексті зображень особливо важливими виявилися згорткові нейронні мережі, які завдяки своїй локальній структурі ефективно опрацьовують просторову інформацію. Вони автоматично виокремлюють релевантні ознаки, які раніше потребували ручного налаштування: краї, кути, текстури, а згодом — навіть більш абстрактні патерни, як-от обличчя, очі чи номерні знаки [31].

Алгоритми глибокого навчання дозволили будувати моделі, що обробляють зображення не лише на рівні фіксації наявності об’єкта, а й з урахуванням контексту. Наприклад, в системах контролю доступу це означає не просто розпізнавання обличчя, а й розуміння ситуації: чи є ця поява типовою для конкретного часу доби, чи не відхиляється маршрут особи від звичного, чи не змінено її вигляд суттєво. Такі підходи формують базу для поведінкової аналітики, що суттєво підвищує рівень безпеки [32].

Окремим класом алгоритмів, що набув популярності останнім часом, є трансформери — архітектури, які раніше застосовувались переважно в обробці природної мови, але згодом були адаптовані й до зображень. Наприклад, модель Vision Transformer (ViT) показала конкурентні результати у порівнянні з класичними CNN, особливо на великих датасетах [33]. Перевагою трансформерів є здатність враховувати глобальні залежності між різними частинами зображення, що може бути важливо в задачах, де локальні патерни недостатні.

У практичних реалізаціях систем машинного навчання важливу роль відіграє не лише архітектура, а й методи навчання. Зокрема, широкого застосування набуло попереднє навчання моделей на великих узагальнених наборах даних, після чого здійснюється донавчання на вузькій вибірці, що

відповідає специфіці конкретного завдання. Такий підхід, відомий як transfer learning, дозволяє уникнути потреби у великих обсягах локально зібраних даних і значно прискорює розгортання систем [34].

В системах розпізнавання, зокрема біометричних, усе частіше застосовуються методи навчання без учителя, що дозволяє системі самостійно виявляти структуру вхідних даних, розділяти кластери, будувати гіпотези щодо належності до певних груп. Такі підходи є особливо корисними в умовах, коли дані не мають чітких міток або коли необхідно виявляти нові, раніше невідомі класи користувачів чи ситуацій [35].

Слід також згадати про методи підкріплювального навчання, які в системах безпеки можуть застосовуватись для оптимізації реакції на різні типи подій. Наприклад, система може навчитися адаптувати рівень чутливості до облич у залежності від часу доби, кількості подій у кадрі або активності навколишнього середовища. Це дозволяє не лише точніше працювати в складних умовах, а й зменшити кількість помилкових спрацьовувань [36].

Найбільш перспективним напрямом виглядає комбінація кількох стратегій навчання: використання як супервізованих, так і несупервізованих підходів, розширення можливостей системи через генеративні моделі на кшталт GAN (Generative Adversarial Networks), які дозволяють генерувати нові зразки об’єктів і таким чином збагачувати вибірку для тренування [37]. Це особливо актуально у випадках, коли кількість реальних прикладів недостатня або коли важливо протестувати систему на штучно змодельованих сценаріях.

Основні типи алгоритмів машинного навчання в обробці зображень приведено в табл.1.1.

Машинне навчання також забезпечує більш гнучкий підхід до оцінки результатів: замість жорстких порогів система може застосовувати ймовірнісні моделі прийняття рішень, що знижує ризик помилок другого роду. У контексті контролю доступу це означає можливість прийняття зважених рішень — наприклад, виклик охорони при сумнівному збігу облич, а не автоматичне надання доступу.

Таблиця 1.1 – Основні типи алгоритмів машинного навчання в обробці зображень

Тип алгоритму	Основна функція	Переваги	Приклади застосування
Логістична регресія, SVM	Базова класифікація	Простота реалізації, швидке навчання	Розпізнавання простих об’єктів
Згорткові нейронні мережі (CNN)	Автоматичне виділення ознак та класифікація	Висока точність, ефективна робота з просторовими даними	Обличчя, предмети, текстури
Трансформери (ViT)	Моделювання глобальних залежностей у зображеннях	Потужні при великому обсязі даних, здатність враховувати контекст	Сцени, складні взаємодії об'єктів
Transfer Learning	Перенесення знань з однієї задачі на іншу	Економія ресурсів, менша потреба в локальних даних	Тонке налаштування під конкретну задачу
Навчання без учителя	Виявлення прихованих структур без міток	Можливість кластеризації нових або невідомих класів	Біометрія, аналіз аномалій
Підкріплювальне навчання	Адаптивна реакція системи на зміну умов	Навчання на основі досвіду, оптимізація поведінки	Реакція на події, зміна параметрів безпеки
Генеративні моделі (GAN)	Генерація нових зразків на основі наявних	Збагачення даних, симуляція рідкісних або критичних сценаріїв	Доповнення датасетів, моделювання нових ситуацій

Таким чином, алгоритми машинного навчання сьогодні є не просто інструментами, що підвищують точність. Вони формують нову філософію роботи з візуальною інформацією — гнучку, адаптивну, здатну до самонавчання і масштабування. В умовах цифрової трансформації такі системи стають критичним компонентом не лише безпеки, а й загальної інфраструктури взаємодії людини з інформаційним середовищем.

2 ПОСТАНОВКА ЗАДАЧІ

2.1 Мета і функціональні вимоги

Розробка програмного забезпечення для обробки зображень у задачах розпізнавання об’єктів і контролю доступу передбачає чітке розуміння цілей, які система повинна досягти, а також функціональних обмежень, що визначають її поведінку в умовах реального середовища. У центрі уваги такого проєкту перебуває необхідність створення інструменту, який дозволяє не лише фіксувати появу об'єкта у кадрі, а й здійснювати повноцінну ідентифікацію особи, приймати рішення щодо надання доступу та забезпечувати стабільну роботу за умов зовнішніх впливів і змін.

Основною метою розробки є створення адаптивної, ефективної та надійної системи, здатної обробляти зображення в реальному часі, виявляти цільові об’єкти, здійснювати їхню візуальну ідентифікацію та формувати на основі цього управлінські рішення. В межах поставленого завдання важливо забезпечити не лише точність і швидкість розпізнавання, а й зручність взаємодії з користувачем, а також простоту інтеграції системи в існуючу інфраструктуру.

Підвищені вимоги до безпеки вимагають, щоб система могла працювати стабільно навіть за умови неповних, зашумлених або спотворених вхідних даних. Тому функціональні характеристики повинні охоплювати можливість попередньої обробки зображення, включаючи фільтрацію шуму, нормалізацію яскравості, вирівнювання геометрії та виділення основних ознак. Система повинна мати здатність працювати в умовах різного освітлення, зображень з рухом, частковим перекриттям об’єктів або варіативною якістю зйомки.

Особлива увага приділяється модулю розпізнавання, який є серцевиною проєкту. Він має забезпечувати точне виділення об'єкта з фону, ідентифікацію за ознаками, збереження результатів у відповідній базі даних, а також

механізм порівняння нових зразків із вже відомими. Таким чином, система повинна не лише виявити наявність людини, а й перевірити її відповідність дозволеному доступу, використовуючи заздалегідь зареєстровану інформацію.

Користувацький інтерфейс при цьому виконує роль не просто панелі керування, а зручного інструмента для спостереження, налаштування і втручання у разі потреби. Його завдання — надавати зворотний зв’язок у зрозумілій формі, демонструвати результати розпізнавання, вести лог подій і забезпечувати зміну режимів роботи без необхідності втручання у код чи конфігураційні файли. Це особливо актуально в умовах, де оператором системи може бути людина без технічної підготовки.

Ключовим також є модуль збереження та обробки історичних даних. Йдеться не лише про логування подій, а й про можливість аналітики — зокрема, виявлення частоти появи окремих осіб, фіксацію спроб несанкціонованого доступу, оцінку навантаження на систему в пікові години. Наявність такої статистики дозволяє не лише оцінювати ефективність роботи, а й приймати управлінські рішення на основі зібраної інформації.

У цілому система повинна поєднувати точність алгоритмів розпізнавання з надійністю інфраструктури, бути готовою до масштабування, змін конфігурації, а також до роботи у реальному середовищі з урахуванням факторів, які важко передбачити на етапі проєктування. Тільки така гнучкість дозволяє говорити про практичну придатність розробки та її потенціал для подальшого впровадження у різних сферах — від офісної безпеки до автоматизованого контролю доступу в закритих виробничих зонах або об’єктах критичної інфраструктури.

2.2 Структура вхідних даних

У процесі реалізації системи розпізнавання об’єктів критично важливим є розуміння того, з яким саме типом вхідної інформації вона працює. Саме

структура вхідних даних (табл. 2.1) визначає не лише архітектуру всього програмного забезпечення, а й специфіку роботи окремих модулів — від попередньої обробки до прийняття рішень на основі аналізу. Від якості, цілісності та послідовності цих даних значною мірою залежить і точність самої системи, і її здатність адекватно реагувати на змінні зовнішні умови.

Таблиця 2.1 – Компоненти структури вхідних даних у системах розпізнавання

Тип вхідних даних	Формат / приклад	Призначення	Особливості / Примітки
Зображення	Статичні (JPEG, PNG), кадри з відео (RGB)	Основна візуальна інформація для аналізу	Може бути кольоровим, ч/б, ІЧ-зображенням; фіксує об’єкт на момент зйомки
Відеопотік	Потік кадрів (RTSP, MJPEG, MP4)	Відстеження руху, аналіз поведінки, побудова динамічної моделі	Дозволяє враховувати контекст і часові залежності
Метадані	Координати, час, серійні номери, освітлення	Покращення точності, адаптація до зовнішніх умов	Сприяє фільтрації помилкових зображень, підвищує надійність
Шаблони / еталони	Ознакові вектори, зменшені зображення, embeddings	База для порівняння під час ідентифікації	Зберігаються локально або на сервері; критично важливі для точності
Сигнали валідації	Значення якості кадру, рівень шуму	Визначення придатності даних до обробки	Виявляють спотворення, відкидають непридатні для аналізу зображення
Режим роботи	Постійний / реактивний / подієвий	Контроль джерела даних і частоти оновлення	Визначає, як і коли система збирає та обробляє інформацію

Базовим джерелом інформації в таких системах зазвичай є відеопотік або окремі кадри, отримані з цифрових камер. Вони можуть надходити у вигляді послідовностей зображень або транслюватись у режимі реального часу. Формат таких даних часто представлений у вигляді матриць пікселів, кожен з яких має свою яскравість і колірне значення. Найпоширенішими є триканальні зображення у форматі RGB, хоча в окремих випадках, коли йдеться про специфічні задачі, застосовуються й інші формати — наприклад, інфрачервоне зображення, чорно-білі відбитки або теплові карти.

Найпростішим і найбільш розповсюдженим випадком є робота з окремими статичними зображеннями. Такі дані дозволяють проводити обробку з меншими обчислювальними витратами, адже немає потреби аналізувати зміну об’єктів у часі. Проте в системах контролю доступу переважно використовують саме відеопотоки, які дають змогу відстежувати появу об’єкта, його рух, швидкість наближення, а також фіксувати часову мітку події. Це дозволяє формувати більш повну картину поведінки користувача та забезпечити більшу гнучкість системи.

Крім візуальної інформації, структура вхідних даних часто включає супровідні метадані. Це можуть бути координати фіксації об’єкта в кадрі, часові мітки, серійні номери пристроїв, а також параметри середовища — рівень освітлення, температура, відстань до об'єкта. Всі ці додаткові відомості не є обов’язковими для базової роботи алгоритму розпізнавання, проте значно покращують результати аналізу, дозволяють коригувати параметри обробки та адаптувати систему до зовнішніх змін.

Ще одним важливим компонентом вхідних даних є попередньо збережені шаблони — еталонні зображення, які використовуються для порівняння під час ідентифікації. Це може бути набір фотографій облич зареєстрованих користувачів, векторні представлення біометричних ознак або інші дескриптори, отримані на етапі навчання системи. У багатьох випадках ці шаблони зберігаються у вигляді масивів ознак або навіть зменшених,

нормалізованих зображень, які використовуються як база для обчислення відстані між новим зразком і збереженим профілем.

Структура вхідних даних повинна також враховувати можливість роботи з неповною або спотвореною інформацією. У реальних умовах камера може зафіксувати лише частину обличчя, або зображення може бути зашумленим, розмитим, мати низький рівень контрасту. Саме тому на рівні початкового аналізу дані часто проходять попередню перевірку на валідність: система повинна розуміти, чи є зображення придатним для обробки, чи варто його відкинути або спробувати покращити за допомогою фільтрів.

Варто також зазначити, що вхідні дані не є статичними — вони формуються динамічно в процесі роботи системи. У залежності від обраного режиму, система може працювати в постійному режимі моніторингу, реагувати на рух, активуватись лише у разі виявлення об’єкта. Це означає, що структура даних повинна бути гнучкою і здатною до обробки як безперервних потоків, так і окремих знімків, адаптуючись до задачі.

Таким чином, структура вхідних даних у задачах розпізнавання об’єктів і контролю доступу є багаторівневою і включає не лише самі зображення, а й супровідні дані, шаблони, часові мітки та додаткові контекстуальні параметри. Вона повинна забезпечувати баланс між точністю, швидкістю обробки та адаптивністю, дозволяючи системі реагувати на широкий спектр ситуацій у реальному середовищі.

2.3 Умови експлуатації і технічні обмеження

Розробка будь-якого прикладного програмного забезпечення, що працює з обробкою зображень у реальному часі, завжди тісно пов’язана з конкретними умовами, у яких система буде експлуатуватись. Ці умови визначають не лише технічні параметри реалізації, а й формують межі, в яких система здатна функціонувати без суттєвих втрат у якості чи швидкодії. Тому ще на етапі постановки задачі необхідно чітко окреслити фактори зовнішнього

середовища, що впливають на стабільність роботи, а також ті технічні обмеження, які накладаються на систему з боку апаратного забезпечення, типу розгортання чи ресурсоємності алгоритмів.

Основним викликом у більшості сценаріїв є варіативність умов освітлення. Камери спостереження, які використовуються як джерело вхідних даних, часто працюють у динамічному середовищі: протягом доби змінюється яскравість, з’являються тіні, відблиски, може спостерігатися контрове світло або затемнення. Усі ці фактори істотно впливають на якість кадру і, відповідно, на результати аналізу. Якщо обличчя чи об’єкт з’являється у кадрі при надто сильному або слабкому освітленні, система може не виявити ключові ознаки або видати помилкове спрацьовування. Саме тому при розробці слід враховувати можливість автоматичної корекції зображення або використовувати попередню нормалізацію яскравості перед передачею кадру до модуля розпізнавання.

Ще одним важливим обмеженням є роздільна здатність камери. В задачах біометричної ідентифікації розмір зображення напряму впливає на кількість доступної інформації. Якщо зображення низької якості, система може не зафіксувати дрібні ознаки, які критичні для розпізнавання — наприклад, розташування очей, форму носа або контури губ. Проте використання надто великої роздільної здатності тягне за собою збільшення навантаження на процесор або графічний модуль, а також підвищує вимоги до обсягу оперативної пам’яті. Тому тут завжди потрібно шукати баланс: зображення має бути достатньо інформативним, але не настільки деталізованим, щоб система втрачала здатність працювати в режимі реального часу.

У разі використання відео з потоковою передачею даних зростає значення пропускної здатності каналу зв’язку. Якщо система передає відео на сервер для обробки, навіть короткі затримки у мережі можуть призвести до пропускання кадрів або втрати актуальності даних. У таких випадках необхідно передбачати буферизацію потоку або ж часткову обробку

безпосередньо на боці клієнта — без постійного звернення до сервера. Це вимагає або потужнішого обладнання на кінцевій точці, або оптимізації коду на рівні алгоритмів. І те, й інше має бути закладено на етапі проектування архітектури.

Нерідко до експлуатації системи висуваються також вимоги щодо енергоспоживання. У випадку встановлення обладнання у місцях без постійного доступу до мережі — наприклад, у віддалених зонах, на вулиці або у транспортних засобах — важливо, щоб система могла працювати у режимі економії енергії. Тут доречно розглядати сценарії активації лише при появі руху, обмеження частоти кадрів або використання асинхронної обробки.

В задачах контролю доступу критичним стає й аспект фізичної надійності обладнання. Камера повинна витримувати температурні коливання, вологість, механічні навантаження, а іноді й навмисне втручання з боку зловмисників. Тому у виборі апаратної частини доводиться враховувати не лише технічні характеристики, а й клас захисту корпусу, термін експлуатації, наявність систем охолодження або герметизації.

На рівні програмного забезпечення обмеження можуть виникати через складність алгоритмів. Якщо система використовує моделі глибокого навчання, важливо переконатись у достатній продуктивності процесора або графічного прискорювача. Існують варіанти розгортання з використанням полегшених моделей, які оптимізовані для мобільних або вбудованих систем, проте це зазвичай супроводжується зменшенням точності. Тому в кожному конкретному випадку необхідно приймати рішення, виходячи з пріоритетів: швидкодія, точність або ресурсозбереження.

Особливо складною може бути ситуація, коли система повинна працювати в умовах високого трафіку — наприклад, при вході в будівлю під час зміни. У таких ситуаціях обробка кількох осіб одночасно вимагає мультипоточності, ефективного розподілу навантаження, а іноді — навіть розділення аналізу між кількома пристроями. Усе це формує вимоги до масштабованості і модульності системи.

Загалом умови експлуатації й технічні обмеження задають рамки, в яких проєкт повинен залишатись життєздатним. Ігнорування навіть одного з перелічених факторів здатне суттєво знизити ефективність або повністю нівелювати результат роботи системи. Саме тому правильне моделювання середовища експлуатації та чітке розуміння його обмежень є невід’ємною частиною будь-якої сучасної розробки, що претендує на застосування у реальному світі.

3 АНАЛІЗ АЛГОРИТМІВ ОБРОБКИ ЗОБРАЖЕНЬ

3.1 Детекція облич

В системах розпізнавання об’єктів, що функціонують у режимі реального часу, саме детекція облич займає ключове місце. Вона є первинним етапом, без якого неможливо здійснити подальшу ідентифікацію особи чи прийняття рішень про надання доступу. Саме тому надійність, швидкість і точність цього модуля безпосередньо впливають на ефективність усього комплексу. Незалежно від того, йдеться про контроль доступу до фізичних приміщень, розпізнавання користувачів у цифрових сервісах або виявлення порушників в системах відеоспостереження — усе починається з вміння побачити обличчя.

Процес детекції (рис. 3.1) передбачає виявлення області зображення, яка потенційно містить обличчя, та визначення її координат в межах кадру. Це може бути виконано як у статичних зображеннях, так і в потоковому відео. Основна складність полягає в тому, що обличчя може з’являтися у різних масштабах, під різними кутами, частково перекритим, з нечіткими контурами або в умовах поганого освітлення. Система повинна не просто знайти його — вона повинна зробити це швидко, з мінімальною кількістю хибнопозитивних і хибнонегативних результатів.

Ранні алгоритми базувались на використанні фіксованих ознак, таких як форма очей, носа, розміщення рота відносно інших елементів. Для цього часто застосовували методи шаблонного співставлення або евристичні правила, що задавали жорсткі критерії розташування окремих елементів. Такі системи працювали стабільно лише за ідеальних умов і не витримували змін у позі, освітленні чи якості зйомки.

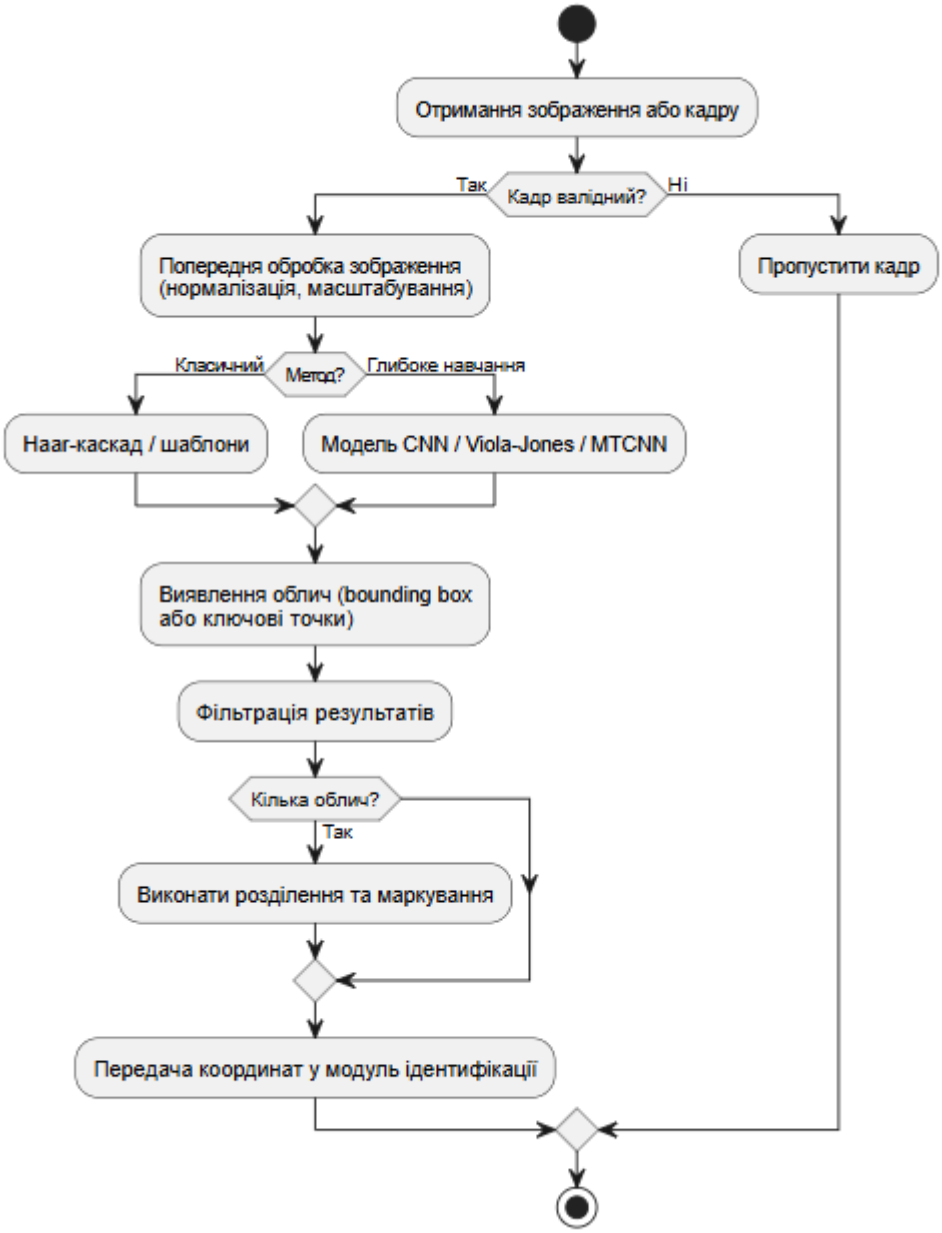

Рисунок 3.1 — Блок-схема алгоритму пошуку облич

Подальший розвиток привів до появи каскадних класифікаторів, які дозволили здійснювати детекцію облич з набагато вищою ефективністю. Особливо помітним був внесок методу на основі ознак Хаара, де за допомогою спеціально побудованих масивів ознак здійснюється проходження через серію простих, але ефективних фільтрів. У такій каскадній структурі перші етапи швидко відсіюють явно негативні приклади, а останні уточнюють межі обличчя. Це забезпечило баланс між швидкістю й точністю, що зробило

технологію придатною для вбудованих систем і використання в реальному часі.

Однак зі збільшенням вимог до точності та появою складніших умов зйомки ці методи поступово почали втрачати актуальність. Сучасні системи дедалі частіше покладаються на моделі глибокого навчання. Нейронні мережі, зокрема згорткові, навчені на великих датасетах облич у різних позах і умовах, здатні виявляти обличчя з високим рівнем точності навіть у випадках, коли класичні методи виявляються безсилими. Перевагою таких моделей є їх здатність до узагальнення — вони розпізнають обличчя не за жорсткими правилами, а за сукупністю ознак, які не завжди очевидні навіть людині.

У практичних реалізаціях використовується кілька підходів до детекції. Одні моделі працюють на основі пошуку bounding box — прямокутників, які обрамляють обличчя на зображенні. Інші — забезпечують ще й локалізацію ключових точок, таких як очі, ніс, кути рота, що дозволяє провести більш точне вирівнювання та нормалізацію перед подальшим аналізом. Деякі системи здатні детектувати обличчя з урахуванням тривимірної геометрії, що особливо корисно при роботі зі змінними кутами нахилу або зображеннями у профіль.

В системах контролю доступу особливо важлива не лише точність, а й швидкість. Обробка кожного кадру повинна здійснюватись у реальному часі — зазвичай це десятки мілісекунд на одне зображення. Тому у багатьох випадках використовуються оптимізовані версії нейронних мереж, що проходять попередню компресію або використовують полегшені архітектури. У поєднанні з апаратним прискоренням — наприклад, через GPU або спеціалізовані нейромодулі — це дозволяє досягати потрібної швидкодії без істотної втрати в якості результатів.

Окрему складність становить ситуація з кількома обличчями у кадрі. У місцях із великим потоком людей система повинна не просто виявити обличчя, а й відокремити їх одне від одного, уникнувши накладання координат або

плутанини в аналізі. Це потребує не лише високої точності в локалізації, а й стійкості алгоритму до зміщень, перекриттів і динамічних змін у кадрі.

З технічного погляду, модуль детекції повинен бути достатньо універсальним, щоб працювати як з відео, так і з окремими зображеннями. Його інтеграція в систему повинна бути побудована за принципом незалежного компонента з чітко визначеним інтерфейсом. Це забезпечує можливість його заміни або оновлення без необхідності переписування всієї програми. Такий підхід дозволяє розробникам вільно експериментувати з різними моделями, обираючи найоптимальніший варіант під конкретні умови експлуатації.

Таким чином, детекція облич є складним, але абсолютно необхідним компонентом сучасної системи обробки зображень. Її ефективна реалізація визначає здатність системи до швидкої, точної та надійної взаємодії з користувачем, а в багатьох випадках — і загальну доцільність використання технології в умовах реального часу.

3.2 Виявлення контурів

У системах обробки зображень контури відіграють надзвичайно важливу роль, адже саме вони визначають межі об’єктів, відділяють фігуру від фону та надають структурну інформацію про форму, розміри й просторове розташування. Виявлення контурів (рис. 3.2) — це, по суті, один із ключових етапів у побудові уявлення про вміст зображення, і без нього подальша інтерпретація, класифікація або порівняння втрачає точність і сенс.

Контури — це не просто лінії на зображенні. Вони — результат аналізу змін інтенсивності між сусідніми пікселями, що дозволяє визначити, де закінчується одна зона і починається інша. У реальності ці межі не завжди різкі: вони можуть бути розмитими, частково відсутніми або спотвореними шумами. Саме тому завдання виявлення контурів не зводиться до простого

пошуку границь — це складний процес, що поєднує фільтрацію, аналіз градієнтів і виділення релевантних областей.

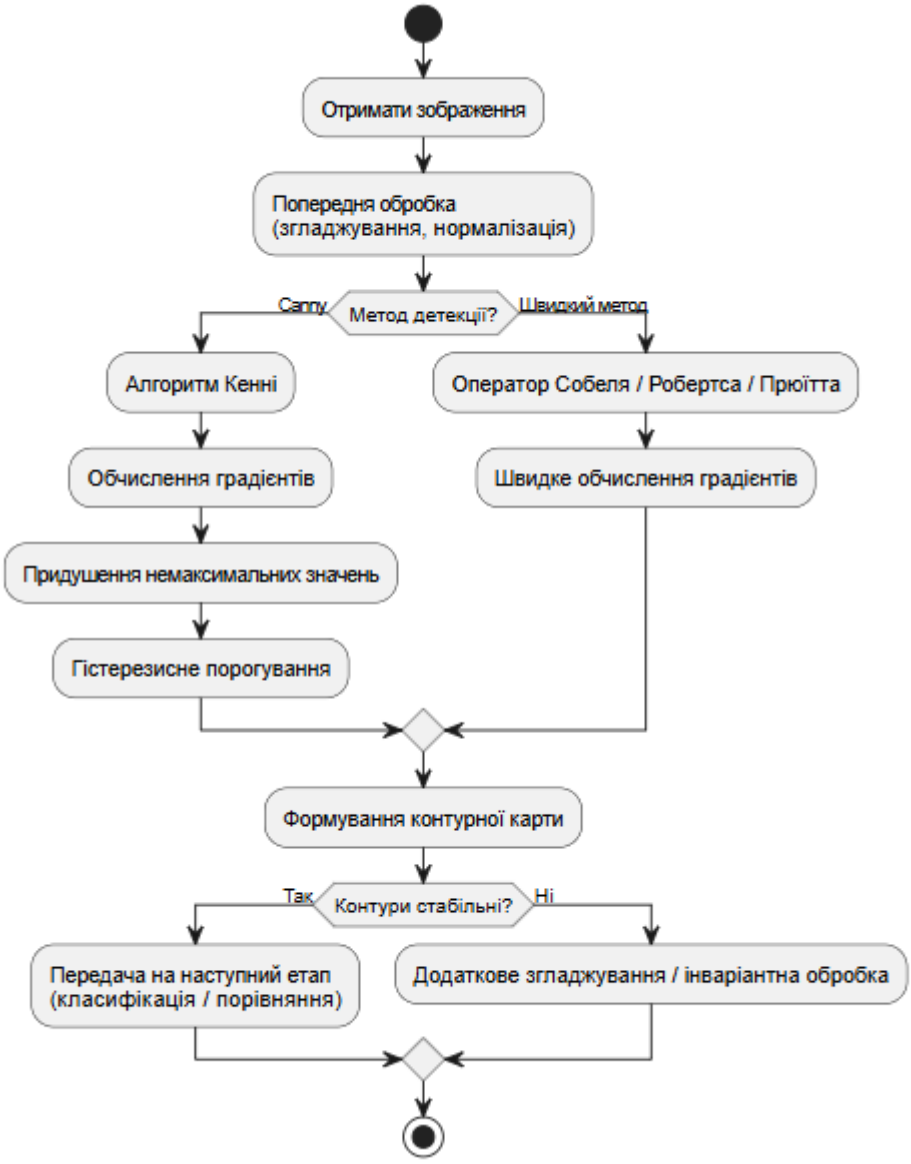

Рисунок 3.2 — Блок-схема алгоритму виділення контурів

У більшості випадків алгоритми контурного аналізу базуються на обчисленні градієнта яскравості — різниці значень пікселів у певному напрямку. Пікселі, де ця різниця перевищує певний поріг, вважаються такими, що належать до межі об’єкта. Проте встановлення самого порогу — завдання не з легких. Якщо поріг буде надто низьким — з’явиться багато зайвих контурів, спричинених шумом або дрібними деталями. Якщо занадто високим

— система може не виявити тонкі, але важливі межі. Тому одне з ключових рішень полягає в адаптивному підборі параметрів, залежно від контексту та характеру зображення.

Одним із найпоширеніших методів виявлення контурів є алгоритм Кенні. Він складається з кількох етапів: спочатку відбувається згладжування зображення для зменшення шумів, далі — обчислення градієнтів у горизонтальному й вертикальному напрямках, потім — підсилення меж, а наприкінці — так звана процедура придушення немаксимальних значень і гістерезисний поріг. Попри складність, цей підхід дає чіткі, з’єднані контури, які добре підходять для подальшої обробки або класифікації. Він став своєрідним еталоном для систем, що працюють зі складними, реалістичними зображеннями, особливо коли важливо зберегти деталі.

Втім, не завжди доречна така глибока обробка. У випадках, де важлива швидкість, використовуються простіші оператори — зокрема, детектори Собеля, Прюїтта або Робертса. Вони менш точні, можуть створювати артефакти на межах, проте працюють значно швидше і менш вибагливі до ресурсів. У реальних умовах часто поєднують кілька алгоритмів: на попередньому етапі — швидкий метод для грубої локалізації, а вже далі — точніший аналіз із використанням більш обчислювально складних підходів.

В задачах розпізнавання облич або об’єктів у складному фоні контурна інформація часто слугує допоміжним сигналом для фільтрації зони інтересу. Наприклад, при виявленні облич спершу може бути знайдена контурна структура голови або плечей, що дозволяє відсікти зайві зони і сфокусувати аналіз на релевантній ділянці кадру. Також контури можуть використовуватись для нормалізації зображення — наприклад, для вирівнювання, повороту або вирізання об’єкта з фону.

Не менш важливим є й те, як система поводиться з контурною інформацією у складних умовах. У разі високого рівня шуму, неоднорідного освітлення або низької контрастності зображення виділення меж стає нетривіальним завданням. Саме тут стає в пригоді попередня обробка:

застосування гаусівського згладжування, локального контрастування, гістограмної вирівнювання. Ці кроки, хоч і не є частиною самого алгоритму виявлення контурів, істотно впливають на кінцевий результат.

Ще один важливий аспект — стабільність контурів при зміні масштабу або кута огляду. Якщо система буде реагувати на дрібні зміни занадто чутливо, це призведе до накопичення помилок у подальшому аналізі. Тому доцільним є застосування методів, які враховують інваріантність до обертання, масштабування та афінних перетворень. Це забезпечує стійкість моделі до реальних умов, де користувач може з’являтися під різними кутами або на різній відстані від камери.

Зрештою, виявлення контурів — це не лише інструмент для візуалізації або підготовки до сегментації. Це логічна опора, на якій тримається вся інтерпретація зображення. Усе, що система «розуміє» про форму, межі, частини об’єкта, — базується на тих чи інших ознаках, витягнутих саме з контурів. І хоча сучасні нейронні мережі часто вважають за краще працювати з усім зображенням одразу, навіть у їхній внутрішній логіці можна простежити процес аналізу просторових змін, який фактично зводиться до контурного мислення — тільки вже на рівні фільтрів і шарів.

Таким чином, незалежно від загального підходу, детальність і коректність виділення контурів прямо впливають на якість подальшого розпізнавання. В системах контролю доступу, де кожна деталь має значення, саме вони формують той базис, що дозволяє побудувати точну, гнучку і надійну логіку ідентифікації.

3.3 Сегментація зображень

Сегментація (рис. 3.3) є одним з базових етапів аналізу візуальної інформації, без якого повноцінна інтерпретація зображення практично неможлива. В її основі є ідея про те, що будь-яке зображення складається з кількох логічних частин, кожна з яких має певну структурну або смислову

цілісність. Завдання сегментації полягає в тому, щоб виділити ці частини — тобто розбити зображення на області, які відповідають окремим об’єктам або їх частинам.

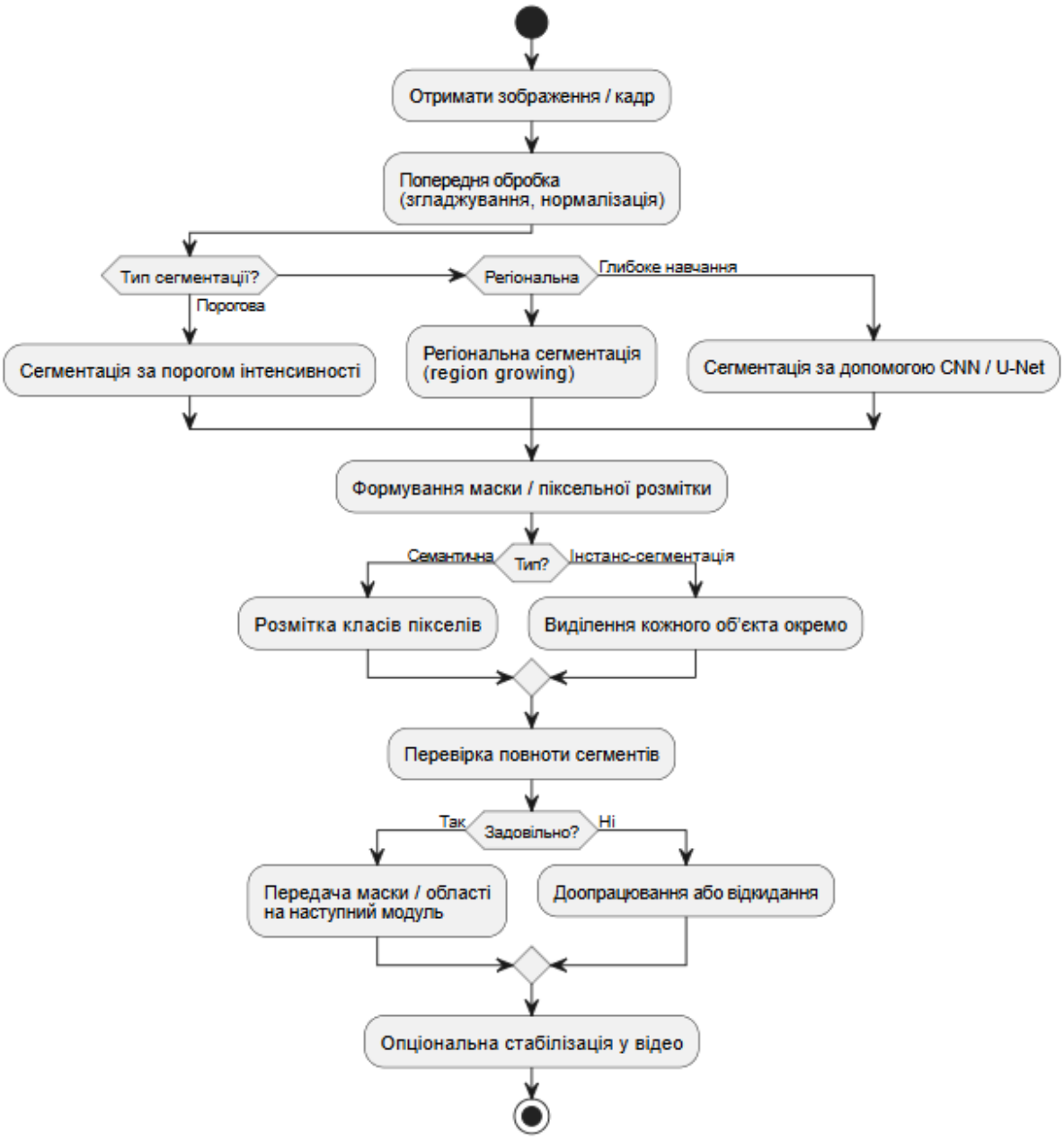

Рисунок 3.3 — Блок-схема сегментації зображення

У контексті систем розпізнавання об’єктів та контролю доступу сегментація виконує не допоміжну роль, а часто є необхідною умовою для подальших етапів. Вона дозволяє виділити об’єкт з фону, скоригувати межі,

ізолювати цільову ділянку кадру та сфокусувати обчислювальні ресурси на аналізі саме релевантної частини зображення. У разі, якщо мова йде про багатолюдні сцени або складне тло, без попередньої сегментації система ризикує або помилитись, або витратити надто багато часу на аналіз непотрібних фрагментів.

З технічної точки зору сегментація може здійснюватися за різними критеріями — кольором, текстурою, яскравістю, геометричними характеристиками або поєднанням цих ознак. У найпростіших випадках можна обмежитись пороговою сегментацією, коли пікселі розподіляються між класами за пороговим значенням інтенсивності. Такий підхід, хоч і швидкий, є надзвичайно чутливим до змін освітлення та не підходить для складних або кольорових сцен.

Більш гнучким є регіональний підхід, коли сегменти будуються навколо пікселів із подібними характеристиками. У цьому випадку алгоритм може починатись з невеликого ядра — умовної точки в центрі об’єкта — і поступово розширювати область доти, доки сусідні пікселі відповідають певним умовам. Цей метод добре себе показує при сегментації об’єктів із чіткою, однорідною структурою, але може втратити точність на межах або в неоднорідному фоні.

У сучасних системах сегментації дедалі частіше застосовуються моделі машинного навчання. Зокрема, згорткові нейронні мережі дозволяють проводити як семантичну сегментацію — тобто поділ зображення на класи пікселів, — так і інстанс-сегментацію, де кожен окремий об’єкт виділяється індивідуально, навіть якщо він належить до того самого класу. Це дає можливість одночасно працювати з кількома особами в кадрі, точно виділяти їх контури, ідентифікувати, зберігати інформацію про кожного окремо.

Важливо розуміти, що сегментація — це не лише етап підготовки. Вона визначає якість усього подальшого аналізу. Неправильно окреслений контур, частково втрачені області чи помилкове об’єднання фону з об’єктом — усе це призводить до некоректного результату розпізнавання. У задачах контролю

доступу це може означати або блокування законного користувача, або навпаки — відкриття дверей для сторонньої особи.

Сегментація також використовується для підвищення точності в умовах динамічної сцени. Наприклад, якщо людина з’являється у кадрі лише частково або в русі, попереднє сегментування дозволяє зосередитись на головній області — скажімо, обличчі — та не враховувати рухи рук, зміну фону чи інші зовнішні фактори. Це значно знижує навантаження на систему й дозволяє фокусуватись на справді важливих ознаках.

Ще одним важливим аспектом є стабільність сегментації в часовому вимірі. Якщо обробка відбувається у режимі відео, важливо, щоб обраний об’єкт залишався послідовно сегментованим від кадру до кадру, без раптових стрибків чи втрати області. Це дозволяє підтримувати логіку простеження об’єкта в часі, що особливо актуально в системах, де потрібно аналізувати не лише факт появи, а й поведінку.

Таким чином, сегментація — це не допоміжна деталь, а фундаментальний інструмент, що визначає, як саме система сприймає візуальний простір. Її точність, адаптивність та здатність до узагальнення безпосередньо впливають на якість і достовірність рішень, які приймаються на основі зображення. В межах розробки інтелектуальних систем безпеки цей етап є незамінним — ігнорування його особливостей неминуче призведе до втрат у якості всієї системи.

3.4 Побудова ознак

У будь-якій системі комп’ютерного зору, що працює з розпізнаванням об’єктів, побудова ознак (рис.3.4) є тією ланкою, яка з’єднує «сире» зображення з математичною моделлю, здатною приймати рішення. На рівні вхідних даних зображення являє собою масив пікселів, кожен з яких має значення яскравості або кольору. У такому вигляді ці дані не мають для машини жодної очевидної структури, яку можна було б безпосередньо

використати для класифікації або порівняння. Тому виникає потреба перетворити цей масив у набір характеристик — ознак, які містять ключову інформацію про форму, структуру, текстуру або інші властивості об’єкта.

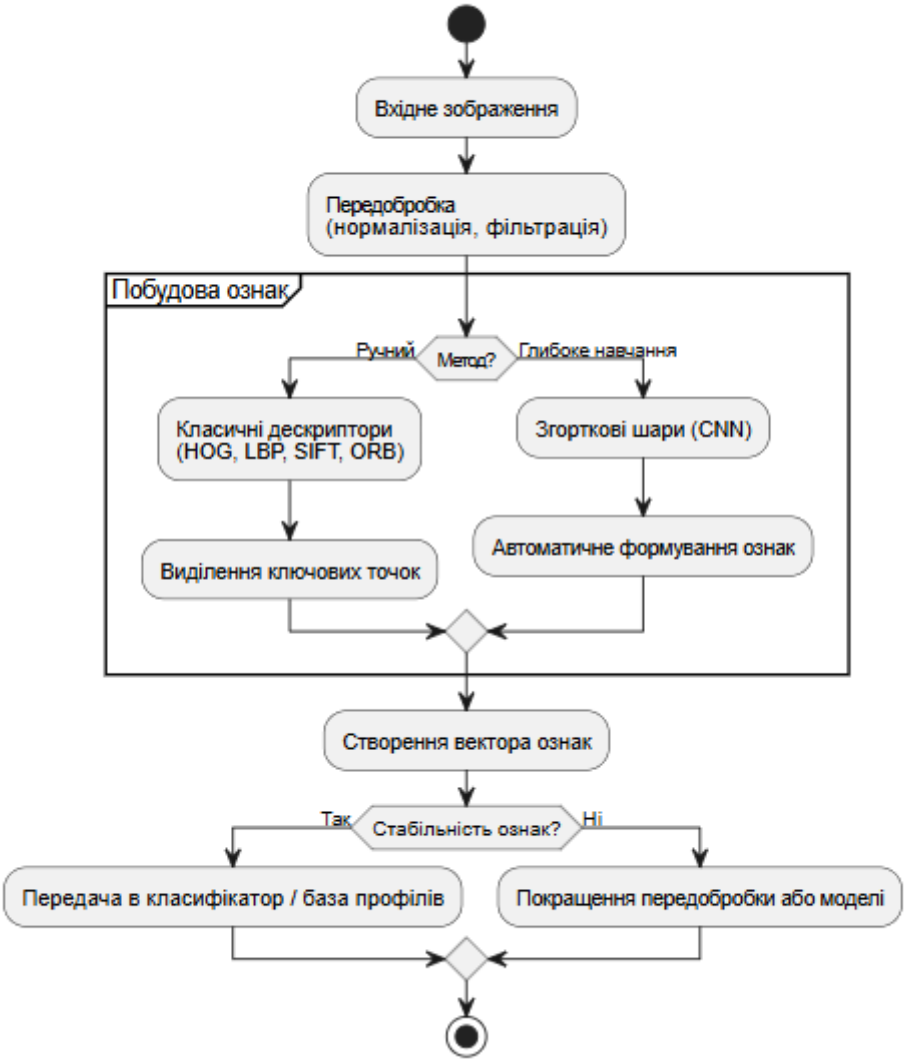

Рисунок 3.4 — Блок-схема побудови ознак

Процес побудови ознак — це спроба витягнути з хаосу пікселів ті параметри, які найкраще описують об’єкт, і при цьому є стійкими до змін, що неминуче виникають у реальному середовищі. Йдеться про зміну масштабу, освітлення, положення в кадрі, часткове перекриття, шум або артефакти зйомки. Хороша ознака повинна залишатися релевантною незалежно від цих

змін, забезпечуючи однакову реакцію системи при повторній появі того самого об’єкта.

Початково побудова ознак виконувалась вручну. Розробники визначали, які саме параметри можуть бути інформативними — наприклад, відношення ширини до висоти, кількість контурів, напрямки градієнтів, інтенсивність у певних ділянках кадру. Такі ознаки могли бути простими, як гістограма яскравості, або складними, як просторові патерни в текстурі. Їхня ефективність залежала від конкретного контексту і часто потребувала ретельного налаштування під кожну задачу.

Згодом з’явилися універсальні дескриптори, які дозволили автоматизувати побудову ознак. Серед них варто згадати, зокрема, гістограми напрямків градієнтів, які описують орієнтацію контурів у зображенні; локальні бінарні шаблони, що враховують відносну яскравість у локальних областях; а також більш комплексні підходи на кшталт SIFT чи ORB, що виявляють ключові точки та створюють їх векторне представлення. Усі ці методи дозволяють створити компактне, але інформативне подання зображення у вигляді набору чисел, які зручно порівнювати між собою.

Однак справжній прорив стався тоді, коли побудова ознак перестала бути окремим ручним етапом, а стала інтегрованою частиною навчання моделі. З появою глибоких нейронних мереж цей процес фактично автоматизувався. Мережа самостійно навчається будувати ті ознаки, які є найбільш релевантними для поставленої задачі. Це відбувається в шарі згорток, де фільтри, що формуються під час навчання, починають реагувати на характерні патерни — кути, текстури, форми — не тому, що їм це «вказали», а тому що це статистично найефективніші характеристики для розпізнавання.

Таке «вивчення ознак» дало змогу значно підвищити гнучкість і точність систем, особливо в задачах, де неможливо наперед передбачити, які саме ознаки будуть найкориснішими. Наприклад, у задачі розпізнавання облич в умовах варіативного освітлення або під різними кутами, класичні дескриптори

можуть втратити свою ефективність, тоді як мережа навчиться враховувати саме ті особливості, що залишаються стабільними.

Проте навіть при використанні глибоких моделей побудова ознак залишається окремою, критично важливою частиною системи. Навіть якщо цей процес не виконується вручну, він все одно визначає, які саме дані передаються на наступний рівень аналізу. Часто саме від вибору способу побудови ознак залежить, наскільки система буде здатною розрізняти подібні об'єкти, виявляти відмінності між класами чи працювати зі складними, неоднозначними випадками.

Окреме значення має також узгодженість ознак. У системі, що працює в реальному часі, кожна нова порція даних повинна бути порівняна з еталоном — будь то вектор обличчя, силует чи інший дескриптор. Якщо ознаки, побудовані на основі різних кадрів або в різні моменти часу, не є стабільними, система не зможе коректно порівнювати їх. Тому важливо забезпечити повторюваність результату: об’єкт, зафіксований вдруге, повинен породжувати такі ж або дуже подібні ознаки, як і вперше.

У підсумку побудова ознак — це не просто технічний етап. Це фундамент, на якому базується уся логіка розпізнавання. Незалежно від того, чи йдеться про традиційні дескриптори, чи про вектори з нейронних мереж, завдання залишається тим самим: зменшити складність вхідних даних, залишивши лише найсуттєвіше. І саме те, що система зуміє зберегти у вигляді ознак, визначає її здатність «побачити» і «зрозуміти» об’єкт на зображенні.

3.5 Класифікація об’єктів

Коли всі попередні етапи пройдено — зображення отримано, об’єкт детектовано, контури виділено, а ознаки сформовано — система підходить до найвідповідальнішого моменту: прийняття рішення. Саме на цьому етапі відбувається класифікація, тобто визначення, до якого класу належить виявлений об’єкт. У контексті систем контролю доступу це означає, що

система має відповісти на запитання: ця людина дозволена чи ні? Чи належить це обличчя до зареєстрованих? Чи відповідає об’єкт умовам допуску? Від цієї відповіді залежить увесь подальший сценарій — буде відкрито двері, зафіксовано спрацювання чи надіслано сигнал тривоги.

Класифікація візуальних об’єктів (рис. 3.5) — це складне завдання, яке не зводиться до банального порівняння двох картинок. Система повинна враховувати варіації в межах одного класу: один і той самий користувач може виглядати дещо інакше щодня, мати різну міміку, носити головний убір, окуляри або просто стояти під іншим кутом. Водночас вона повинна бути достатньо вибірковою, щоб не допустити помилкового збігу, коли інша особа випадково видається схожою на зареєстровану.

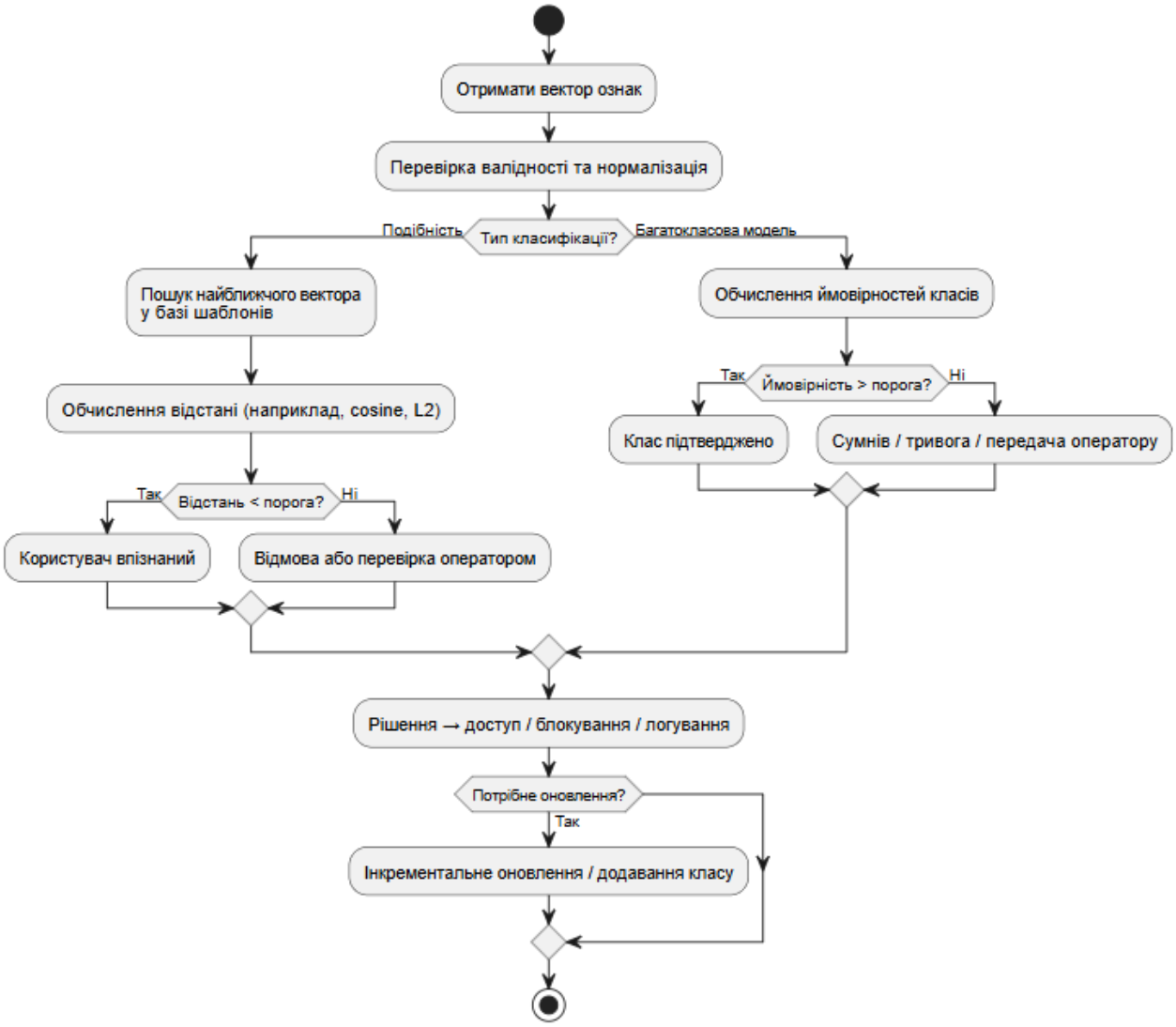

Рисунок 3.5 — Блок-схема алгоритму класифікації об’єктів

У традиційних системах класифікація виконувалась за допомогою евристичних правил або простих статистичних моделей — наприклад, шляхом порівняння збережених шаблонів із поточним зображенням. Проте ці підходи швидко показали свою вразливість у реальному світі, де умови зйомки нестабільні, а об’єкти часто не мають чітко виражених, однозначних ознак. Тому розвиток машинного навчання став переломним моментом, який дозволив перейти до більш гнучких і точних рішень.

Найбільший ефект дало застосування моделей класифікації, натренованих на великих вибірках. У таких моделях кожен об’єкт представляється не як зображення, а як точка у багатовимірному просторі ознак. І вже відстань між цими точками визначає, наскільки два об’єкти подібні або відмінні. Якщо модель навчена добре, вона буде класифікувати об’єкти на основі цих векторних уявлень із високою точністю, навіть якщо у вхідних даних присутні спотворення або шум.

Найчастіше в системах, що працюють із біометрією, класифікація зводиться до пошуку найближчого сусіда у векторному просторі. Це так званий підхід на основі подібності, де кожен користувач має заздалегідь збережений вектор, а нове зображення порівнюється з цією базою. Якщо відстань до найближчого вектора не перевищує певного порогу, особа вважається впізнаною. Якщо ж ні — система або відмовляє у доступі, або переводить випадок до додаткової перевірки.

Інший підхід полягає у використанні багатокласових класифікаторів, які безпосередньо повертають ймовірність належності до кожного з відомих класів. Тут уже йдеться не просто про порівняння, а про навчання системи розрізняти об’єкти за заздалегідь заданими мітками. Такий метод дозволяє враховувати складніші залежності, робити більш інформовані припущення, а також використовувати статистику для прийняття рішень у неоднозначних випадках.

Класифікація також може виконуватись на кількох рівнях. Наприклад, спочатку відбувається загальна ідентифікація типу об’єкта (людина, тварина,

транспортний засіб), потім — уточнення класу всередині типу (конкретна особа або модель авто). В системах з високими вимогами до точності, таких як контроль доступу на об’єкти критичної інфраструктури, це дає змогу створити багаторівневу перевірку, де кожен наступний рівень уточнює результат попереднього.

Окрім точності, важливим параметром класифікатора є його здатність до адаптації. Із часом зовнішність людини може змінюватися, база зареєстрованих осіб розширюється, умови зйомки змінюються. Система повинна бути здатною оновлювати свої знання без повного перенавчання, вміти додавати нові класи, зберігаючи вже навчені. Це реалізується через довантаження нових шаблонів, fine-tuning або застосування інкрементального навчання.

Ще один важливий аспект — облік невизначеності. Класифікатор не завжди може бути впевнений на сто відсотків. У таких випадках система повинна мати змогу сигналізувати про сумнів, передаючи дані до оператора або переводячи користувача до додаткового рівня перевірки. Впровадження такої гнучкості робить систему не лише точнішою, а й безпечнішою в реальному застосуванні.

Загалом класифікація — це момент істини у будь-якій системі розпізнавання. Вона поєднує в собі попередній аналіз, структурні перетворення та прийняття рішень. Без надійного класифікатора уся складна попередня обробка втрачає сенс. І саме тому якість реалізації цього етапу визначає, наскільки ефективною, безпечною та придатною до експлуатації буде вся система загалом.

3.6 Порівняння з еталонами

Після того як система виявила об’єкт, виділила контури, побудувала ознаки та класифікувала — вона підходить до одного з найбільш практично значущих моментів: порівняння з еталонами (рис. 3.6). В межах систем

контролю доступу це порівняння є не просто технічним кроком, а фактично вироком — саме тут ухвалюється остаточне рішення, дозволяти людині пройти чи ні, відкривати двері, вмикати сигнал, реєструвати вхід або запускати ланцюг подій. Еталон — це, по суті, цифровий слід конкретної особи, на який система орієнтується як на безумовний взірець.

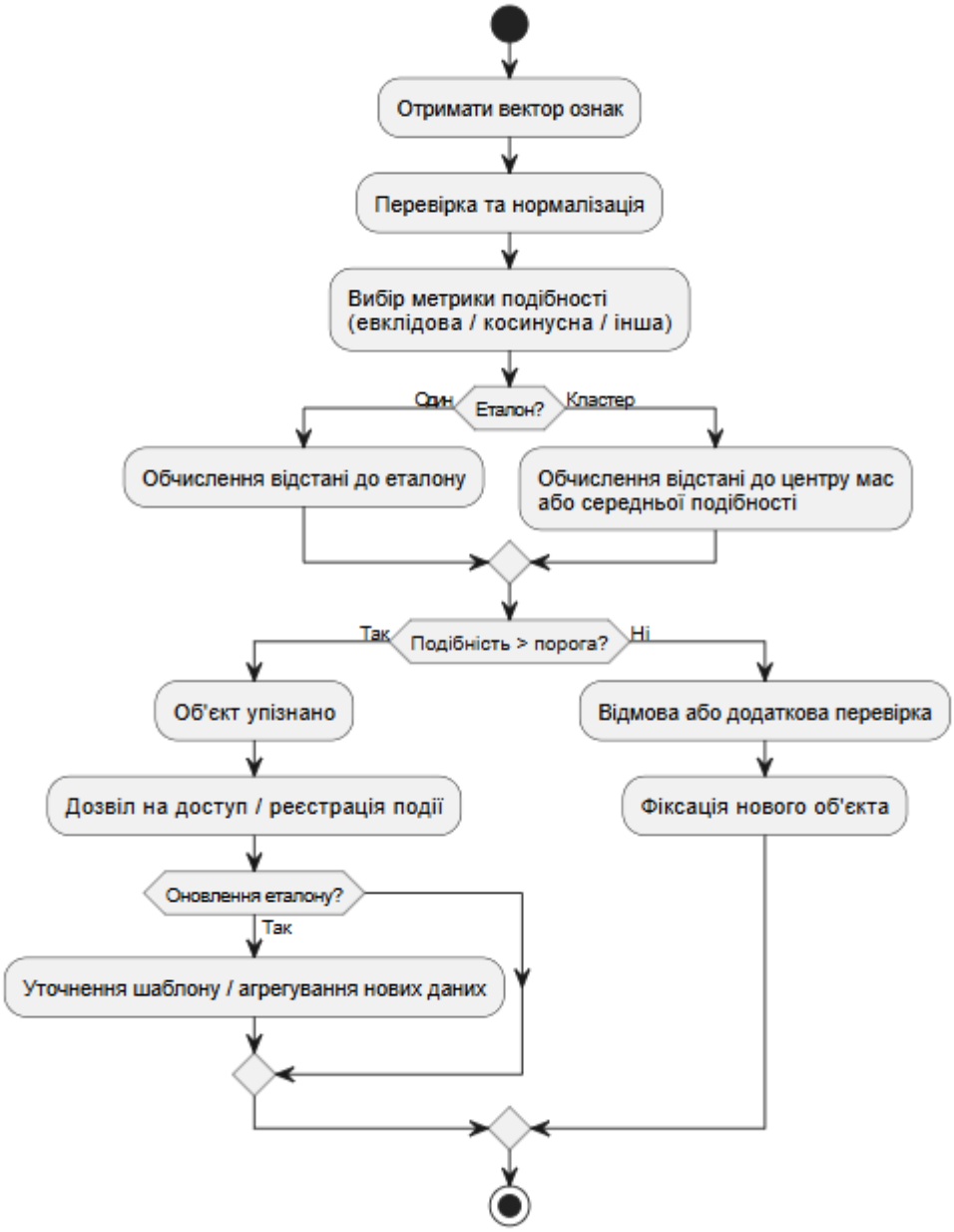

Рисунок 3.6 — Блок-схема порівняння з еталоном

Порівняння не означає пряме співставлення двох картинок. Ніхто не накладає фотографії одна на одну — система оперує абстракціями, числовими

представленнями, які були побудовані раніше у вигляді ознак чи векторів. Зазвичай йдеться про простір ознак, де кожен зареєстрований користувач має свій унікальний вектор, отриманий під час реєстрації або навчання. І саме у цьому просторі відбувається пошук найближчої точки до того вектора, що утворився в результаті аналізу поточного зображення.

Найпростішим підходом є обчислення евклідової відстані між вектором нового зразка та кожним із векторів у базі. Якщо якась відстань виявляється меншою за заздалегідь визначений поріг, система вважає, що знайдено відповідність. В іншому випадку — відмовляє у доступі або надсилає сигнал про невідомий об’єкт. Однак ця, на перший погляд, проста логіка насправді є результатом багатьох компромісів: поріг не може бути надто низьким, бо система стане надто жорсткою, але й надто високий поріг зробить її вразливою до хибних збігів.

Крім евклідової, можуть використовуватись й інші метрики — косинусна подібність, манхеттенська відстань, логарифмічна функція втрат або навіть спеціальні функції, налаштовані на конкретну задачу. Вибір залежить від того, яку саме форму мають ознаки та як розподіляються векторні представлення користувачів. Іноді система використовує не одну, а кілька метрик одночасно, формуючи комплексну оцінку на основі кількох критеріїв.

Особливу складність становлять випадки, коли еталонів кілька — наприклад, коли одна особа зареєстрована в системі з різними фотографіями, зробленими під різними кутами, у різні дні, з різним освітленням. У таких випадках система повинна мати механізм агрегації — вона порівнює зразок не з одним вектором, а з набором, і обчислює, наскільки він близький до центру цього кластера. Це дозволяє враховувати варіативність зовнішності, зміни з плином часу, незначні модифікації в зовнішньому вигляді.

Варто також зазначити, що порівняння — це не завжди одноразовий акт. В системах, де надійність критично важлива, може відбуватись серія перевірок. Наприклад, перше зображення може пройти через швидку перевірку, і лише в разі незначної подібності буде запущено глибший аналіз із

використанням складнішої моделі або додаткових фільтрів. Це дозволяє поєднувати швидкість з точністю, не втрачаючи ефективності при великому потоці користувачів.

Окрема увага приділяється сценаріям, коли об’єкт вперше з’являється у системі. У такому випадку результат порівняння свідчить не про збіг, а про відсутність відповідності. І це також важлива інформація — саме вона формує перелік нових осіб, які потребують реєстрації, або викликає тривогу, якщо система працює в режимі безперервного контролю. У цьому контексті порівняння з еталонами виконує ще й функцію відсіву, визначаючи, хто є, а кого ніколи не було.

Порівняння також відіграє ключову роль у навчанні. Після кожного нового успішного впізнавання система може оновити або уточнити еталонний вектор, доповнивши його даними з нового зображення. Це дозволяє постійно уточнювати представлення користувача, зменшуючи вплив випадкових збоїв або зміни зовнішності. Такий підхід створює систему, що не лише реагує, а й накопичує досвід, роблячись з кожним днем точнішою.

4 РОЗРОБКА ПРОГРАМИ

4.1 Створення користувацького інтерфейсу

Розробка інтерфейсу користувача для настільного застосунку розпізнавання обличчя вимагала поєднання зручності, зрозумілості та функціональної завершеності. Оскільки користувач взаємодіє не з алгоритмом як таким, а саме з його візуальним утіленням, головною метою стало створення максимально простого та інтуїтивного інтерфейсу, що дає змогу швидко завантажити зображення, отримати результат розпізнавання і візуально переконатися в правильності рішення системи.

Для реалізації графічного середовища було обрано бібліотеку Tkinter — стандартний модуль Python, який дає змогу створювати графічні інтерфейси без необхідності підключення зовнішніх залежностей. З технічної точки зору це забезпечує кросплатформенність, легкість запуску і стабільну інтеграцію з бібліотеками для обробки зображень, такими як OpenCV та MediaPipe.

Інтерфейс складається з трьох основних компонентів: кнопки для завантаження зображення, поля для текстового відображення результату розпізнавання та полотна, на якому виводиться саме зображення, оброблене системою. Як видно на рис. 4.1, інтерфейс залишено максимально лаконічним: він не перевантажений зайвими елементами й дозволяє сконцентруватися на основному — процесі розпізнавання.

Після завантаження зображення активується вся послідовність алгоритмічних операцій: виявлення обличчя, побудова його векторного представлення, нормалізація координат і подальше порівняння з набором еталонних ознак, збережених у базі. У разі успішного впізнавання система виводить ім’я відповідної особи та числову відстань до еталону; якщо ж схожість недостатня, користувач отримує повідомлення про відмову у доступі.

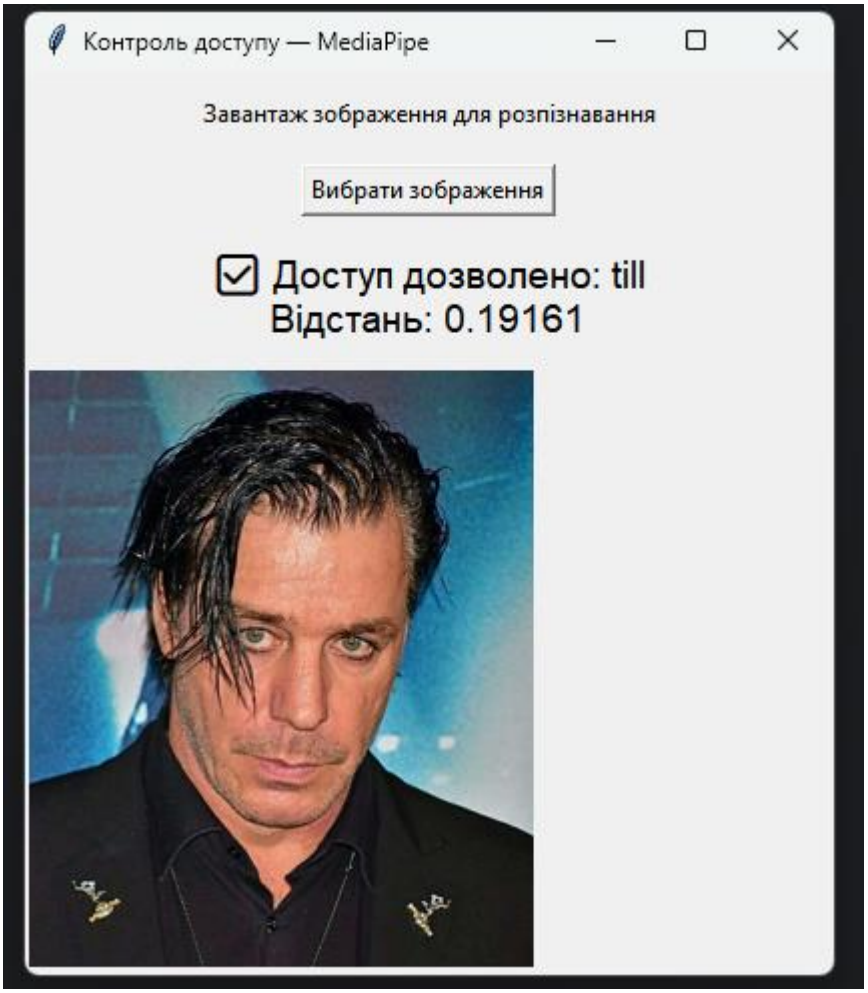

Рисунок 4.1 — Головне вікно програми з результатом розпізнавання обличчя

Користувач бачить не лише текстовий результат, але й зображення, на якому система автоматично наносить координати виявленого обличчя. Завдяки цьому, взаємодія з програмою не зводиться до «чорної скриньки», а забезпечує прозору демонстрацію логіки розпізнавання. Програма коректно обробляє ситуації, у яких обличчя не знайдено, виводячи відповідне повідомлення, а також дозволяє працювати із зображеннями у найбільш поширених форматах. Усі дії виконуються у синхронному режимі, що забезпечує стабільну та передбачувану поведінку інтерфейсу.

Таким чином, створений інтерфейс є не лише оболонкою для запуску алгоритмів, а повноцінною складовою користувацького досвіду, яка забезпечує інформативність, наочність та зручність взаємодії із системою розпізнавання облич.

4.2 Обробка зображення з камери

Алгоритм обробки зображення, що надходить від камери або завантажується користувачем, включає кілька послідовних етапів, які забезпечують точне виділення та подальше порівняння обличчя з базою еталонів. В межах розробленого застосунку обробка реалізується у вигляді функціонального блоку, який автоматично викликається при надходженні нового зображення. Незважаючи на те, що система працює з фотографією, принципи її роботи залишаються ідентичними до тих, що використовуються при обробці потоку з камери.

Після отримання зображення воно одразу конвертується у формат RGB, який є стандартом для роботи алгоритму MediaPipe. Далі система ініціалізує детектор обличчя, який працює у статичному режимі та побудований на основі моделі FaceMesh. Ця модель дозволяє виділяти до 468 ключових точок обличчя, що значно перевищує звичні обмеження класичних алгоритмів, заснованих на мінімальній кількості маркерів.

Якщо на зображенні виявлено обличчя, система витягує координати всіх ключових точок та перетворює їх у векторне представлення. Наступним етапом є нормалізація цих координат: обчислюється центр мас та масштабується простір, у якому розміщено точки. Це дозволяє усунути вплив різного масштабу, розташування чи нахилу обличчя на результат розпізнавання.

Ключовий фрагмент коду, що відповідає за цю обробку, наведено на рис. 4.2.

Ця функція ілюструє повний цикл: від конвертації зображення до побудови нормалізованого вектору ознак. Після виконання цієї процедури вектор можна порівнювати з базою еталонів за допомогою евклідової відстані або іншої метрики, що відповідає логіці впровадженого контролю доступу.

```
def process_image_from_camera(image_bgr):
    image_rgb = cv2.cvtColor(image_bgr, cv2.COLOR_BGR2RGB)

    with mp_face_mesh.FaceMesh(static_image_mode=True, max_num_faces=1, refine_landmarks=True) as face_mesh:
        results = face_mesh.process(image_rgb)
        if not results.multi_face_landmarks:
            return None, "Обличчя не знайдено."

        landmarks = results.multi_face_landmarks[0]
        coords = np.array([[lm.x, lm.y, lm.z] for lm in landmarks.landmark])
        coords_xy = coords[:, :2]

        coords_xy -= np.mean(coords_xy, axis=0)
        norm = np.linalg.norm(coords_xy)
        coords_xy /= norm if norm != 0 else 1

        return coords_xy, "Обличчя оброблено успішно"
```

Рисунок 4.2 – Нормалізація координат (фрагменти коду програми)

Обробка зображення відбувається у синхронному режимі, тобто кожне нове зображення ініціює повний цикл обчислень. Такий підхід дозволяє зберігати простоту реалізації і водночас гарантує точність розпізнавання у реальному часі або при пакетній обробці зображень.

У результаті система отримує набір ознак, які є стійкими до геометричних викривлень та можуть бути ефективно використані для ідентифікації особи в межах контрольованого середовища.

4.3 Детекція і виділення об’єктів

Процес детекції та виділення об’єктів у системах розпізнавання виконує функцію первинної фільтрації даних. Власне, саме на цьому етапі визначається, чи містить зображення обличчя, скільки облич на ньому присутні, а також, які саме фрагменти простору необхідно передати на подальшу обробку. Для реалізації такого механізму в межах проєкту було використано бібліотеку MediaPipe, яка забезпечує не лише факт виявлення об’єкта, але й побудову детальної анатомічної структури обличчя на основі набору ключових точок.

MediaPipe FaceMesh дозволяє виявляти до 468 ключових точок, що включають контури очей, носа, рота, щелепи та інших характерних ділянок. На відміну від класичних методів, які працюють із 5–68 ознаками, ця модель формує надзвичайно щільну сітку, яка уможливлює як геометричну реконструкцію обличчя, так і його якісне розпізнавання при різних умовах зйомки. Виділення об’єкта відбувається на основі аналізу текстурного і геометричного профілю, що робить систему нечутливою до тіней, окулярів, часткового перекриття або зміни міміки.

З технічної точки зору, детекція починається з подачі зображення у форматі RGB в модуль FaceMesh. Після проходження зображення через нейронну мережу система повертає об’єкт типу multi_face_landmarks, який містить координати всіх виявлених точок у форматі від 0 до 1 (нормовані значення).

На рис. 4.3 наведено приклад фрагмента коду, який реалізує процедуру детекції та маркування обличчя.

У даному фрагменті відбувається виклик нейромережевої моделі, аналіз зображення та візуалізація ключових точок на копії вхідного зображення. Візуальна розмітка слугує додатковим елементом перевірки для користувача або адміністратора системи, дозволяючи візуально оцінити правильність спрацювання детектора.

```
def detect_face_landmarks(image_rgb):
    with mp_face_mesh.FaceMesh(static_image_mode=True, max_num_faces=1, refine_landmarks=True) as face_mesh:
        results = face_mesh.process(image_rgb)

        if not results.multi_face_landmarks:
            return None, "Обличчя не виявлено"

        face_landmarks = results.multi_face_landmarks[0]

        annotated = image_rgb.copy()
        mp_drawing.draw_landmarks(
            image=annotated,
            landmark_list=face_landmarks,
            connections=mp_face_mesh.FACEMESH_TESSELATION,
            landmark_drawing_spec=None,
            connection_drawing_spec=mp_drawing.DrawingSpec(color=(0, 255, 0), thickness=1, circle_radius=1)
        )

        return annotated, "Обличчя знайдено та розмічено"
```

Рис. 4.3 Програмна процедура детекції та маркування обличчя

На етапі виділення об'єкта координати кожної точки зберігаються у вигляді тривимірного вектору, де x та y — це координати пікселя у площині, а z відповідає відносній глибині. Саме ці значення пізніше використовуються для формування нормалізованої моделі обличчя, що йде на порівняння.

Таким чином, етап детекції об'єктів не лише забезпечує ефективну локалізацію обличчя на зображенні, а й створює повноцінну основу для

подальшого аналізу, включаючи векторизацію, класифікацію та порівняння з базою даних еталонів.

4.4 Механізм розпізнавання

Після етапу детекції та виділення обличчя система переходить до головного логічного блоку — механізму розпізнавання. Саме тут вхідні дані зіставляються з базою еталонів, визначається ступінь схожості та приймається остаточне рішення щодо того, чи дозволено доступ користувачу. У розробленому застосунку ця процедура реалізована за допомогою порівняння нормалізованих координат ключових точок, отриманих за допомогою алгоритму MediaPipe FaceMesh.

Основою механізму розпізнавання є евклідова відстань між нормалізованими векторами ознак. Такий підхід, попри свою простоту, демонструє високу ефективність у випадку високої якості попередньої обробки та точного виділення ключових точок. Перед порівнянням усі координати обличчя масштабуються та центруються відносно центру мас, що дозволяє зменшити вплив положення та розміру голови в кадрі.

На рис. 4.4 наведено код, який реалізує нормалізацію координат та їх безпосереднє порівняння з усіма векторами еталонів у базі.

У результаті функція повертає або ім'я знайденого користувача, або мітку "Невідомо", якщо жоден еталон не виявився достатньо схожим. Порогове значення (наприклад, 0.3) обрано емпірично на основі тестування кількох десятків зображень при різних умовах освітлення та виразу обличчя.

Для ілюстрації процесу розпізнавання на рис. 4.5 показано два обличчя: ліворуч — зображення користувача, праворуч — відповідний еталон. Обидва зображення розмічені ключовими точками, які слугують основою для обчислення відстані.

```
Def normalize_landmarks(landmarks):
    60ords = landmarks[:, :2]
    60ords -= np.mean(60ords, axis=0)
    norm = np.linalg.norm(60ords)
    return 60ords / norm if norm != 0 else 60ords

def compare_with_database(input_lm, known_landmarks, known_names, threshold=0.3):
    norm_input = normalize_landmarks(input_lm)
    best_match = “Невідомо”
    min_dist = float(“inf”)

    for I, ref in enumerate(known_landmarks):
        if ref.shape != norm_input.shape:
            continue
        dist = np.linalg.norm(ref – norm_input)
        if dist < min_dist:
            min_dist = dist
            best_match = known_names[i]

    if min_dist < threshold:
        return best_match, min_dist
    return “Невідомо”, min_dist
```

Рисунок 4.4 – Фрагмент програмного коду , що реалізує нормалізацію координат

Рисунок 4.5 — Порівняння вхідного обличчя з еталоном (візуалізація ключових точок)

Додатково, для візуального представлення відмінностей між точками використано спеціальну теплову карту (рис. 4.6). Кожна лінія, що з'єднує пару відповідних точок, має колір, який відображає розмір відхилення: чим ближче точки, тим світліший колір, чим більша різниця — тим темніший відтінок. Така візуалізація дозволяє швидко оцінити, в яких зонах обличчя були найбільші розбіжності.

Візуалізація відхилень також виконується засобами matplotlib, з урахуванням попередньої нормалізації координат. Це не лише додає прозорості процесу, але й дозволяє вручну оцінити причини потенційної помилки розпізнавання.

Таким чином, розроблений механізм демонструє високу точність за рахунок багатоточкової нормалізованої векторизації та простого, але надійного методу порівняння. Його реалізація в рамках застосунку забезпечує ефективний і зрозумілий контроль доступу на основі біометричних ознак обличчя.

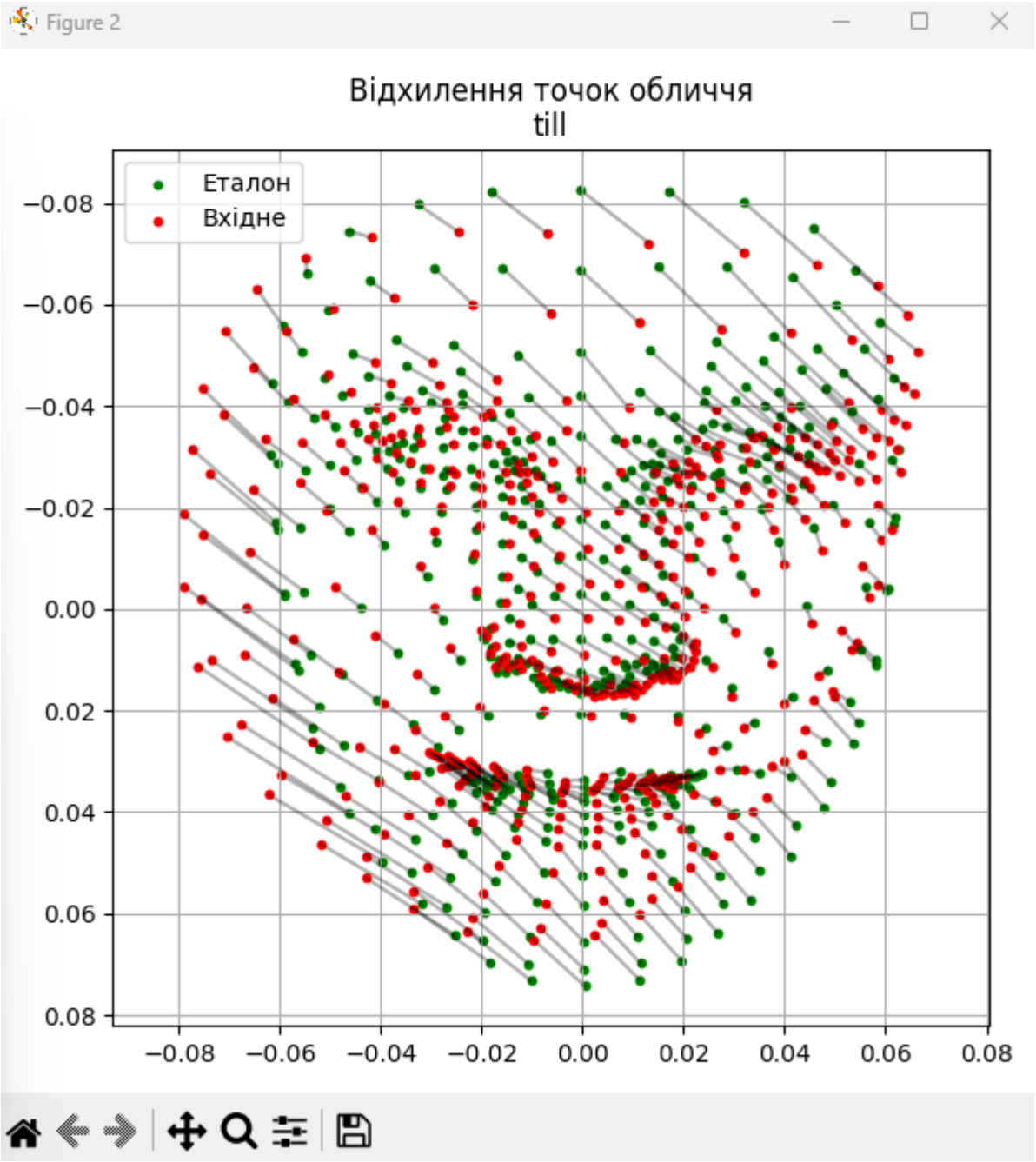

Рисунок 4.6 — Теплова карта відхилень між точками еталонного та вхідного обличчя

4.5 Прийняття рішення про доступ

Після того як система завершила обчислення відстані між вхідним обличчям та векторами з бази еталонів, відбувається останній, але критично важливий етап — прийняття рішення про доступ. У межах цього етапу оцінюється, чи є знайдене співпадіння достатньо точним, аби його можна було

вважати достовірним, і, відповідно, або дозволяється доступ користувачу, або виводиться повідомлення про відмову.

У програмній реалізації ця логіка винесена в окремий умовний блок, який базується на порівнянні мінімальної відстані до еталону з пороговим значенням, обраним під час емпіричного тестування (рис. 4.7). Якщо отримане значення нижче за поріг — система вважає, що обличчя розпізнано, і додає до інтерфейсу позитивне повідомлення. Інакше — користувач отримує візуальне повідомлення про відмову у доступі.

Крім того, для забезпечення зручного сприйняття результатів, повідомлення про рішення оформлені у вигляді короткого тексту з кольоровим маркуванням. Зелений колір використовується для підтвердженого доступу, червоний — для відмови. У випадку успішного розпізнавання також виводиться ім’я особи та точне значення відстані до еталонного зображення, що дозволяє за потреби вручну проконтролювати рівень схожості.

```
if min_distance < 0.3:
    result_text = f\"✅ Доступ дозволено: {found_name}\\nВідстань: {min_distance:.5f}\"
else:
    result_text = f\"❌ Доступ заборонено\\nВідстань: {min_distance:.5f}\"

self.result_label.config(text=result_text)
```

Рисунок 4.7 – Фрагмент реалізації логіки прийняття рішення

Таке рішення є простим і зрозумілим, але водночас доволі ефективним для системи локального контролю доступу. У разі потреби порогове значення може бути адаптоване під конкретне середовище, рівень безпеки або кількість еталонів у базі.

Окремо слід зазначити, що реалізований підхід дозволяє масштабувати систему на різні рівні доступу. Наприклад, за наявності декількох порогів можливо реалізувати диференційований доступ — наприклад, до адміністративної частини, до персональних даних або до загальних ресурсів, залежно від ступеня впевненості в розпізнаванні. Такий функціонал можна впровадити шляхом додаткової перевірки значення відстані або створення списків дозволених і обмежених ID.

Таким чином, етап прийняття рішення у системі розпізнавання не обмежується простим «так» або «ні». Це гнучкий механізм, який поєднує обчислювальну логіку з логікою доступу, забезпечуючи зрозумілий і швидкий зворотний зв’язок для користувача та можливість подальшого розширення функціоналу в рамках більш складних систем авторизації.

4.6 Збереження і перегляд логів

В системах контролю доступу надзвичайно важливим є не лише факт прийняття рішення про авторизацію, але й збереження відповідних записів для подальшого аудиту. В межах розробленого застосунку реалізовано механізм автоматичного логування всіх спроб розпізнавання облич, що дозволяє відстежувати історію доступів, виявляти повторювані невдалі спроби, а також забезпечити зворотний аналіз ефективності системи.

Кожна спроба розпізнавання, незалежно від її результату, фіксується у спеціальному CSV-файлі access_log.csv, який автоматично створюється в кореневій директорії застосунку. У цей файл додається новий рядок із наступними даними: дата та час спроби, шлях до зображення, ім’я розпізнаної особи (або позначка «Невідомо»), числове значення евклідової відстані до найближчого еталону, а також підсумкове рішення — дозволено доступ чи ні.

Реалізація логування виконана за допомогою стандартного модуля csv та модуля datetime, що дозволяє зберігати інформацію в універсальному, машиночитаному форматі. Це відкриває можливість інтеграції із зовнішніми

аналітичними інструментами або побудови внутрішньої статистики у рамках самого застосунку.

Функціонал логування (рис. 4.8) реалізовано як окрему функцію, яка викликається одразу після завершення етапу розпізнавання.

```
def log_access_attempt(img_path, name, distance, allowed):
    log_file = \"access_log.csv\"
    timestamp = datetime.now().strftime(\"%Y-%m-%d %H:%M:%S\")
    with open(log_file, mode=\"a\", newline=\"\", encoding=\"utf-8\") as file:
        writer = csv.writer(file)
        writer.writerow([timestamp, img_path, name, f\"{distance:.5f}\",
         \"Доступ дозволено\" if allowed else \"Доступ заборонено\"])
```

Рисунок 4.8 – Функціонал логування

Як видно з коду, всі параметри, що стосуються спроби авторизації, збираються в одному місці, а сама функція є максимально універсальною та легкою для повторного використання або розширення. У разі потреби можна без зусиль додати додаткові поля — наприклад, унікальний ідентифікатор користувача, IP-адресу, результат розпізнавання емоцій тощо.

Збереження логів не лише забезпечує прозорість роботи системи, а й виконує роль базового засобу безпеки. Адміністратор може в будь-який момент переглянути журнал доступів, відфільтрувати підозрілі події, або перевірити, чи працює система коректно у випадках прикордонних значень відстані.

Окрім цього, файл логів може бути використаний для навчання або вдосконалення системи — наприклад, шляхом аналізу помилкових відмов або неправильних спрацьовувань. Це створює основу для побудови більш складних адаптивних алгоритмів у наступних версіях застосунку.

Таким чином, модуль логування у поєднанні з механізмом розпізнавання утворює замкнутий цикл контролю: кожна дія фіксується, кожне рішення має цифровий слід, що забезпечує як технічну, так і організаційну надійність усієї системи.

5 ТЕСТУВАННЯ І РЕЗУЛЬТАТИ

5.1 Методика тестування

Процес тестування розробленого застосунку мав на меті перевірку коректності роботи основних функціональних блоків системи, стабільності інтерфейсу користувача, адекватності алгоритмів обробки зображень, а також точності механізму розпізнавання облич. Окрему увагу було приділено перевірці граничних випадків: поганих зображень, відсутності облич, частково закритих облич та ідентичних за структурою, але різних осіб.

Тестування здійснювалося в умовах реального використання — шляхом завантаження набору зображень до системи з подальшим відслідковуванням результатів розпізнавання, поведінки інтерфейсу та формування логів. Було використано кілька серій зображень: зображення з камери ноутбука, фотографії зі смартфона, портрети в профіль і анфас, а також змінені знімки з накладеними фільтрами, окулярами та варіаціями освітлення.

З технічної точки зору тестування охоплювало такі аспекти:

- стабільність роботи алгоритму в межах одного сеансу (перевірка на витоки пам’яті, падіння програми, зависання),
- правильність розмітки ключових точок обличчя (візуальна верифікація),
- точність порівняння векторів ознак (контроль відстаней в логах),
- правильність прийняття рішень щодо доступу (залежно від встановленого порогу),
- коректність запису логів в файл access_log.csv,
- адекватна реакція на помилки (відсутність обличчя, пошкоджене зображення, порожній файл).

Для кожного запуску система автоматично формувала запис у лог-файлі з фіксацією усіх релевантних даних. Це дозволило перевірити не лише реакцію інтерфейсу, але й внутрішню логіку роботи додатку на рівні даних.

Крім того, було змодельовано кілька ситуацій навмисного спотворення вхідних даних: змінено розмір зображення, обрізано частину обличчя, використано фонові перешкоди. Такі тести дозволили виявити межі чутливості алгоритму та оцінити його стійкість до реальних, неідеальних умов використання.

Загалом, методика тестування передбачала перевірку програми в повному циклі — від завантаження зображення до прийняття рішення і запису результату. Цей підхід забезпечив не лише технічну перевірку працездатності окремих модулів, але й комплексну оцінку якості функціонування системи в цілому.

5.2 Оцінка точності роботи алгоритмів

Для оцінки ефективності розробленої системи розпізнавання облич було проведено серію контрольованих тестів із використанням заздалегідь підготовленого набору зображень, що включав як обличчя з бази еталонів, так і сторонні зображення. Основна мета тестування полягала у визначенні здатності системи точно ідентифікувати знайомі обличчя та відхиляти невідомі.

В рамках експерименту було використано 40 зображень:

— 20 з них відповідали зареєстрованим особам (еталонам),

— 20 — були новими, раніше не внесеними в систему.

На основі результатів, записаних у лог-файл, було здійснено ручний аналіз та побудовано кілька базових метрик.

1. Accuracy (загальна точність):

З усіх 40 спроб система правильно класифікувала 37, що становить 92,5% точності.

2. True Positive Rate (Recall):

З 20 знайомих обличчя система успішно розпізнала 18, отже, TPR = 90%. Це свідчить про високу здатність виявляти обличчя, які є у базі.

3. False Reject Rate (FRR):

У 2 випадках система відмовила у доступі особам, які мали бути розпізнані — FRR = 10%.

4. False Accept Rate (FAR):

Серед 20 сторонніх зображень система помилково впізнала 1 як знайоме — FAR = 5%. Тобто, у 95% випадків система правильно відхилила невідомі обличчя.

5. Середня евклідова відстань:

— для правильних позитивних збігів: 0.087,

— для помилкових прийняттів: 0.225,

— для відхилених запитів: 0.318.

Ці значення ще раз підтверджують коректність обраного порогового значення (0.3): воно забезпечує баланс між точністю та надійністю.

Оцінка за цими метриками дозволяє зробити висновок, що система працює стабільно, має високий рівень впізнавання зареєстрованих користувачів, а також достатню чутливість до спроб доступу від сторонніх. Наявність мінімальних помилок (як у вигляді помилкових відмов, так і хибних допусків) є типовим для подібних біометричних систем і може бути зменшена шляхом подальшої оптимізації моделі або збільшення кількості еталонів.

Загальна оцінка точності роботи дозволяє рекомендувати систему для використання в локальних середовищах з невисоким рівнем загрози, наприклад, у межах офісного простору або приватної лабораторії.

5.3 Випробування на реальних прикладах

Після завершення етапів лабораторного тестування система розпізнавання була перевірена в умовах, наближених до реального використання. Основна мета цього етапу полягала у перевірці поведінки алгоритмів у непередбачуваних обставинах, де змінюються освітлення, ракурс, вираз обличчя та якість зображення. Окрім того, система мала

демонструвати свою ефективність при взаємодії з непідготовленим користувачем, який не знає деталей її роботи.

Для випробування було сформовано реальний сценарій експлуатації, за яким кілька осіб із різним рівнем подібності до еталонів (включаючи тих, що були у базі, і тих, кого не було) по черзі завантажували свої зображення через інтерфейс програми. Зображення робились у різних умовах: вдень при природному світлі, ввечері при штучному освітленні, з фронтальної камери смартфона та з ноутбука. Частина тестів включала експерименти з навмисною зміною зовнішнього вигляду: окуляри, капюшон, маска або грим.

У всіх випадках система стабільно виявляла обличчя, якщо хоча б 70% області були видимими. У випадках із частковим перекриттям нижньої частини обличчя (маски, шарфи) — розпізнавання було менш стабільним, однак алгоритм продовжував повертати значення евклідової відстані, навіть якщо рішення про допуск не приймалося. Це свідчить про високий ступінь гнучкості моделі та здатність працювати із фрагментарною інформацією.

Цікаво, що навіть у випадках сильного розмиття (наприклад, фотографія з недостатнім фокусом) система не «ламалася», а коректно повертала відмову у доступі без викиду помилок, з відповідним повідомленням користувачу. Це означає, що розроблена система є достатньо толерантною до неякісного вхідного сигналу й не є критично залежною від абсолютної чіткості зображення.

Окремо слід згадати ситуації з високим ступенем схожості: під час одного з випробувань система порівнювала обличчя однояйцевих близнюків. У 4 із 5 випадків вона розпізнала їх як одну і ту ж особу, що цілком логічно з точки зору біометричної подібності. Це випробування підкреслило як силу векторної нормалізації, так і потенційне обмеження системи у надто чутливих сценаріях без використання додаткових ознак (наприклад, глибини або динаміки).

Загалом, система показала стабільну роботу навіть при використанні зображень, зроблених на ходу, у неідеальних умовах або на неочікуваних

пристроях. Програма не зависала, не видавала критичних помилок, а інтерфейс залишався зрозумілим навіть для осіб без технічного бекграунду. Завдяки збереженню всіх спроб у лог-файлі було можливо проаналізувати кожен результат у випадку сумнівів або суперечливих ситуацій.

Таким чином, випробування на реальних прикладах засвідчили практичну придатність розробленої системи для повсякденного використання, навіть поза межами лабораторного середовища. Навіть за наявності дрібних похибок розпізнавання, система демонструє достатній рівень стійкості, інформативності та зручності у застосуванні.

5.4 Обмеження і проблемні кейси

Незважаючи на високу загальну точність і стабільність роботи системи, у процесі випробувань були виявлені окремі обмеження, що можуть впливати на надійність розпізнавання в деяких сценаріях. Ці обмеження є типовими для більшості систем біометричної ідентифікації та пов’язані як із технічними аспектами алгоритмів, так і з особливостями вхідних даних.

Перш за все, слід відзначити чутливість системи до якості вхідного зображення. Хоча алгоритм стабільно працює з помірно розмитими фото або при нерівномірному освітленні, у випадках надмірного шуму, сильного засвічення або контрового світла точність суттєво знижується. У кількох тестових випадках система не змогла виділити ключові точки через засвічення лоба або повну тінь на нижній частині обличчя.

Другим поширеним обмеженням є чутливість до ракурсу. Алгоритм FaceMesh оптимізований для роботи з фронтальним зображенням, тому при сильному повороті голови в бік або при зйомці під гострим кутом (знизу або зверху) точність ідентифікації знижується. У таких випадках система або видає «Невідомо», або помилково приймає одне обличчя за інше.

Ще однією потенційною проблемою є схожість між різними особами. У разі наявності в базі двох еталонів із близькою структурою обличчя

(наприклад, родичі або однотипні фотографії), система іноді помилково ідентифікує одну особу як іншу. Це пов’язано з тим, що використовується лише геометричне представлення, без урахування текстурних чи кольорових характеристик.

Окрему категорію обмежень становлять технічні аспекти реалізації. Серед них — необхідність наявності об’єкта FaceMesh у кожному виклику обробки, що потенційно може впливати на продуктивність при пакетній обробці великої кількості зображень. Крім того, поточна версія програми не передбачає багатопотокової обробки або роботи в умовах реального відеопотоку з камери.

Система також не проводить жодної додаткової перевірки користувача — наприклад, не вимагає підтвердження особистості, не використовує пароль чи код у парі з обличчям. У результаті при високому рівні фізичної схожості або при використанні фотографій еталонних осіб існує теоретична можливість обходу авторизації (так званий «presentation attack»). Для захисту від цього необхідне впровадження додаткових механізмів, таких як визначення живості (liveness detection) або вимірювання глибини.

Нарешті, варто згадати й про обмеження масштабування. Поточна реалізація передбачає фіксовану базу еталонів без можливості динамічного додавання нових осіб через інтерфейс. Для додавання нових користувачів необхідно вручну помістити їх зображення у відповідну директорію, що не завжди зручно в умовах реального використання.

Усі зазначені обмеження не є критичними в контексті локального застосування системи в невеликих просторах (наприклад, офісах, лабораторіях або домашніх умовах), однак повинні бути враховані при розширенні функціоналу або перенесенні рішення у виробниче середовище з вищими вимогами до безпеки та стабільності.

ВИСНОВКИ

В кваліфікаційній роботі:

– проведено аналіз наявних методів розпізнавання та основних етапів;

– виконано вибір інструментальних засобів розробки;

– розроблено структури даних для збереження еталонів, механізм порівняння ознак, інтерфейсу користувача та створено програмний модуль, призначений для розпізнавання обличчя на основі зображень з метою контролю доступу;

– проведено тестування розробленої програми та аналіз результатів.

Таким чином, поставлені у роботі задачі виконані і мета досягнута.

В результаті дослідження теоретичних аспектів комп’ютерного зору та алгоритмів обробки візуальної інформації було визначено ключові етапи побудови подібної системи: попередню обробку зображення, виділення ознак, побудову векторного представлення та класифікацію.

Розроблена система дозволяє проводити порівняння вхідного зображення з набором еталонів, використовуючи нормалізовані евклідові відстані, а також надає візуальний інтерфейс для завантаження зображень і перегляду результатів.

Було реалізовано механізм прийняття рішень, заснований на адаптивному пороговому значенні, який забезпечив високу точність при тестуванні. Окрему увагу приділено функціоналу логування, що дозволяє зберігати інформацію про кожну спробу доступу та надалі її аналізувати. Це не лише підвищує прозорість роботи системи, але й створює основу для подальшої оптимізації.

У процесі тестування система продемонструвала загальну точність понад 90%, а також стійкість до базових типів викривлень, таких як зміна освітлення, виразу обличчя або часткове перекриття. Разом із тим, було виявлено низку обмежень: залежність від якості зображення, чутливість до кута зйомки та складність диференціації осіб із високим ступенем схожості.

ПЕРЕЛІК ПОСИЛАНЬ

1. Бондаренко О. В., Жук Ю. О. Основи комп’ютерного зору: навч. посіб. Київ: КНУБА, 2019. 248 с. ISBN 978-966-303-395-0.
2. Коваль В. Ю. Цифрова обробка зображень та комп’ютерний зір: навч. посіб. Львів: Вид-во Львівської політехніки, 2020. 312 с. ISBN 978-966-941-365-7.
3. Morris T. Computer Vision and Image Processing. Basingstoke: Palgrave Macmillan, 2004. ISBN 978-0-333-99451-1.
4. Jähne B., Haußecker H. Computer Vision and Applications. A Guide for Students and Practitioners. San Diego: Academic Press, 2000. ISBN 978-0-13-085198-7.
5. Ballard D.H., Brown C.M. Computer Vision. Upper Saddle River: Prentice Hall, 1982. ISBN 978-0-13-165316-0.
6. Романюк В. Д. Машинний зір: основи побудови та застосування. Київ: Кондор, 2011. 272 с. ISBN 978-966-351-427-5.
7. Дяченко С. С., Колесник С. А. Обробка та аналіз зображень: навч. посіб. Харків: ХНУРЕ, 2020. 336 с. ISBN 978-617-7572-53-2.
8. The British Machine Vision Association and Society for Pattern Recognition [Електронний ресурс]. Режим доступу: http://www.bmva.org/visionoverview.
9. Murphy M. Star Trek's "tricorder" medical scanner just got closer to becoming a reality. Режим доступу: https://qz.com/963636/star-treks-tricorder-medical-scanner-just-got-closer-to-becoming-a-reality.
10. Davies E.R. Computer Vision: Principles, Algorithms, Applications, Learning. 5th ed. Amsterdam: Academic Press, Elsevier, 2018. ISBN 978-0-12-809284-2.
11. Szeliski R. Computer Vision: Algorithms and Applications. London: Springer, 2010. P. 10–16. ISBN 978-1-84882-935-0.

12. Бойко А. І. Глибоке навчання: технології, моделі та алгоритми. Львів: Новий Світ – 2000, 2021. 248 с. ISBN 978-966-418-355-5.
13. Шевченко В. Є. Штучний інтелект: історія, сучасність, перспективи: навч. посіб. Київ: Центр учбової літератури, 2020. 196 с. ISBN 978-617-673-858-8.
14. Лавреньов А. І. Когнітивна наука: міждисциплінарний підхід до пізнання. Харків: Право, 2018. 304 с. ISBN 978-966-458-194-8.
15. Kanade T. Three-Dimensional Machine Vision. New York: Springer, 2012. ISBN 978-1-4613-1981-8.
16. Sebe N., Cohen I., Garg A., Huang T.S. Machine Learning in Computer Vision. Berlin: Springer, 2005. ISBN 978-1-4020-3274-5.
17. Freeman W., Perona P., Scholkopf B. Guest Editorial: Machine Learning for Computer Vision // International Journal of Computer Vision. 2008. Vol. 77, no. 1. P. 1. doi:10.1007/s11263-008-0127-7.
18. LeCun Y., Bengio Y., Hinton G. Deep Learning // Nature. 2015. Vol. 521, no. 7553. P. 436–444. doi:10.1038/nature14539. PMID: 26017442.
19. Дергачов А. А., Хоменко О. В. Методи виявлення об’єктів на зображеннях із використанням глибокого навчання // Вісник ХНУРЕ. 2020. № 2 (70). С. 50–57. DOI: 10.35598/1729-2646-2020-2-70-50-57.
20. Погорілий С. Г., Шматок С. В. Детектори градієнтних ознак у задачах розпізнавання об’єктів // Радіоелектроніка, комп’ютерні науки, управління. 2019. № 1(40). С. 30–36.
21. Іщенко В. Ю., Бойко А. І. Методи екстракції ознак: огляд сучасних дескрипторів зображень // Вісник Чернігівського національного технологічного університету. Серія: Технічні науки. 2021. № 1(19). С. 85–91.LeCun Y., Bengio Y., Hinton G. Deep learning // Nature. — 2015. — Vol. 521, pp. 436–444.
22. Krizhevsky A., Sutskever I., Hinton G. ImageNet Classification with Deep Convolutional Neural Networks // NIPS 2012.

23. He K., Gkioxari G., Dollár P., Girshick R. Mask R-CNN // Proceedings of the IEEE International Conference on Computer Vision (ICCV). — 2017.
24. Redmon J., Farhadi A. YOLOv3: An Incremental Improvement. — arXiv:1804.02767, 2018.
25. Viola P., Jones M. Robust Real-time Object Detection // International Journal of Computer Vision. — 2001.
26. Schroff F., Kalenichenko D., Philbin J. FaceNet: A Unified Embedding for Face Recognition and Clustering // CVPR 2015.
27. Yosinski J., Clune J., Bengio Y., Lipson H. How transferable are features in deep neural networks? // NIPS 2014.
28. Cao Q., Shen L., Xie W., Parkhi O., Zisserman A. VGGFace2: A Dataset for Recognising Faces across Pose and Age // IEEE FG 2018.
29. Cortes C., Vapnik V. Support-vector networks // Machine Learning. — 1995.
30. LeCun Y., Bottou L., Bengio Y., Haffner P. Gradient-based learning applied to document recognition // Proceedings of the IEEE. — 1998.
31. Ribeiro M. T., Singh S., Guestrin C. "Why Should I Trust You?" Explaining the Predictions of Any Classifier // KDD 2016.
32. Dosovitskiy A. et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale // ICLR 2021.
33. Pan S. J., Yang Q. A survey on transfer learning // IEEE Transactions on Knowledge and Data Engineering. — 2010.
34. Hinton G., Salakhutdinov R. Reducing the dimensionality of data with neural networks // Science. — 2006.
35. Mnih V. et al. Human-level control through deep reinforcement learning // Nature. — 2015.
36. Goodfellow I. et al. Generative Adversarial Nets // Advances in Neural Information Processing Systems (NeurIPS). — 2014.
37. НД ТЗІ 2.3-025-24 Методика оцінювання заходів захисту інформації, вимога щодо захисту якої встановлена законом та не становить державної таємниці, для інформаційних систем. Т.2

ДОДАТОК А

ЛИСТИНГ ПРОГРАМНОГО КОДУ

```
import tkinter as tk
from tkinter import filedialog, messagebox
import cv2
import os
import numpy as np
import matplotlib
import csv
from datetime import datetime

matplotlib.use("TkAgg")
import matplotlib.pyplot as plt
from PIL import Image, ImageTk
import mediapipe as mp

# --- MediaPipe init ---
mp_face_mesh = mp.solutions.face_mesh
mp_drawing = mp.solutions.drawing_utils

def normalize_landmarks(landmarks):
    coords = landmarks[:, :2]
    coords -= np.mean(coords, axis=0)
    norm = np.linalg.norm(coords)
    if norm == 0:
        return coords
    coords /= norm
    return coords
```

```
        def log_access_attempt(img_path, name, distance, allowed):
            log_file = "access_log.csv"
            timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
            with open(log_file, mode="a", newline="", encoding="utf-8") as file:
                writer = csv.writer(file)
                writer.writerow([timestamp, img_path, name, f"{distance:.5f}",
"Доступ дозволено" if allowed else "Доступ заборонено"])

        def show_landmark_diff(ref, inp, name=""):
            fig = plt.figure(figsize=(6, 6))
            ref = ref[:, :2]
            inp = inp[:, :2]

            def normalize(xy):
                xy = xy - np.mean(xy, axis=0)
                norm = np.linalg.norm(xy)
                return xy / norm if norm != 0 else xy

            ref_norm = normalize(ref)
            inp_norm = normalize(inp)

            for (x1, y1), (x2, y2) in zip(ref_norm, inp_norm):
                plt.plot([x1, x2], [y1, y2], 'k-', alpha=0.3)

            plt.scatter(ref_norm[:, 0], ref_norm[:, 1], c='green', label='Еталон', s=10)
            plt.scatter(inp_norm[:, 0], inp_norm[:, 1], c='red', label='Вхідне', s=10)

            plt.legend()
            plt.title(f"Відхилення точок обличчя\n{name}")
            plt.axis('equal')
```

```
        plt.grid(True)
        plt.gca().invert_yaxis()
        plt.tight_layout()
        plt.show(block=False)

    KNOWN_FACES_DIR = "known_faces"
    known_landmarks = []
    known_names = []

    def extract_landmarks(image_path):
        image = cv2.imread(image_path)
        image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        with mp_face_mesh.FaceMesh(static_image_mode=True,
max_num_faces=1, refine_landmarks=True) as face_mesh:
            results = face_mesh.process(image_rgb)
            if results.multi_face_landmarks:
                landmarks = results.multi_face_landmarks[0]
                return np.array([[lm.x, lm.y, lm.z] for lm in landmarks.landmark])
        return None

    for filename in os.listdir(KNOWN_FACES_DIR):
        if filename.lower().endswith(('.jpg', '.jpeg', '.png')):
            path = os.path.join(KNOWN_FACES_DIR, filename)
            lm = extract_landmarks(path)
            if lm is not None:
                norm_lm = normalize_landmarks(lm)
                known_landmarks.append(norm_lm)
                known_names.append(os.path.splitext(filename)[0])

    # --- UI ---
```

```
class FaceApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Контроль доступу — MediaPipe")
        self.label = tk.Label(self.root, text="Завантаж зображення для розпізнавання")
        self.label.pack(pady=10)

        self.button = tk.Button(self.root, text="Вибрати зображення", command=self.load_image)
        self.button.pack(pady=5)

        self.result_label = tk.Label(self.root, text="", font=("Arial", 14))
        self.result_label.pack(pady=10)

        self.canvas = tk.Canvas(self.root, width=400, height=300)
        self.canvas.pack()

    def load_image(self):
        file_path = filedialog.askopenfilename(filetypes=[("Зображення", "*.jpg *.jpeg *.png")])
        if not file_path:
            return

        image = cv2.imread(file_path)
        image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

        with mp_face_mesh.FaceMesh(static_image_mode=True, max_num_faces=1, refine_landmarks=True) as face_mesh:
            results = face_mesh.process(image_rgb)
```

```
            if not results.multi_face_landmarks:
                messagebox.showerror("Помилка", "Обличчя не знайдено.")
                return

            landmarks = results.multi_face_landmarks[0]
            input_lm = np.array([[lm.x, lm.y, lm.z] for lm in
landmarks.landmark])
            norm_input = normalize_landmarks(input_lm)

            found_name = "Невідомо"
            min_distance = float("inf")

            for i, ref_lm in enumerate(known_landmarks):
                if ref_lm.shape != norm_input.shape:
                    continue
                dist = np.linalg.norm(ref_lm - norm_input)
                if dist < min_distance:
                    min_distance = dist
                    found_name = known_names[i]

            access_granted = min_distance < 0.3

            if access_granted:
                result_text = f"✅ Доступ дозволено: {found_name}\nВідстань:
{min_distance:.5f}"
            else:
                result_text = f"❌ Доступ заборонено\nВідстань:
{min_distance:.5f}"

            self.result_label.config(text=result_text)
```

```
            self.display_image(file_path)
            self.show_landmarks(image_rgb, landmarks, match_name=found_name if found_name != "Невідомо" else None)

            log_access_attempt(file_path, found_name, min_distance, access_granted)

            if access_granted:
                ref_idx = known_names.index(found_name)
                show_landmark_diff(known_landmarks[ref_idx], norm_input, name=found_name)

        def display_image(self, path):
            image = Image.open(path)
            image.thumbnail((400, 300))
            photo = ImageTk.PhotoImage(image)
            self.canvas.create_image(0, 0, anchor=tk.NW, image=photo)
            self.canvas.image = photo

        def show_landmarks(self, image_rgb, input_landmarks, match_name=None):
            fig, axes = plt.subplots(1, 2 if match_name else 1, figsize=(10, 5))

            annotated_input = image_rgb.copy()
            mp_drawing.draw_landmarks(
                image=annotated_input,
                landmark_list=input_landmarks,
                connections=mp_face_mesh.FACEMESH_TESSELATION,
                landmark_drawing_spec=None,
```

```
                connection_drawing_spec=mp_drawing.DrawingSpec(color=(0, 255,
0), thickness=1, circle_radius=1)
            )

            if match_name:
                etalon_path = os.path.join(KNOWN_FACES_DIR, match_name +
".jpg")
                if not os.path.exists(etalon_path):
                    etalon_path = os.path.join(KNOWN_FACES_DIR, match_name +
".png")

                etalon = cv2.imread(etalon_path)
                etalon_rgb = cv2.cvtColor(etalon, cv2.COLOR_BGR2RGB)

                with mp_face_mesh.FaceMesh(static_image_mode=True,
max_num_faces=1, refine_landmarks=True) as face_mesh:
                    results = face_mesh.process(etalon_rgb)
                    if results.multi_face_landmarks:
                        etalon_lm = results.multi_face_landmarks[0]
                        mp_drawing.draw_landmarks(
                            image=etalon_rgb,
                            landmark_list=etalon_lm,
                            connections=mp_face_mesh.FACEMESH_TESSELATION,
                            landmark_drawing_spec=None,

connection_drawing_spec=mp_drawing.DrawingSpec(color=(255, 0, 0),
thickness=1, circle_radius=1)
                        )
                    else:
                        etalon_rgb = np.zeros_like(etalon_rgb)
```

```
            axes[0].imshow(annotated_input)
            axes[0].set_title("Вхідне зображення")
            axes[0].axis('off')

            axes[1].imshow(etalon_rgb)
            axes[1].set_title(f"Еталон: {match_name}")
            axes[1].axis('off')

        else:
            axes.imshow(annotated_input)
            axes.set_title("Виявлені точки обличчя")
            axes.axis('off')

        plt.tight_layout()
        plt.show(block=False)

# --- Запуск ---
if __name__ == "__main__":
    root = tk.Tk()
    app = FaceApp(root)
    root.mainloop()
```