Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматизації, комп'ютерно-інтегрованих технологій та систем
(повна назва)
Кафедра Комп'ютерно-інтегрованих технологій, автоматизації, робототехніки та безпекової інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)
Розроблення програмного забезпечення для обробки відеоінформації у системі технічного зору промислового робота
(тема)

Виконав:
здобувач 4 року навчання,
групи АКТАКІТ-22-2
Руслан МИХАЙЛОВ
(власне ім'я, прізвище)

Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Автоматизація та комп'ютерно-інтегровані технології
(повна назва освітньої програми)

Керівник доцент Наталія ДЕМСЬКА
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри КІТАРБІ _____ Ігор НЕВЛЮДОВ
(підпис) (власне ім'я, прізвище)

2026 р.

Я, Михайлов Руслан Артемович, як здобувач вищої освіти ХНУРЕ, розумію і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Я не використовував штучний інтелект для підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

"01" липня 2026 р.  Руслан МИХАЙЛОВ

Харківський національний університет радіоелектроніки

Факультет АКІТС
Кафедра КІТАРБІ
Рівень вищої освіти перший (бакалаврський)
Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
(код і повна назва)
Тип програми освітньо-професійна
Освітня програма Автоматизація та комп'ютерно-інтегровані технології
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
« 27 » квітня 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Здобувачеві Михайлову Руслану Артемовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення програмного забезпечення для обробки відеоінформації у системі технічного зору промислового робота
затверджена наказом університету від 04.06.2026 р. № 472 Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 17.07.2026 р.
3. Вихідні дані до роботи
3.1 Основу реалізованої системи становить модель YOLO, яка забезпечує швидке та точне розпізнавання об'єктів у реальному часі
3.2 Середовище реалізації: мова програмування Python, бібліотеки OpenCV та Ultralytics YOLOv8
3.3 Для підготовки та експорту набору даних використано платформу Roboflow
3.4 Цифрові зображення та відеопотік для виявлення й розпізнавання об'єктів
4. Перелік питань, що потрібно опрацювати в роботі
4.1 Вступ
4.2 Визначення мети, об'єкта та предмета розробки
4.3 Аналіз предметної області: системи технічного зору, методи обробки зображень та сучасні алгоритми розпізнавання об'єктів
4.4 Обґрунтування вибору програмних засобів та нейромережевої моделі YOLOv8 для реалізації системи
4.5 Реалізація програмного забезпечення для обробки відеоінформації та детекції об'єктів із використанням OpenCV і YOLOv8
4.6 Тестування системи та аналіз отриманих результатів
4.7 Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Демонстраційний матеріал в форматі (*.ppt), А4 – 10 сторінок

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Актуальність роботи, постановка задачі	27.04.26 – 30.04.26	викон.
2	Визначення мети об'єкту і предмету розробки	01.05.26 – 03.05.26	викон.
3	Аналіз предметної області: системи технічного зору, методи обробки зображень та сучасні алгоритми розпізнавання об'єктів	04.05.26 – 17.05.26	викон.
4	Обґрунтування вибору програмних засобів та нейромережевої моделі YOLOv8 для реалізації системи	18.05.26 – 25.05.26	викон.
5	Розробка системи детекції об'єктів із використанням OpenCV і YOLOv8	26.05.26 – 26.06.26	викон.
6	Оформлення пояснювальної записки	27.06.26	викон.
7	Подання роботи на перевірку Інтернетсервісом StrikePlagiarisma	28.06.26	викон.
8	Подання роботи на рецензію	29.06.26	викон.
9	Подання роботи на підпис зав. кафедри	30.06.26	викон.
10	Подання кваліфікаційної роботи в ЕК	01.07.26	викон.

Дата видачі завдання 27.04.2026 р.

Здобувач _____ Руслан МИХАЙЛОВ
(підпис) (власне ім'я, прізвище)

Керівник роботи _____ доцент Наталія ДЕМСЬКА
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 61 с., 23 рис., 1 дод., 28 джерел.

СИСТЕМА ТЕХНІЧНОГО ЗОРУ, ОБРОБКА ВІДЕОІНФОРМАЦІЇ, КОМП’ЮТЕРНИЙ ЗІР, РОЗПІЗНАВАННЯ ОБ’ЄКТІВ, ПРОМИСЛОВИЙ РОБОТ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, OPENCV, YOLO.

Об’єкт розробки – процеси обробки відеоінформації в системах технічного зору.

Предмет розробки – алгоритмічне та програмне забезпечення для обробки зображень і розпізнавання об’єктів у системах технічного зору промислових роботів.

Методи дослідження: аналіз сучасних систем технічного зору та алгоритмів обробки зображень, програмна реалізація алгоритмів комп’ютерного зору, тестування розробленого програмного забезпечення.

Мета роботи – дослідження та практичне застосування сучасних алгоритмів комп'ютерного зору й розпізнавання об'єктів для автоматичного виявлення об'єктів у режимі реального часу в системі технічного зору промислового робота та оцінка можливостей їх використання в умовах автоматизованого виробництва.

У кваліфікаційній роботі виконано аналіз систем технічного зору та сучасних методів обробки зображень. Розглянуто алгоритми виявлення та розпізнавання об’єктів, обґрунтовано вибір програмних засобів для реалізації системи. Розроблено програмне забезпечення на базі Python та OpenCV із використанням сучасних алгоритмів комп’ютерного зору.

Результати роботи можуть бути використані в робототехнічних комплексах, системах автоматизованого контролю та навчальних лабораторіях.

ABSTRACT

Explanatory note: 61 pages, 23 figures, 1 appendices, 28 references.

MACHINE VISION SYSTEM, VIDEO INFORMATION PROCESSING, COMPUTER VISION, OBJECT RECOGNITION, INDUSTRIAL ROBOT, SOFTWARE, OPENCV, YOLO.

The object of development – video information processing in machine vision systems.

The subject of the development – algorithmic and software support for image processing and object recognition in machine vision systems of industrial robots.

Research methods: analysis of modern machine vision systems and image processing algorithms, software implementation of computer vision algorithms, testing of the developed software.

The purpose of the work is to increase the accuracy and speed of video information processing in the machine vision system of an industrial robot by using modern computer vision algorithms and object recognition, which ensure their automatic detection in real time and contribute to increasing the efficiency of the robotic complex in automated production.

The qualification work analyzes machine vision systems and modern image processing methods. Object detection and recognition algorithms are considered, and the choice of software tools for implementing the system is justified. Software based on Python and OpenCV is developed using modern computer vision algorithms.

The results of the work can be used in robotic complexes, automated control systems, and training laboratories.

ЗМІСТ

Перелік скорочень 7
Вступ 8
1 Аналіз вимог технічного зору 10
1.1 Системи технічного зору 10
1.2 Класифікація СТЗ 11
1.3 Методи розпізнавання, що використовуються у СТЗ 15
1.4 Вимоги до розроблюваної системи 18
2 Дослідження існуючих методів побудови алгоритмів обробки зображень та відповідних математичних моделей 19
2.1 Немаксимальне придушення 19
2.2 Гістерезис 20
2.3 Метод: подання орієнтації та кутів 21
2.4 Детектор контурів Canny 22
2.5 Алгоритм SUSAN 24
2.6 Інші методи виявлення контурів 26
3 Програмні засоби та нейромережеві технології обробки зображень 28
3.1 Мова програмування Python 28
3.1.1 Переваги і недоліки Python 28
3.1.2 Порівняння Python з іншими мовами програмування 29
3.2 Бібліотека OpenCV 30
3.2.1 Функції OpenCV 31
3.3 Бібліотека YOLOv8 36
3.4 Платформа Roboflow 38
3.5 Порівняння нейромереж 40
4 Розробка та реалізація системи детекції об’єктів 42
4.1 Програмна реалізація 42
4.2 Аналіз умов праці під час роботи з програмним забезпеченням 52

4.3 Вимоги безпеки під час роботи за персональним комп’ютером.....53
4.4 Пожежна безпека.....53
4.5 Висновки 54
Висновки 55
Перелік джерел посилання 56
Додаток А Демонстраційний матеріал.....59

ПЕРЕЛІК СКОРОЧЕНЬ

ПЗС – прилади із зарядовим зв’язком;

СТЗ – системи технічного зору;

AI (Artificial Intelligence) – штучний інтелект;

Focal Loss – функція втрат для покращення навчання моделі при незбалансованих даних;

OpenCV (Open Source Computer Vision Library) – бібліотека комп’ютерного зору з відкритим кодом;

Python – високорівнева мова програмування;

RetinaNet – нейронна мережа для виявлення об’єктів;

SSD (Single Shot Detector) – алгоритм одноетапного виявлення об’єктів;

YOLO (You Only Look Once) – алгоритм виявлення об’єктів у реальному часі;

YOLOv8 (You Only Look Once version 8) – восьма версія алгоритму YOLO.

ВСТУП

На сьогоднішній день обробка відеоінформації є одним із найбільш актуальних напрямів розвитку комп’ютерного зору. Такі технології використовуються в системах автоматичного контролю, робототехніці, системах безпеки, на виробництві та в багатьох інших сферах. Використання відеокамер і програмних засобів для аналізу зображень дозволяє автоматизувати процес виявлення та розпізнавання об’єктів, що спрощує виконання багатьох практичних завдань [1].

Якість роботи систем технічного зору значною мірою визначається алгоритмами, які використовуються для обробки зображень. Сучасні програмні засоби дають можливість не лише отримувати відеодані, а й виконувати їхню фільтрацію, аналіз та подальше розпізнавання об’єктів у кадрі

Метою даної роботи є дослідження та практичне застосування сучасних засобів комп'ютерного зору для обробки відеоінформації та автоматичного розпізнавання об'єктів у системі технічного зору.

Об’єктом розробки є процеси отримання, обробки та аналізу відеоінформації в системах технічного зору промислових роботів.

Предметом дослідження є методи, алгоритми та програмні засоби комп’ютерного зору, призначені для обробки цифрових зображень і розпізнавання об’єктів.

Предметом розробки є програмне забезпечення для обробки відеоінформації та автоматичного розпізнавання об’єктів у системі технічного зору промислового робота.

Для досягнення поставленої мети необхідно вирішити такі задачі:

– провести аналіз сучасних систем технічного зору та особливостей їх застосування у промисловій робототехніці;

– виконати аналіз існуючих методів і алгоритмів обробки зображень та розпізнавання об’єктів;

– обґрунтувати вибір програмних засобів та алгоритмів для реалізації системи;

– розробити програмне забезпечення для обробки відеоінформації;

– реалізувати алгоритми виявлення та розпізнавання об’єктів;

– провести тестування розробленого програмного забезпечення та оцінити ефективність його роботи.

Результати роботи відповідають Цілям сталого розвитку (ЦСР), а саме цілі № 9: Створити гнучку інфраструктуру, сприяти всеосяжній і стійкій індустріалізації і заохочувати інновації [2].

Кваліфікаційну роботу виконано згідно з [3–4]. Результати роботи опублікованя в [1].

1 АНАЛІЗ ВИМОГ ТЕХНІЧНОГО ЗАВДАННЯ

1.1 Системи технічного зору

Система технічного зору (СТЗ) розширює можливості автоматичного керування технологічними процесами, дозволяючи визначати розмір, позицію та орієнтацію вузлів на лінії складання. Однак для реалізації всього потенціалу таких систем велике значення має об’єктив камери, що використовується [5-6].

Для робототехнічних систем, які застосовуються під час складання або контролю великої кількості предметів, наприклад в автомобільній промисловості та на лініях моніторингу, конвеєрна стрічка є контрольним пунктом. На ній роботи виконують дві функції: ідентифікацію та переміщення.

Для більшості систем технічного зору освітлення є однією з найважливіших характеристик. Система повинна забезпечувати високу повторюваність даних, тому для отримання чіткого зображення необхідно мінімізувати вібрацію [7].

На лініях контролю великих партій продукції, наприклад у фармакології, система технічного зору повинна ідентифікувати пошкодження упаковки, нерозбірливі етикетки та відсутність медикаментів, швидко розпізнавати і проводити вимірювання квадратних, круглих і прямокутних об’єктів з високою точністю. Підтримка однакового зовнішнього вигляду та кольору упаковки дозволяє підвищити точність системи. У системах контролю харчових продуктів розмір, колір, щільність і форма визначаються багатокомпонентним обстеженням об’єкта. Багатоелементні системи технічного зору можуть мати кольорові та монохромні камери, використовувати структуроване освітлення для визначення профілю та внутрішнього складу продукту.

Функціональні завдання СТЗ, характерні для робототехнічних застосувань, можна умовно поділити за рівнем їх відносної складності. До стандартних завдань належать: виявлення наявності об’єкта; вимірювання

відстані до об’єкта, його лінійних або кутових переміщень, швидкості; вимірювання геометричних параметрів об’єкта; визначення фізичних характеристик випромінювання об’єкта тощо [8].

1.2 Класифікація СТЗ

Ефективність функціонування СТЗ різних типів значною мірою визначається досконалістю їх структурних схем, які в реальному масштабі часу повинні виділяти зображення окремих об’єктів досліджуваної сцени, здійснювати їх класифікацію, вимірювати відстань між ними, визначати розміри тощо. Залежно від складності аналізованої сцени та типу розташованих на ній об’єктів використовують різні принципи побудови візуальних аналізаторів: активні, пасивні, монокулярні, бінокулярні, такі, що формують далекомірну картину тощо [8].

Під час класифікації СТЗ можуть використовуватися різні класифікаційні схеми. Так, СТЗ можуть бути поділені таким чином:

а) двовимірні, тривимірні та такі, що мають розмірність 2,5 (зі структурованим підсвічуванням);

б) бінарні та багатотональні;

в) активні (з підсвічуванням) і пасивні (без підсвічування) тощо.

Як перетворювачі світло-сигналу, що реалізують процес зчитування зображення, у СТЗ застосовують як вакуумні телевізійні передавальні трубки, так і твердотільні матричні перетворювачі зображення – прилади із зарядовим зв’язком (ПЗС). Методи збору та передавання інформації здебільшого запозичені з телебачення: використовуються растрові способи розгортки зображень. Однак для отримання зображень потрібних відомостей застосовують методи обробки відеоінформації, зокрема ідентифікацію та розпізнавання образів, інтерпретацію сцен та інші методи, не характерні для звичайної телевізійної техніки. Відмінною особливістю задач аналізу, контролю та автоматичної класифікації зображень є їх багатовимірність. У

найпростішому випадку зображення характеризується двома просторовими координатами, до яких додається третя – часова. Зображення, отримані в різних діапазонах електромагнітного спектра (видимому, інфрачервоному та ультрафіолетовому), як правило, містять дуже великий обсяг інформації, яку необхідно обробити для їх ідентифікації, виявлення відмінностей і прийняття рішень. Це зумовлює жорсткі вимоги до систем візуального контролю та класифікації щодо роздільної здатності й продуктивності. Крім того, більшість зображень характеризується наявністю завадового фону, а також невизначеністю положення й орієнтації окремих елементів, що призводить до значної надлишковості інформації. Це зумовлює необхідність використання методів попередньої обробки зображень: фільтрації, нормування, згладжування тощо.

Процес автоматичного аналізу візуальної інформації, як правило, є складною багатоступеневою процедурою. Після завершення операції розгортки зображення у відеодатчику інформація в аналоговій або цифровій формі надходить на наступні етапи обробки. Аналіз візуальної інформації є процесом, що базується на використанні аналізуючої структури, чутливої до різних рівнів деталізації зображення. При цьому застосовують різноманітні методи обробки, ідентифікації та розпізнавання двовимірних і тривимірних зображень, зокрема стереоскопічні методи, методи активного зондувального підсвічування тощо. Під час реалізації процесу ідентифікації об’єктів у реальному масштабі часу дуже важливим питанням є виділення оптимальних ознак ідентифікації об’єктів. Існує багато способів розпізнавання об’єктів, застосування яких пов’язане з необхідністю виконання великої кількості машинних процедур і, відповідно, наявністю ЕОМ із великою ємністю пам’яті. У СТЗ використовують спрощені методи ідентифікації об’єктів за їх геометричними ознаками, характерними конструктивними елементами тощо. Ознаки ідентифікації визначаються в режимі навчання системи та зберігаються в пам’яті ЕОМ як еталонні, з якими порівнюються ті самі параметри, що визначаються під час роботи системи. Перспективними для використання в СТЗ є різні алгоритми

самонавчання, які визначають ознаки ідентифікації на етапі навчання під час кількох пред'явлень об'єкта. Іноді еталонами виступають не окремі ознаки, а шаблони зображення [8].

Положення об'єкта визначається за координатами його бінарного силуету. Орієнтацію можна визначити, наприклад, за напрямком, що відповідає найбільшому радіусу, або за напрямком на найбільший отвір. У зв'язку зі значним прогресом у галузі обчислювальної техніки нині в автоматичному розпізнаванні образів досягнуто певних успіхів. Роботи ведуться переважно за двома напрямами. Перший напрям – це створення систем обробки й аналізу зображень для різних автоматичних виробничих комплексів. Ці системи призначені для розв'язання таких завдань, як ідентифікація, класифікація та сортування деталей, контроль якості їх поверхонь тощо. Другий напрям пов'язаний зі створенням автоматичних пристроїв для зчитування етикеток, маркувань, чеків тощо. Інтенсивні роботи ведуться зі створення технічних засобів для систем аналізу двовимірних і тривимірних сцен. Однак основна частина досліджень зосереджена на проблемах обробки, аналізу та використання візуальної інформації (розробка програм, алгоритмів і засобів їх апаратної реалізації) [9].

Алгоритм обробки двовимірної візуальної інформації за допомогою СТЗ зазвичай включає такі процедури: отримання даних про зображення, бінаризацію зображення, сегментацію, виділення характерних ознак, ідентифікацію або класифікацію об'єктів, визначення положення об'єктів і передачу даних до виконавчих органів. Таким чином, функції, що виконуються СТЗ, можна поділити на три групи: ідентифікація форми; визначення просторового положення; виділення сигналу контролю або керування.

Ідентифікація форми необхідна для того, щоб відрізнити одну деталь від іншої, визначити, чи потрібна дана деталь для виконання технологічної операції. Для розв'язання цього завдання часто використовується метод зіставлення образів. Алгоритми визначення положення об'єкта зазвичай ґрунтуються на аналізі геометричних характеристик зображення, таких як

геометричний центр, центр мас, головні осі інерції об’єкта тощо. Іноді для цієї мети використовують характерні особливості контуру або його ділянки (злами, прямолінійні відрізки тощо) [9].

В даний час найбільш розробленими є системи аналізу двовимірних зображень, що зазвичай працюють з одним відеодатчиком. Такі системи, як правило, мають у своїй архітектурі чотири ієрархічні рівні, які взаємодіють між собою [10]:

а) відновлення зображення – отримання достовірного зображення, корекція спотворень, внесених приймальними пристроями та завадами, фільтрація тощо;

б) сегментація – перетворення зображення у бінарну форму, розкладання зображення на фрагменти, що відповідають окремим об’єктам або їх частинам тощо;

в) опис зображень – визначення узагальнюючих ознак, наприклад морфометричних, текстурних, мультиспектральних тощо;

г) аналіз – узгодження з моделлю, класифікація, варіювання ознак за допомогою мов високого рівня, інтерпретація зображення тощо.

Перші два ієрархічні рівні є процедурами попередньої обробки зображень і реалізуються за допомогою спеціальних засобів математичного та апаратного забезпечення. Для реалізації вищих ієрархічних рівнів СТЗ необхідні засоби, технічний рівень яких дозволяє віднести їх до систем штучного інтелекту.

Розв’язання завдань зазначених рівнів потребує використання проблемно-орієнтованих мов, застосування моделей простору, банків даних тощо.

Як було зазначено вище, СТЗ можна класифікувати залежно від того, яким чином у них здійснюється обробка зображень, на бінарні та багатотонові. У бінарних двовимірних системах кожна точка зображення описується в термінах двох тонів – чорного та білого, а в багатоградаційних системах точки зображення можуть мати напівтони. Перехід від напівтонових зображень до бінарних здійснюється за допомогою операції порівняння з порогом. У бінарних системах об’єкти спостерігаються або у відбитих, або у прохідних

променях, при цьому похиле освітлення в деяких випадках використовується для отримання тіней, за якими можна оцінити висоту об’єкта. Процедури обробки відеоінформації в бінарних системах простіші і, відповідно, швидкість обробки вища. Такі системи завдяки меншій інформаційній ємності зображення мають підвищену швидкодію порівняно з напівтоновими системами, у яких яскравість кодується значно більшою кількістю рівнів [10].

Водночас багатоградаційні системи дозволяють отримати значно більше інформації про об’єкт, ніж бінарні. Якщо відбивна здатність, текстура або форма деталі такі, що її важко виділити з фону, необхідно обробляти напівтонові зображення. Такі СТЗ дають змогу здійснювати більш детальний аналіз відеоінформації, однак у реальному масштабі часу ці системи поки що працюють лише у відносно вузьких галузях [11].

1.3 Методи розпізнавання, що використовуються у СТЗ

Розпізнавання двовимірних зображень може бути виконане на основі використання таких методів:

а) зіставлення з еталоном;

б) заснованих на теорії прийняття рішень;

в) синтаксичних.

У методах першої групи під час аналізу зображень використовується набір еталонних зображень – представників кожного типу об’єктів. У процесі класифікації аналізоване зображення порівнюється з кожним еталоном.

Результати порівняння дозволяють зробити висновок про належність об’єкта до того чи іншого класу. Такий підхід є найбільш простим під час розв’язання задач розпізнавання, оскільки вихідні зображення не піддаються обробці з метою виділення певних характеристик, а виконується лише попередня обробка для очищення зображення від шумів. Процес прийняття рішення полягає у порівнянні з усіма еталонами та обчисленні критерію узгодженості (наприклад, кореляційної функції). Однак такий підхід потребує

значних обчислювальних витрат, оскільки порівняння необхідно виконувати для всіх можливих положень об’єкта в полі зображення. Розроблено різні алгоритми порівняння з еталоном бінаризованих зображень.

Під час бінаризації в діапазоні яскравості зображення обирається поріг, порівняння з яким дає «чорні» силуети об’єктів на «білому» фоні. У тому випадку, якщо деталь може займати невелику кількість фіксованих положень і орієнтацій (що може бути досягнуто, наприклад, за допомогою механічних упорів), для її розпізнавання можна успішно застосувати метод зіставлення з еталоном [12].

У методах розпізнавання, заснованих на теорії прийняття рішень, використовуються класичні підходи до розпізнавання. За таких підходів об’єкти представляються у вигляді N-вимірних векторів у просторі ознак, а процес прийняття рішення базується на обчисленні міри близькості спостережуваного вектора до області визначення шуканого об’єкта. Ця міра близькості може бути виражена як відстань, імовірнісна характеристика тощо. Як правило, при використанні такого підходу попередньо виконується навчання системи, обираються найбільш інформативні ознаки, які використовуватимуться під час прийняття рішення. Оскільки в методах цієї групи зображення описується в термінах певних кількісних характеристик, немає необхідності зберігати еталонне зображення [12].

Класифікація здійснюється відповідно до статистичних методів за багатовимірними функціями щільності умовної ймовірності. Основними етапами обробки відеоінформації при використанні таких методів є сегментація зображення, виділення ознак і класифікація. Завдання етапу сегментації полягає у розбитті зображення на області, що відповідають різним об’єктам.

Після сегментації зображення виконується виділення ознак, які повинні бути інваріантними до положення й орієнтації об’єктів і водночас однозначно їх розрізняти. Розпізнавання ґрунтується на морфологічних ознаках, у ролі яких використовуються такі характеристики форми, як площа силуету, його периметр, відношення площі до периметра, відношення моментів інерції,

максимальний і мінімальний радіуси, площа та кількість отворів, протяжність тощо. Сукупність цих характеристик утворює опис об’єкта, за яким здійснюється класифікація [13].

Третю групу методів розпізнавання зображень становлять синтаксичні методи. При цьому підході зображення подається як речення (конструкція мови, яка визначається введеною граматикою); для опису об’єктів і сцен використовуються формальні граматики, а процес прийняття рішення являє собою синтаксичний аналіз.

Як уже було зазначено, СТЗ можуть бути поділені на двовимірні, тривимірні та системи з розмірністю 2,5. Розглянуті вище двовимірні системи призначені для ідентифікації плоскої робочої сцени, форма об’єктів у якій однозначно визначається за двовимірною проєкцією. Більшість існуючих СТЗ працює з двовимірними зображеннями. Такі СТЗ реалізуються переважно за допомогою відеодатчиків, що працюють при розсіяному освітленні. Системи з розмірністю 2,5 додатково забезпечують визначення окремих просторових параметрів; для цих цілей у них використовуються допоміжні освітлювальні засоби [13].

Тривимірні системи дозволяють ідентифікувати будь-які просторові властивості об’єктів незалежно від вибору позиції спостереження.

Найбільший економічний ефект системи технічного зору дозволяють отримати у тих випадках, коли:

а) можливо однозначно визначити форму та габарити виробу;

б) для опису виробу використовується обмежений набір характеристик;

в) візуальний контроль виконується в обмеженому обсязі;

г) розміри виробу допускають використання візуального контролю;

д) існує чіткий контраст між виробом і фоном.

1.4 Вимоги до розроблюваної системи

Вся система при цьому повинна відповідати таким вимогам:

- обробка інформації повинна відбуватися в режимі реального часу;
- виконувати автоматичне виявлення та розпізнавання заданого об’єкта у кадрі;
- обробка інформації повинна відбуватися в режимі реального часу;
- відображати результати розпізнавання на відеозображенні;
- мати низьке споживання обчислювальних ресурсів;
- бути надійною;
- використовувати доступні програмні засоби та апаратне забезпечення.

Для реалізації системи технічного зору використовується персональний комп’ютер, на якому виконується обробка відеоінформації та робота алгоритмів розпізнавання об’єктів.

2 ДОСЛІДЖЕННЯ ІСНУЮЧИХ МЕТОДІВ ПОБУДОВИ АЛГОРИТМІВ ОБРОБКИ ЗОБРАЖЕНЬ ТА ВІДПОВІДНИХ МАТЕМАТИЧНИХ МОДЕЛЕЙ

2.1 Немаксимальне придушення

За відомими оцінками величини градієнта потрібно знайти крайові точки. Величину градієнта можна розглядати як ланцюг невисоких пагорбів. Локальні екстремуми дають окремі точки – за аналогією з вершинами пагорбів. Найкращий критерій – розбити градієнт на маленькі ділянки вздовж напрямку градієнта, який має бути перпендикулярним до краю, і відмітити точки ділянки, в яких ця величина максимальна [14]. У результаті вийде ряд точок уздовж вершин початкового ланцюга пагорбів; цей процес називається немаксимальним пригніченням, алгоритм роботи якого такий:

а) поки є неперевірені точки з великим значенням градієнта;

б) знайти початкову точку, яка є локальним максимумом у напрямку, перпендикулярному до градієнта, виключаючи вже перевірені точки;

в) поки можливо, розширюємо ланцюг, що проходить через поточну точку, для цього:

1) передбачаємо набір наступних точок за допомогою напрямку, перпендикулярного до градієнта;

2) знаходимо точку (якщо така є) локального максимуму в напрямку градієнта;

3) перевіряємо, чи є величина градієнта в максимумі достатньо великою;

4) відмічаємо, що ця точка та її околиці вже перевірені;

г) фіксуємо наступну точку, яка стає поточною.

Як правило, очікується, що крайові точки будуть розташовані вздовж кривої у вигляді ланцюжка. Перелічимо важливі етапи немаксимального пригнічення:

а) визначити, чи є дана точка крайовою;

б) якщо це так, знайти наступну крайову точку.

Після обмірковування цих етапів неважко виділити всі ланцюжки крайових точок. Знаходимо першу крайову точку, відмічаємо її, продовжуємо всі ланцюги, що проходять через цю точку, відмічаючи всі точки на цих ланцюгах, і робимо те саме для всіх невідмічених крайових точок.

Два основні етапи дуже прості. Припустимо поки, що краї слід відмічати за положеннями пікселів (а не за якимось більш дрібним поділом сітки пікселів). Визначити, чи є величина градієнта в даному пікселі максимальною, можна, порівнюючи її зі значеннями в точках праворуч і ліворуч уздовж напрямку градієнта. Це функція відстані вздовж напрямку градієнта; як правило, на наступному кроці виконується перехід праворуч на наступний рядок (або стовпець) пікселів і ліворуч на попередній, щоб визначити, чи є величина градієнта в даному пікселі більшою. Вектор градієнта зазвичай не проходить через наступний піксель, тому виконують інтерполяцію, щоб визначити значення градієнта в потрібних точках; інтерполяція, як правило, лінійна [12].

Якщо піксель виявляється крайовою точкою, то наступну крайову точку кривої можна передбачити, зробивши крок перпендикулярно до градієнта. У загальному випадку цей крок не закінчується на пікселі; природна реакція – розглянути сусідні пікселі, що знаходяться поблизу цього напрямку. Такий підхід дає набір кривих, які можна представити, зобразивши їх на чорному або білому фоні [7].

2.2 Гістерезис

Занадто багато з отриманих кривих дозволяють адекватно описати межі об’єкта. Частково це зумовлено тим, що максимуми величини градієнта визначалися незалежно від того, наскільки велике значення цього максимуму.

Частіше застосовують порогову перевірку, щоб переконатися, що максимум перевищує певну нижню межу. Це, своєю чергою, призводить до розриву кривої краю. У такому випадку зазвичай застосовується концепція гістерезису: існують два порогові значення, і на початку ланцюга використовується більше, а потім, рухаючись уздовж нього, – менше. Цей прийом допомагає покращити отриманий результат [10].

2.3 Метод подання орієнтації та кутів

Добре відомо, що визначати краї на кутах складно, оскільки тут припущення про те, що для оцінки орієнтації градієнта достатньо оцінити частинні похідні за x та y, не спрацьовує. У гострих або невдало накладених кутах ці оцінки частинних похідних зазвичай виявляються неправильними, оскільки їх носій перетинатиме кут. Існує багато спеціальних детекторів країв на кутах, у яких шукаються області зображення з сильними коливаннями градієнта. У більш загальному випадку дійсно корисний опис області зображення дає статистика градієнта області [8]. Вікна зображення умовно можна поділити на чотири типи:

– з гладкою вибіркою, коли рівень яскравості приблизно дорівнює константі;

– зі ступінчастою вибіркою, де в межах вікна є різкі зміни яскравості зображення вздовж одного напрямку;

– з анізотропною текстурою, де в межах вікна є кілька дрібних ліній (наприклад, волосся або шерсть);

– з ізотропною текстурою, де в межах вікна є якась двовимірна текстура (наприклад, плями або кут).

Усі ці ситуації відповідають різним типам поведінки градієнта зображення. У вікнах із гладкою вибіркою вектор градієнта короткий; у вікнах зі ступінчастою вибіркою існує невелика кількість довгих векторів градієнта, які всі спрямовані в один бік; у вікнах з анізотропною текстурою існує багато векторів градієнта, що вказують у двох напрямках; а у вікнах з ізотропною

текстурою вектор градієнта коливається. Ці відмінності можна досить легко зобразити, розглянувши коливання орієнтації в межах вікна. Зокрема, матриця добре описує поведінку орієнтації в межах вікна. У вікні з гладкою вибіркою обидва власні значення цієї матриці малі, оскільки всі елементи малі. У вікні зі ступінчастою вибіркою можна очікувати, що одне власне значення, пов'язане з градієнтами краю, буде великим, а інше – малим, оскільки кілька градієнтів спрямовані в інші боки. У вікні з анізотропною текстурою можна очікувати подібні власні значення, за винятком того, що велике значення буде ще більшим, оскільки країв, що роблять внесок, буде більше. Нарешті, у вікні з ізотропною текстурою обидва власні значення будуть великими.

Поведінку цієї матриці найлегше зрозуміти, побудувавши еліпси

$$(x, y)^T H^{-1}(x, y) = \varepsilon. \tag{2.1}$$

для деякої малої константи ε. Ці еліпси накладаються на вікна зображення. Їхні велика та мала осі спрямовані вздовж власних векторів матриці H, а довжина великої та малої осей відповідає величинам власних значень. Це означає, що велике коло відповідає вікну з ізотропною текстурою, а вузький витягнутий еліпс вказує на вікно зі ступінчастою вибіркою. Таким чином, кути можна позначати, відмічаючи точки, у яких площа цих еліпсів має екстремальне та велике значення. Точність локалізації, яку пропонує цей метод, обмежена розміром вікна та поведінкою градієнта. Більш точної локалізації можна досягти за рахунок більш детальної моделі шуканого кута [9].

2.4 Детектор контурів Canny

John Canny описав алгоритми виявлення контурів, які з того часу стали одними з найширше використовуваних. Можна сказати, що вони стали класикою у галузі виявлення контурів. Canny виходив із трьох критеріїв, яким повинен задовольняти детектор контурів:

а) добре виявлення – Canny трактував цю властивість як підвищення співвідношення сигнал/шум;

б) хороша локалізація (правильне визначення положення контуру);

в) єдиний відгук на один контур.

З цих критеріїв потім будувалася цільова функція вартості помилок, мінімізацією якої знаходять «оптимальний» лінійний оператор для згортки зображення.

Було показано, що оптимальний фільтр є дуже близьким наближенням до першої похідної Гаусіана, тобто являє собою поєднання операторів градієнта та гаусового згладжування. Причому, якщо автори алгоритму LoG виходили з того, що згладжуючий і диференціюючий оператори можна об’єднати, то Canny будував оптимальний детектор контурів і прийшов до того ж об’єднання. Формула оптимального за Canny оператора для виявлення контуру (в одновимірному випадку) має вигляд:

$$f(x) = -x \cdot \exp\left[-\frac{x^2}{2s^2}\right]. \qquad (2.2)$$

У двовимірному випадку похідна має братися у напрямку, перпендикулярному до контуру, який попередньо оцінюється за напрямком градієнта згладженого зображення.

Алгоритм детектора контурів Кенні не обмежується обчисленням градієнта згладженого зображення. У контурі контуру залишаються лише точки максимуму градієнта зображення, а немаксимальні точки, що лежать поруч із контуром, видаляються. Тут також використовується інформація про напрямок контуру, щоб видаляти точки саме поруч із контуром і не розривати сам контур біля локальних максимумів градієнта. Потім за допомогою двох порогів видаляються слабкі контури. Фрагмент контуру при цьому обробляється як ціле. Якщо значення градієнта десь на відстежуваному фрагменті перевищує верхній поріг, то цей фрагмент залишається «допустимим» контуром, а в тих місцях, де значення градієнта падає нижче цього порогу, він залишається до тих

пір, поки не стане нижче нижнього порогу. Якщо ж на всьому фрагменті немає жодної точки зі значенням, більшим за верхній поріг, то він видаляється. Такий гістерезис дозволяє зменшити кількість розривів у вихідних контурах [12].

Масштаб Гаусіана sigma визначає величину придушення шуму: чим ширший Гаусіан, тим більший ефект згладжування. Проте збільшення цього масштабу знижує точність локалізації контуру [13].

Включення в алгоритми LoG та Canny шумопригнічення, з одного боку, підвищує стійкість результатів, а з іншого – збільшує обчислювальні витрати та призводить до спотворення і навіть втрати деталей контурів. Так, наприклад, такими алгоритмами скруглюються кути об’єктів і руйнуються контури в точках з’єднань. Проте досягнення повної зв’язності в з’єднаннях буває необхідним для правильного подання сцени. Тому були запропоновані евристичні доповнення до алгоритму Canny, які дозволяють з’єднувати незамкнений кінець контуру з близькими контурами. Однак у деяких випадках це призводить до появи хибних контурів [8].

2.5 Алгоритм SUSAN

Один із новітніх алгоритмів детектора контурів був розроблений авторами з оборонного дослідницького агентства Великої Британії у 1995 році та опублікований у 1996 році. Його назва – SUSAN є абревіатурою англійських слів Smallest Univalue Segment Assimilating Nucleus (найменший однорідний сегмент, асимільований ядром) [10].

Smith & Brady звернули увагу на те, що сусіди кожної точки в однорідній області мають яскравість, близьку до неї, а поблизу контуру число сусідів з однаковою яскравістю зменшується. Відповідно до такого трактування контурів був розроблений алгоритм детектора контурів SUSAN. Крім контурів, цей метод також виявляє інші особливості на зображенні (кути, тонкі лінії тощо).

Навколо кожного пікселя зображення будується маска, центральний піксель якої називається ядром (у роботі використовується кругла маска з

радіусом 3,4 пікселя, яка включає 37 пікселів, та традиційна квадратна маска 3×3). Пікселі в межах маски, які мають яскравість, порівнянну з ядерним, утворюють область USAN (Univalue Segment Assimilating Nucleus – однорідний сегмент, асимільований ядром). Для виявлення двовимірних особливостей і контурів використовують розмір, центр маси та другі моменти USAN. Такий підхід до виявлення особливостей відрізняється від відомих методів тим, що не використовує похідних зображення і, відповідно, не потребує попереднього придушення шуму [14].

Площа USAN максимальна, коли ядро знаходиться в однорідній (або майже однорідній) області зображення, зменшується до половини цього максимуму поблизу прямого контуру і зменшується ще більше поблизу кута, досягаючи локальних мінімумів точно на контурі та в кутах. Це властивість площі USAN використовується як головний критерій присутності контурів і двовимірних особливостей.

Яскравість кожного пікселя в межах маски порівнюється з яскравістю ядра (середньої точки) відповідно до виразу (2.7):

$$c(r, r_0) = \left[\frac{l(r)-l(r_0)}{t}\right]. \quad (2.3)$$

де r_0 – положення ядра;

r – положення будь-якої іншої точки в межах маски;

l(r) – яскравість пікселя r;

c – результат порівняння.

Результати порівняння для всіх пікселів у межах маски сумуються:

$$n(r_0) = \sum_{i=1}^{N} c(r_i, r_0). \quad (2.4)$$

Отримана сума n являє собою число пікселів у USAN, тобто її площу. Потім n порівнюється з фіксованим геометричним порогом g. При виявленні контурів за відсутності шуму цей поріг не потрібен.

2.6 Інші методи виявлення контурів

Один із підходів згладжування зображення ґрунтується на представленні зміни зображення як еволюції його в штучному часі та розв’язанні відповідних рівнянь у частинних похідних:

$$\frac{dl}{dt} = D(l). \tag{2.5}$$

Для виявлення контурів дифузійний член D повинен бути невозростаючою функцією градієнта згладженого зображення, що прямує до нуля на нескінченності. Такою, наприклад, є функція 1/x. Там, де модуль градієнта великий, дифузія мала, і зображення практично не змінюється. Там, де модуль градієнта малий, дифузія велика, і зображення згладжується [15].

Задача виявлення контурів значно спрощується, коли заздалегідь відомо, що потрібно знайти на сцені. Lakshmanan та інші розглянули задачу виявлення контурів, що являють собою дві паралельні прямі лінії. Зображення сцени отримуються в міліметровому діапазоні і, залежно від діапазону, можуть або містити багато дрібних непотрібних деталей (вузький діапазон), або бути слабоконтрастними (широкий діапазон). Стандартні детектори типу Canny в таких умовах не дають хороших результатів. Враховуючи, що контури на сцені являють собою дві паралельні прямі лінії:

$$\begin{aligned} r \cdot \sin(TET) &= k \cdot \cos(TET) + C_1, \\ r \cdot \sin(TET) &= k \cdot \cos(TET) + C_2. \end{aligned} \tag{2.6}$$

Задається деякий початковий шаблон kC_1C_2, який потім «деформують» шляхом зміни kC_1C_2 (виконується 1000 ітерацій за алгоритмом Метрополіса) і перевіряють «схожість» отриманого зображення з вихідним.

Noble у своїй роботі використовує математичну морфологію для знаходження структури зображення [16]. Описується кілька різних морфологічних операцій, які застосовуються для покращення контуру та виявлення двовимірних особливостей. Оператор «erode-dilate» подібний до похідної першого порядку, а оператор «open-close» – до похідної другого порядку. Контури хорошої якості виявляються при відстеженні з обох боків кожного контуру, після чого ці «половини контурів» зшиваються разом.

Зв’язність у з’єднаннях, як правило, добра, хоча іноді з’являються хибні короткі «хвости» у структурах типу Т-з’єднань. Алгоритм, а особливо частина відстеження контуру, потребує досить великих обчислювальних витрат [17].

3 ПРОГРАМНІ ЗАСОБИ ТА НЕЙРОМЕРЕЖЕВІ ТЕХНОЛОГІЇ ОБРОБКИ ЗОБРАЖЕНЬ

3.1 Мова програмування Python

Python – це високорівнева мова програмування загального призначення, яка широко використовується в сучасних інформаційних технологіях, зокрема в галузях обробки даних, штучного інтелекту та комп'ютерного зору.

Основною перевагою Python є його простий та зрозумілий синтаксис, що дозволяє швидко розробляти програмне забезпечення навіть для складних задач. Це особливо важливо при створенні систем обробки відеоінформації, де необхідно працювати з великою кількістю даних та алгоритмів.

Python також підтримує інтеграцію з сучасними фреймворками машинного навчання, такими як PyTorch, що дозволяє ефективно навчати нейронні мережі навіть на персональних комп'ютерах.

3.1.1 Переваги і недоліки Python

Переваги Python:

- простий синтаксис – код читається майже як англійська мова, легко вчити навіть новачкам;
- швидка розробка – менше коду для тих самих задач у порівнянні з C++ чи Java;
- велика кількість бібліотек – є готові рішення для AI, вебу, аналізу даних, автоматизації;
- популярність – величезна спільнота, багато прикладів і документації;
- кросплатформеність – працює на Windows, Linux, macOS без суттєвих змін коду;
- гнучкість – підходить для різних сфер: від скриптів до штучного

інтелекту.

Недоліки Python:

– низька швидкість виконання – повільніший за компільовані мови (C/C++);

– велике споживання пам’яті – не найкращий вибір для слабких пристроїв;

– динамічна типізація – помилки можуть з’являтися лише під час виконання програми;

– обмеження у мобільній розробці – рідше використовується для Android/iOS;

– GIL (Global Interpreter Lock) – ускладнює повноцінну багатопоточність.

3.1.2 Порівняння Python з іншими мовами програмування

У сучасному світі програмування існує велика кількість мов, кожна з яких має свої особливості, переваги та сфери застосування. Вибір мови програмування залежить від типу задачі, вимог до швидкодії, складності проєкту та рівня підготовки розробника [18].

Python vs C++ – Python простіший у вивченні та написанні коду, має більш читабельний синтаксис, але працює повільніше. C++ значно швидший і використовується там, де важлива продуктивність (ігри, системне ПЗ), але він складніший у навчанні та вимагає більше контролю з боку програміста.

Python vs Java – Python дозволяє писати код швидше і з меншою кількістю рядків, підходить для прототипування та штучного інтелекту. Java більш строгий, має статичну типізацію, краще підходить для великих корпоративних систем і Android-розробки.

Python vs JavaScript – Python є універсальною мовою і використовується в різних сферах: AI, автоматизація, backend. JavaScript в основному застосовується для веб-розробки і працює у браузері, а також на сервері через Node.js. JavaScript більш спеціалізований для інтернет-додатків.

3.2 Бібліотека OpenCV

OpenCV – це відкрита бібліотека комп'ютерного зору, яка використовується для обробки зображень та відео в реальному часі. Вона є однією з найпопулярніших бібліотек у сфері аналізу зображень і широко застосовується в наукових дослідженнях та промислових системах (рис. 3.1).

Основне призначення OpenCV полягає у наданні готових інструментів для роботи з цифровими зображеннями, що значно спрощує розробку програм, пов'язаних із відеоаналізом та розпізнаванням об'єктів [14-19].

Бібліотека підтримує мову Python, що дозволяє швидко інтегрувати її у програмні проєкти. Вона забезпечує ефективну обробку даних завдяки оптимізованим алгоритмам та можливості роботи в реальному часі.



Рисунок 3.1 – Логотип Open CV

OpenCV включає широкий набір функцій, серед яких:

- зчитування та збереження зображень;
- обробка відеопотоку з камери;
- зміна розмірів і геометрії зображень;
- фільтрація та покращення якості зображення;
- виявлення об'єктів і облич;
- накладання графічних елементів і тексту.

Важливою перевагою OpenCV є її універсальність – вона може використовуватись як для простих задач (наприклад, зміна кольору

зображення), так і для складних систем комп’ютерного зору, включаючи інтеграцію з нейронними мережами.

3.2.1 Функції OpenCV

Для роботи із зображеннями бібліотека OpenCV застосовує різні структури даних, зокрема скалярні значення, вектори, матриці та діапазони. Завдяки цьому стає можливим виконання різноманітних математичних операцій над зображеннями, їх обробка, аналіз та застосування перетворень. Такі інструменти забезпечують гнучкість у роботі з піксельними даними та дозволяють реалізовувати широкий спектр функцій комп’ютерного зору.

Результати перетворення зображень представлені на рис. 3.2 – рис. 3.9 та відповідні описи програмної реалізації.



Рисунок 3.2 – Кадрування зображення

```
import cv2
image = cv2.imread("screen/scr2.png")
cropped = image[10:500, 250:1500]
cv2.imshow("Фото після кадрування", cropped)
cv2.waitKey(0)
```

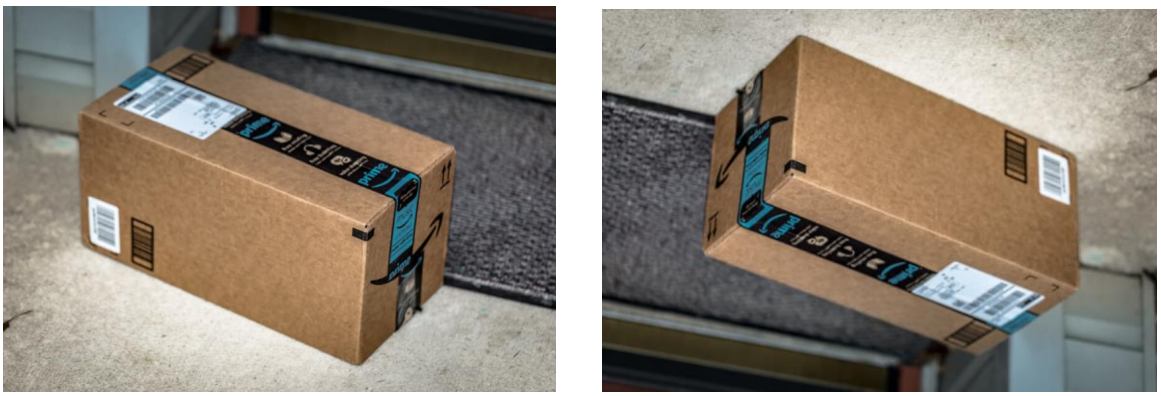


Рисунок 3.3 – Обертання зображення

```
import cv2
image = cv2.imread("screen/scr2.png")
rotated = cv2.rotate(image, cv2.ROTATE_180)
cv2.imshow("Поворот 180", rotated)
cv2.waitKey(0)
```

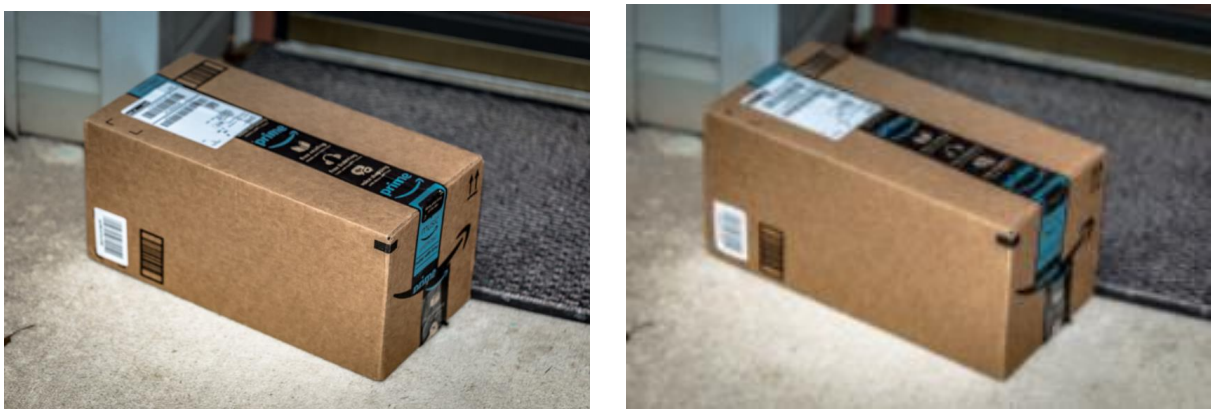


Рисунок 3.4 – Зміна зображення на 30%

```
import cv2
img = cv2.imread("screen/scr2.png")
if img is None:
    print("Зображення не знайдено! Перевір шлях до файлу.")
else:
    scale_percent = 30
    width = int(img.shape[1] * scale_percent / 100)
```

```
    height = int(img.shape[0] * scale_percent / 100)
    dim = (width, height)
    resized = cv2.resize(img, dim, interpolation=cv2.INTER_AREA)
    print("Зображення успішно змінено (ресайз завершено)")
    cv2.imshow("Змінене зображення", resized)
    cv2.waitKey(0)
```



Рисунок 3.5 – Переведення зображення у сірий градієнт

```
import cv2
img = cv2.imread("screen/scr2.png")
if img is None:
    print("Зображення не знайдено! Перевір шлях до файлу.")
else:
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    print("Зображення успішно перетворено в градації сірого")
    cv2.imshow("Сіре зображення", gray)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

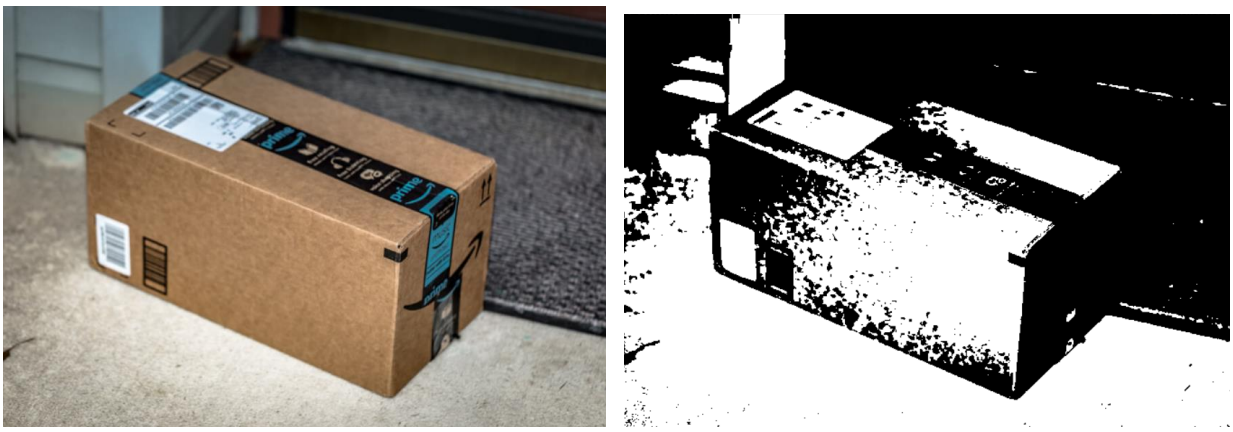


Рисунок 3.6 – Переведення зображення у чорно-біле

```
import cv2
img = cv2.imread("screen/scr2.png")
if img is None:
    print("Зображення не знайдено! Перевір шлях до файлу.")
else:
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    _, binary = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)
    print("Зображення успішно перетворено в чорно-біле (binary)")
    cv2.imshow("Чорно-біле зображення", binary)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
```



Рисунок 3.7 – Додавання лінії на зображення

```
import cv2
image = cv2.imread("screen/scr2.png")
if image is None:
    exit()
output = image.copy()
cv2.line(output, (350, 20), (350, 400), (0, 255, 0), 5)
cv2.imshow("2 собаки, розділенні лінією", output)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



Рисунок 3.8 – Накладання прямокутної області на зображення

```
import cv2
image = cv2.imread("screen/scr2.png")
if image is None:
    exit()
output = image.copy()
cv2.rectangle(output, (100, 70), (550, 400), (0, 255, 0), 5)
cv2.imshow("Квадрат на зображенні", output)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



Рисунок 3.9 – Накладання тексту на зображення

```
import cv2
image = cv2.imread("screen/scr2.png")
output = image.copy()
cv2.putText(
    output,
    "boxes",
    (250, 450),
    cv2.FONT_HERSHEY_SIMPLEX,
    2,
    (0, 255, 0),
    3
)
cv2.imshow("Зображення з текстом", output)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

3.3 Бібліотека YOLOv8

YOLOv8 – це версія фреймворку штучного інтелекту YOLO, який застосовується для вирішення різноманітних задач комп'ютерного зору, а саме виявлення, сегментації, класифікації об’єктів та розпізнавання стану (рис. 3.10).



Рисунок 3.10 – Візуалізація задач комп’ютерного зору

Розпізнавання об’єктів. Це основне завдання, яке вирішує фреймворк YOLOv8. Воно включає в себе виявлення об'єктів на зображенні або відеокадрі та виділення їх рамкою. Виявлені об'єкти класифікуються по різним категоріям на основі їх характеристик. YOLOv8 може виявляти кілька об'єктів на одному зображенні або відеокадрі з високою точністю та швидкістю [20].

Сегментація. Це функціонал, який включає поділ зображення на різні області на основі його вмісту. Кожній області призначається мітка відповідно до її вмісту. Використання цієї функції фреймворку доцільно в застосунках, які використовують сегментацію зображень у медичній сфері, обробці супутникових знімків. Також цей функціонал може бути використаний у системах автономного керування автомобілем, розважальних застосунках віртуальної реальності тощо.

Класифікація. Це завдання, що включає віднесення зображення до певної категорії із заданого набору. Воно знаходить широке застосування в різних сферах. У медичній діагностиці класифікація може бути використана для автоматизованої ідентифікації патологій, у системах безпеки – для розпізнавання облич та виявлення подій, в автономному керуванні автомобілем – для ідентифікації різних транспортних засобів, у сільському господарстві – для розрізнення рослин і шкідників тощо.

Оцінювання стану. Це завдання, що передбачає виявлення конкретних точок на зображенні або відеокадрі, які використовуються для відстеження

руху або оцінки стану. Воно має широке застосування в різних галузях – анімації та ігрової індустрії, фітнес-технологіях, віртуальній реальності, відеоспостереженні медичній діагностиці та розробці автономних систем.

На рисунку 3.11 зображено схему етапів роботи нейронної мережі для обробки відеопотоку.



Рисунок 3.11 – Блок-схема роботи системи комп’ютерного зору

3.4 Платформа Roboflow

Roboflow – це сучасна хмарна платформа, призначена для спрощення процесу створення, підготовки та розгортання моделей комп’ютерного зору. Вона об’єднує в собі інструменти для роботи з наборами зображень, їх розмітки, а також подальшого навчання та експорту моделей машинного

навчання. Для підготовки даних до навчання нейронної мережі використовувалося середовище Roboflow. Воно забезпечує виконання основних етапів обробки датасету, починаючи від завантаження зображень та їх анотації до експорту даних у форматі YOLO.

На рисунку 3.12 зображено послідовність виконання цих етапів.

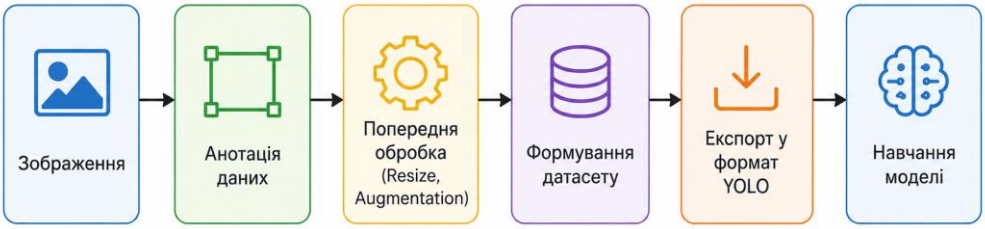


Рисунок 3.12 – Етапи формування моделі

YOLOv8 часто використовується у поєднанні з бібліотекою OpenCV, яка забезпечує базові інструменти для обробки зображень та відеопотоків. У такій зв’язці OpenCV виконує роль проміжного шару, що відповідає за захоплення відео з камери, попередню обробку кадрів, зміну їх розміру та формату, а також відображення результатів детекції на екрані [21-27].

Користувач завантажує набір зображень або відео на сайт, після чого система виконує їх організацію, обробку та підготовку до подальшого використання в моделях машинного навчання.

На першому етапі зображення зберігаються у хмарному сховищі та автоматично структуруються в проєкт. Далі користувач або вбудовані інструменти розмітки створюють анотації, тобто позначають об’єкти на зображеннях за допомогою bounding box, полігонів або інших типів розмітки. (рис. 3.13).

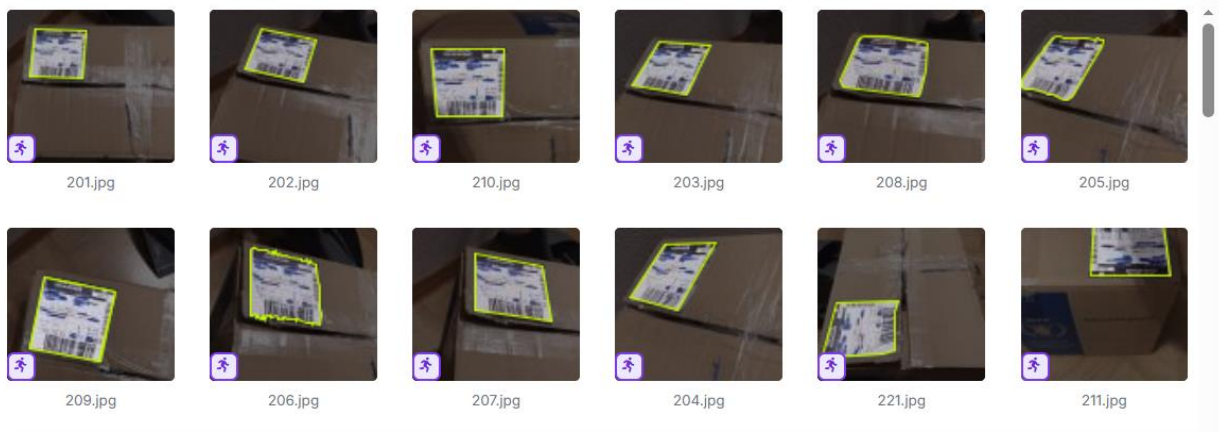


Рисунок 3.13 – База зображень

Після завершення розмітки платформа виконує обробку датасету: перевірку якості даних, балансування класів, а також аугментацію зображень, що дозволяє штучно збільшити обсяг навчальної вибірки шляхом зміни яскравості, поворотів, масштабування або додавання шуму. Це покращує здатність моделі до узагальнення.

На фінальному етапі користувач отримує готовий датасет, який можна експортувати у різні формати, сумісні з популярними фреймворками машинного навчання, такими як YOLO, TensorFlow або PyTorch. Після цього датасет використовується для навчання моделей детекції або класифікації об’єктів.

3.5 Порівняння нейромереж

YOLO (You Only Look Once) у порівнянні з іншими нейронними мережами для детекції об’єктів має суттєві відмінності в підході, швидкості та області застосування. На відміну від двоетапних моделей, таких як Faster R-CNN, яка спочатку генерує регіони інтересу за допомогою Region Proposal Network, а потім виконує їх класифікацію, YOLO здійснює повну детекцію за один прохід нейронної мережі, що забезпечує значно вищу швидкість роботи. Подібний двоетапний підхід також використовується в Mask R-CNN, яка є розширенням Faster R-CNN і додатково виконує сегментацію об’єктів, проте

вона є ще більш обчислювально складною і повільнішою.

У порівнянні з одностадійними моделями, такими як SSD (Single Shot Detector), YOLO зазвичай демонструє кращий баланс між швидкістю та точністю, особливо у сучасних версіях. SSD також виконує детекцію за один прохід, але може поступатися YOLO у складних сценах та при розпізнаванні дрібних об’єктів. Інша модель, RetinaNet, використовує механізм Focal Loss для покращення роботи з незбалансованими даними та підвищення точності, однак вона зазвичай є повільнішою у порівнянні з YOLO (рис. 3.14).

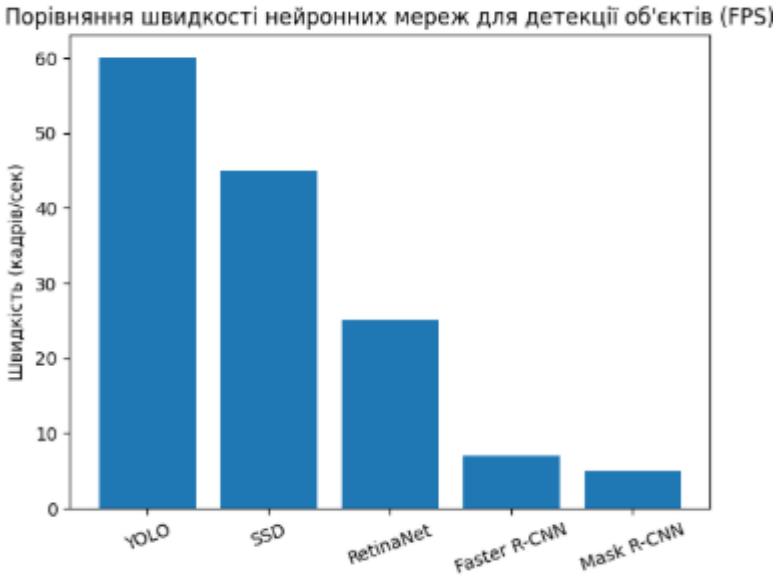


Рисунок 3.14 – Порівняння швидкості роботи різних нейронних мереж

Завдяки своїй архітектурі YOLO забезпечує високу швидкість обробки зображень у реальному часі, що робить його більш придатним для практичних застосувань, таких як відеоспостереження, автономні транспортні системи та робототехніка. У той час як Faster R-CNN, Mask R-CNN і RetinaNet частіше використовуються в задачах, де пріоритетом є максимальна точність, YOLO залишається оптимальним вибором для систем, де важлива оперативність і стабільна робота в реальному часі.

4 РОЗРОБКА ТА РЕАЛІЗАЦІЯ СИСТЕМИ ДЕТЕКЦІЇ ОБ’ЄКТІВ

Метою кваліфікаційної роботи було дослідження та практичне застосування сучасних алгоритмів комп'ютерного зору для автоматичного виявлення та розпізнавання об'єктів на зображеннях і у відеопотоці з використанням нейронної мережі YOLO. Реалізована система забезпечує швидке та точне розпізнавання об’єктів у реальному масштабі часу, що сприяє підвищенню ефективності роботи робототехнічного комплексу в умовах автоматизованого виробництва.

4.1 Програмна реалізація

У процесі підготовки даних було використано платформу Roboflow. Для кожного зображення за допомогою вбудованих інструментів анотації було виділено область розташування об’єкта та призначено відповідну мітку класу.

Розмітка даних є важливим етапом підготовки датасету, оскільки саме на основі цієї інформації нейронна мережа навчається знаходити та розпізнавати об’єкти на зображеннях.

Процес розмітки зображення наведено на рисунку 4.1.

Після завантаження зображень до платформи Roboflow було виконано їх розмітку, на якій створено власний датасет. Для цього було зібрано 504 зображення, частину з яких було зроблено самостійно, а частину знайдено в інтернеті.



Рисунок 4.1 – Розмітка об'єкту

Підготовлена база зображень для навчання моделі YOLOv8 зображена на рисунку 4.2.

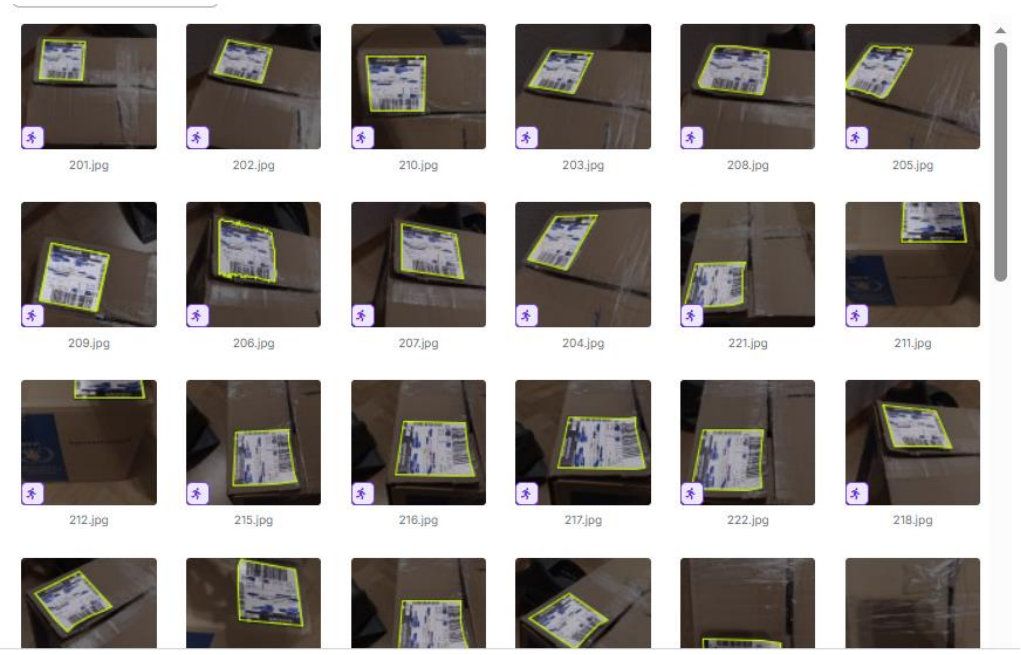


Рисунок 4.2 – Набір зображень

Далі виконано процес навчання нейронної мережі на підготовлених даних, а також проведено тестування отриманої моделі для оцінки її точності та продуктивності (рис. 4.3).

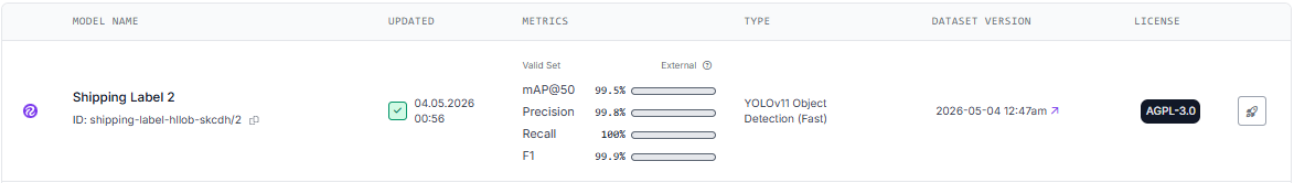


Рисунок 4.3 – Сформований датасет

Під час навчання моделі здійснювався постійний контроль зміни функції втрат, показників точності та стабільності процесу навчання.

На практиці під час обробки відеопотоку можуть виникати різні фактори, які впливають на якість детекції: зміна освітлення, цифрові шуми, часткове перекриття об'єктів, зміна положення камери або погіршення якості зображення. Якщо за таких умов система продовжує коректно виконувати свої функції, її роботу можна вважати стійкою. У теорії автоматичного керування стійкою називають систему, яка після появи зовнішнього збурення повертається до нормального режиму роботи без втрати працездатності. Для класичних систем автоматичного керування така властивість оцінюється за допомогою математичних моделей та критеріїв стійкості, серед яких найбільш відомими є критерії Гурвіца, Найквіста та Михайлова. Вони дозволяють визначити, чи буде система працювати стабільно після зміни вхідних параметрів або виникнення збурень.

Розроблена система детекції об'єктів не є класичною системою автоматичного керування, оскільки не містить регулятора та об'єкта керування у традиційному розумінні. Проте основний принцип стійкості залишається актуальним і для програмних систем комп'ютерного зору. У даному випадку під стійкістю розуміється здатність алгоритму зберігати правильність детекції та стабільну швидкодію навіть за зміни умов роботи.

Одним із головних факторів, що впливає на якість роботи системи, є

освітлення. При недостатньому або надмірному освітленні контури та окремі елементи об'єктів можуть втрачатися, через що зменшується точність розпізнавання. Для зменшення впливу цього фактора використовується бібліотека OpenCV, яка дозволяє виконувати попередню обробку зображень, змінювати їх розмір, виконувати перетворення кольорового простору, а також застосовувати інші операції, необхідні для підготовки кадрів перед подачею до нейромережі.

Ще одним фактором є наявність шумів або розмиття зображення. Подібні спотворення можуть виникати через низьку якість камери, недостатнє освітлення або рух об'єктів. Незважаючи на це, сучасні моделі сімейства YOLO здатні виконувати детекцію навіть у випадках, коли зображення містить незначні шуми або окремі дефекти. Це досягається завдяки навчанню моделі на великій кількості різноманітних прикладів, що дозволяє нейромережі краще узагальнювати отримані дані.

Важливою особливістю є також робота системи в режимі реального часу. Якщо швидкість обробки кадрів буде недостатньою, виникатимуть затримки між отриманням відео та відображенням результату детекції. Це негативно впливає на ефективність усієї системи, особливо якщо вона використовується для керування роботизованими комплексами або автоматизованими виробничими процесами. Саме тому під час розробки було обрано модель YOLO, яка поєднує високу точність розпізнавання з достатньою швидкістю роботи.

Окрему увагу було приділено забезпеченню стабільності програмної реалізації. У процесі роботи система безперервно отримує відеокадри, передає їх до нейромережевої моделі, виконує аналіз отриманих результатів та відображає знайдені об'єкти. Така послідовність дій дозволяє підтримувати безперервний цикл обробки інформації без необхідності повторного запуску програми або додаткового втручання користувача. Крім того, використання бібліотеки OpenCV забезпечує стабільну роботу із відеопотоком та сумісність із різними типами камер.

Під час тестування було встановлено, що система зберігає працездатність при незначних змінах освітлення, зміні відстані до об'єкта та частковому його перекритті. У більшості випадків модель продовжує правильно визначати необхідний клас об'єкта та відображати результати детекції. Це свідчить про достатній рівень стійкості розробленої системи до найбільш поширених зовнішніх впливів, які можуть виникати під час практичної експлуатації.

За результатами навчання було отримано графіки, що відображають процес збіжності нейронної мережі та зміну показників якості.

На рисунку 4.4 зображені результати навчання моделі в середовищі Roboflow.

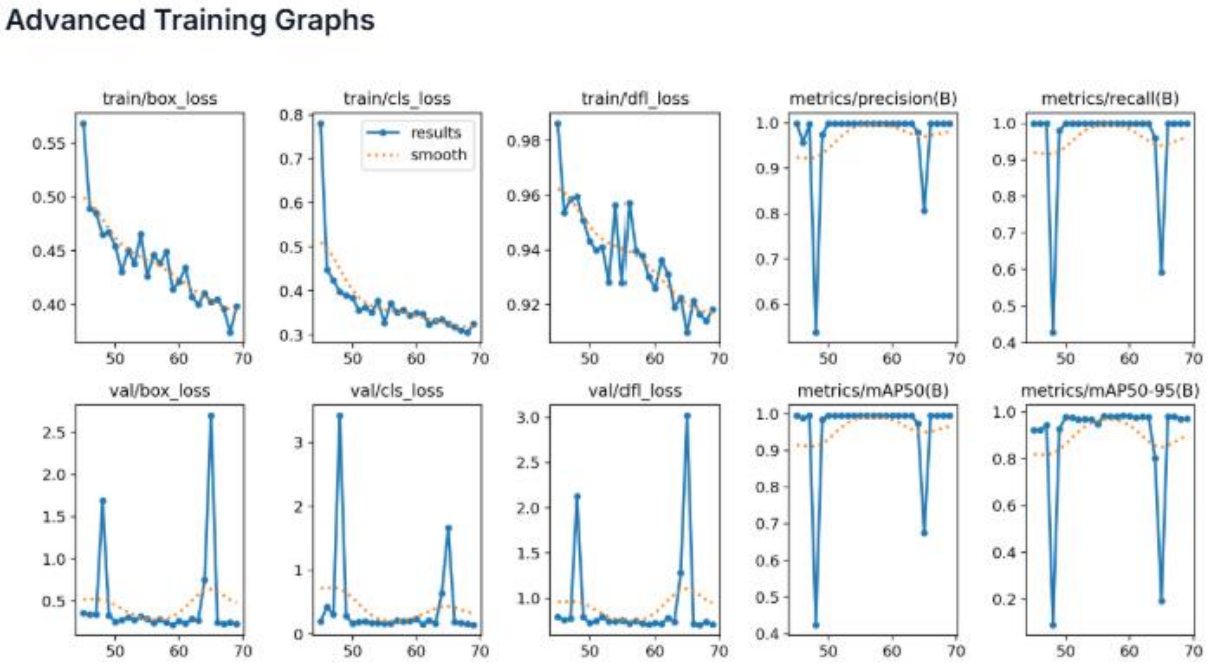


Рисунок 4.4 – Графіки результатів навчання моделі в Roboflow

Для реалізації системи детекції об’єктів у середовищі Python було підключено необхідні бібліотеки.

1. Імпорт необхідних бібліотек:

```
import cv2
from ultralytics import YOLO
import numpy as np
```

2. Завантаження датасету до проекту Python (рис. 4.5).

Після підготовки та розмітки зображень датасет було завантажено до проєкту Python. Для цього з платформи Roboflow було експортовано набір даних у форматі YOLOv8, який містить зображення, файли розмітки та конфігураційний файл data.yaml. Отримані файли були додані до структури проєкту та використані для подальшого навчання нейронної мережі.

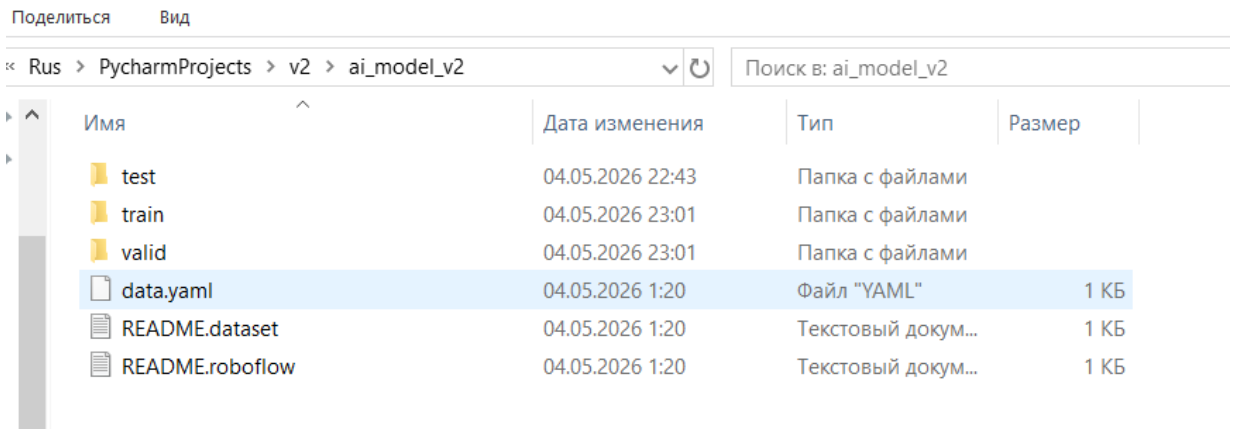


Рисунок 4.5 – Проект ai_model_v2

3. Створення файлу для навчання нейронної мережі train.py:

```
from ultralytics import YOLO
model = YOLO("yolov8n.pt")

model.train(
        data="ai_model_v2/data.yaml",
        epochs=50,
        imgsz=640
)
```

Файл train.py призначений для запуску процесу навчання нейронної мережі (рис. 4.6). У ньому використовується бібліотека Ultralytics YOLO, за допомогою якої завантажується попередньо натренована модель yolov8n.pt. Далі викликається функція навчання train(), у якій вказується шлях до файлу

датасету data.yaml, що містить інформацію про класи та структуру даних.

Важливим параметром під час навчання нейронної мережі є кількість епох. Під епохою розуміють один повний прохід моделі через усі зображення навчального датасету. Після кожної епохи відбувається коригування параметрів моделі, що дозволяє покращувати якість розпізнавання об’єктів.

У роботі для навчання моделі було встановлено значення 50 епох та розмір зображень (imgsz=640), який використовується для масштабування вхідних даних. Цієї кількості достатньо для навчання нейронної мережі на підготовленому датасеті та отримання задовільних результатів. Зі збільшенням кількості епох може покращуватися точність розпізнавання, проте одночасно збільшується час навчання. Тому було обрано значення, яке забезпечує достатню якість роботи моделі без суттєвого збільшення тривалості навчання.

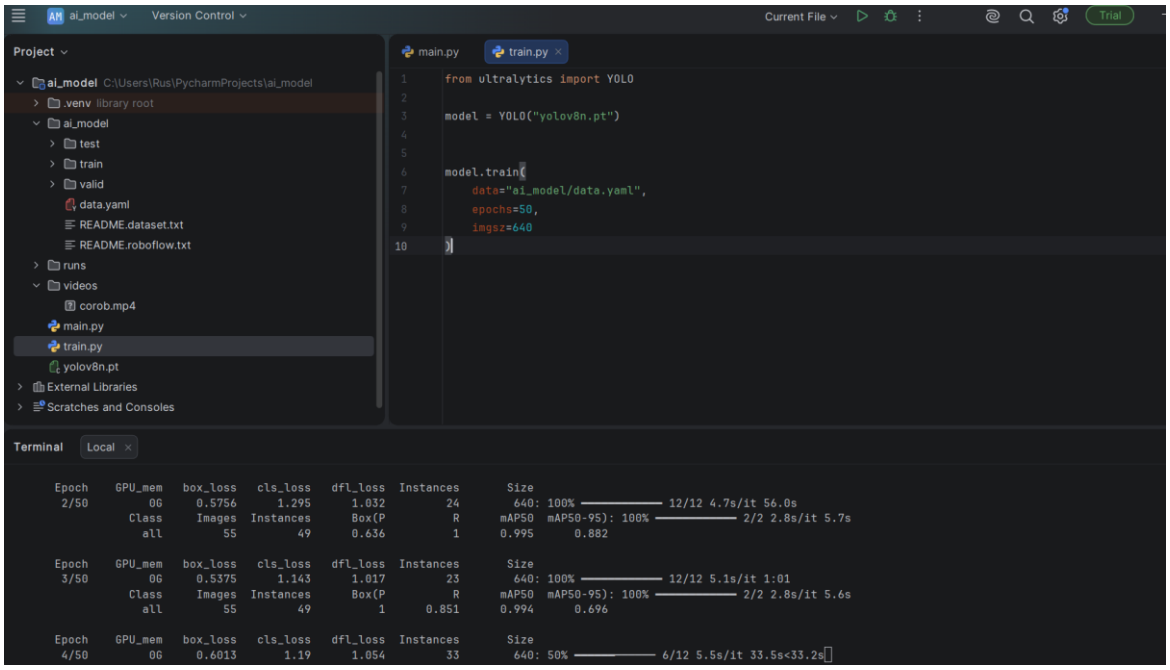


Рисунок 4.6 – Навчання нейронної мережі

Після завершення процесу навчання нейронної мережі формуються вихідні файли моделі, які містять отримані ваги та параметри навченої системи. Саме ці файли є результатом навчання і представляють собою готову нейронну мережу, яка надалі використовується для виконання задачі виявлення об’єктів.

4. Створення директорії screen для зображень з етикеткою (рис. 4.7):

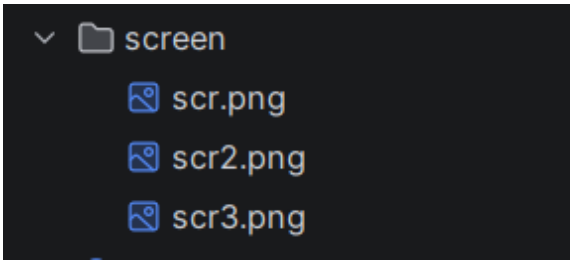


Рисунок 4.7 – Директорія із зображеннями

Ці зображення використовуються для перевірки роботи вже навченої нейронної мережі. На їх основі модель виконує виявлення об’єктів та визначає відповідні мітки (класи), демонструючи результат розпізнавання після завершеного навчання.

5. Створення файлу ai_foto.py для розпізнавання label з фото:

```
from ultralytics import YOLO
import cv2
model=YOLO(r"C:\Users\Rus\PycharmProjects\v2\runs\detect\train\weights\best.pt")
img = cv2.imread(r"C:\Users\Rus\PycharmProjects\v2\screen\scr3.png")
if img is None:
    print("Помилка: зображення не знайдено")
    exit()
results = model(img, conf=0.1)
print("Знайдено об'єктів:", len(results[0].boxes))
frame = results[0].plot()
cv2.imshow("Виявлення", frame)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

На рисунку 4.8 наведено приклад розпізнавання етикеток за допомогою навченої нейронної мережі.



Рисунок 4.8 – Результат розпізнавання з фото

6. Створення файлу camera.py для розпізнавання label з камери у реальному часі:

```
from ultralytics import YOLO
import cv2

model=YOLO(r"C:\Users\Rus\PycharmProjects\v2\runs\detect\train\weights\best.pt")
cap = cv2.VideoCapture(0)  # 0 – це основна камера

if not cap.isOpened():
    print("Помилка: камера не знайдена")
    exit()

while True:
```

```
    ret, frame = cap.read()
    if not ret:
        print("Помилка: не вдалося отримати кадр")
        break

    results = model(frame, conf=0.1)
    print("Знайдено об'єктів:", len(results[0].boxes))
    output_frame = results[0].plot()
    cv2.imshow("Виявлення з камери", output_frame)

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

На рисунку 4.9 зображено розпізнавання етикетки у реальному часі з камери.



Рисунок 4.9 – Результат розпізнавання з камери

Під час тестування системи було проведено демонстрацію роботи моделі з використанням камери. У режимі реального часу нейронна мережа аналізує відеопотік, отримує кадри з камери та виконує розпізнавання об’єктів. Результати роботи системи відображаються на екрані у вигляді підписаних областей з виявленими об’єктами, аналогічно до обробки статичних зображень.

4.2 Аналіз умов праці під час роботи з програмним забезпеченням

Під час виконання кваліфікаційної роботи розробка та тестування програмного забезпечення здійснювалися на персональному комп’ютері. Робота була пов’язана з тривалим перебуванням за монітором, аналізом результатів роботи моделі YOLOv8 та обробкою відеоінформації [28-29].

Основними шкідливими факторами є підвищене навантаження на зір, тривале статичне положення тіла, а також розумове навантаження під час налагодження та тестування програмного забезпечення. Для зменшення негативного впливу цих факторів необхідно правильно організувати робоче місце та дотримуватись режиму праці й відпочинку [28-29].

4.3 Вимоги безпеки під час роботи за персональним комп’ютером

Перед початком роботи необхідно перевірити справність комп’ютера, монітора та інших пристроїв. Не допускається використання обладнання з пошкодженими кабелями живлення або несправними розетками [28-29].

Під час роботи користувач повинен сидіти у зручному положенні, а монітор має знаходитися на відстані 50–70 см від очей. Забороняється самостійно виконувати ремонт електрообладнання, що перебуває під напругою.

Після завершення роботи необхідно завершити роботу програм та вимкнути обладнання відповідно до правил експлуатації [28-29].

4.4 Пожежна безпека

Під час роботи з комп’ютерною технікою необхідно дотримуватися правил пожежної безпеки. Забороняється перевантажувати електричну мережу шляхом підключення великої кількості пристроїв до однієї розетки або подовжувача [28-29].

Не допускається використання несправних електроприладів та пошкоджених проводів. Робоче місце повинно утримуватися в чистоті, без накопичення паперу та інших легкозаймистих матеріалів поблизу комп’ютерної техніки [28-29].

У приміщенні повинні бути наявні засоби пожежогасіння та забезпечений вільний доступ до шляхів евакуації у разі виникнення надзвичайної ситуації.

4.5 Висновки

Дотримання розглянутих вимог охорони праці дає змогу зменшити ризик виникнення небезпечних ситуацій під час роботи з комп’ютерною технікою та програмним забезпеченням. Виконання правил електробезпеки та пожежної безпеки забезпечує безпечні умови праці, збереження працездатності користувача та надійну роботу обладнання під час обробки відеоінформації і розпізнавання об’єктів. Правильна організація робочого місця сприяє зниженню втомлюваності, підвищенню продуктивності праці та створенню комфортних умов роботи.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було проведено дослідження сучасних методів обробки зображень та розпізнавання об’єктів у системах технічного зору. Розглянуто принципи роботи алгоритмів комп’ютерного зору, особливості використання бібліотеки OpenCV та сучасних нейромережевих моделей детекції об’єктів.

У процесі виконання роботи було підготовлено набір даних для навчання моделі, виконано його розмітку та проведено навчання нейромережі YOLOv8. Також було розроблено програмне забезпечення мовою Python для обробки відеопотоку та автоматичного виявлення об’єктів у кадрі в режимі реального часу.

Було реалізовано систему технічного зору, яка забезпечує отримання відеозображення з камери, обробку відеопотоку та розпізнавання заданого об’єкта. Для реалізації системи використано бібліотеку OpenCV та модель YOLOv8, що дозволило забезпечити достатню швидкість і точність роботи.

Проведене тестування показало працездатність розробленої системи та коректність роботи алгоритмів розпізнавання. Отримані результати підтвердили можливість використання створеного програмного забезпечення для задач автоматичного виявлення об’єктів. Розроблена система може бути використана як основа для подальших досліджень у сфері комп’ютерного зору, а також як навчальний або демонстраційний стенд для студентів під час вивчення сучасних технологій обробки зображень та нейромережевих методів розпізнавання об’єктів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Михайлов Р. А. Аналіз методів обробки відеоінформації у системі технічного зору промислового робота //«Автоматизація та приладобудування» ADED-2026, Випуск 1, С. 209-215

2. Указ Президента України №722/2019 «Про Цілі сталого розвитку України на період до 2030 року». – Режим доступу: https://www.president.gov.ua/documents/7222019-29825.

3. Методичні вказівки з підготовки кваліфікаційної роботи для здобувачів першого (бакалаврського) рівня вищої освіти денної і заочної форми навчання спеціальності 151 – Автоматизація та комп'ютерно-інтегровані технології освітньої [Електронний ресурс] / упоряд.: І. Ш. Невлюдов, О. І. Филипенко, О. В. Токарєва, С. П. Новоселов, О.В Сичова ; М-во освіти і науки України, Харків. нац. ун-т радіоелектроніки. – електрон. вид. – Харків : ХНУРЕ, 2023. – 760 Kb

4. ДСТУ 3008-15. Документація. Звіти у сфері науки та техніки. структура та правила оформлення. Введ. 2015-06-22. К. Держстандарт України, 2017. 29 с.

5. Технічне та програмне забезпечення розробки малогабаритного мобільного робота: монографія / І.Ш. Невлюдов, В.В. Євсєєв, Д.В. Гурін. Кривий Ріг: КФК ДНП «Державний університет «Київський авіаційний інститут», 2025. –355с.

6. BEAM робототехніка [Електронний ресурс] : Навчальний посібник / І. Ш. Невлюдов, В. В. Євсєєв, С. С. Максимова ; Харків. нац. ун-т радіоелектроніки. Харків : Видавець Чернявський Д. О., 2024. 276 с.

7. Bilozerskyi, V., Dergachov K., Krasnov L. Аналіз і попередня обробка відеоданих для підвищення якості роботи систем технічного зору. International Scientific Technical Journal" Problems of Control and Informatics" 68.2 (2023): 50-66.

8. Lyashenko, V., Abu-Jassar, A. T., Yevsieiev, V., & Maksymova, S. (2023).