Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки
(повна назва)

# КВАЛІФІКАЦІЙНА РОБОТА
# Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Розроблення інтелектуальної системи для класифікації виробів на сортувальній лінії із використанням технології машинного навчання
(тема)

Виконав:
здобувач 4 року навчання,
групи АКТАКІТ-21-1
Софія ДРІГА
(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 доц. Оксана СИЧОВА
(посада, власне ім'я, прізвище)

Допускається до захисту

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

2025 р.

Я, Дріга Софія Ігорівна, як здобувачка вищої освіти ХНУРЕ, розумію і підтримую політику закладу із академічної доброчесності. Я не надавала і не одержувала недозволену допомогу під час підготовки кваліфікаційної роботи. Я не використовувала штучний інтелект для підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на віповідне джерело.

«02» червня 2025 р.  Софія ДРІГА

Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки

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)

«30» квітня 2025 р.

# ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Дрізі Софії Ігорівні

(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення інтелектуальної системи для класифікації виробів на сортувальній лінії із використанням технології машинного навчання

затверджена наказом університету від 19 травня 2025 р. № 390 Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 18 червня 2025 р.

3. Вихідні дані до роботи Мова програмування C#
Навчальний макет робота маніпулятора
Модель розпізнавання об'єктів в реальному часі YOLOv8
Веб-камера з роздільною здатністю 1080p (1920x1080)

4. Перелік питань, що потрібно опрацювати в роботі Аналіз методів використання машинного навчання з метою підвищення ефективності виробничих процесів; вибір та Обґрунтування засобів програмної реалізації; вибір апаратної складової проєкта та розроблення програмного забезпечення з використанням мови програмування C#; оформлення пояснювальної записки кваліфікаційної роботи.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)
Демонстраційний матеріал представлений у форматі презентації PowerPoint в кількості 16 слайдів з розширенням .pptx.

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 | Примітка |
|---|---|---|---|
| 1 | Аналіз технічного завдання | 30.04.2025 | виконано |
| 2 | Аналіз літератури за темою роботи | 04.05.2025 | виконано |
| 3 | Виконання аналізу актуальності використання машинного навчання для підвищення ефективності виробничого процесу | 08.05.2025 | виконано |
| 4 | Вибір і обґрунтування обраних засобів програмної реалізації | 11.05.2025 | виконано |
| 5 | Вибір апаратної складової і розробка програмного забезпечення | 13.05.2025 | виконано |
| 6 | Оформлення пояснювальної записки | 29.05.2025 | виконано |
| 7 | Подання роботи на перевірку Інтернет-сервісом StrikePlagiarism | 03.06.2025 | виконано |
| 8 | Подання роботи на рецензію | 05.06.2025 | виконано |
| 9 | Подання роботи на підпис зав. кафедри | 16.06.2025 | виконано |
| 10 | Подання атестаційної роботи в ЕК | 18.06.2025 | виконано |

Дата видачі завдання 30 квітня 2025 р.

Здобувач _____ Софія ДРІГА
(підпис) (власне ім'я, прізвище)

Керівник роботи _____ доц. Оксана СИЧОВА
(підпис) (посада, власне ім'я, прізвище)

# РЕФЕРАТ

Пояснювальна записка: 66 с., 1 табл., 25 рис., 3 дод., 25 джерел.

YOLO, МАШИННЕ НАВЧАННЯ, РОБОТ-МАНІПУЛЯТОР, НЕЙРОННІ МЕРЕЖІ, INDUSTRY 4.0, ВИЯВЛЕННЯ ОБ'ЄКТІВ.

Об'єкт розробки – процеси сортування компонентів на інтелектуальному виробництві.

Предмет розробки – методи використання комп'ютерного зору і машинного навчання в контексті управління роботом-маніпулятором.

Мета дослідження – покращення функціоналу макета робота-маніпулятора за рахунок створення програмного забезпечення комп'ютерного зору із використанням моделі машинного навчання.

В першому розділі було проведено аналіз актуальності автоматизації процесів розпізнавання й класифікації дрібних об'єктів у виробничих і переробних галузях із застосування алгоритмів машинного навчання, зокрема моделі YOLO.

В другому розділі було обрано та обґрунтовано засоби реалізації проєкту, проведено аналіз переваг й сумісності обраних засобів згідно з вимогами проєкту.

В третьому розділі було розглянуто апаратну складову проєкту, обрано макет робота-маніпулятора, розроблено програмний модуль із використанням моделі машинного навчання YOLO.

Розроблене програмне забезпечення може використовуватись у системах комп'ютерного зору роботизованих комплексів для автоматизованого сортування, контролю якості, ідентифікації об'єктів у реальному часі на виробничих або переробних лініях.

# ABSTRACT

Explanatory note: 66 pages, 1 tables, 25 figures, 3 appendixes, 25 references.

YOLO, MACHINE LEARNING, ROBOTIC MANIPULATOR, NEURAL NETWORKS, INDUSTRY 4.0, OBJECT DETECTION.

The object of the development is the component sorting processes in intelligent manufacturing.

The development subject – methods of using computer vision and machine learning in the context of controlling a manipulator robot.

The purpose of the development is to improve the functionality of the manipulator robot model by creating computer vision software using a machine learning model.

In the first chapter, an analysis was conducted of the relevance of automating the processes of recognition and classification of small objects in the production and processing industries using machine learning algorithms, in particular the YOLO model.

The second chapter presents the selected tools for project implementation, provides justification for their choice, and analyzes their advantages and compatibility in accordance with the project requirements.

Chapter three presents the selected robot manipulator prototype, discusses the hardware components of the project, and describes the development of a software module using the YOLO machine learning model.

The developed software can be used in computer vision systems of robotic complexes for automated sorting, quality control, and object identification in real-time on production or recycling lines.

# ЗМІСТ

# ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ЛАЧХ – логарифмічна амплітудно-фазова частотна характеристика;

САУ – система автоматичного управління;

ЦСР – цілі сталого розвитку;

AI – artificial Intelligence;

CNN – convolutional neural network;

IoU – intersection over union;

mAP – mean average precision;

ML – machine learning.

# ВСТУП

У сучасних умовах інтенсивного розвитку технологій, одним із пріоритетних напрямів удосконалення виробничих процесів є впровадження інтелектуальних рішень, зокрема, машинного навчання задля підвищення ефективності розумного виробництва [1].

Використання машинного навчання дає змогу автоматизувати аналіз даних у реальному часі, здійснювати прогнозування виробничих процесів, покращити контроль якості продукції та зменшити потребу у людському втручанні. Такий підхід стає особливо актуальним в умовах зростаючих вимог до ефективності й гнучкості сучасного виробництва [1].

Рішення, впроваджені у виробничий процес за допомогою машинного навчання, також відкривають перспективи для досягнення Цілей сталого розвитку (ЦСР), передбачених програмою ООН до 2030 року. Зокрема, Ціль № 12 «Відповідальне споживання й виробництво», п. 12.5, який передбачає скорочення обсягу утворення відходів й збільшення обсягу їх перероблення [2], може бути реалізований шляхом застосування систем комп’ютерного зору на основі технологій машинного навчання для подальшого впровадження безпосередньо на сортувальні лінії.

Таким чином, метою кваліфікаційної роботи є покращення функціоналу макета робота-маніпулятора за рахунок створення програмного забезпечення комп’ютерного зору із використанням моделі машинного навчання.

Об’єкт розробки – процеси сортування компонентів на інтелектуальному виробництві.

Предмет розробки – методи використання комп’ютерного зору і машинного навчання в контексті управління роботом-маніпулятором.

Методи дослідження – аналіз проблеми, тестування й порівняння ефективності моделей машинного навчання YOLO, розробка програмного

модуля з використанням мови програмування C# для подальшої інтеграції з роботом-маніпулятором.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– виконати аналіз наявних підходів й алгоритмів комп’ютерного зору на основі машинного навчання, які застосовуються для виявлення та розпізнавання об’єктів у виробничому середовищі;

– визначити вимоги до проєктування моделі комп’ютерного зору з урахуванням особливостей та специфіки виробничого процесу;

– підготувати навчальний датасет та здійснити навчання моделі на основі обраного алгоритму машинного навчання;

– розробити структуру програмного забезпечення для реалізації та функціонування моделі комп’ютерного зору.

Робота виконана згідно з рекомендаціями [3] та вимогами ДСТУ 3008:2015 [4]. Матеріали кваліфікаційної роботи були апробовані у [5].

Результати роботи відносяться до ЦСР 12 «Відповідальне споживання й виробництво», п. 12.5 про скорочення утворення відходів завдяки запобіганню, зменшенню, переробці та повторному використанню відходів.

# 1 АНАЛІЗ АКТУАЛЬНОСТІ ВИКОРИСТАННЯ МАШИННОГО НАВЧАННЯ ДЛЯ ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ ВИРОБНИЧОГО ПРОЦЕСУ

## 1.1 Використання машинного навчання для автоматизації процесів у Smart Manufacturing

Industry 4.0, або четверта промислова революція (англ. The fourth industrial revolution), є головним напрямом розвитку як для галузі промисловості, так і для загального наукового прогресу. В Industry 4.0 традиційне виробництво, тобто виробництво без здатності до самостійної адаптації, замінюється «розумним» (англ. Smart manufacturing) [1].

Національний інститут стандартів і технологій США (NIST) визначає поняття Smart Factory або Smart Manufacturing як повністю інтегровані корпоративні виробничі системи, які здатні в реальному масштабі часу реагувати на мінливі умови виробництва, вимоги мереж постачання і задоволення потреб клієнтів [6].

Для досягнення попередньо поставленої мети «розумного» виробництва, необхідно використовувати дивергенцію інформаційно-комунікаційних технологій, операційних технологій і кіберфізичних систем на всіх етапах виробництва високотехнологічної продукції [6].

Дивергенція інформаційно-комунікаційних технологій, своєю чергою, містить в собі великі дані (англ. Big data), механізми штучного інтелекту (AI), аналіз даних на межі мереж (туманні й приграничні обчислення), мобільну передачу даних, мережеві технології, інтерфейси користувачів, SCADA системи (Supervisory control and data acquisition), системи управління, програмовані логічні контролери, датчики й виконавчі механізми, автоматизовані робототехнічні процеси, автономні роботи [6].

Як показано на рисунку 1.1, Smart Manufacturing представляє нове покоління виробничої системи в концепціях індустрії 4.0.

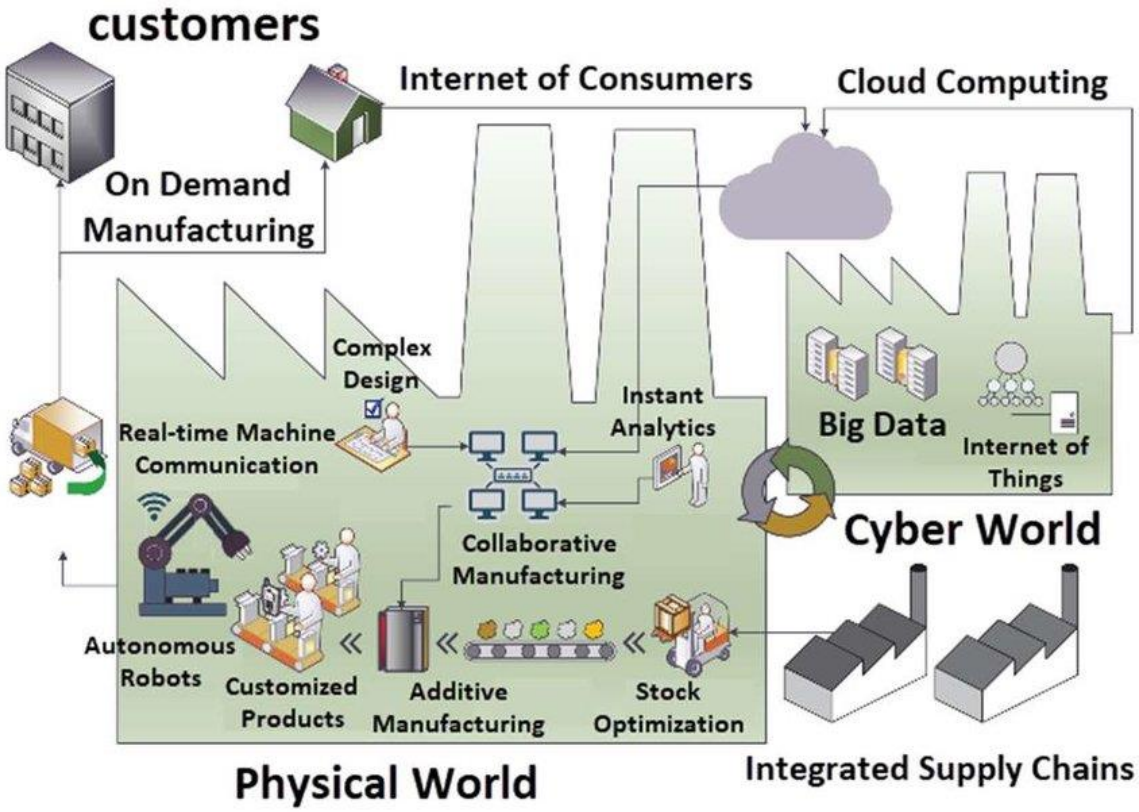



Рисунок 1.1 – Схематичне представлення Smart Manufacturing в компонентах Industry 4.0 [7]

Отже, Smart Factory підтримує передові технології, наприклад: комп'ютеризація виробництва, кіберфізичні системи, великі дані, інтернет речей, хмарні обчислення, а також автоматизовані й роботизовані системи [7].

Впровадження «розумних» технологій в рамках Industry 4.0 дозволяє досягти високого рівня автоматизації промисловості, де машини вчаться розуміти процеси виробництва, взаємодіяти з навколишнім середовищем та адаптувати свою поведінку відповідно вимог виробництва. Зокрема, Big Data та штучний інтелект роблять машини в промисловому виробництві значно «розумнішими», ніж раніше, що надає можливість глибше розглянути проблему створення

комп’ютерів, які автоматично вдосконалюються на основі отриманого досвіду [1].

Питання створення машин, що навчаються, головним чином вирішується машинним навчанням (ML), як підгалуззю AI. ML стало головним рушієм інновацій в контексті створення систем, які можуть автоматично адаптуватися до змін у виробничих процесах та вдосконалювати свою продуктивність без втручання людини [1].

ML є важливим методом для прогнозування та класифікації складних задач у виробничих системах. Машинне навчання використовує потужні обчислювальні ресурси та різноманітне програмне забезпечення для отримання інформації й знань з великих даних, які збираються з навколишнього середовища, а також здатне використати ці дані безпосередньо для навчання, в результаті виробляючи штучний або «обчислювальний» інтелект [1].

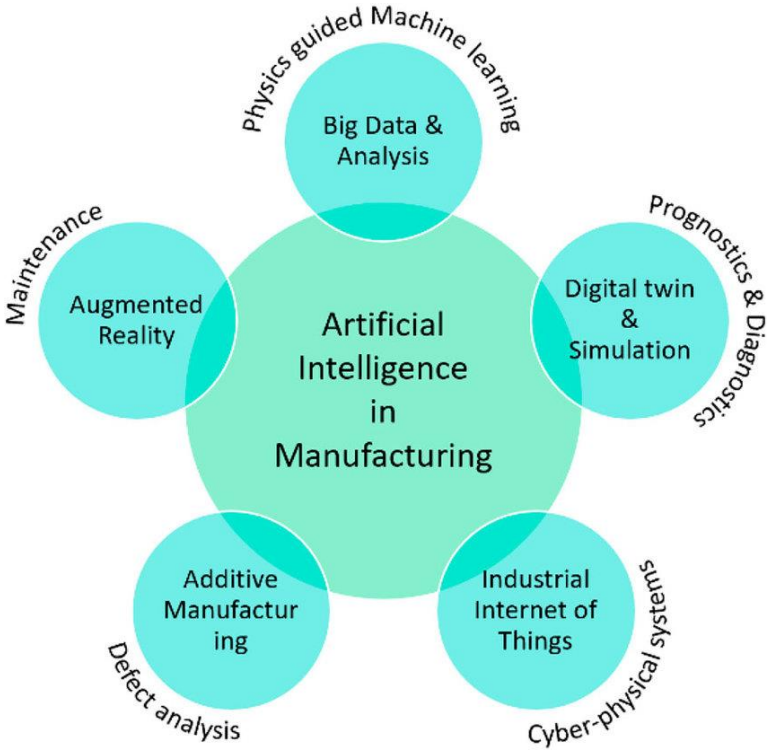



Рисунок 1.2 – Багатогранне використання AI та ML у виробничій сфері [8]

Як показано на рис. 1.2, машинне навчання та AI загалом мають великий спектр використання у виробничій сфері. Зокрема, основним рушієм AI у виробничих умовах є Промисловий Інтернет речей, який дозволяє ефективно використовувати великі дані. Великі дані, своєю чергою, потребують застосування AI для прийняття рішень у реальному часі [8].

Основні підходи AI, такі як глибоке навчання (англ. Deep learning) та комп'ютерний зір (англ. Computer vision), вже активно використовуються в виробничих процесах. Наприклад, AI може бути застосований для аналізу даних, що генеруються під час проведення виробничих процесів, таких як адитивне виробництво, для реального моніторингу, контролю та мінімізації дефектів [8].

Виявлення несправностей та прогнозування стану обладнання в цьому випадку здійснюється за допомогою машинного навчання, тоді як додаткові можливості для налаштування процедур технічного обслуговування на ходу забезпечують пристрої доповненої реальності (англ. Augmented reality devices) [8].

## 1.2 Роль машинного навчання в досягненні цілей сталого розвитку

Завдяки використанню штучного інтелекту, машинне навчання здатне забезпечити основу для структурного вдосконалення методів перероблення відходів – теми, що з кожним роком набуває все більшої ваги в рамках досягнення цілей сталого розвитку [9].

Цілі сталого розвитку, також відомі як Глобальні цілі, були ухвалені Організацією Об’єднаних Націй у 2015 році як універсальний заклик до дій щодо скорочення бідності, захисту планети та забезпечення того, щоб до 2030 року усі люди жили в мирі й достатку [10].

ЦСР складається з 17 цілей, які взаємодоповнюють одна одну в рамках досягнення поставленої мети. Так, для забезпечення зниження ресурсомісткості економіки та екологічної безпеки, існує ціль під номером 12: відповідальне

споживання та виробництво (англ. Ensure sustainable consumption and production patterns [2], [10].

Одним з рішень проблеми надмірного обсягу утворення відходів є перероблення, яке забезпечує можливість повторного використання матеріалів на основі інноваційних технологій та виробництва [10]. Рівень перероблення постійно зростає, однак прогнози показують, що кількість відходів має тенденцію збільшуватись, в той час як сама технологія перероблення має свої неточності, які можуть потенційно знизити загальний рівень ефективності перероблення [9].

Однією з перешкод для проведення ефективного перероблення відходів є насамперед проблема відокремлення відходів, які підлягають переробленню, від відходів, що не підлягають утилізації [9]. Наявна проблема перероблення зумовлена складністю й високими витратами на процес сортування, що потребує значних людських і технологічних ресурсів. Однак сучасні досягнення в галузі машинного навчання відкривають нові можливості для підвищення ефективності перероблення відходів, а також вирішення проблеми їх попереднього сортування [9].

Одним з алгоритмів глибокого навчання є згорткова нейронна мережа (CNN) – алгоритм, який є загальним підходом до створення моделей з використанням розпізнавання зображень для класифікації та виявлення об’єктів [9]. CNN складається з нейронів, які отримують кілька вхідних сигналів і забезпечують вихідні дані шляхом обчислення суми вхідної інформації за допомогою зображень на численних каналах. Зображення проходять через кілька згорткових шарів із фільтрами, зокрема: згортковий шар, шар активації (англ. Activation layer), пулінг-шар (англ. Pooling layer) і повнозв’язний шар (англ. Fully connected layer) [9].

Згортковий шар виконує основні обчислення, об’єднуючи дані з фільтрами. Шар активації використовує ректифікаційну функцію для корекції нелінійності зображення. Пулінг-шар обмежує пошук зображень за характеристиками, такими

як, наприклад, розміри зображення. Наприкінці, повнозв'язний шар перетворює зібрані дані у стовпець і передає їх у нейронну мережу для подальшої обробки. У кінцевому підсумку функція активації сортує вихідні дані [9].

Приклад алгоритму машинного навчання CNN наведено на рис. 1.3.

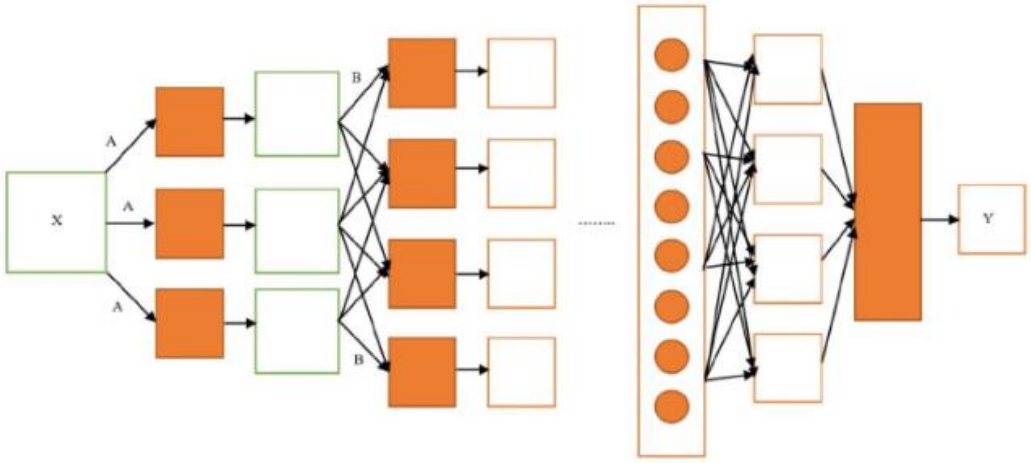



Рисунок 1.3 – Приклад алгоритму машинного навчання CNN [9]

Алгоритм CNN є першим по-справжньому успішним алгоритмом машинного навчання для тренування багатошарових мережевих структур, що спрямований на мінімізацію вимог до попередньої обробки даних [9].

Прикладом безпосереднього використання технології машинного навчання є автономні роботизовані системи для сортування відходів. Наприклад, система з вакуумним захватом і маніпулятором, розроблена італійськими дослідниками Кім Дж., Ночентіні О. та Скафуро М.. Система може захоплювати об'єкти та сортувати їх за матеріальним складом за допомогою RGB-камери та алгоритму машинного навчання CNN з модифікованою моделлю LeNet. Для зразків картону та пластику успішність захоплення вакуумом становила 86,09% і 90% відповідно, а для маніпулятора – 86,67% і 78,57% відповідно. Загальна точність класифікації склала 96% [9].

Іншим прикладом є розробка методу флуоресцентної класифікації 14 різних типів чорного пластикового сміття та 12 пластикових зразків з

використанням флуоресцентного спектрометра з додатковим освітленням у певному спектральному діапазоні. Особливістю розробки є також використання в ході дослідження різних алгоритмів машинного навчання, таких як kNN (англ. k-nearest Neighbor Method), SVM (англ. Support Vector Machine), CNN. Дослідження показало, що алгоритм CNN забезпечував найвищу точність класифікації порівняно з іншими застосованими алгоритмами, яка склала 93,5% [9].

Таким чином, алгоритм машинного навчання CNN показує високу ефективність в рамках розв'язання проблем сортування відходів і класифікації матеріалів, що може бути зеленим сигналом на поширення використання штучного інтелекту в галузі перероблення загалом [9].

## 1.3 Застосування CNN для автоматизації сортування

Перероблення пластику, – матеріалу, що відрізняється найбільшим внеском у проблему забруднення навколишнього середовища, – є надзвичайно важливим кроком до сталого розвитку. Широке використання пластику одноразового використання з кожним роком збільшується в обсягах. Так, індустрія бутильованої води, за результатами досліджень в 109 країнах світу, зросла в обсягах продаж на 73% в період з 2010 по 2020 роки, що зробило її однією з найшвидше зростаючих індустрій світу [11].

Бутильована вода є популярною по всьому світу, включно з США, Китаєм та Індонезією в якості головних споживачів. Для країн з середнім чи низьким рівнем доходу бутильована вода є базовою необхідністю, оскільки доступ до питної води з-під крану в таких країнах обмежений. Попит на бутильовану воду привів до вироблення близько 600 мільярдів пластикових пляшок і контейнерів у 2021 році, що призвело до утворення близько 25 мільйонів тонн пластикових відходів, більшість з яких не переробляється і потрапляє на звалища [11].

Незважаючи на розвиток технологій перероблення пластикових пляшок, які дозволяють переробляти їх у нові пляшки або інші продукти, пластикові кришки все ще часто потрапляють на звалища або в навколишнє середовище [12].

Це пов’язано з відсутністю економічно вигідних методів перероблення кришок, а також складністю їх збору та обробки через малі розміри і різноманітність матеріалів, що використовуються при виготовленні кришок. Іншою проблемою повторного використання кришок пляшок є той факт, що кришки часто відокремлюються від пляшок під час сортування та подальшого перероблення [12].

Факт недостатньої уваги перероблення кришкам пляшок разом з невеликим значенням проценту перероблення пластикових пляшок загалом [11], що є прямою загрозою як клімату так і досягненню поставлених цілей сталого розвитку, підкреслює необхідність впровадження більш ефективних систем збору та перероблення пластикових відходів.

Доречним рішенням проблеми сортування пластикових кришок, які часто потрапляють на звалища також через свої дрібні розміри [12], може стати навчання та впровадження згорткових нейронних мереж для безпосередньої автоматизації процесів сортування та класифікації пластикових кришок в рамках перероблення на основі зображень, отриманих за допомогою камер або сенсорів.

## 1.4 Досвід класифікації малих об’єктів за допомогою CNN у межах сучасних досліджень

З метою вивчення можливостей згорткових нейронних мереж для розв’язання задач детекції й класифікації об’єктів різних розмірів, зокрема малогабаритних та середньогабаритних, і матеріалів, було проаналізовано низку сучасних наукових досліджень.

Особливу увагу приділено прикладам класифікації малих металевих об'єктів [13], а також об'єктів різних розмірів з різноманіттям матеріалів у

контексті виробничого перероблення [14]. Аналіз було проведено для формування подальшої оцінки адаптивності й ефективності підходів на основі CNN для подальшого застосування в реальних системах сортування та ідентифікації відходів виробництва.

### 1.4.1 Застосування алгоритмів CNN для детекції малих металевих об’єктів

Одним з прикладів застосування алгоритмів на основі згорткових нейронних для детекції невеликих металевих об’єктів, таких як цвяхи, болти, гайки, дроти та інші малогабаритні металеві деталі, є наукове дослідження при університеті UniMAP, Малайзія [13].

Для проведення дослідження було зібрано набір даних із зображеннями малих металевих об’єктів, які пройшли попередню обробку: зміну розміру, нормалізацію та аугментацію. Для подальшої класифікації було використано згорткову нейронну мережу на базі архітектури ResNet-50, та проведено порівняння ефективності трьох оптимізаторів, або алгоритмів, які використовуються для оновлення ваг (англ. Weights) нейронної мережі під час навчання: RMSprop, Adam та SGD [13].

За результатами дослідження було виявлено, що найефективнішим підходом для класифікації малих металевих об'єктів є використання згорткової нейронної мережі ResNet-50 у поєднанні з оптимізатором Adam, що дозволило досягти 86% точності в процесі детекції заданих металевих предметів з врахуванням факту, що графіки точності та втрат (рис. 1.4) продемонстрували стабільне зростання точності моделі під час навчання, а також мінімізацію втрат без ознак значного переобучення [13].

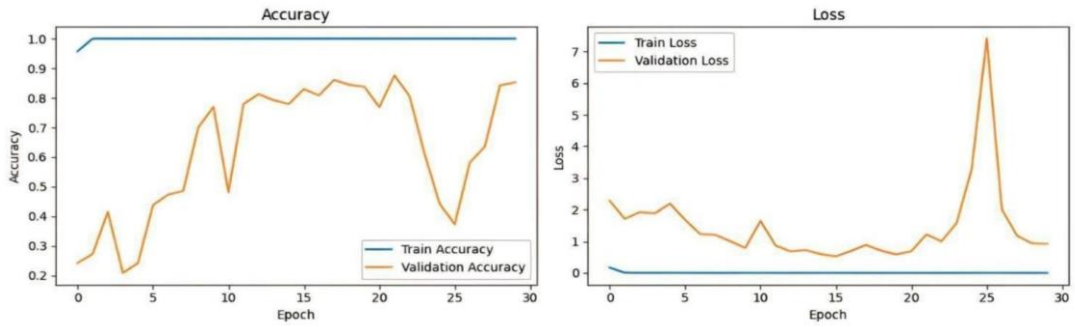



Рисунок 1.4 – Графіки точності та втрат для моделі з оптимізатором Adam [13]

Для повноти аналізу роботи моделі необхідно також звернути увагу на показанні матриці плутанини (англ. Confusion Matrix). Так, на рис. 1.5 можна побачити, що матриця плутанини для запропонованої моделі демонструє 100% точності виявлення класів «Гайка» і «Скріпка», тоді як класи «Ключ» та «Гвинт» виявилися складнішими для виявлення через схожість форми або наявності неточної репрезентації поданих класів на зображеннях в датасеті [13].

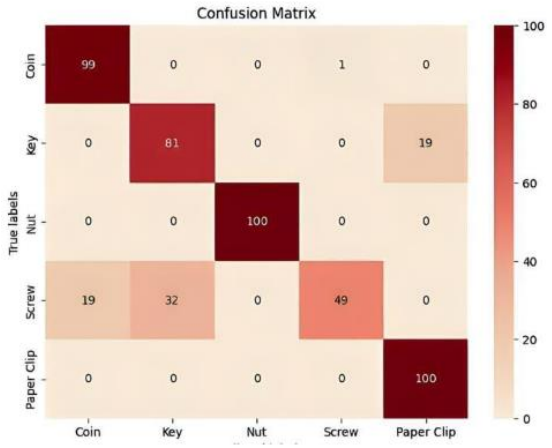



Рисунок 1.5 – Матриця плутанини моделі з оптимізатором Adam [13]

Таким чином, результати дослідження підтверджують, що використання згорткових нейронних мереж, зокрема ResNet-50, може бути ефективним підходом для класифікації малогабаритних об'єктів, оскільки досягнута точність 86% при використанні оптимізатора Adam [13]. Отриманий результат дослідження свідчить про здатність нейромережі точно розпізнавати дрібні

об’єкти при належній підготовці даних датасету: аугментація, нормалізація, зміна розміру зображення [13].

З огляду на отримані результати, підхід з використанням згорткових нейронних мереж можна адаптувати для автоматизованого розпізнавання й сортування пластикових кришок, оскільки кришки мають схожі характеристики з малими металевими деталями, які розглядались в дослідженні, а саме: невеликі розміри, різноманіття форм, різноманіття кольорів.

### 1.4.2 Застосування CNN для детекції відходів із різних матеріалів у виробничому переробленні

З метою детального аналізу можливостей згорткових нейронних мереж в рамках детекції виробів в системах сортування відходів, отриманих в результаті проведення виробничих процесів, було проаналізовано наукове дослідження студентів Вроцлавського університету науки й технологій [14].

Метою дослідження було вивчення сучасних підходів до автоматизованої ідентифікації потоків відходів. Так, в роботі було запропоновано модель ідентифікації в реальному часі на основі CNN, задля створення якої було зібрано відповідний набір даних для більш стійкого датасету [14]. Автоматична ідентифікація потоку була розроблена з використанням CNN, зокрема моделі YOLOv5.

Замість того, щоб навантажувати одну модель усіма класами, було обрано шлях розділення загального масиву даних на підкласи для трьох менших моделей, що дозволило більш точно адаптувати модель до конкретних характеристик відходів виробництва [14]. Так, робота загальної моделі починається з того, що зображення обробляється двома паралельними моделями (1a і 1b). Кожна з них виявляє ті класи відходів, які було обрано для кожної моделі. Архітектура запропонованої моделі проілюстрована на рис. 1.6 [14].

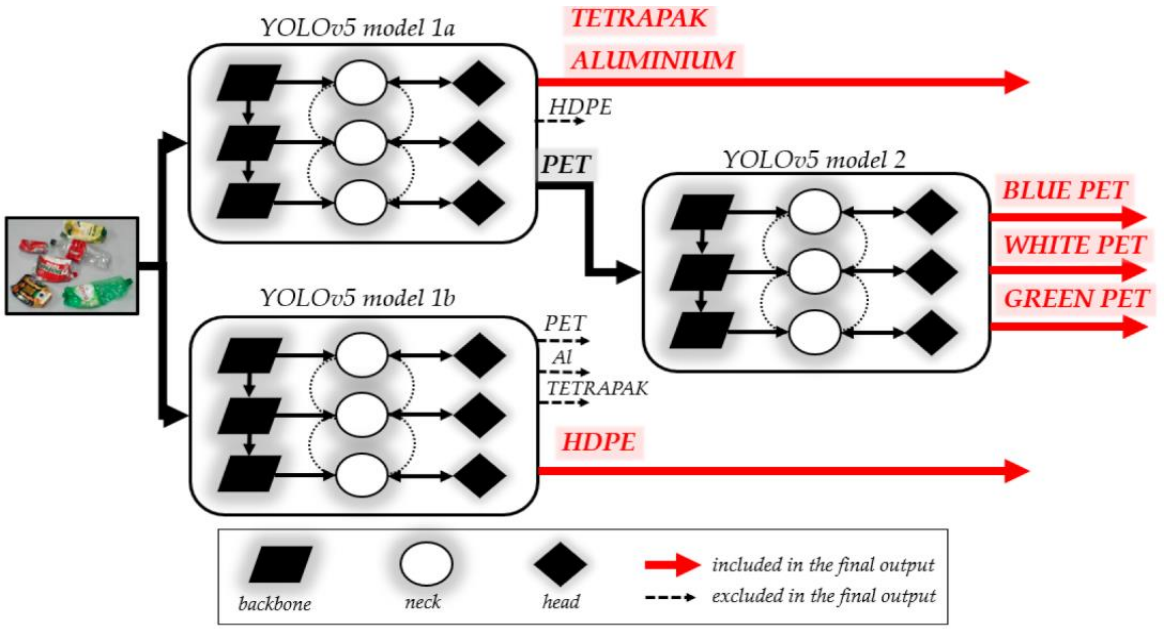



Рисунок 1.6 – Архітектура запропонованої моделі автоматичної ідентифікації потоку відходів на основі YOLOv5 [14]

Присвоєні моделям класи відходів містять PET (Polyethylene terephthalate, укр. Поліетилентерефталат) різних кольорів, HDPE (High-density polyethylene, укр. Поліетилен високої щільності), тетрапак і алюміній [14]. Для кожного класу було забезпечено відповідну кількість проілюстрованих екземплярів на фотографіях задля досягнення збалансованого у всіх категоріях датасету. Для імітації ефекту реальних умов, датасет для кожного з класів містив відповідні зображення з перешкодами, що нерідко трапляється на реальних лініях перероблення відходів виробництва [14].

Продуктивність кожної з трьох моделей оцінювалася за допомогою точності (англ. Precision), відкликання (англ. Recall) та mAP (англ. Mean average precision), які зазвичай використовуються для оцінювання проблем виявлення запропонованих класів в умовах тестування готової моделі [14]. Прогрес метрик моделей представлено на рис. 1.7.

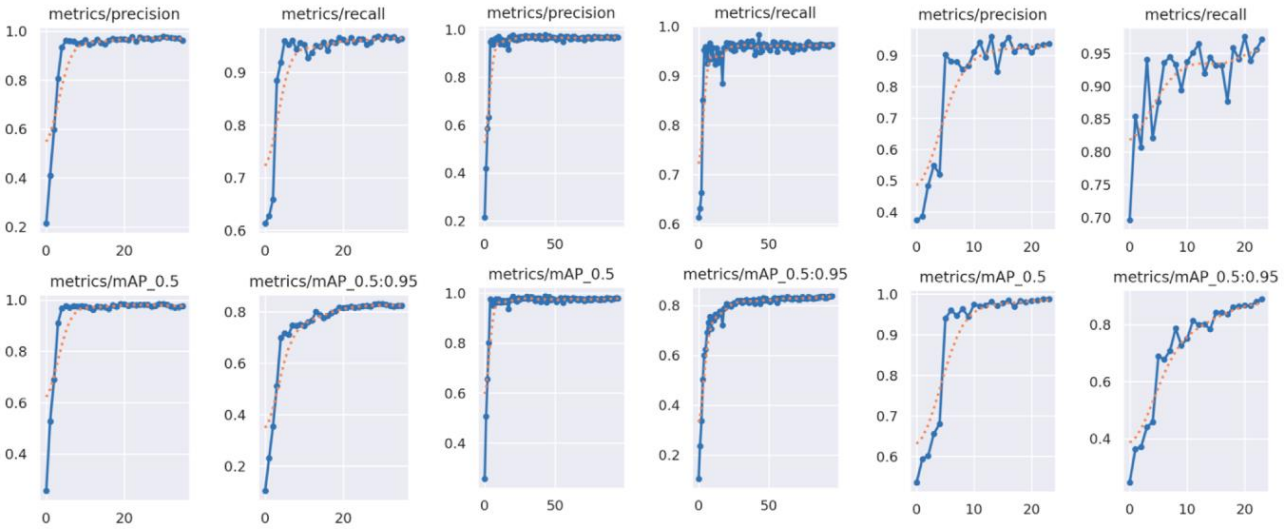


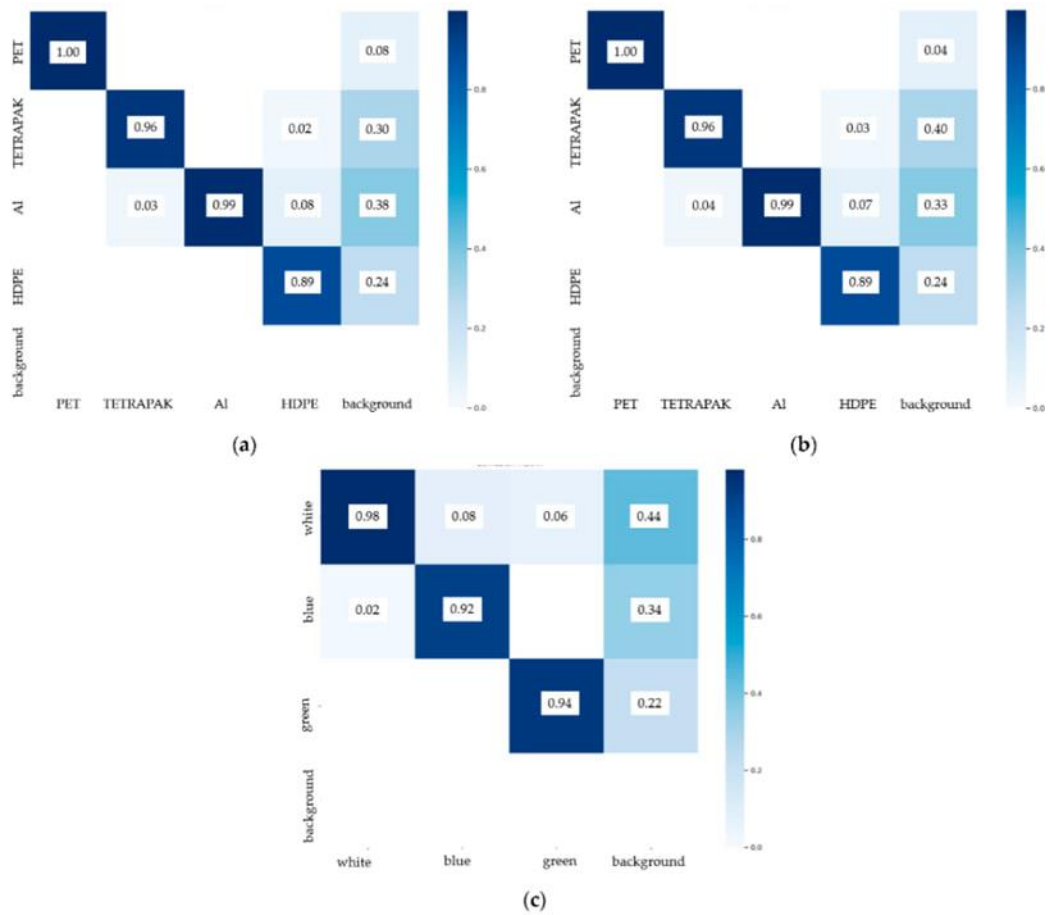

Рисунок 1.7 – Прогрес метрик трьох використаних моделей в ході навчання для моделі a1, 1b та моделі 2 [14]

Порівняння кінцевих показників ефективності для кожної з трьох моделей представлено на рис. 1.8.

| Model | Precision | Recall | mAP_0.5 | mAP_0.5:0.95 |
|---|---|---|---|---|
| 1a | 0.961 | 0.964 | 0.977 | 0.824 |
| 1b | 0.967 | 0.963 | 0.979 | 0.837 |
| 2 | 0.936 | 0.972 | 0.988 | 0.899 |

Рисунок 1.8 – Порівняння кінцевих показників ефективності моделей [14]

Крім того, на рис. 1.9 представлені матриці плутанини, створені навченими моделями YOLO.




а) модель 1a; б) модель 1b; в) модель 2

Рисунок 1.9 – Матриці плутанини створених моделей YOLO [14]

Основні показники ефективності моделей 1a, 1b і 2 показують, що модель 2 має найкращі показники, досягаючи найвищих значень відкликання та mAP, що робить її найбільш точною та стійкою [14]. Модель 1b пропонує хороший баланс між точністю та відкликанням, незначно перевершуючи 1a в характеристиці mAP. Поєднання високого відкликання (1a), високої точності (1b) і високої точності класифікації (2) дозволяє створити оптимізовану конвеєрну систему, збалансовану для детекції відходів виробництва з високою точністю класифікації [14].

Таким чином, проведене дослідження демонструє потенціал та можливість застосування згорткових нейронних мереж як ефективного підходу для ідентифікації та класифікації об'єктів різних розмірів і матеріалів у процесах виробничого перероблення.

# 2 ОБҐРУНТУВАННЯ ВИБОРУ ЗАСОБІВ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

Оскільки метою роботи є створення програмного забезпечення комп’ютерного зору із використанням моделі машинного навчання для подальшої інтеграції з макетом робота-маніпулятора для виконання дій на сортувальній лінії, основними вимогами до такого програмного забезпечення є:

а) висока точність виявлення та класифікації об’єктів в межах робочої зони;

б) можливість отримання й обробки зображень у реальному часі з мінімальною затримкою;

в) стійкість до змін зовнішніх умов освітлення та/або фонового шуму;

г) можливість інтеграції з системою керування маніпулятором.

Задля досягнення зазначених вимог було здійснено підбір оптимальних засобів програмної реалізації.

## 2.1 Вибір мови програмування

Програмну реалізацію проєкту виконано мовою програмування C# у середовищі розробки Visual Studio 2022 (версія 17.13.6), з використанням платформи .NET 8.0, яка забезпечила необхідні можливості для створення всієї програмної частини.

Технологія .NET – порівняно новий інструмент для розробки веб-застосунків і програмного забезпечення, орієнтованого на операційну систему Windows. Платформа .NET 8.0 вирізняється підвищеною продуктивністю завдяки оптимізованому виконанню коду, скороченому часу завантаження застосунків і зниженому споживанню пам’яті [15].

Описані характеристики платформи є ключовими для ефективної реалізації ресурсомістких задач, таких як обробка зображень або використання нейронних мереж, що активно застосовуються в межах даного проєкту. Крім того, .NET 8

має статус версії з довготривалою підтримкою (LTS, англ. Long-term support), що гарантує її актуальність і стабільність упродовж тривалого часу.

Платформа .NET, на якій базується розробка, також підтримує міжплатформенність, багатомовність через CLR (англ. Common Language Runtime) [15], має велику стандартну бібліотеку класів і пропонує широкий набір технологій для створення різноманітних типів програм: веб-додатки (ASP.NET, Blazor), мобільні інтерфейси (Xamarin, MAUI), десктопні інтерфейси (WPF, WinUI). В роботі було використано можливості .NET для розробки десктопного інтерфейсу WPF.

Суттєвими перевагами також є висока продуктивність .NET-застосунків, автоматичне управління пам’яттю через вбудований збирач сміття, а також зручність і ефективність розробки в обраному середовищі Visual Studio.

Мова C# – нова алгоритмічна мова, розроблена для написання програм у середовищі .NET. Основою C# є мови Java та C++. C# поєднує найкращі риси мов Java та C++, має зрозумілий Сі-подібний синтаксис і підтримує повноцінну об’єктно-орієнтовану модель, що забезпечує масштабованість і гнучкість розроблюваних застосунків [15].

C# використовується як для створення невеликих десктопних програм, так і для побудови великих веб-сервісів, що обслуговують мільйони користувачів. Водночас C# не є надлишковою мовою й містить лише необхідні конструкції [15].

C# активно розвивається, з кожним оновленням додаючи нові можливості для розробників. Таким чином, подана мова програмування постійно вдосконалюється, що дозволяє використовувати її для вирішення як класичних, так і новітніх задач [15]. Завдяки підтримці асинхронного програмування, C# ідеально підходить для створення високопродуктивних застосунків, що потребують паралельної обробки великої кількості даних або взаємодії з численними користувачами в реальному часі.

## 2.2 EmguCV, .NET обгортка для бібліотеки обробки зображень OpenCV

OpenCV (Open Source Computer Vision Library) – це бібліотека з відкритим кодом для комп'ютерного зору та машинного навчання, створена для забезпечення спільної платформи для розробки програм і прискорення впровадження технологій машинного сприйняття у комерційні продукти. Вона містить понад 2500 оптимізованих алгоритмів, серед яких є як класичні рішення, так і сучасні розробки в галузі комп'ютерного зору та машинного навчання [16].

OpenCV дозволяє розпізнавати та виявляти обличчя, ідентифікувати об'єкти, класифікувати дії людини на відео, відстежувати рух камери та об'єктів, будувати тривимірні моделі, створювати 3D-хмари точок за допомогою стереокамер, зшивати зображення для отримання панорам високої роздільної здатності [16].

Бібліотека також використовується для пошуку схожих зображень у базах даних, видалення ефекту червоних очей на фотографіях зі спалахом, відстеження руху очей, розпізнавання пейзажів та розміщення маркерів для доповненої реальності.

Завдяки своїй гнучкості й широкому функціоналу OpenCV стала незамінним інструментом як у наукових дослідженнях, так і в розробці промислових і комерційних застосунків [16].

Оскільки в проєкті використовується мова програмування C#, для роботи з комп'ютерним зором було обрано EmguCV – кросплатформенну .NET обгортку для бібліотеки OpenCV, яка дозволяє викликати функції OpenCV із будь-яких мов програмування, сумісних із платформою .NET. EmguCV забезпечує підтримку різних операційних систем, зокрема Windows, Android, iOS, macOS та Linux, що робить її зручною для розробки кросплатформених застосунків [17].

У даному проєкті EmguCV використовується для роботи із зображеннями, отриманими з веб-камери, оскільки тематика проєкту вимагає наявності обробки поточного зображення з камери. Бібліотека дозволяє здійснювати захоплення

відеопотоку, попередню обробку кадрів і виконання складних трансформацій зображень. Завдяки її функціональності стало можливим проводити перспективні перетворення, необхідні для корекції геометричних спотворень, що виникають при зміні кута огляду камери. Крім того, EmguCV була застосована для кодування й декодування зображень у формати, зручні для подальшого відображення чи обробки, а також для роботи з координатами ключових точок на об’єктах.

Використання цієї бібліотеки дозволило ефективно вирішити завдання комп’ютерного зору у проєкті без потреби у складній інтеграції сторонніх інструментів або розробці власних рішень для базових операцій із зображеннями. Завдяки широкій підтримці різних платформ і активній спільноті розробників, EmguCV також гарантує стабільність роботи застосунку й можливість його розширення в майбутньому [17].

## 2.3 Виявлення об'єктів за допомогою алгоритму YOLO

YOLO (You Only Look Once) – це збірка систем для виявлення й класифікації об’єктів у реальному часі, яка дозволяє виконати обидві операції в рамках одного перегляду зображення. Таким чином, моделі YOLO достатньо лише одного перегляду зображення, або ж одного «проходу» зображення через алгоритм, щоб виявити й розпізнати об’єкти. Завдяки такому підходу модель працює з надзвичайною швидкістю, що дає змогу ефективно застосовувати її в режимі реального часу [18].

Термін «виявлення об'єктів» у контексті YOLO означає, що модель здатна не лише точно визначати місцезнаходження об’єктів на зображенні, а й класифікувати їх відповідно до класових знань, отриманих під час навчання. Це працює шляхом пропускання зображення через нейронну мережу, яка дозволяє моделі виявляти всі об'єкти одночасно [18]. Завдяки прогнозуванню на основі сітки (англ. Grid-based prediction) та використанню обмежувальних рамок (англ.

Bounding box prediction), модель може визначити, чи потрапляє об’єкт у певну комірку або обмежувальну рамку, на основі чого зробити припущення про наявність й розташування об’єкта. Крім того, YOLO використовує ймовірність належності до класу для класифікації об’єктів, що дозволяє точно визначити, який саме об’єкт представлено на зображенні [18].

YOLO має низку переваг у порівнянні з попередніми методами виявлення об’єктів на зображенні. Через те, що модель YOLO надзвичайно швидка, вона підходить для низки застосувань із різними вимогами до моделі, наприклад: автономне керування транспортом або використання моделі в системі відеоспостереження. Крім того, модель забезпечує високу точність, особливо при роботі з природними зображеннями, й демонструє меншу кількість хибнопозитивних результатів, оскільки аналізує все зображення цілісно, в результаті чого підвищує контекстуальну точність [18].

### 2.3.1 Загальний принцип роботи YOLO

Серія систем YOLO, розроблена Джозефом Редмоном і вперше представлена в 2015 році, представляє процес виявлення об’єктів як задачу регресії до просторово розділених обмежувальних рамок і пов’язаних з ними ймовірностей класів. YOLO розглядає повне зображення під час тестування, а отже прогнози залежать від глобального контексту зображення. Таким чином, системи YOLO, завдяки принципу своєї роботи, придатні для успішного застосування в реальному часі [18].

Принцип роботи моделі полягає в тому, що алгоритм YOLO використовує одну згорткову нейронну мережу, яка розділяє зображення на сітку. Кожна комірка сітки передбачає певну кількість обмежувальних рамок, разом з якими кожна комірка в сітці також передбачає певну ймовірність класу, яка вказує на ймовірність присутності певного об’єкта в рамці [18].

Принцип роботи моделі YOLO представлено на рис. 2.1.

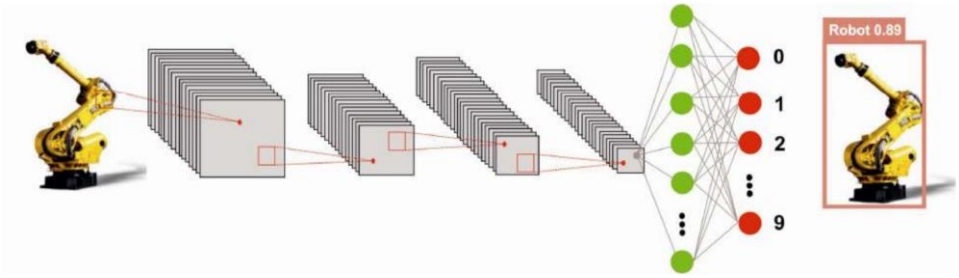



Рисунок 2.1 – Принцип роботи моделі YOLO [19]

2.3.2 Версія моделі YOLOv8

Починаючи випуском моделі YOLOv1 в 2014 році, з кожною новою версією YOLO відбувалося поступове покращення оригінальної моделі завдяки впровадженню нових аспектів, підвищенню середньої точності (mAP) й удосконалень наявних аспектів моделі для покращення її стабільності й продуктивності [18].

Так, з виходом YOLOv8 в 2023 році, було представлено нову архітектуру магістралі, CSPDarknet-AA – покращену версію серії CSPDarknet, відомої своєю ефективністю та продуктивністю у завданнях виявлення об'єктів.

Одним з ключових методів, представлених у YOLOv8, є багатомасштабне виявлення об'єктів, який дозволяє моделі виявляти об'єкти різних розмірів на зображенні [18]. Ще одним суттєвим удосконаленням у YOLOv8 в порівнянні з попередніми версіями є використання функції активації ELU (Exponential Linear Unit). ELU, або експоненціальна лінійна одиниця, допомагає пришвидшити навчання в глибоких нейронних мережах, пом'якшуючи проблему зникнення градієнта, що призводить до швидшої конвергенції [18].

Додатково, YOLOv8 застосовує втрату узагальнений перетин над об'єднанням (англ. Generalized Intersection over Union, GIoU). GIoU, або узагальнений перетин над об'єднанням, є більш просунутою версією метрики перетину над об'єднанням (англ. Intersection over Union, IoU), яка враховує форму та розмір обмежувальних рамок, покращуючи точність локалізації об'єктів [18].

Алгоритм YOLOv8 демонструє збільшення середньої точності mAP на 1,2% порівняно з YOLOv7, що є значним покращенням в контексті оцінки

ефективності машинного навчання. Цього вдалося досягти, зменшивши розмір файлу ваги моделі на 80,6 МБ, що своєю чергою робить модель ефективнішою та простішою у розгортанні в середовищах з обмеженими ресурсами [18].

Архітектура YOLOv8 використовує модифіковану основу CSPDarknet53. В порівнянні з іншими версіями моделі, в архітектурі YOLOv8 модуль C2f замінює CSPLayer, що використовується в YOLOv5, тоді як шар просторової пірамідальної пулінгової швидкості (SPPF) прискорює обчислення шляхом об'єднання ознак у карту фіксованого розміру. Кожна згортка має пакетну нормалізацію та активацію SiLU. Головка відокремлена для незалежної обробки завдань об'єктності, класифікації та регресії [20].

Загальна архітектура моделі YOLOv8 представлена на рис. 2.2.

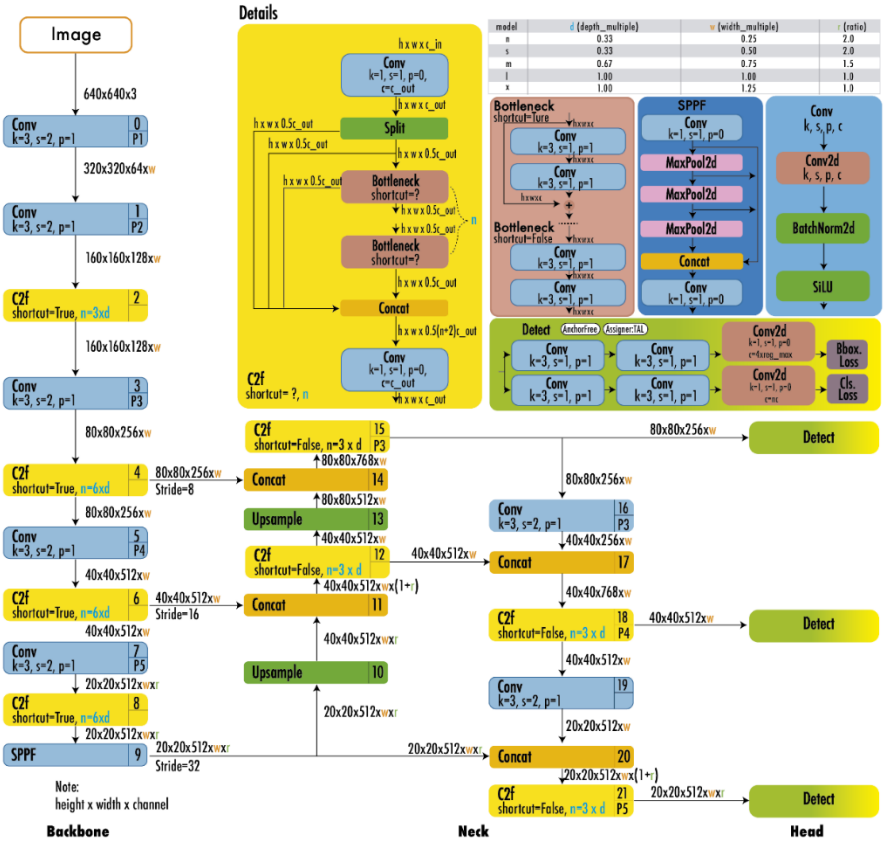



Рисунок 2.2 – Архітектура YOLOv8 [20]

### 2.3.3 Переваги використання моделі YOLO

Модель YOLO має ряд переваг над традиційними методами виявлення об’єктів. Як було попередньо зазначено, моделі YOLO є надзвичайно швидкими та ефективними в контексті виявлення об’єктів. Крім того, для формування прогнозів, YOLO, на відміну від розсувного вікна та методів заснованих на пропозиції, обробляє вхідне зображення глобально, а отже бачить ціле зображення під час навчання й тестування, завдяки чому неявно кодує контекстну інформацію про класи, а також їх зовнішній вигляд [19], що сприяє підвищенню точності виявлення об'єктів, особливо в умовах складного фону або часткового перекриття.

Модель YOLO здатна формувати узагальнені уявлення про об’єкти, що забезпечує високу ефективність навіть за умов зміни домену або появи нових, неочікуваних вхідних даних. Під час навчання на природних зображеннях та тестування на ілюстраціях вона демонструє перевагу над класичними методами виявлення об’єктів, такими як DPM та R-CNN [19].

YOLO використовує ознаки з усього зображення для одночасного передбачення обмежувальних рамок і класів, що дозволяє моделі глобально аналізувати сцену. Такий підхід забезпечує наскрізне навчання, високу швидкість роботи в реальному часі та конкурентну середню точність виявлення об’єктів [19].

Таким чином, поєднання високої швидкості, узагальнюваності, глобального аналізу сцени та наскрізного навчання робить модель YOLO особливо придатною для застосування в системі комп’ютерного зору реального часу, яка моделюється в роботі.

# 3 АПАРАТНА СКЛАДОВА ТА ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ

## 3.1 Об’єкт керування

Задля виконання кінцевої мети роботи, а саме створення програмного забезпечення для інтеграції моделі машинного навчання з функціоналом робота-маніпулятора для виконання виробничих дій, до обраного в ході роботи маніпулятора були висунуті наступні вимоги:

а) наявність надійного захоплювального механізму (хвату);

б) достатня вантажопідйомність для переміщення дрібних об’єктів;

в) висока маневреність і точність позиціонування;

г) можливість інтеграції з системою копм’ютерного зору для коректного розпізнавання й захоплення об’єктів.

Таким чином, згідно з поставленими вимогами до об’єкту керування, для подальшої роботи було обрано учбовий макет робота маніпулятора, зовнішній вигляд якого представлено на рис. 3.1.

Рисунок 3.1 – Макет обраного робота маніпулятора

Представлений на рис. 3.1 робот маніпулятор має хват для захвату й переміщення деталей, можливість обертання навколо своєї осі та два рухомих суглоби.

У конструктивній основі маніпулятора використовуються три крокові двигуни, кожен з яких забезпечує окрему ступінь свободи, кожна з яких оснащена кінцевим датчиком положення. Керування двигунами здійснюється за допомогою модуля управління, побудованого на базі контролера Arduino Mega.

На рисунку 3.2 представлена кінематична схема маніпулятора.

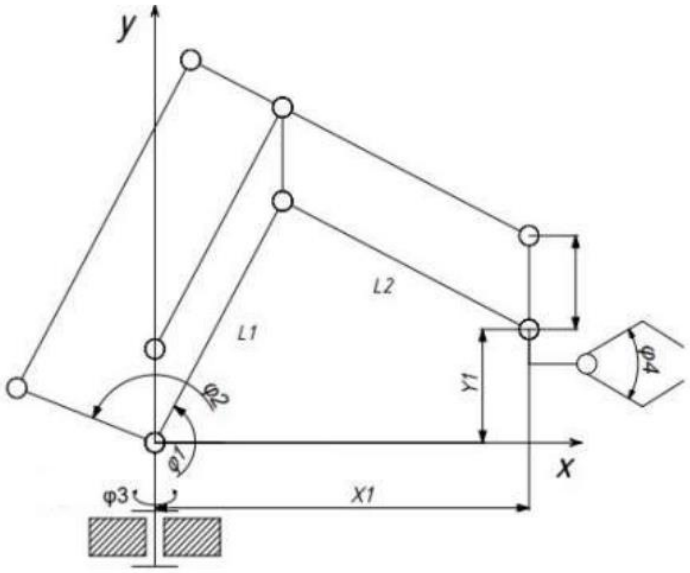



Рисунок 3.2 – Кінематична схема робота маніпулятора

Особливістю конструкції є можливість переміщення рейок вліво та вправо. Кожен крокоий двигун, задіяний в конструкції маніпулятора, реалізує певний ступінь свободи [21].

Так, конструкція містить дві основні ланки: L1 та L2, а також дві обертальні кінематичні пари, що функціонують у межах однієї площини. Переміщення забезпечується шляхом повороту ланок на кути φ1 і φ2. Додаткове обертання на кут φ3 навколо осі Y дозволяє здійснювати просторове позиціонування.

Перша ланка L1 закріплена на основі й обертається на кут φ1. Друга ланка L2 з’єднана з кінцем першої та має можливість обертання відносно неї на кут φ2. Робочий орган розташовується на кінці другої ланки.

### 3.1.1 Розрахунок ПІД-регулятора для системи позиціонування маніпулятора

Система автоматичного управління (САУ) – це сукупність керованого об'єкта й автоматичних вимірювальних і керуючих пристроїв, яка без участі людини формує певні впливи на об'єкт управління, необхідних та достатніх для одержання цілеспрямованого його функціонування із заданою точністю.

Оскільки в системі позиціонування маніпулятора необхідно точно перемістити робочого інструменту в задану точку простору, що вимагає постійного порівняння бажаного положення з фактичним і відповідної корекції помилки, схемою управління подібної системи буде САУ з одиничним негативним зворотним зв'язком.

Так, типова схема управління з одиничним негативним зворотним зв'язком наведена на рис. 3.3.


Рисунок 3.3 – Класична схема САУ [22]

Ефективним методом регулювання параметрів системи є ПІД-регулятор, який приймає дані від різних елементів і генерує відповідний керуючий сигнал.

Вид передавальної функції пристрою управління визначає закон управління. На сьогодні у промисловості розрізняють чотири основні закони

керування: пропорційний, інтегральний, пропорційно-інтегральний, пропорційно-інтегро-диференціальний [22].

При пропорційно-інтегро-диференціальному управлінні пристрій управління формує сигнал, що дорівнює сумі трьох складових: пропорційної похибки, інтегралу від похибки та похідної похибки.

Таким чином, передавальна функція пристрою управління або ПІД-регулятора визначається за формулою (3.1) [22]:

$$W_{\text{ПУ}}(s) = k_1 + \frac{k_2}{5} + k_3 s, \tag{3.1}$$

де $k_1$, $k_2$, $k_3$ – коефіцієнти пропорційної, інтегральної та диференціальної складових відповідно.

У контексті позиціонування маніпулятора подані коефіцієнти підбираються з метою забезпечення швидкого й точного переміщення без перерегулювання й коливань. Для цього необхідно врахувати динамічні характеристики виконавчих елементів, зокрема, крокових двигунів, які приводять у рух ланки маніпулятора.

Для подальшого налаштування ПІД-регулятора, необхідно побудувати математичну модель об'єкта керування, а отже отримати передавальну функцію двигуна, яка пов’язує вхідний сигнал (керуючу напругу) із вихідним сигналом (кут повороту).

Оскільки в роботі розглядається система позиціонування, де вихідним параметром є кутове положення виконавчого інструмента, яке отримується шляхом інтегрування кутової швидкості, а також враховується інерційність електромеханічної системи, об’єкт управління доцільно описати узагальненою моделлю другого порядку. Така модель поєднує інтегруючу ланку та аперіодичну ланку першого порядку, що дозволяє точно відобразити динаміку системи при типовому навантаженні.

Отже, для крокового двигуна узагальнена передавальна функція має вигляд (3.2):

$$W(s) = \frac{K}{s(Ts+1)}, \tag{3.2}$$

де $K$ – коефіцієнт підсилення;

$s$ – оператор Лапласа;

$T$ – постійна часу системи.

За правилами Циглера-Ніколса було експериментально визначено критичні значення $K_{\text{кр}} = 8{,}5$ й $T_{\text{кр}} = 0{,}8$, коефіцієнта підсилення й періоду коливань відповідно, при яких система починає коливатися.

З отриманими значеннями $K_{\text{кр}}$ й $T_{\text{кр}}$ були використані формули Циглера-Нікольса для розрахунків параметрів ПІД-регулятора (3.3) [22]:

$$\begin{cases} K_p = 0{,}6K_{\text{кр}} \\ T_i = 1T_{\text{кр}} \\ T_d = 3T_{\text{кр}} \end{cases}, \tag{3.3}$$

де $K_p$ – пропорційний коефіцієнт;

$T_i$ – інтегральний час;

$T_d$ – диференціальний час.

Таким чином, були отримані наступні значення параметрів ПІД-регулятора: $K_p = 5{,}1$, $T_i = 0{,}4$, $T_d = 0{,}1$.

За умови, що коефіцієнт підсилення системи було прийнято $K = 0{,}001$, тоді як постійна часу системи $T = 0{,}1$, за допомогою пакета прикладних програм для числового аналізу MATLAB було розраховано відповідні передавальні функції розімкненої та замкнутої систем автоматичного управління.

Так, передавальна функція для розімкнутої системи буде мати такий вигляд:

$$W_{\text{роз}}(s) = \frac{0{,}000204s^2 + 0{,}00204s + 0{,}0051}{0{,}04s^3 + 0{,}4s^2}. \qquad (3.4)$$

Передавальна функція для замкнутої системи має такий вигляд:

$$W_{\text{зам}}(s) = \frac{0{,}000204s^2 + 0{,}00204s + 0{,}0051}{0{,}04s^3 + 0{,}4002s^2 + 0{,}00204s + 0{,}0051}. \qquad (3.5)$$

Використаний код MATLAB:

```
s = tf('s');
K = 0.001; T = 0.1;
w = K / (s*(T*s + 1));
Kp = 5.1; Ti = 0.4; Td = 0.1;
pidm = Kp + (Kp / (Ti * s)) + Kp * Td * s;
opensys = pidm * w;
closedsys = feedback(opensys, 1);
```

Для перевірки стійкості системи було побудовано ЛАЧХ розімкнутої системи (рис. 3.4) на основі виразу (3.4) за допомогою функції bode().

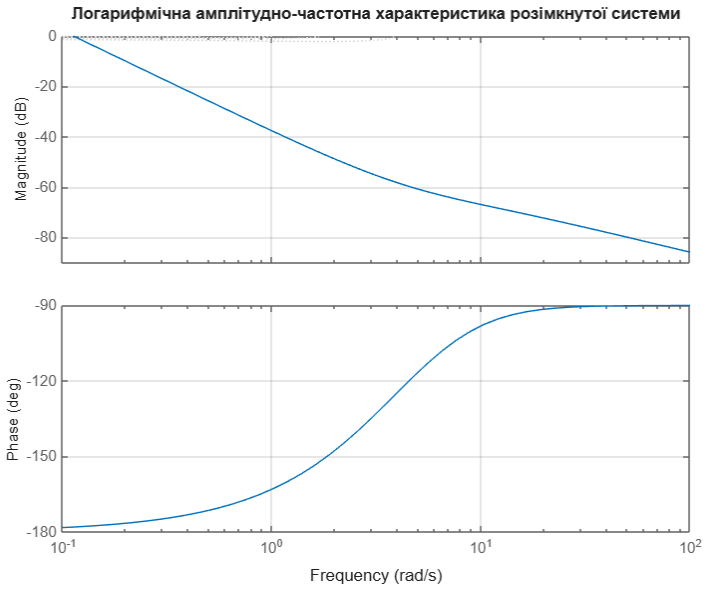



Рисунок 3.4 – ЛАЧХ розімкнутої системи

Для перевірки системи на стійкість було використано критерій Найквіста за допомогою функції nyquist(). Так, на рис. 3.5 можна побачити, що система є стійкою, оскільки вона не перетинає точку (-1; j0).

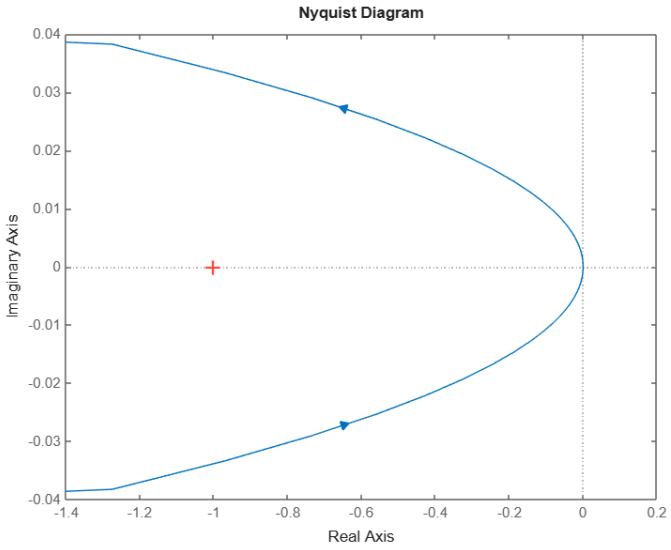



Рисунок 3.5 – Критерій Найквіста

Таким чином, були розраховані параметри ПІД-регулятора з використанням методу Циглеса-Ніколса, побудовано передавальні функції розімкненої та замкнутої систем з врахуванням параметрів ПІД-регулятора,

проведено перевірку стійкості системи. Отримані результати підтверджують ефективність налаштованого регулятора для забезпечення точного й стабільного позиціонування маніпулятора без перерегулювання або коливань.

## 3.2 Процес тренування моделі YOLO

З врахуванням розмірів датасету, використаного для тренування та валідації, а також кількості класів, задля задоволення вимог проєкту було обрано модель YOLOv8s, яка добре підходить для роботи в реальному часі й поєднує в собі баланс продуктивності та точності. Окрім того, використання поданої моделі забезпечує швидке розгортання на пристроях із обмеженими обчислювальними ресурсами та дозволяє досягати прийнятної якості розпізнавання без потреби у великих обсягах обчислень [5].

В якості платформи для анотації зображень датасету було обрано сервіс Roboflow – комплексну платформу комп'ютерного зору, яка спрощує процес побудови моделей комп'ютерного зору й налічує широкий набір інструментів для завантаження, анотації, попередньої обробки й експорту даних [5]. Безпосереднє тренування моделі YOLO відбувалося за використання ресурсів інтегрованого середовища розробки для мови програмування Python, PyCharm (версія 2024.3.4), з подальшою конвертацією отриманої моделі з розширення .pt в .ONNX для сумісності й зручного використання моделі в середовищі .NET [5].

### 3.2.1 Підготовка датасету

Перед тим, як розпочати безпосереднє тренування моделі YOLO, було зібрано відповідний датасет. Загальний об’єм початкового датасету склав 380 зображень, тоді як загальна кількість класів, анотованих на зображеннях склала 4 основних класи для загального виявлення об’єктів, а також 1 додатковий клас для виявлення контрольних точок поверхні, в рамках якої проводиться загальна детекція об’єктів [5]. Таким чином, чотири основні класи призначені для детекції

малогабаритних об’єктів, що розрізняються за геометричною формою (_piece або _king) та кольором (white_ або black_), тоді як п’ятий клас відповідає за визначення опорних точок, необхідних для перспективної трансформації й нормалізації координат вхідного зображення [5].

Оскільки кількість класів датасету є відносно невеликою в рамках тренування моделі, з метою запобігання перетренування (англ. Overfitting), яке виникає тоді, коли модель надто точно запам’ятовує навчальні дані й втрачає здатність узагальнювати на нові приклади, до зображень було застосовано набір ручних аугментацій [5].

Зокрема, близько 13% зображень були віддзеркалені (горизонтально або вертикально), 20% зображень були зроблені в умовах зниженого освітлення, ще 15% – з жовтим світловим відтінком, а приблизно 10% зображень містили візуальні перешкоди (наприклад, часткове перекриття об’єктів, відблиски або тіні), що моделюють реальні експлуатаційні умови [5].

Таким чином, кожне з 380 зображень датасету пройшло етап попередньої обробки та аугментації, після цього анотовані засобами платформи Roboflow з використанням обмежувальних рамок, як показано на рис. 3.6 [5].

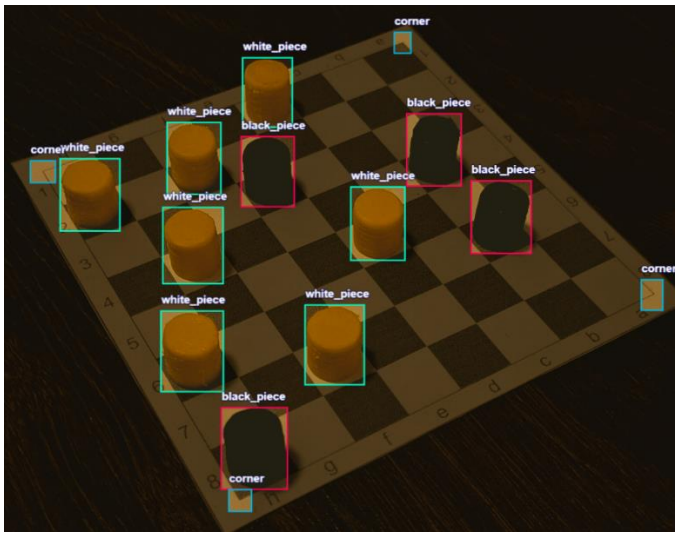



Рисунок 3.6 – Процес анотації датасету методом обмежувальних рамок [5]

За результатами проведеного анотування зображень було отримано сумарно 2686 анотацій серед всіх п’яти класів, що дорівнює 7,1 анотації на одне зображення [5].

Так, на рис. 3.7 представлено огляд кількості анотацій для кожного класу датасету.


Рисунок 3.7 – Загальна кількість анотацій для п’яти класів датасету [5]

### 3.2.2 Тренування моделі YOLO засобами середовища PyCharm

Одразу після анотації зображень та фінальної перевірки розмітки, сформований датасет було використано для навчання моделі YOLOv8s в інтегрованому середовищі розробки PyCharm. Для забезпечення об'єктивної оцінки продуктивності моделі, датасет було розділено у співвідношенні 75/25, де 75% зображень використовувалися для навчання, а 25% – для валідації [5].

З метою підвищення середньої точності mAP й загальної продуктивності моделі, за основу була взята попередньо натренована модель yolov8s.pt, що дозволило пришвидшити процес тренування та покращити узагальнення на нових даних [5].

Навчання моделі відбувалося протягом 30 епох (англ. Epochs). Кількість епох була обрана з огляду на помірний розмір датасету й обмежену кількість класів, що не потребують надмірно тривалого процесу навчання.

Для налаштування процесу було використано конфігураційний файл data1.yaml, у якому були зазначені шляхи до навчального та валідаційного наборів зображень, а також перелік класів, які підлягали детекції [5].

Нижче наведено фрагмент коду, що був використаний для ініціалізації процесу тренування моделі:

```
from ultralytics import YOLO
model = YOLO("../yolov8s.pt")
results = model.train(data="data1.yaml", epochs=30)
```

По завершенню тренування моделі була автоматично створена тека runs в корені проєкту , яка містила результати навчання, зокрема: збережену модель, графіки втрат (cls_loss, dfl_loss, box_loss), метрики точності (precision, recall, mAP) та допоміжні візуалізації [5].

На рис. 3.8 представлено графіки втрат, а також метрики точності.

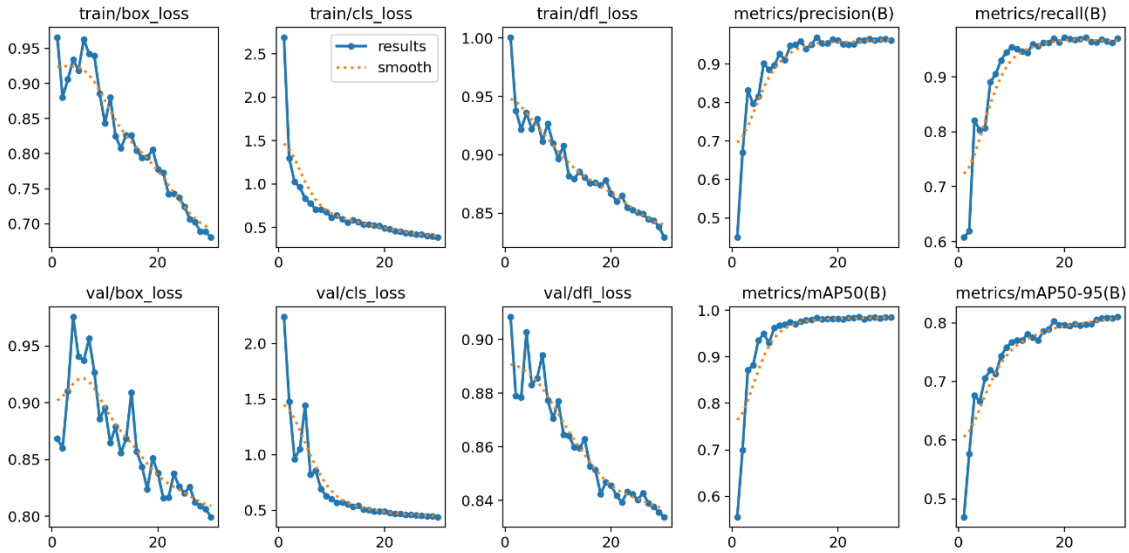



Рисунок 3.8 – Результати тренування моделі YOLOv8s: графіки втрат і метрики точності [5]

Як видно на графіках, значення втрат поступово зменшуються з кожною епохою, що свідчить про зниження помилок під час навчання, тоді як метрики точності стабільно зростають, що вказує на покращення здатності моделі

правильно класифікувати об'єкти. Звідси можна зробити висновок про відсутність перетренування й позитивні результати навчання моделі [5].

Наступним кроком, модель була оцінена на валідаційному датасеті, який складався з 94 зображень або 632 анотованих об'єктів відповідно. Результати оцінки продемонстрували високу ефективність, що закріплює попередні висновки з графіків: точність (англ. Precision) становила 0,962, повнота (англ. Recall) – 0,97, середня точність при порозі IoU = 0,5 (mAP@50) – 0,985, а середня точність у діапазоні IoU від 0,5 до 0,95 (mAP@50-95) склала 0,81 [5], як показано на рис. 3.9.

| Class | Images | Instances | Box(P | R | mAP50 | mAP50-95) |
|---|---|---|---|---|---|---|
| all | 94 | 632 | 0.962 | 0.97 | 0.985 | 0.81 |
| black_king | 44 | 81 | 0.924 | 0.951 | 0.984 | 0.872 |
| black_piece | 54 | 105 | 0.942 | 0.971 | 0.981 | 0.886 |
| corner | 94 | 317 | 0.967 | 0.937 | 0.969 | 0.5 |
| white_king | 21 | 41 | 0.988 | 1 | 0.995 | 0.907 |
| white_piece | 41 | 88 | 0.986 | 0.989 | 0.994 | 0.886 |

Рисунок 3.9 – Результат оцінки моделі на валідаційному датасеті [5]

Для додаткового підтвердження ефективності моделі була проаналізована матриця плутанини (англ. Confusion matrix), представлена на рис. 3.10.

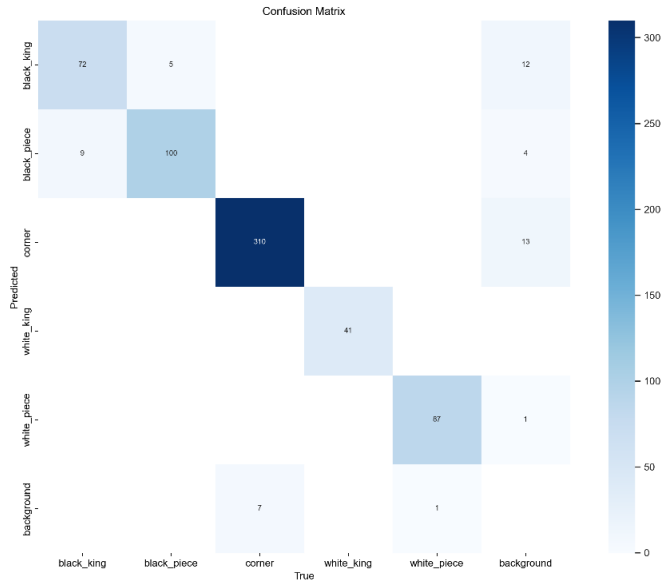



Рисунок 3.10 – Матриця плутанини [5]

Матриця невідповідностей є табличною формою, яка демонструє кількість істинно позитивних (англ. True Positives, TP), хибно позитивних (англ. False Positives, FP), істинно негативних (англ. True Negatives, TN) та хибно негативних (англ. False Negatives, FN) спрацювань [5]. Аналіз матриці надає більш глибоке розуміння здатності моделі правильно класифікувати об’єкти кожного класу, а також дозволяє оцінити ступінь помилкових класифікацій між окремими класами.

Так, з рис. 3.10 видно, що загальний рівень продуктивності моделі залишається високим, однак певна плутанина все ж спостерігається у випадку з додатковим класом corner [5]. З огляду на допоміжну роль цього класу у загальній задачі детекції, його відносно нижче значення за суворішим критерієм mAP@50-95 не критичне для загального функціонування системи і є прийнятними в контексті поставлених цілей проєкту. Високі значення інших характеристик, таких як точність і повнота, підтверджують, що модель здатна ефективно розпізнавати цей клас [5].

Додатково, модель було протестовано на відеофрагментах і статичних зображеннях із реальних умов (рис. 3.11).

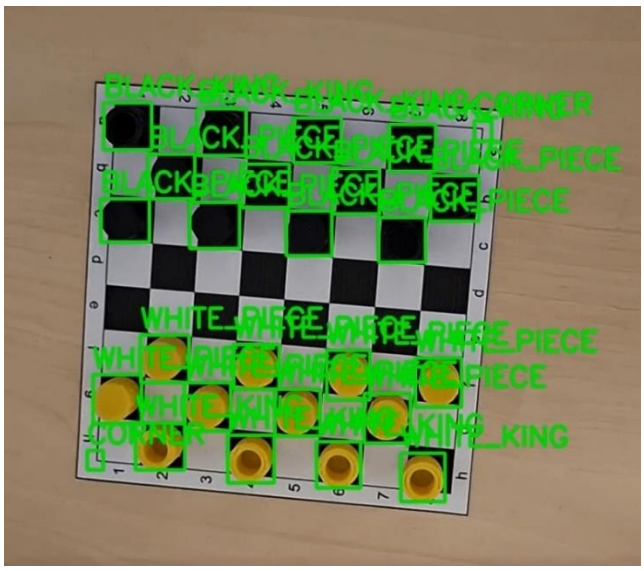


Рисунок 3.11 – Тестування моделі [5]

Модель продемонструвала стабільну роботу й високу здатність до генералізації, коректно виявляючи об'єкти різних класів за умов змінного фону та освітлення [5].

### 3.2.3 Порівняння з альтернативною моделлю на розширеному датасеті

Оскільки початкова модель тренувалася на датасеті, що складався з 380 зображень, з метою перевірки впливу обсягу даних на якість розпізнавання було прийнято рішення розширити навчальний набір до 950 зображень.

Для формування нового датасету за допомогою інструментів Roboflow було застосовано низку аугментацій, що дозволяють урізноманітнити вхідні дані й підвищити здатність моделі до генералізації. Зокрема, використовувалися такі трансформації:

а) горизонтальне та вертикальне віддзеркалення застосовувалось до частини зображень для симуляції змін орієнтації об'єктів;

б) перетворення в градації сірого застосовано до 5% зображень;

в) додана зміна яскравості в діапазоні від -10% до +10%;

г) додане розмиття до 2 пікселів;

г) додано шум до 0,5% пікселів на зображенні.

Значення кількості епох залишилося незмінним, як і загальні параметри навчання, з метою забезпечення коректного порівняння результатів обох моделей в однакових умовах.

Так, отримані результати оцінки другої моделі наведені на рис. 3.12.

| Class | Images | Instances | Box(P | R | mAP50 | mAP50-95): |
|---|---|---|---|---|---|---|
| all | 94 | 632 | 0.958 | 0.984 | 0.989 | 0.828 |
| black_king | 44 | 81 | 0.941 | 0.984 | 0.99 | 0.901 |
| black_piece | 54 | 105 | 0.92 | 0.981 | 0.99 | 0.889 |
| corner | 94 | 317 | 0.972 | 0.968 | 0.98 | 0.546 |
| white_king | 21 | 41 | 0.984 | 1 | 0.995 | 0.901 |
| white_piece | 41 | 88 | 0.973 | 0.989 | 0.992 | 0.903 |

Рисунок 3.12 – Результат оцінки другої моделі на валідаційному датасеті

Як видно з рис. 3.12, суттєвих змін у показниках точності та загальної продуктивності моделі не спостерігалося. Різниця у значеннях метрик для обох моделей коливається в межах ±0,1, що вказує на обмежений вплив збільшення обсягу датасету за поточних умов навчання.

З огляду на отримані результати обох моделей, було прийнято рішення використовувати першу версію моделі, оскільки вона забезпечує зіставну якість детекції при меншому обсязі навчальних даних, що знижує обчислювальні витрати та пришвидшує процес розгортання в рамках реального застосування.

### 3.2.4 Порівняння отриманих результатів з попередньо розглянутими роботами

Оскільки в науковому дослідженні, проведеному в університеті UniMAP [13], для оцінки ефективності моделей використовується характеристика F1 (англ. F1-score), для об’єктивного порівняння загальної продуктивності моделей було розраховано відповідну метрику F1 для моделі, представленої в даній роботі, на основі попередньо викладених даних (рис. 3.9).

Оцінка F1 широко використовується в машинному навчанні для оцінки продуктивності моделей класифікації. Вона забезпечує збалансоване уявлення про точність і повноту, що робить її особливо корисною в сценаріях, де обидві метрики є критично важливими [23].

Так, метрику F1 можна розрахувати за наступною формулою (3.6) [23]:

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}. \quad (3.6)$$

Результати розрахунків, а також значення оцінки F1 для кожного класу розглянутих моделей наведено в табл. 3.1.

Таблиця 3.1 – Метрика F1 для розглянутих моделей

| № класу | Назва класу | Назва моделі | | |
|---|---|---|---|---|
| | | UniMAP | NURE | WUST |
| | | Значення характеристики F1 | | |
| 1 | Coin / black_king / blue PET | 0,91 | 0,93 | 0,94 |
| 2 | Key / black_piece / green PET | 0,76 | 0,96 | 0,91 |
| 3 | Nut / corner / white PET | 1,00 | 0,95 | 0,91 |
| 4 | Screw / white_king / tetrapak | 0,65 | 0,99 | 0,89 |
| 5 | Paper Clip / white_piece / AI | 0,91 | 0,98 | 0,91 |
| 6 | HDPE | - | - | 0,84 |
| Середнє значення: | | 0,86 | 0,96 | 0,9 |

Оскільки кількість класів в кожній моделі відрізняється, використані класи позначені номерами згідно з порядком їх розташування в таблицях досліджень. В табл. 3.1, моделі позначені наступним чином:

а) UniMAP – модель, представлена в науковій роботі при університеті UniMAP, [13];

б) NURE – модель, представлена в даній роботі;

в) WUST – модель, представлена в науковій роботі при Вроцлавському університеті науки й технологій, [14].

Таким чином, за даними табл. 3.1 було побудовано графік для наглядної оцінки рівня продуктивності моделей на основі даних кожного з досліджуваних класів (рис. 3.13).

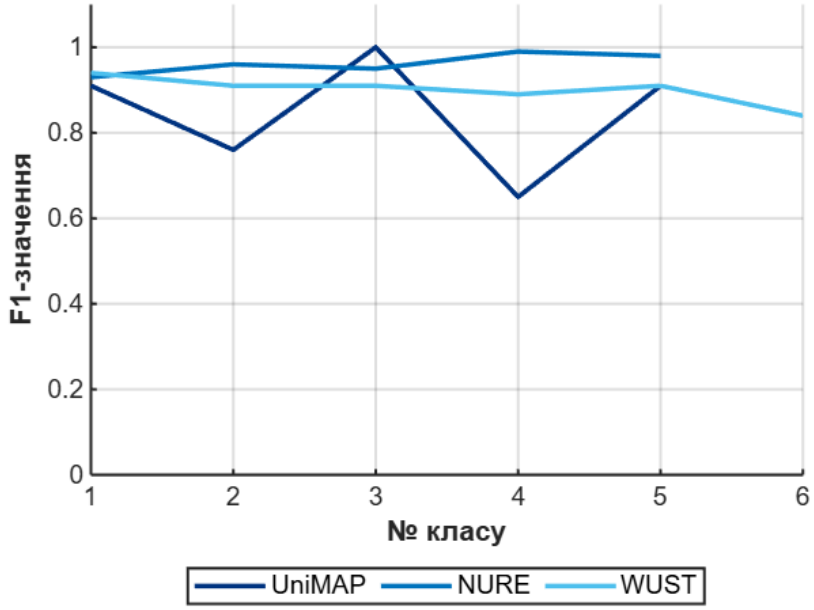



Рисунок 3.13 – Порівняння F1-метрики для моделей UniMAP, NURE та WUST

Загальна оцінка середнього значення F1-метрики серед трьох моделей представлена на рис. 3.14.

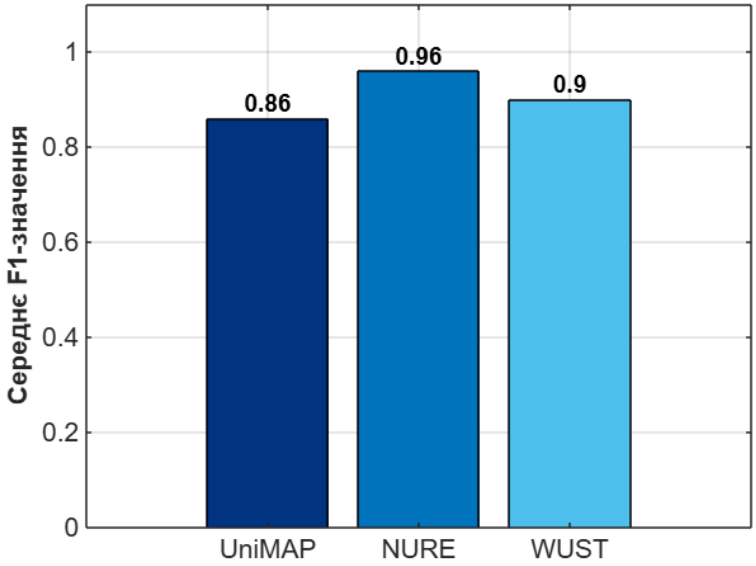



Рисунок 3.14 – Порівняння середніх значень F1-метрики моделей UniMAP, NURE та WUST

В підсумку проведеного дослідження, було розроблено модель, здатну ефективно класифікувати дрібні об’єкти, що підтверджується високим

значенням F1-метрики й середньої точності. Отримані результати свідчать про те, що запропонований підхід не лише перевершує існуючу модель, розроблену при університеті UniMAP та орієнтовану на виявлення дрібних об’єктів, а й демонструє конкурентоспроможність порівняно з більш універсальною моделлю, розробленою при Вроцлавському університеті науки й технологій.

## 3.3 Алгоритм роботи програмного забезпечення

Фрагмент коду розробленої програми, поданий нижче, є ключовим елементом системи та відповідає за захоплення відеопотоку з вебкамери, обробку кадрів у реальному часі з використанням нейромережі YOLO для виявлення об’єктів, а також за відображення результатів детекції в інтерфейсі користувача.

Так, на рис. А.1 (Додаток А) представлено загальний алгоритм роботи поданого фрагменту програми.

Конструктор MainWindow() відповідає за початкову ініціалізацію головного вікна програми за допомогою методу InitializeComponent(), задає стан й розміри вікна, а також відповідає за підготовку ресурсів, необхідних для обробки відеопотоку та запуску моделі YOLO.

_yolo = new Yolo(new YoloOptions { ... }) ініціалізує модель YOLO для задачі розпізнавання об’єктів з використанням CPU.

_dispatcher = Dispatcher.CurrentDispatcher ініціалізує графічний диспетчер з подальшою ініціалізацією кадрів для їх збереження в _currentFrame та подальшого отримання трансформованого зображення (_transformedFrame).

Наступним кроком викликаються методи ініціалізації додаткових компонентів, як-от: InitializePdnMappings(), InitializeDatabaseManager(), які забезпечують зв’язок між координатами розпізнаних об’єктів і внутрішнім представленням позицій об’єктів на обмеженому полі, а також забезпечують підключення до бази даних для зберігання або обробки інформації про роботу програми.

Змінна _cameraActive встановлюється у значення false, що свідчить про початковий неактивний стан відеопотоку.

Загальний код конструктора MainWindow() представлено нижче:

```
public MainWindow()
{
    InitializeComponent();
    this.WindowState = WindowState.Normal;
    this.Width = 960;
    this.Height = 650;
    _yolo = new Yolo(new YoloOptions
    {
        OnnxModel = @"C:\path\model.onnx",
        ModelType = ModelType.ObjectDetection,
        Cuda = false,
        GpuId = 0,
        PrimeGpu = false,
    });
    _dispatcher = Dispatcher.CurrentDispatcher;
    _currentFrame = SKImage.FromBitmap(new
SKBitmap(WEBCAM_WIDTH, WEBCAM_HEIGHT));
    _transformedFrame = SKImage.FromBitmap(new SKBitmap(BOARD_SIZE,
BOARD_SIZE));
    _webcamRect = new SKRect(0, 0, WEBCAM_WIDTH,
WEBCAM_HEIGHT);
    _transformedRect = new SKRect(0, 0, BOARD_SIZE, BOARD_SIZE);
    InitializePdnMappings();
    InitializeDatabaseManager();
    _cameraActive = false;
```

```
}
```

Метод WebcamAsync() реалізує асинхронне захоплення відеопотоку з вебкамери у режимі реального часу. Він запускається в окремому потоці при активації камери й працює доти, доки не буде отримано сигнал про скасування обробки кадрів відеопотоку через CancellationToken.

Камера ініціалізується за допомогою бібліотеки Emgu.CV, з налаштуванням частоти й розміру кадрів, які постійно читаються у циклі, доки не найде запит на скасування операції. Задля запобігання надмірного навантаження на CPU, була встановлена відповідна затримка між кадрами, що відповідає бажаній частоті (FPS).

Так, захоплений кадр зчитується в об’єкт Mat й конвертується у формат SKImage для подальшої обробки та відображення за допомогою SkiaSharp.

В разі, якщо активований режим детекції, кадр обробляється моделлю YOLO. Для постійного оновлення інтерфейсу у головному потоці WPF використовується диспетчер інтерфейсу _dispatcher.

Код методу WebcamAsync() наведено нижче:

```
private async Task WebcamAsync(CancellationToken cancellationToken)
        {
            try
            {
                _capture = new VideoCapture(0, VideoCapture.API.DShow);
                _capture.Set(CapProp.FrameCount, FPS);
                _capture.Set(CapProp.FrameWidth, WEBCAM_WIDTH);
                _capture.Set(CapProp.FrameHeight, WEBCAM_HEIGHT);
                using var mat = new Mat();
                using var buffer = new VectorOfByte();
                while (!cancellationToken.IsCancellationRequested)
```

```
                {
                    _capture.Read(mat);
                    mat.CopyTo(_latestMat);
                    CvInvoke.Imencode(FRAME_FORMAT_EXTENSION, mat,
buffer);
                    buffer.Position = 0;
                    _currentFrame = SKImage.FromEncodedData(buffer);
                    buffer.Clear();
                    if (_runDetection)
                    {
                        var results = _yolo.RunObjectDetection(_currentFrame);
                        ProcessDetectionResults(results);
                        DrawDetectionResults(_currentFrame, results);
                    }
                    await _dispatcher.InvokeAsync(() =>
WebCamFrame.InvalidateVisual());
                    await Task.Delay(1000 / FPS, cancellationToken);
                }
            }
            catch (OperationCanceledException)
            {
            }
            catch (Exception ex)
            {
                await _dispatcher.InvokeAsync(() =>
                    MessageBox.Show($"Помилка камери: {ex.Message}",
"Помилка", MessageBoxButton.OK, MessageBoxImage.Error));
            }
            finally
```

```
        {
            _capture?.Dispose();
            _capture = null;
        }
    }
```

Метод ProcessDetectionResults() обробляє результати виявлення об'єктів, отримані від моделі YOLO. Основною метою методу є фільтрація і обробка результатів для виявлення і класифікації ключових об'єктів, заданих в класах моделі.

В даному випадку обробляється п'ятий, додатковий клас, для визначення кутів, що формують межі робочої поверхні для розпізнавання інших об'єктів всередині цієї зони. Метод здійснює перевірку на наявність усіх чотирьох кутових об'єктів, після чого виконується обчислення їхніх центрів і впорядкування координат для коректної геометричної інтерпретації зони інтересу.

Код методу ProcessDetectionResults() наведений нижче:

```
    private void ProcessDetectionResults(List<ObjectDetection> results)
    {
        var cornerDetections = results.FindAll(d => d.Label.Name == "corner");
        if (cornerDetections.Count == 4)
        {
            _cornerPoints.Clear();
            foreach (var detection in cornerDetections)
            {
                float centerX = detection.BoundingBox.Left +
(detection.BoundingBox.Width / 2);
```

```
                float centerY = detection.BoundingBox.Top +
(detection.BoundingBox.Height / 2);
                _cornerPoints.Add(new PointF(centerX, centerY));
            }
            var orderedPoints = OrderCornerPoints(new List<PointF>(_cornerPoints));
            _cornerPoints.Clear();
            _cornerPoints.AddRange(orderedPoints);
            _cornersDetected = true;
        }
    }
```

Метод DrawDetectionResults() відповідає за візуалізацію результатів детекції, малюючи на кадрі прямокутники, що обмежують виявлені об'єкти, а також лінії для виявлених кутів.

Для того, щоб малювати на зображенні, воно перетворюється у формат SKBitmap, який дозволяє здійснювати малювання за допомогою об'єкта SKCanvas. Після перетворення для кожного виявленого об'єкта малюється прямокутник з певним кольором, залежно від типу об'єкта за його класом. Для цього використовується об'єкт SKPaint, який визначає колір і товщину лінії.

Далі, зображення оновлюється й зберігається в _currentFrame.

Код методу DrawDetectionResults() представлено нижче:

```
    private void DrawDetectionResults(SKImage image, List<ObjectDetection>
results)
    {
        using SKBitmap bitmap = SKBitmap.FromImage(image);
        using SKCanvas canvas = new(bitmap);
        foreach (var detection in results)
        {
```

```
            SKColor boxColor = detection.Label.Name switch
            {
                "black_king" => SKColors.DarkRed,
                "black_piece" => SKColors.Red,
                "white_king" => SKColors.DarkBlue,
                "white_piece" => SKColors.Blue,
                _ => SKColors.Red
            };
            using SKPaint boxPaint = new()
            {
                Color = boxColor,
                StrokeWidth = 2,
                Style = SKPaintStyle.Stroke
            };
            canvas.DrawRect(
                detection.BoundingBox.Left,
                detection.BoundingBox.Top,
                detection.BoundingBox.Width,
                detection.BoundingBox.Height,
                boxPaint);
        }
        _currentFrame = SKImage.FromBitmap(bitmap);
    }
```

Таким чином, представлена частина коду є ключовим елементом програми, що забезпечує інтеграцію потокового відео з вебкамери, реального розпізнавання об’єктів за допомогою нейромережі YOLO та динамічного відображення результатів у графічному інтерфейсі. Реалізовано окрему обробку спеціального

класу для визначення меж робочої області, що дозволяє підвищити точність розпізнавання основних об'єктів лише в межах цієї зони.

Такий підхід покращує загальну продуктивність системи й зменшує кількість хибних спрацювань.

## 3.4 Охорона праці

Згідно з постановою Верховної Ради № 2695-XII від 14.10.92, охорона праці – це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності [24].

Роботодавець зобов'язаний створити на робочому місці в кожному структурному підрозділі умови праці відповідно до нормативно-правових актів, а також забезпечити додержання вимог законодавства щодо прав працівників у галузі охорони праці [24].

Відповідно до стандарту ДСТУ EN ISO 10218-2:2018, який визначає вимоги щодо безпеки в разі інтеграції промислових роботів і промислових роботизованих систем, визначених в ISO 10218-1, а також роботизованих модулів, основнимі вимоги для забезпечення охорони праці описані в п. 5.2 «Характеристики систем керування, пов’язаних із безпекою (апаратні засоби та програмне забезпечення)» [25].

До таких вимог, згідно з п. 5.2 стандарту ДСТУ EN ISO 10218-2:2018, відносяться:

а) системи керування, що відповідають за безпеку, повинні мати визначений рівень продуктивності або рівень цілісності безпеки відповідно до стандартів ISO 13849-1 або IEC 62061. У разі відмови системи керування, пов'язаної з безпекою, має відбуватися зупинка категорії 0 або 1 згідно з IEC 60204-1 [25];

б) система робота та захисні заходи повинні бути спроєктовані з урахуванням умов навколишнього середовища, таких як температура, вологість, електромагнітні перешкоди та освітлення приміщення. Компоненти системи повинні витримувати очікувані експлуатаційні та екологічні умови [25];

в) елементи керування та обладнання (наприклад, контролер зварювання, пневматичні клапани тощо), що потребують доступу під час автоматичної операції повинні бути розташовані поза захищеним простором. Таким чином, людина яка використовує виконавчі механізми керування, повинна знаходитися поза межами захищеного простору. Елементи керування та обладнання повинні бути розміщені та сконструйовані таким чином, щоб забезпечити чіткий огляд обмеженого простору робота [25];

г) роботизована система не повинна реагувати на будь-які зовнішні дистанційні команди або умови, які можуть спричинити небезпечні ситуації.

Стандартом ДСТУ EN ISO 10218-2:2018 також передбачаються вимоги до джерел живлення, їх ізоляції та контролю [25]:

а) усі джерела живлення для роботів та іншого обладнання (наприклад, пневматичне, гідравлічне, механічне, електричне) повинні відповідати вимогам, зазначеним виробниками машин та компонентів. Електричні установки повинні відповідати вимогам IEC 60204-1. Гідравлічні установки повинні відповідати вимогам ISO 4413, а пневматичні установки повинні відповідати вимогам ISO 4414 [25];

б) захисне з'єднання та функціональне з'єднання повинні відповідати вимогам IEC 60204-1;

в) повинні бути передбачені засоби для ізоляції небезпечних джерел енергії без нараження персоналу на небезпеку. Ці засоби повинні бути замкненими та/або закріпленими лише у знеструмленому положенні. Роботизована система повинна мати один пристрій відключення живлення для кожного типу джерела енергії. Для кількох роботів або великих установок може знадобитися кілька пристроїв відключення для кожного типу енергії [25];

г) повинні бути передбачені засоби для контролю та/або контрольованого вивільнення накопиченої небезпечної енергії. Повинна бути наклеєна етикетка для ідентифікації небезпеки, пов'язаної зі накопиченою енергією [25].

Таким чином, під час роботи з промисловими роботами й роботизованими системами важливо дотримуватися вимог чинного законодавства задля забезпечення охорони праці, передбачених як національними нормативно-правовими актами, так і міжнародними стандартами.

Дотримання вказаних стандартами вимог дозволяє забезпечити безпечні умови праці, знизити ймовірність виробничого травматизму та підвищити надійність функціонування автоматизованих систем.

# ВИСНОВКИ

В ході виконання кваліфікаційної роботи було проаналізовано сучасні методи застосування машинного навчання в системах автоматизації виробничих процесів, зокрема для виявлення, класифікації та сортування об’єктів, проведено тренування двох моделей YOLOv8 з врахуванням розглянутих прикладів моделей машинного навчання, адаптованих для задач розпізнавання об’єктів малих габаритів у реальному часі, розроблено та реалізовано програмне забезпечення комп’ютерного зору, яке забезпечує точне виявлення об’єктів і їхню передачу до системи керування робота-маніпулятора.

В результаті роботи було розроблено інтелектуальну систему класифікації виробів з використанням технології машинного навчання, яка забезпечує автоматичне виявлення, ідентифікацію та передачу координат об’єктів до системи керування робота-маніпулятора, а також забезпечує прийняття рішень щодо подальших дій з виявленими об’єктами в режимі реального часу.

Модель машинного навчання, отримана в результаті проведеного навчання, досягла значення 0,96 показника метрики F1, середня точність mAP@50 склала 0,985, тоді як середня точність у діапазоні IoU від 0,5 до 0,95 (mAP@50-95) склала 0,81. Отримані результати свідчать про високу ефективність виявлення та класифікації об’єктів, а також про придатність моделі для застосування в умовах роботи системи в реальному часі.

В ході виконання поставлених задач кваліфікаційної роботи було виконано наступні завдання:

– проведено аналіз літератури за темою кваліфікаційної роботи, визначено сучасні підходи до використання комп’ютерного зору на основі машинного навчання, переваги використання машинного навчання в контексті Цілей сталого розвитку;

– визначено вимоги до апаратної частини системи, моделі машинного навчання з врахуванням специфіки виробничого процесу, зокрема необхідності підтримки стабільної роботи системи в умовах реального часу;

– розглянуто та проаналізовано існуючі рішення моделей машинного навчання для виявлення об’єктів малих габаритів, а також моделі з більш узагальненими умовами використання задля досягнення оптимальних результатів в ході тренування моделі машинного навчання під потреби проєкту;

– підібрано збалансований навчальний та валідаційний датасети, на основі яких було навчено дві моделі YOLOv8s з різним розміром використаних датасетів;

– розроблено програмне забезпечення з використанням мови програмування C#, нейромережі YOLOv8 та бібліотеки EmguCV, обгортки для бібліотеки обробки зображень OpenCV.

Досягнені результати роботи можуть бути використані в навчальному процесі в університеті як практичний приклад застосування методів комп’ютерного зору та машинного навчання в задачах автоматизації виробничих процесів.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Machine learning in manufacturing and industry 4.0 applications / R. Rai et al. *International Journal of Production Research*. 2021. Vol. 59, no. 16. P. 4773–4778. URL: https://doi.org/10.1080/00207543.2021.1956675 (дата звернення: 14.03.2025).

2. Goal 12 | Department of Economic and Social Affairs. | *Sustainable Development*. URL: https://sdgs.un.org/goals/goal12 (дата звернення: 14.03.2025).

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 програми «Автоматизація та комп'ютерно-інтегровані технології» / Упоряд.: І.Ш. Невлюдов, О.І. Филипенко, О.В. Токарєва, С.П. Новоселов, О.В Сичова. – Харків: ХНУРЕ, 2023. – 64 с.

4. ДСТУ 3008: 2015. Інформація та документація. Звіти у сфері науки і техніки. Структура і правила оформлення. К.: ДП «УкрНДНЦ». 2016. 30 с.

5. Driha S. Automated Waste Classification for Efficient Recycling Using Machine Learning. «Automation and Development of Electronic Devices» ADED-2025 [Електронний ресурс] : збірник студентських наукових статей – Харків : ХНУРЕ, 2025. – Вип. 1. С.51-55.

6. Невлюдов І.Ш., Євсєєв В.В., Андрусевич А.О., Максимова С.С. Моделі та методи кіберфізичних виробничих систем в концепції Industry 4.0 Монографія. – Харків: 2023. – 321 с. (с.17)

7. The Potential of Additive Manufacturing in the Smart Factory Industrial 4.0: A Review / M. Mehrpouya et al. *Applied Sciences*. 2019. Vol. 9, no. 18. P. 3865. URL: https://doi.org/10.3390/app9183865 (дата звернення: 16.03.2025).

8. Machine learning in manufacturing and industry 4.0 applications / R. Rai et al. *International Journal of Production Research*. 2021. Vol. 59, no. 16. P. 4773–4778. URL: https://doi.org/10.1080/00207543.2021.1956675 (дата звернення: 16.03.2025).

9. Applying machine learning approach in recycling / M. Erkinay Ozdemir et al. *Journal of Material Cycles and Waste Management*. 2021. Vol. 23, no. 3. P. 855–871. URL: https://doi.org/10.1007/s10163-021-01182-y (дата звернення: 19.03.2025).

10. 17 Цілей сталого розвитку | Global Compact. *Global Compact*. URL: https://globalcompact.org.ua/tsili-stijkogo-rozvytku/ (дата звернення: 19.03.2025).

11. Ramirez R. The plastic water bottle industry is booming. Here's why that's a huge problem | CNN. *CNN*. URL: https://edition.cnn.com/2023/03/16/world/plastic-water-bottles-un-report-climate/index.html (дата звернення: 24.03.2025).

12. Plastic Bottle Cap Recycling–Characterization of Recyclate Composition and Opportunities for Design for Circularity / M. Gall et al. *Sustainability*. 2020. Vol. 12, no. 24. P. 10378. URL: https://doi.org/10.3390/su122410378 (дата звернення: 24.03.2025).

13. Small Metal Objects Classification Based on The Deep Learning Approach / Nur Safariah Inani M. Tahir et al. *Journal of Engineering Research and Education (JERE)*. 2025. Vol. 16. P. 46–56. URL: https://doi.org/10.58915/jere.v16.2024.1661 (дата звернення: 25.03.2025)

14. Giel R., Fiedeń M., Dąbrowska A. Real-Time Automatic Identification of Plastic Waste Streams for Advanced Waste Sorting Systems. *Sustainability*. 2025. Vol. 17, no. 5. P. 2157. URL: https://doi.org/10.3390/su17052157 (дата звернення: 25.03.2025).

15. Комплекс навчально-методичного забезпечення навчальної дисципліни «Технологія розробки програмного забезпечення комп'ютерно-інтегрованих систем» підготовки бакалавра спеціальності 151 «Автоматики і комп'ютеризованих технологій», освітня програма «Автоматики і комп'ютеризованих технологій» [Електронний ресурс] / ХНУРЕ ; розроб. С.П. Новоселов, О.В. Сичова. – Харків, 2022. – 257 с. URL: http://catalogue.nure.ua/knmz

16. Конспект лекцій з дисципліни «Основи комп’ютерно-інтегрованого управління» для студентів спеціальності 151 «Автоматизація та комп’ютерно-інтегровані технології» освітньої програми «Автоматизація та комп’ютерно-інтегровані технології» [Електронне видання] / Упоряд.: А.І. Бронніков. − Харків: ХНУРЕ, 2023. − 358 с.

17. GitHub - emgucv/emgucv: Emgu CV is a cross platform .Net wrapper to the OpenCV image processing library. *GitHub*. URL: https://github.com/emgucv/emgucv (дата звернення: 6.04.2025).

18. YOLO Object Detection Explained: Evolution, Algorithm, and Applications. *Encord | Label & Curate Multimodal Data for AI*. URL: https://encord.com/blog/yolo-object-detection-guide/ (дата звернення: 8.04.2025).

19. Невлюдов І. Ш., Андрусевич А. О., Євсєєв В. В., Новоселов С. П., Демська Н. П. Проектування мобільних маніпуляційних роботів: Монографія. – Х. :, 2022. – 427 с.

20. Terven J., Córdova-Esparza D.-M., Romero-González J.-A. A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Machine Learning and Knowledge Extraction*. 2023. Vol. 5, no. 4. P. 1680–1716. URL: https://doi.org/10.3390/make5040083 (дата звернення: 8.04.2025).

21. Nevliudov I., Novoselov S., Sychova O., Mospan D. Multithreaded Software Control of Industrial Manipulator Movement. IEEE 4th International Conference on Modern Electrical and Energy System (MEES), Kremenchuk, Ukraine, 2022, pp. 1-6, doi: 10.1109/MEES58014.2022.10005675 (дата звернення: 10.04.2025).

22. Методичні вказівки до лабораторних робіт з дисципліни «Теорія автоматичного управління» для студентів усіх форм навчання спеціальності 151 «Автоматизація та комп’ютерно-інтегровані технології» освітня програма Автоматизація та комп’ютерно-інтегровані технології [Електронне видання] / Упоряд.: О.В. Токарєва.– Харків: ХНУРЕ, 2022. – 86 с.

23. F1 Score in Machine Learning: Intro & Calculation. *V7 | AI Document Processing & Data Labelling*. URL: https://www.v7labs.com/blog/f1-score-guide (дата звернення: 10.04.2025).

24. Закон Уряїни «Про охорону праці». Офіційний вебпортал парламенту України. Режим доступу: https://zakon.rada.gov.ua/laws/show/2694-12#Text (дата звернення: 15.04.2025).

25. ДСТУ EN ISO 10218-2:2018 Роботи та роботизовані пристрої. Вимоги щодо безпечності промислових роботів. Частина 2. Роботизовані системи та їхні поєднання (EN ISO 10218-2:2011, IDT; ISO 10218-2:2011, IDT). *БУДСТАНДАРТ Online - нормативні документи будівельної галузі України*. URL: https://online.budstandart.com/ua/catalog/doc-page.html?id_doc=81746 (дата звернення: 15.04.2025).