Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Навчально-науковий
інститут електроенергетики
(навчально-науковий інститут)
Факультет інформаційних технологій
(факультет)
Кафедра інформаційних технологій та комп'ютерної інженерії
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня магістра

Здобувача вищої освіти _____Торгольського Андрія Олександровича_____
(ПІБ)
академічної групи _____123м-24-1_____
(шифр)
спеціальності _____123 Комп'ютерна інженерія_____
(код і назва спеціальності)
за освітньо-професійною програмою ___«Комп'ютерна інженерія»_____
(офіційна назва)

на тему «Обґрунтування структури та параметрів кіберфізичної системи віддаленого керування маніпулятором з використанням технологій комп'ютерного зору»
(назва за наказом ректора)

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	доц. Бешта Д.О.			
розділів:				
синтез системи	доц. Бешта Д.О.			
розроблення програмного забезпечення	ас. Панферова Я.В.			
Рецензент				
Нормоконтролер	проф. Цвіркун Л.І.			

Дніпро
2025

ЗАТВЕРДЖЕНО:
завідувач кафедри
інформаційних технологій
та комп'ютерної інженерії
(повна назва)
_____ _В.В. Гнатушенко
(підпис) (ініціали, прізвище)

«_____» _____ 2025 року

ЗАВДАННЯ
на кваліфікаційну роботу
ступеня магістра
(бакалавра, магістра)

здобувача вищої освіти ___Торгольського А.О.__ **академічної групи** ____123м-24-1____
(прізвище та ініціали) (шифр)

спеціальності _____123 Комп'ютерна інженерія_____

за освітньою-професійною програмою _____«Комп'ютерна інженерія»_____
(офіційна назва)

на тему «Обґрунтування структури та параметрів кіберфізичної системи віддаленого керування маніпулятором з використанням технологій комп'ютерного зору»,
затверджену наказом ректора НТУ «Дніпровська політехніка» від 13 жовтня 2025 р. №1165-с

Розділ	Зміст	Термін виконання
Стан питання та постановка завдання	На основі матеріалів практик, інших науково-технічних джерел сформулювати наукове завдання, конкретизувати предмет та мету досліджень	12.10.2025
Теоретичний	Обґрунтування теоретичної базу побудови КФС з віддаленим керуванням та обґрунтування вибору програмних та апаратних компонентів системи	28.10.2025
Синтез системи	Розробка архітектури кіберфізичної системи з підсистемами відеозахоплення, обробки та керування	14.11.2025
Розроблення програмного забезпечення	Розробка програмного забезпечення плати керування, бездротового модулю та мобільного застосунку з модулем комп'ютерного зору для розпізнавання об'єктів	29.11.2025
Експериментальний розділ	Проведення і обробка результатів експериментів з ефективності системи розпізнавання об'єктів в різних умовах рівня освітленості	07.12.2025

Завдання видано _____ доц. Д. О. Бешта
(підпис керівника) (ініціали, прізвище)

Дата видачі 05 вересня 2025 р.

Дата подання до екзаменаційної комісії 10.12.2025 р.

Прийнято до виконання _____ Торгольський А.О.
(підпис здобувача вищої освіти) (ініціали, прізвище)

РЕФЕРАТ

Пояснювальна записка 96 с., 38 рис., 10 табл., 1 дод., 14 джерел.

КІБЕРФІЗИЧНА СИСТЕМА, МАНІПУЛЯТОР, ESP32-CAM, МОБІЛЬНИЙ ДОДАТОК, НЕЙРОННІ МЕРЕЖІ, РОЗПІЗНАВАННЯ ОБ’ЄКТІВ.

Об’єкт розробки – кіберфізична система віддаленого керування маніпулятором, що включає апаратний модуль захоплення відеопотоку, мобільним застосунком та розпізнаванням об’єктів у режимі реального часу.

Мета роботи – розробка та дослідження кіберфізичної системи віддаленого керування маніпулятором, що включає апаратний модуль захоплення відеопотоку, мобільним застосунком та алгоритмами комп'ютерного зору для автоматизації процесів розпізнавання та обробки об'єктів у режимі реального часу.

Методи дослідження – для досягнення поставленої мети використовувались алгоритми згорткової нейронної мережі для розпізнавання об’єктів, експериментальне тестування працездатності системи, інструменти мобільної розробки та протоколи бездротової передачі даних.

У роботі виконаний аналіз існуючих платформ для побудови кіберфізичних систем, які підтримують передачу відеопотоку з можливістю інтеграції алгоритмів розпізнавання об’єктів. Обґрунтований вибір моделі нейронної мережі, сформульовано завдання дослідження.

У розділі «Синтез системи» сформовані технічні вимоги до створюваної системи, побудована структурна схема кіберфізичної системи.

У розділі «Розроблення програмного забезпечення» виконано проєктування та навчання нейронної мережі для розпізнавання об’єктів, реалізовано віджет камери у мобільному додатку.

В експериментальному розділі поставлена задача експерименту і проведено тестування розробленої системи з оцінкою якості передачі відеопотоку та точністю розпізнавання об’єктів.

Практична цінність отриманих результатів полягає у створені практичної моделі, яка може бути використана, наприклад, у навчальних цілях.

ЗМІСТ

Перелік скорочень, умовних познак, одиниць і термінів 7
Вступ 8
1 Стан питання та постановка завдання 10
1.1 Стан питання 10
1.2 Інтеграція комп'ютерного зору в кіберфізичні системи 12
1.2.1 Існуючі комерційні системи 13
1.3 Кінематична модель маніпулятору 15
1.3.1 Типи маніпуляторів за геометрією робочої зони 15
1.4 Аналіз існуючих апаратних платформ 18
1.4.1 Мікроконтролерні платформи для захоплення відеопотоку 18
1.4.2 Платформи для обробки візуальної інформації 20
1.5 Аналіз методів та алгоритмів нейронних мереж 23
1.5.1 Двоетапні детектори об'єктів 24
1.5.2 Одноетапні детектори об'єктів 26
1.6 Проблеми сучасних систем 27
1.7 Постановка завдання дослідження 28
2 Теоретичний розділ 30
2.1 Загальна характеристика об'єкта дослідження 30
2.2 Структура об'єкта дослідження 31
2.3 Математична модель руху маніпулятора 32
2.3.1 Рівняння прямої задачі кінематики 32
2.3.2 Рівняння зворотної задачі кінематики 34
2.4 Обґрунтування і вибір методів синетзу системи 36
2.4.1 Метод порівняльного аналізу для вибору апаратних компонентів 36
2.4.2 Метод порівняльного аналізу для вибору програмних компонентів 36
2.4.3 Метод математичного моделювання руху маніпулятору 37
2.5 Обґрунтування і вибір методів експериментальних досліджень 37
2.5.1 Метод вимірювання затримок 37
2.5.2 Метод оцінки точності розпізнавання об'єктів 38
2.6 Висновки до розділу 37

3 Синтез системи 40
3.1 Цілі впровадження системи 40
3.2 Формування технічних вимог до системи 40
3.2.1 Вимоги до реалізації системи 41
3.2.2 Вимоги до показників призначення 41
3.2.3 Вимоги до функцій (задач), виконуваним системою 42
3.2.4 Вимоги до видів забезпечення 43
3.2.4.1 Вимоги до математичного забезпечення 43
3.2.4.2 Вимоги до інформаційного забезпечення 43
3.2.4.3 Вимоги до лінгвістичного забезпечення 44
3.2.4.4 Вимоги до технічного забезпечення 44
3.3 Синтез структурної схеми системи 45
3.4 Вибір та обґрунтування застосування апаратних засобів 46
3.5 Синтез схеми підключення апаратних засобів 48
3.6 Вибір та обґрунтування застосування програмних засобів 49
3.7 Висновки до розділу 50
4 Розробка програмного забезпечення системи 52
4.1 Призначення та область застосування ПЗ 52
4.2 Обґрунтування технічних характеристик програми 52
4.2.1 Постановка завдання на розробку програми 52
4.2.2 Опис алгоритму та функціонування програми 54
4.2.3 Опис та обґрунтування вибору методів організації вхідних та вихідних даних 56
4.3 Опис розробленої програми 58
4.3.1 ПЗ плати керування Arduino Uno 58
4.3.1.1 Загальні відомості 58
4.3.1.2 Функціональне призначення 59
4.3.1.3 Опис логічної структури 59
4.3.1.4 Використовувані технічні засоби 59
4.3.1.5 Виклик і завантаження програми 60
4.3.1.6 Вхідні та вихідні дані 59
4.3.2 ПЗ бездротового модулю ESP32-CAM 60
4.3.2.1 Загальні відомості 60
4.3.2.2 Функціональне призначення 60

4.3.2.3 Опис логічної структури 61
4.3.2.4 Використовувані технічні засоби 62
4.3.2.5 Виклик і завантаження програми 62
4.3.2.6 Вхідні та вихідні дані 62
4.3.3 ПЗ мобільного застосунку 63
4.3.3.1 Загальні відомості 63
4.3.3.2 Функціональне призначення 63
4.3.3.3 Опис логічної структури 64
4.3.3.4 Використовувані технічні засоби 64
4.3.3.5 Виклик і завантаження програми 64
4.3.3.6 Діаграма класів 65
4.4 Вікна мобільного застосунку 65
4.5 Очікувані техніко-економічні показники 68
4.6 Висновки до розділу 69
5 Експериментальний розділ 70
5.1 Мета і завдання експерименту 70
5.2 Методика проведення експерименту 70
5.3 Вимоги до експерименту 72
5.3.1 Вимоги до вимірювальної апаратури 72
5.3.2 Вимоги до об'єктів розпізнавання 72
5.3.3 Вимоги до експериментального стенду 72
5.3.4 Вимоги до фіксації результатів дослідження 73
5.3.5 Критерії оцінювання результатів дослідження 73
5.4 Результати експерименту 74
5.4.1 Сутність експерименту 74
5.4.2 Результат експерименту в цифрах і фактах 79
5.4.3 Аналіз відповідності теоретичних та експериментальних досліджень 82
5.4.4 Характеристика новизни результатів 82
5.5 Висновки до розділу 83
Висновки 84
Перелік посилань 86
Додаток А. Текст програми мобільного застосунку 86

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАК, ОДИНИЦЬ І ТЕРМІНІВ

КФС – кіберфізична система.

КЗ – комп’ютерний зір.

НМ – нейронна мережа.

ПЗ – програмне забезпечення.

API (Application Programming Interface) – набір правил взаємодії двох систем.

APK (Android Package Kit) – формат архівних файлів-додатків для ОС Android.

COCO (Common Objects in Context) – набір даних для навчання нейронних мереж.

CNN (Convolutional Neural Network) – згорткова нейронна мережа.

HTTP (HyperText Transfer Protocol) – протокол передачі гіпертексту.

PWM (Pulse Width Modulation) – широтно-імпульсна модуляція.

RoI (Region of Interest) – область інтересу на зображені.

R-CNN (Region-based Convolutional Neural Network) – регіональна згорткова нейронна мережа.

mAP (Mean Average Precision) – середня точність розпізнавання.

SDK (Software Development Kit) – набір засобів для розробки програмного забезпечення.

YOLO (You Only Look Once) – алгоритм для розпізнавання об'єкті.

ВСТУП

Маніпулятори знаходять широке коло застосування у різних сферах людської діяльності від логістичної, промислової та закінчуючи медичною чи освітньою. Виконуючи операції різного рівня складності, вони забезпечують точність та стабільність виконання процесів.

Поточний стан у розвитку кіберфізичних систем (КФС) демонструє тенденції до інтеграції маніпуляторів із технологіями комп’ютерного зору (КЗ) чи штучного інтелекту. Використання таких взаємоінтегрованих систем відкриває нові можливості для ефективного ведення підприємницької діяльності. Впровадженням подібних інтеграційних рішень активно займаються компанії KUKA, FANUC, ABB, а також українські підприємства, зокрема компанія «Самміт», м. Дніпро, яка пропонує роботизовані рішення для оптимізації зварювальних процесів.

Не зважаючи на значний розвиток кіберфізичних систем з використанням комп’ютерного зору існують проблеми, вирішення яких потребує уваги.

Мета і завдання дослідження. *Метою роботи* розробка кіберфізичної системи віддаленого керування маніпулятором з комп'ютерним зором для розпізнавання об'єктів у режимі реального часу.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

– провести комплексний аналіз сучасного стану розвитку кіберфізичних систем та методів інтеграції із системами комп’ютерного зору;

– дослідити апаратні та програмні платформи, придатні для створення системи керування маніпулятором із передачею відеопотоку.;

– розробити архітектуру кіберфізичної системи з урахуванням можливості передачі відеопотоку з мінімальними затримками та можливістю розпізнавання об’єктів;

– розробити методи та алгоритми обробки відеопотоку для розпізнавання об’єктів у режимі реального часу з урахуванням обмеженості обчислювальних ресурсів;

– створити програмне забезпечення для керування маніпулятором;

– провести експериментальні дослідження ефективності системи розпізнавання об’єктів.

Об’єкт дослідження – кіберфізична система віддаленого керування маніпулятором з алгоритмом комп’ютерного зору.

Предмет дослідження – параметри та структура кіберфізичної системи віддаленого керування маніпулятором з алгоритмами комп'ютерного зору.

Методи дослідження. Для досягнення поставленої мети використовувались методи порівняльного аналізу апаратних і програмних платформ, математичного моделювання кінематики маніпулятора та методи експериментального тестування точності розпізнавання об'єктів в режимі реального часу.

Наукові положення:

1. Встановлено, що при рівні освітленості робочої зони нижче значення 300-350 люкс, точність розпізнавання об’єктів нейронною мережею EfficientDet-Lite0 знижується до показників значення F1-score до 33-40%.

2. Розпізнавання об’єктів у системі забезпечується шляхом інтеграції нейронної мережі EfficientDet-Lite0 в мобільний застосунок через фреймворк MLKit без використання додаткових обчислювальних потужностей.

Наукові результати:

1. Встановлено залежність точності розпізнавання об’єктів нейронною мережею EfficientDet-Lite0 від рівня освітленості робочої зони, що дозволяє визначити мінімальні вимоги до освітленості та стабільності роботи системи.

Обґрунтованість і достовірність наукових положень, висновків і рекомендацій підтверджуються тим, що в роботі використані: апробовані методи математичного моделювання кінематики руху маніпулятору, стандартні метрики оцінки якості розпізнавання (Precision, Recall, F1-score), експериментальні підтвердження результатів теоретичних досліджень на реальному обладнанні.

Практичне значення отриманих результатів полягає в розробці економічно ефективної кіберфізичної системи віддаленого керування маніпулятором з інтегрованим комп'ютерним зором, яка може бути використана у навчальних лабораторіях закладів вищої освіти.

1 СТАН ПИТАННЯ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Стан питання

Впровадження кіберфізичних систем (КФС) на сьогоднішній день демонструє стабільну тенденцію до зростання, ставши одним із основних факторів ефективного розвитку сфер людської діяльності, таких як, логістика, медицина, освіта та інші. Підтвердженням цієї тенденції є діаграма динаміка глобального впровадження роботизованих КФС у логістиці, яка демонструє стабільний розвиток у роки з 2015 по 2025, що наведена на рисунку 1.1 [1].

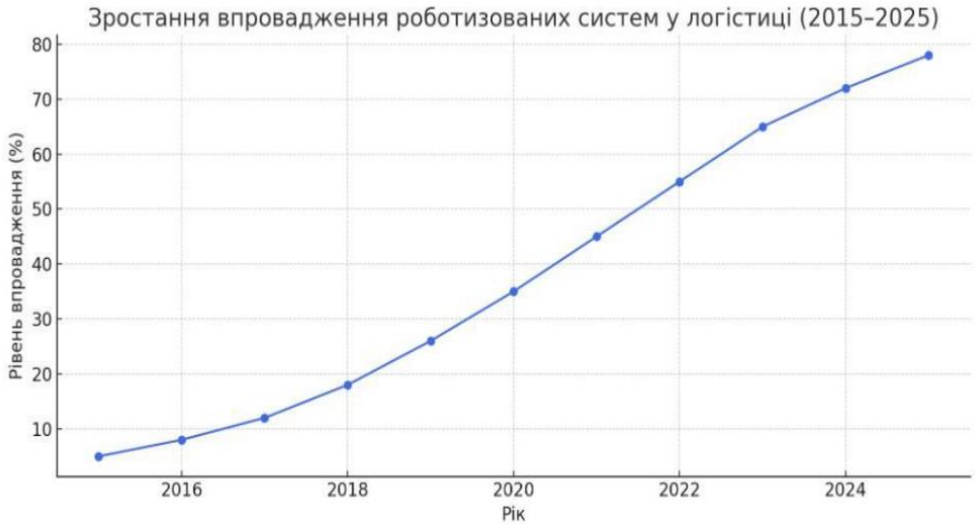


Рисунок 1.1 – Графік зростання впровадження роботизованих систем у логістиці

Роботизовані КФС призначені для автоматизації виконання повторюваних та рутинних процесів. Застосування їх забезпечує суттєве скорочення витрат на оплату праці, в довгостроковій перспективі, та підвищення показників ефективності та якості виконання процесів [2]. На рисунку 1.2 продемонстрована діаграма порівняльного аналізу ефективності виконання операцій людиною та автоматизованою системою, з виокремленням основних показників продуктивності, а саме: кількість виконаних операцій за годину, час на виконання однієї операції в секундах, частота виникнення помилки за 1000 операцій та залученість персоналу при виконанні операції.

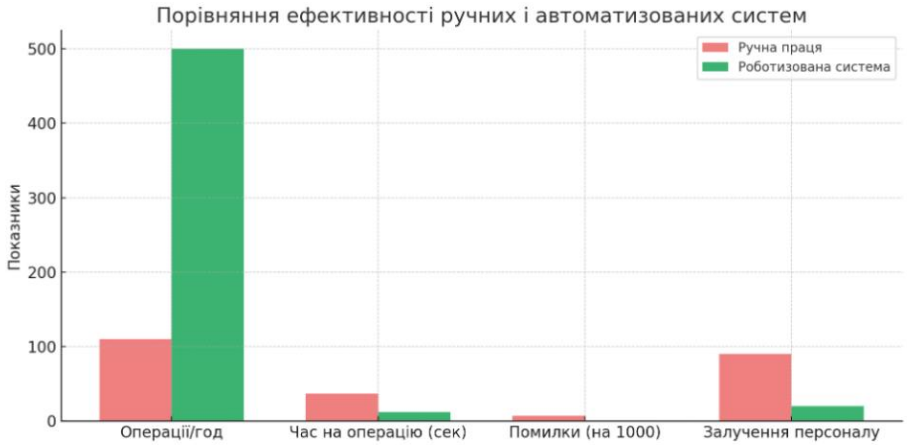


Рисунок 1.2 – Порівняльна діаграма ефективності ручних та автоматизованих систем

Згідно зі статистикою AIPRM (2024), глобальний ринок робототехніки стрімко зростає, перевищивши 45 млрд доларів у 2024 році та прогнозуючи збільшення до понад 73 млрд доларів до 2030 року, див. рисунок 1.3 [3].

В порівнянні зі світовими показниками, український ринок сервісної роботехніки демонструє поступове зростання: за прогнозами, дохід цього сегменту у 2025 році може становити близько 32,37 млн доларів США.

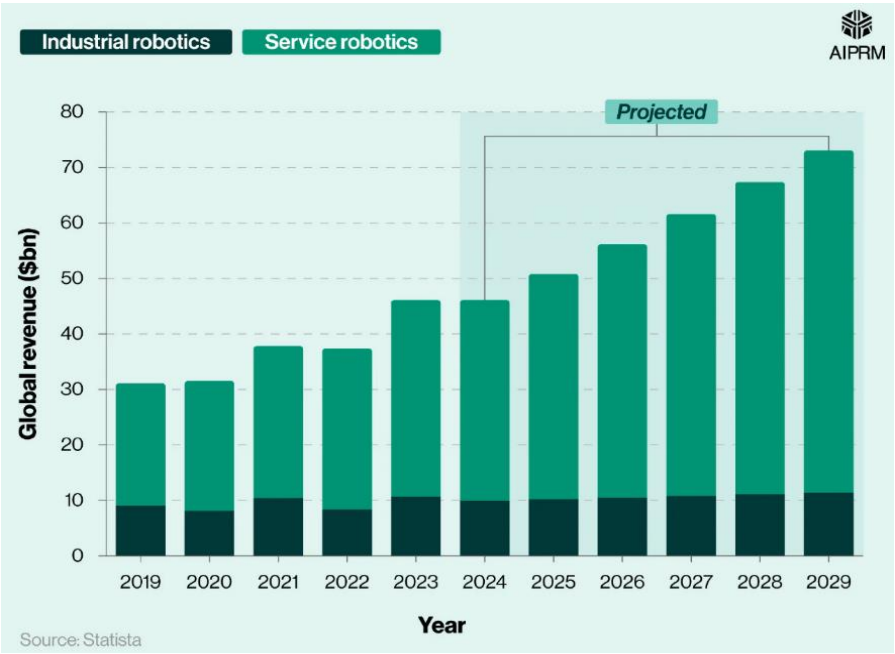


Рисунок 1.3 – Діаграма глобального доходу від промислової та сервісної робототехніки

1.2 Інтеграція комп’ютерного зору в кіберфізичні системи

Інтеграція КФС із технологіями комп’ютерного зору (КЗ) та штучного інтелекту (ШІ) надає додаткову перевагу під час виконання процесів. Завдяки такій взаємній інтеграції ці системи здатні не лише виконувати наперед запрограмовані операції, але й адаптуватися до змін умов навколишнього середовища, ідентифікувати об’єкти, оптимізувати траєкторії руху та приймати рішення в реальному часі [4].

Типова архітектура таких систем складається з наступних підсистем:

− підсистема захоплення візуальної інформації (модулі камер, датчики глибини), що безпосередньо отримують зображення навколишнього середовища;

− підсистема обробки та аналізу візуальної інформації (апаратна платформа для виконання алгоритмів комп’ютерного зору), що приймає рішення на основі отриманої інформації;

− підсистема керування актуаторами (серводвигуни, захоплювачі), що фізично взаємодіють з об’єктам на основі обробленої інформації.

Важливим у побудові систем таких систем є правильний розподіл обчислювальних потужностей. Апаратні платформи з інтегрованими камерними модулю дозволяють захоплювати та попередньо оброблювати відеопотік, але зазвичай мають обмежені ресурси для виконання складних алгоритмів КЗ, тоді як розпізнаванням об’єктів, плануванням траєкторій руху, прийняттям рішень займаються більш потужні обчислювальні пристрої (від edge-пристроїв до хмарних серверів).

В залежності від вимог до системи виокремлюють три основні підходи для реалізації:

− апаратний підхід, головною метою якого є використання спеціальних обчислювальних модулів, таких як GPU, TPU, FPGA з метою прискорення обробки візуальних даних;

− програмний підхід, який зосереджується на оптимізації методів обробки, замість концентрації на потужності обчислювального обладнання;

− гібридний підхід, в свою чергу є поєднанням використання спеціальних обчислювальних модулів та оптимізований програмний комплекс.

1.2.1 Існуючі комерційні системи

KUKA.VisionTech є однією з поширених систем інтеграції комп'ютерного зору в промислові роботизовані маніпуляторні системи, див. рисунок 1.4. Програмного технічні рішення підтримують використання як камер 2D, так і 3D, дозволяючи охоплювати розпізнавання великої кількість об’єктів, з визначенням їх орієнтації та позиціонування в просторі. Більше 200 видів різних об’єктів може бути розпізнано за одиницю часу. Вартість базової конфігурації починається від 50 тис доларів США, що включає програмне забезпечення та апаратне обладнання.

Рисунок 1.4 – Маніпулятор KUKA з камерою для розпізнавання об’єктів

ABB Integrated Vision пропонує архітектуру, що дозволяє поєднувати різні види камер для розпізнавання, детекції країв та розпізнавання символів, див. рисунок 1.5. Особливістю наявного рішення є його взаємодія з програмною системою RAPID, яка надає можливості для створення власних бібліотек. Підтримується інтеграція з системами управління виробничими процесами для обміну даними про виробництво. Залежно від складності, час за який може відбуватися захоплення та розпізнавання об’єкту може коливатися в межах від 0.8 до 1.2 секунди. Вартість базової конфігурації починається від 25 тис доларів США, в залежності від типу маніпулятору, камери та програмного забезпечення.

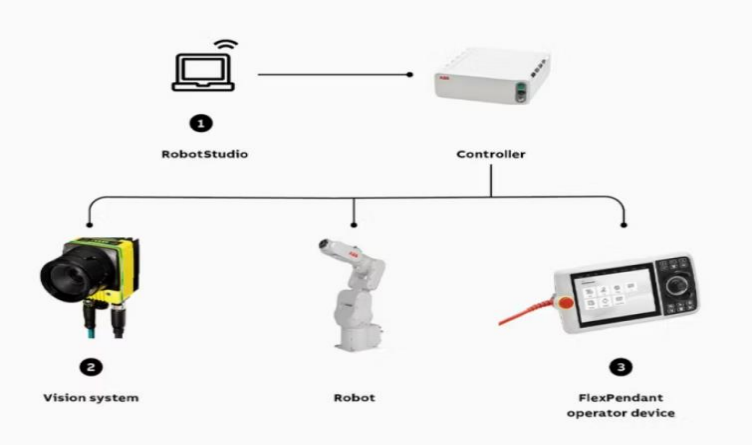


Рисунок 1.5 – Інтеграція КЗ від ABB

Більшість виробників промислових маніпуляторів пропонують мобільні рішення для моніторингу та базового керування:

KUKA smartPAD, див. рисунок 1.6 дозволяє відстежувати стан робота, переглядати діагностичну інформацію та виконувати прості операції керування. Відсутня інтеграція з системою комп'ютерного зору в реальному часі.

Рисунок 1.6 – KUKA smartPAD

Universal Robots PolyScope надає базові можливості керування через планшет, але обробка відеопотоку не інтегрована в мобільний застосунок (див. рисунок 1.7).

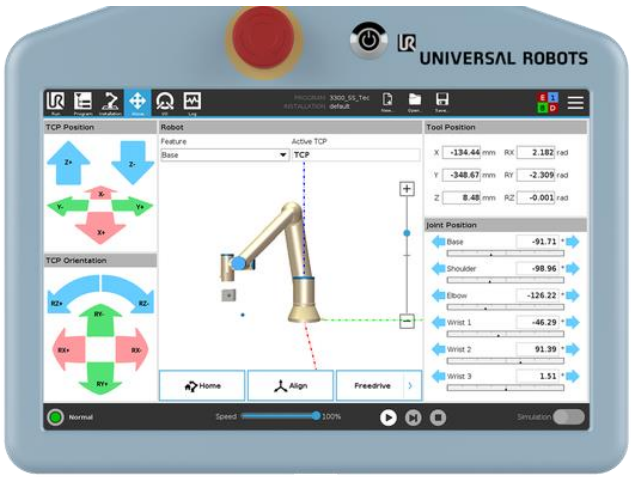


Рисунок 1.7 – Universal Robots PolyScope

1.3 Кінематична модель маніпулятору

Кінематична модель визначає взаємозв’язок між ланками маніпулятору та їх положенням, що дає змогу взаємодіяти з об’єктами. Визначення кінематичної моделі маніпулятору є фундаментальною етапом при побудові алгоритмів керування та розрахунку траєкторій руху [5].

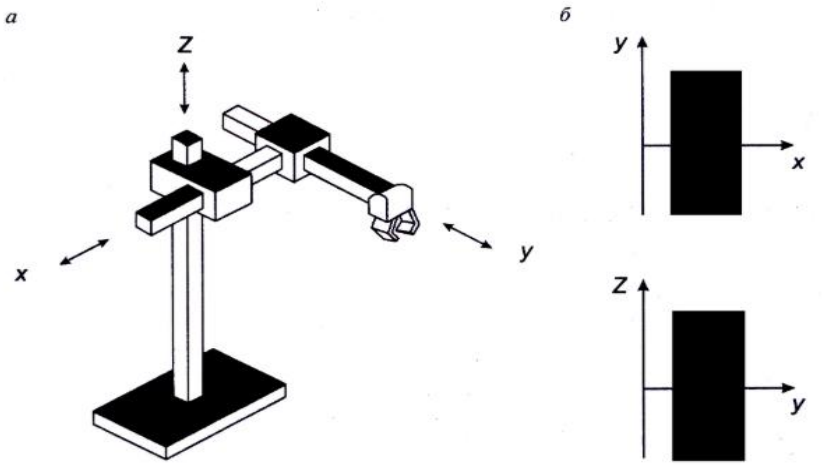
Ланки маніпулятору послідовно з’єднані та обертаються і переміщуються в просторі за допомогою серводвигунів. Кожен серводвигун забезпечує один ступінь рухливості системи. Мінімальна кількість ступенів рухливості, необхідна для позиціювання виконавчого органу в тривимірному просторі, становить три. Додавання додаткових ступенів рухливості підвищує гнучкість та точність маніпуляцій.

Робоча зона маніпулятору визначається як множина всіх точок простору, які може досягати виконавчий орган при зміні кутів у з’єднаннях в межах їх діапазонів. Форма робочої зони залежить від кінематичної моделі, розмірів ланок та обмежень кутів повороту серводвигунів.

1.3.1 Типи маніпуляторів за геометрією робочої зони

Класифікація маніпуляторів за геометрією робочої зони базується на типах з’єднань та їх розташування у просторі. До основних типів маніпуляторів відносять: маніпулятори декартового, циліндричного, сферичного та ангулярного.

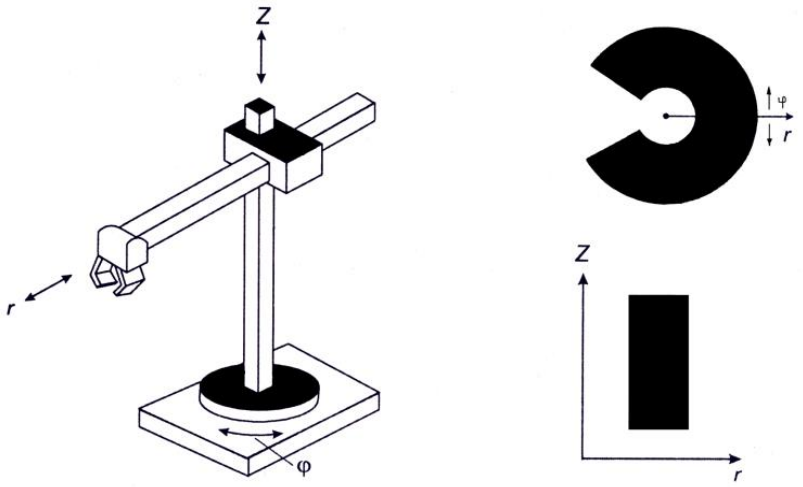
Маніпулятор декартового типу, функціонує в декартовій (прямокутній) системі координат (див. рисунок 1.8, а), робоча зона якого має форму паралелепіпеда (див. рисунок 1.8, б). Переміщення виконавчого органу відбувається вздовж трьох осей (x, y, z).



а – Маніпулятор декартового типу; б – робоча зона

Рисунок 1.8 – Маніпулятор декартового типу

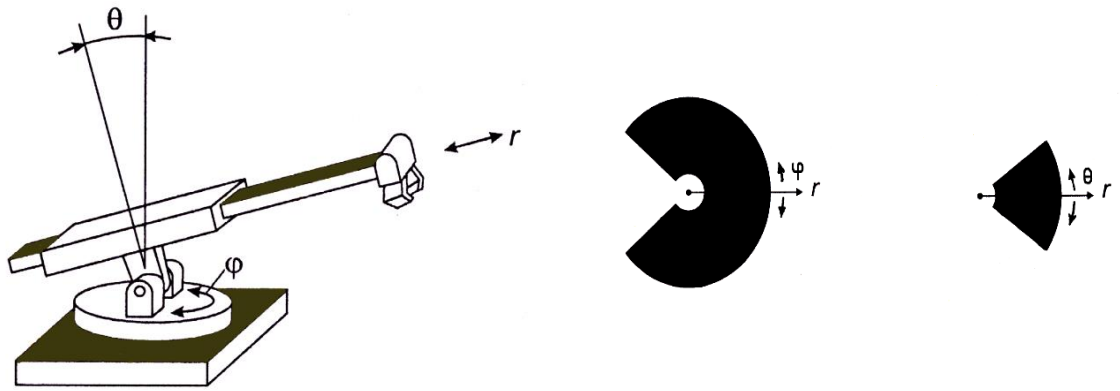
Маніпулятор циліндричного типу, функціонує в циліндричній системі координат (див. рисунок 1.9, а), робоча зона якого обмежена циліндричними поверхнями (див. рисунок 1.9, б). Переміщення виконавчого органу відбувається вгору, вниз по осі z, висуватись та втягуватись та може переміщуватись навколо осі підставки.



а – Маніпулятор циліндричного типу; б – робоча зона

Рисунок 1.9 – Маніпулятор циліндричного типу

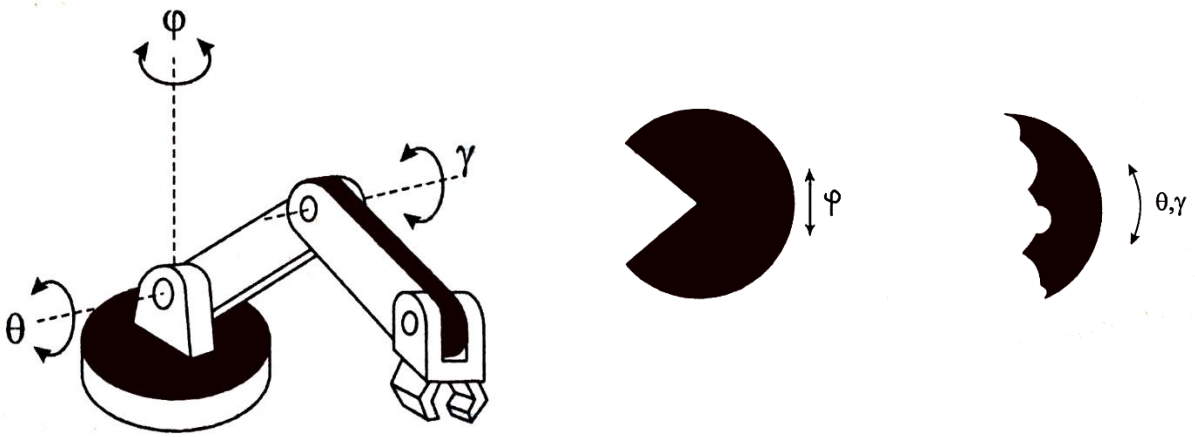
Маніпулятор сферичного типу, функціонує в сферичній (полярній) системі координат (див. рисунок 1.10, а), робоча зона якого обмежена сферичними поверхнями (див. рисунок 1.10, б). Переміщення виконавчого органу відбувається за рахунок обертання навколо вертикальної осі, зміни кута нахилу плеча, висування та втягування.



а – Маніпулятор сферичного типу; б – робоча зона

Рисунок 1.10 – Маніпулятор сферичного типу

Маніпулятор ангулярного типу, функціонує в ангулярній (сферичній) системі координат (див. рисунок 1.11, а), робоча зона якого має форму частини сфери складної конфігурації (див. рисунок 1.11, б). Переміщення виконавчого органу відбувається за рахунок послідовного обертання ланок у «суглобах», що подібно до руху людської руки.



а – Маніпулятор ангулярного типу; б – робоча зона

Рисунок 1.11 – Маніпулятор ангулярного типу

1.4 Аналіз існуючих апаратних платформ

КФС з технологіями КЗ передбачає апаратну платформу, яка поєднує в собі такі компоненти: модуль захоплення відеопотоку, обчислювальна платформа для обробки візуальних інформації та контролера актуаторів.

1.4.1 Мікроконтролерні платформи для захоплення відеопотоку

Велика кількість мікроконтролерних платформ підтримують інтеграцію камер через інтегровані або зовнішні модулі для захоплення відеопотоку [6]. Це дозволяє використовувати такі апаратні рішення у робототехніці або у технологіям розумних речей (IoT).

ESP32-CAM – це модуль, що поєднує плату розробки ESP32 та камеру OV2640 на 2 Мп (див. рис. 1.12). Модуль надає можливості створення HTTP/RSTP серверу для трансляції відеопотоку, підтримує операції оброки зображення, наприклад, стиснення JPEG.

Технічні характеристики ESP32-CAM:

− процесор: Dual-core Tensilica Xtensa LX6, 240 МГц;

− стандарт Wi-Fi: 802.11 b/g/n;

− камера: OV2640 з роздільною здатністю до 1600×1200 (UXGA);

− живлення: 5 В (через micro-USB або контакт +5V);

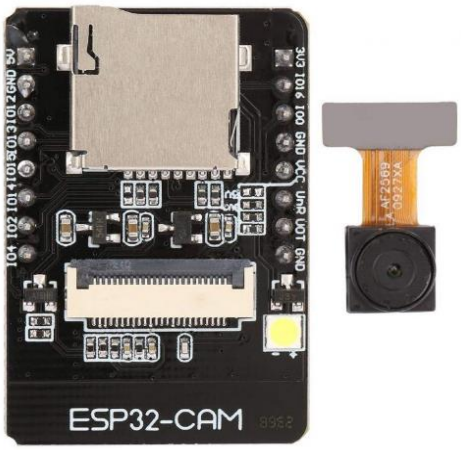


Рисунок 1.12 – Модуль ESP32-CAM

Raspberry Pi Zero W – одноплатний комп’ютер, з підтримкою підключення модуля камери через роз’єм CSI (див. рисунок 1.13). Наявність власної операційної системи (Raspberry Pi OS) дозволяє використовувати бібліотеки обробки зображень, дозволяє створювати RSTP/HTTP сервер для передачі відеопотоку.

Технічні характеристики Raspberry Pi Zero W:

− процесор: ARM11 (BCM2835), 1 ГГц;

− стандарт Wi-Fi: 802.11 b/g/n;

− камера: Raspberry Pi Camera Module v2 (8 Мп);

− живлення: 5 В (через micro-USB або контакт +5V);

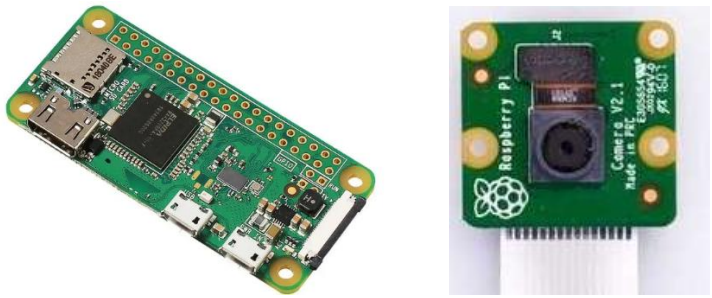


Рисунок 1.13 – Одноплатний комп’ютер Raspberry Pi Zero W та модуль камери Raspberry Pi Camera Module

Arduino Nano 33 BLE Sence – модуль, який поєднує мікроконтролерну плату розробки та сенсори, але без вбудованого модулю камери (див. рисунок 1.14). Має обмежений функціонал у роботі з візуальною інформацією, зазвичай використовується як шлюз для передачі окремих кадрів на зовнішній пристрій обробки.

Технічні характеристики Arduino Nano 33 BLE Sence:

− процесор: ARM Cortex-M4F (nRF52840), 64 MHz;

− стандарт Bluetooth: BLE 5.0;

− камера: ArduCAM Mini 2MP, OV7670 (2 МП);

− живлення: 3,3 В (через micro-USB або контакт +3,3V);

Рисунок 1.14 – Модуль Arduino Nano 33 BLE Sence та модулем камери ArduCAM Mini OV7670

За підсумками проведеного аналізу мікроконтролерних платформ з можливістю захоплення відеопотоку була сформована порівняльна таблиця 1.1.

Таблиця 1.1 – Порівняльна таблиця мікроконтролерних платформ з можливістю захоплення відеопотоку

Платформа	Характеристика						
	Процесор	ОЗП	Модуль камери	Роздільна здатність	Wi-Fi	Bluetooth	Вартість
1	2	3	4	5	6	7	8
ESP32-CAM	Tensilica Xtensa LX6 (240 МГц)	520 КБ	OV2640 (інтегрована)	До 1600×1200	Є (2.4 ГГц)	Є (4.2)	3-5$
Raspberry Pi Zero W	ARM 11 (1 ГГц)	512 МБ	Pi Camera v2	До 1920×1080	Є (2.4 ГГц)	Є (4.1)	30-35$
Arduino Nano 33 BLE Sence	ARM Cortex-M4F (64 МГц)	256 КБ	ArduCAM Mini (OV7670)	До 640×480	Відсутній	Є (BLE 5.0)	40-45$

1.4.2 Платформи для обробки візуальної інформації

Обробка візуальної інформації в реальному часі потребує значних обчислювальних ресурсів, тому необхідність достатньої процесорної потужності є однією з основних для виконання алгоритмів КЗ.

Оскільки за даними асоціації GSM (GSMA) за 2023 рік, кількість власників смартфонів дорівнювала 4,3 млрд, саме смартфони на сьогоднішній день є найбільш доступною обчислювальною технікою для виконання задач КЗ.

Технічні характеристики сучасних Android смартфонів:

− процесор: 8-ядерний ARM;

− GPU: Adreno 619/620 або Mali-G68;

− NPU/AI Engine: Hexagon DSP (Qualcomm) або APU (MediaTek);

− RAM: 8-12 Гб LPDDR4X/5;

− підтримка бібліотек: TensorFlow Lite, PyTorch Mobile;

Технічні характеристики сучасних iOS смартфонів:

− процесор: 6-ядерний Apple A14/A15 Bionic;

− GPU: 4-ядерна Apple GPU;

− Neural Engine: 16-ядерний;

− RAM: 6 Гб;

− підтримка бібліотек: TensorFlow Lite, Core ML, PyTorch Mobile;

Наявність продуктивних GPU/CPU/NPU, що оптимізовані для реалізації машинного навчання, наявність дисплею та батареї є значною перевагою для мобільних пристроїв.

NVIDIA Jetson Orin Nano – одноплатний комп’ютер, який оптимізований для виконання задач КЗ та ШІ на граничних пристроях (edge-пристроях), див. рисунок 1.15. Можливість підключення одразу декількох камер, підтримка бібліотек для глибоко навчання та достатня продуктивність при виконанні задач з розпізнавання об’єктів є перевагою цієї моделі.

Технічні характеристики NVIDIA Jetson Orin Nano:

− процесор: Quad-core ARM Cortex-A57 (1,43 ГГц);

− GPU: NVIDIA Maxwell;

− RAM: 2-4 Гб LPDDR4;

− підтримка бібліотек: TensorFlow, TensorRT, PyTorch;

Рисунок 1.15 – Одноплатний комп’ютер NVIDIA Jetson Orin Nano

Intel NUC – настільний комп’ютер (див. рисунок 1.16), який попри невеликі габаритні розміри забезпечує максимальні обчислювальні ресурси для виконання задач КЗ, серед розглянутих варіантів. Можливість встановлення дискретних графічних плат тільки збільшує його перевагу над варіантами, наведеними вище.

Технічні характеристики Intel NUC:

− процесор: Intel Core i3/i5/i7 (4-8 ядер, до 4.5 ГГц);

− GPU: Intel Iris Xe або дискретна;

− RAM: до 64 Гб DDR4;

− підтримка бібліотек: TensorFlow, TensorRT, PyTorch;

Рисунок 1.16 – Настільний комп’ютер Intel NUC

За підсумками проведеного аналізу платформ для обчислення візуальної інформації була сформована порівняльна таблиця 1.2.

Таблиця 1.2 – Порівняльна таблиця обчислювальних платформ

Платформа	Характеристика					
	Процесор	ОЗП	ШІ прискорювач	Мобільність	Дисплей	Вартість
1	2	3	4	5	6	7
Смартфон (Android)	8-ядер ARM (2.2-2.4 ГГц)	6-8 Гб	NPU (1-3 TOPS)	Висока	Вбудований	0-800 $
NVIDIA Jetson Orin Nano	4-ядра ARM (1.43 ГГц)	2-4 Гб	GPU (472 GFLOPS FP16)	Середня	Зовнішній (HDMI)	100-150 $
Intel NUC	4-8 ядер x86 (до 4.5 ГГц)	До 64 Гб	iGPU/dGPU + OpenVINO	Низька	Зовнішній (HDMI)	400-800 $

1.5 Аналіз методів та алгоритмів нейронних мереж

Алгоритми розпізнавання об’єктів поєднують дві основні технології – класифікацію об’єкта на зображені та визначення його місцезнаходження [7]. Перша технологія відповідає за визначення всіх об’єктів на зображені, тоді як друга всі знайдені об’єкти окреслює межами (bounding boxes).

Кожен об’єкт характеризується набором візуальних ознаки, за якими його можна в подальшому класифікувати, до прикладу, квадрати мають рівні сторони та перпендикулярні кути. Алгоритми розпізнавання об’єктів використовують ці ознаки, щоб відрізняти об’єкти одне від одного.

Алгоритми нейронних мереж розпізнавання об’єктів поділяються на два види: двоетапні та одноетапні детектори (див. рисунок 1.17). Ефективність реалізація системи розпізнавання об’єктів значною мірою залежить від вибору відповідного алгоритму нейронної мережі, що відповідає вимогам розроблюваної системи.

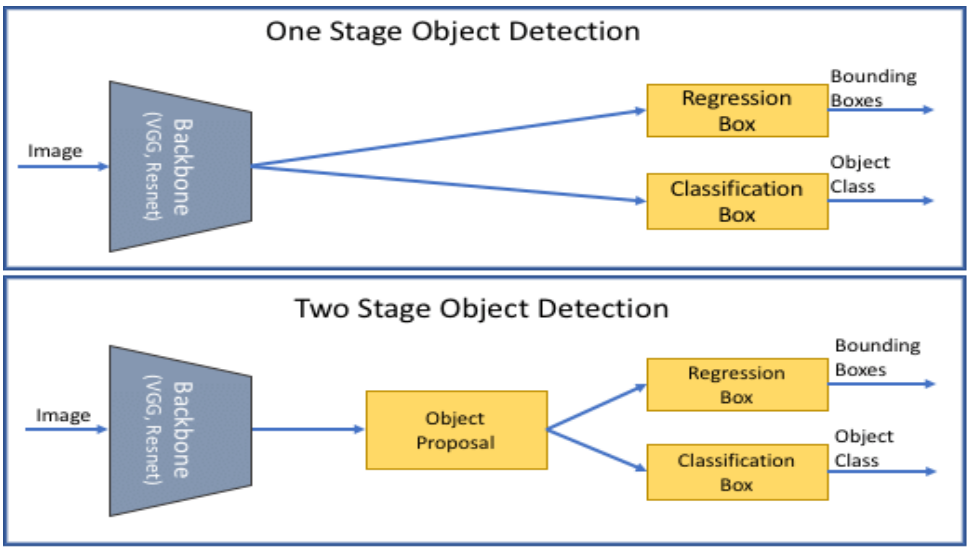


Рисунок 1.17 – Одноетапні та двоетапні детектори

1.5.1 Двоетапні детектори об’єктів

Двоетапні детектори об’єктів працюють в два послідовні процеси, забезпечуючи високу точність ідентифікації об’єкту, проте, вимагають більше обчислювальних ресурсів.

На першому етапі відбувається генерація потенційних регіонів (Region Proposal). Вхідне зображення аналізується і визначаються набір потенційних регіонів (RoI, regions of interest), де з певною ймовірністю можуть знаходитись об’єкти.

На другому етапі відбувається класифікація та визначення меж об’єктів. Кожен з потенційних регіонів вирівнюється та оброблюється згортковою нейронною мережею для точного визначення меж об’єктів та їх класифікацію.

R-CNN, запропонована у 2014 році, стала однією з перших де використовувались глибокі згорткові мережі для розпізнавання об’єктів (див. рисунок 1.18).

Принцип роботи:

1. Генерація потенційних регіонів. Алгоритм Selective Search аналізує вхідне зображення та генерує близько 2000 потенційних регіонів, які можуть містити об’єкти. Цей метод базується на ієрархічному групуванні пікселів за подібністю кольору, текстури чи розміру.

2. Операція вирівняння. Кожен з 2000 потенційних регіонів стискається до розміру 227×227 пікселів, незалежно від вихідного розміру.

3. Визначення ознак за допомогою CNN. Кожен вирівняний регіон подається на вхід до згорткової нейронної мережі.

4. Класифікація об’єкту. Після визначення ознак, регіони подаються на вхід до набору бінарних класифікаторів, що визначають наявність об’єкта в регіоні.

5. Окреслення меж об’єкту Визначенням координатної рамки навколо об’єкту займається лінійний регресор.

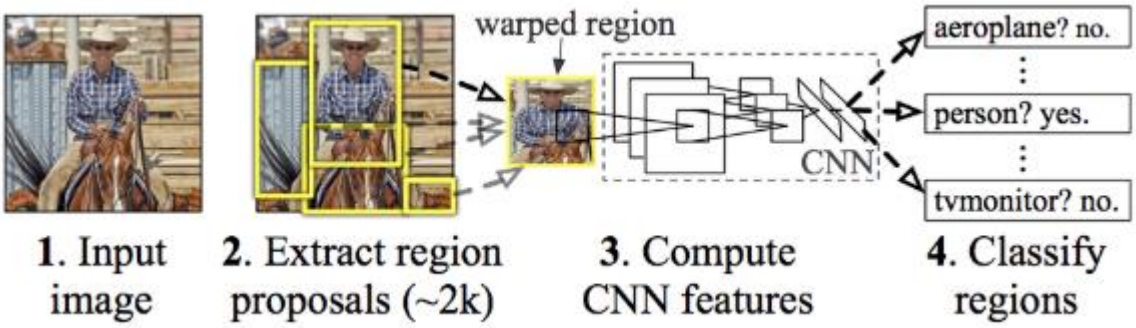


Рисунок 1.18 – Алгоритм роботи R-CNN

Переваги:

− висока точність розпізнавання об’єктів;

− можливість використання попередньо навчених CNN.

Недоліки:

− низька швидкість;

− високі вимоги до обчислювальних ресурсів;

− потенційні регіони можуть накладатись або перехрещуватись, що призводить до надлишкових обчислень;

− деформація регіонів під час операції вирівнювання може спотворювати об'єкти.

Наступний алгоритм, Fast R-CNN, мала краще оптимізовану архітектуру, що дозволила вирішити проблему продуктивності, яка була в R-CNN.

Покращення нової версії:

− замість обробки кожної обробки окремого регіону, все зображення тільки один раз подається на вхід до згорткової мережі і формує карту ознак;

− класифікація об’єктів і визначення рамок виконуються разом у межах однієї функції втрат.

1.5.2 Одноетапні детектори об’єктів

Одноетапні детектори об’єктів виконують класифікацію та локалізацію об’єкту за один прохід, оминаючи етап генерації потенційних регіонів. Не зважаючи на незначну втрату точності такий підхід забезпечує високу швидкодію.

YOLO (You Only Look Once) – сімейство моделей, одноетапних детекторів (див. рисунок 1.19).

Принцип роботи:

1. Розбиття зображення. Вхідне зображення розбивається на сітку розміром S×S., кожна з клітинок відповідає за розпізнавання об’єкту.

2. Передбачення меж та класу об’єкту. Кожній клітинці прогнозується певна кількість рамок. Одночасно з цим відбувається класифікація об’єкту.

3. Об’єднання результатів. Всі прогнозовані дані з клітинок об’єднуються, створюючи набір можливих об’єктів.

4. Окреслення меж. Використовуючи алгоритм Non-Maximum Suppression (NMS) з клітинок видаляються рамки, що перекриваються.

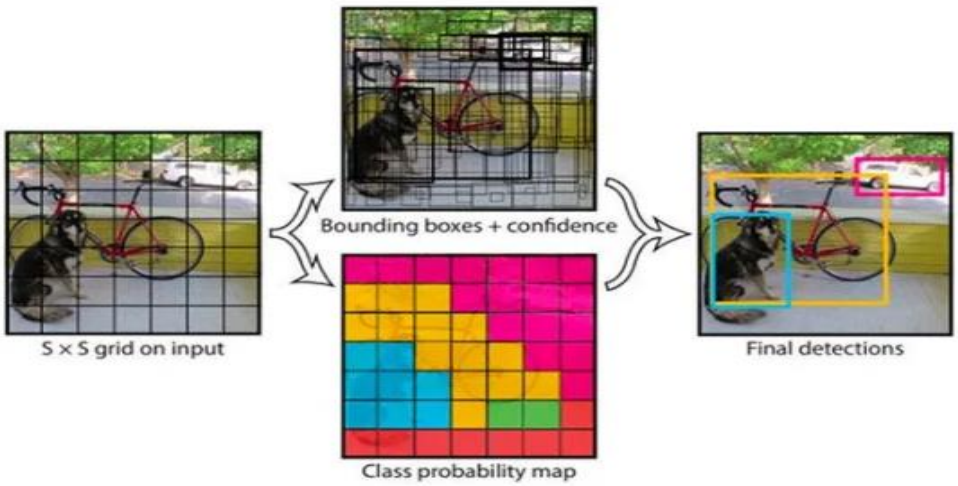


Рисунок 1.19 – Алгоритм роботи YOLO

На рисунку 1.20 наведена порівняльна діаграма моделей YOLO на датасеті COCO з середньою точність розпізнавання (mAP) [8].

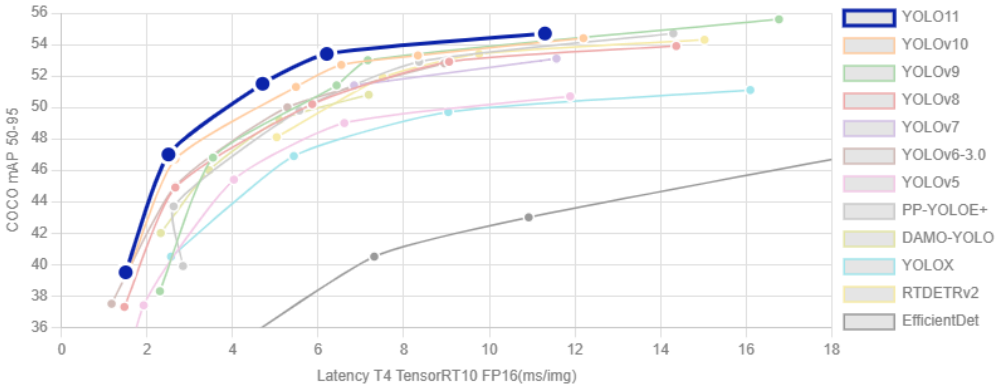


Рисунок 1.20 – Порівняння моделей YOLO

1.6 Проблеми сучасних систем

Сучасні КФС з інтеграцією алгоритмів КЗ стикаються з рядом технічних, організаційних та економічних проблем [9]:

1. Продуктивність. Обробка відеопотоку високої роздільної здатності з використанням сучасних нейромережевих алгоритмів вимагає значних обчислювальних ресурсів. Моделі сімейства YOLO для досягнення частоти в 30 кадрів на секунду потребують графічних прискорювачів рівня NVIDIA RTX 3060 або вище.

Моделі MobileNet чи ShuffleNet не такі вибагливі до обчислювальних ресурсів, але їх використання у якості розпізнавання об’єктів в середньому на 10-15% гріше, ніж повноцінні моделі.

2. Точність. Лабораторні дослідження демонструють точність розпізнавання 90-96% на стандартних датасетах (COCO, Pascal VOC), проте в реальних виробничих умовах цей показник знижується до 75-85%. Основними факторами зниження цих показників у виробничому середовищі є:

– нестабільне освітлення;

– часткове перекриття об'єктів один одного;

– об'єкти розташовані у не природньому положенні;

– подібність об'єктів.

3. Передача візуальної інформації. При побудові система, яка буде мати розподілену архітектуру, де камера розташована на маніпуляторі, обробка виконується на сервері, а керування здійснюється з мобільного пристрою, виникає проблема затримок при передачі.

4. Обмежена масштабованість. Існуючі системи зазвичай налаштовані для розпізнавання фіксованого набору об'єктів. Додавання нових типів об'єктів вимагає:

– збору великого кількості нових зображень;

– перенавчання або донавчання нейромережевої моделі;

– перевірка у виробничому середовищі;

– можливе погіршення розпізнавання минулих об’єктів.

5. Вартість. Комерційні системи від провідних виробників коштують від $30,000 що робить їх недоступними для використання малим або середнім бізнесом, чи в навчальних закладах, дослідницьких лабораторіях або стартапах.

1.7 Постановка завдання дослідження

В умовах сьогодення зростає потреба у мобільних системах віддаленого керування, здатних забезпечити автоматичне розпізнавання об’єктів у реальному часі. Що є особливо актуальним при використанні в таких галузях промисловості як, логістика, медична робототехніка та інші.

Метою кваліфікаційної роботи є розробка та дослідження кіберфізичної системи віддаленого керування маніпулятором із використанням технології комп’ютерного зору, що здатна розпізнавати ефективно розпізнавати об’єкти, попри обмеженість обчислювальних ресурсів у режимі реального часу.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– провести комплексний аналіз сучасного стану розвитку кіберфізичних систем та методів інтеграції із системами комп’ютерного зору. Це передбачає дослідження існуючих підходів у реалізації систем віддаленого керування, оцінка перевага та проблем застосування технології розпізнавання об’єктів у таких системах;

– дослідити апаратні та програмні платформи, придатні для створення системи керування маніпулятором із передачею відеопотоку. Це передбачає оцінку продуктивності, можливості реалізації алгоритмів КЗ та вартості цих рішень з метою виокремлення найбільш оптимального варіанту конфігурації системи;

– розробити архітектуру кіберфізичної системи з урахуванням можливості передачі відеопотоку з мінімальними затримками та можливістю розпізнавання об’єктів. Архітектурне рішення повинно забезпечувати модульність, масштабованість та передбачати можливості майбутньої інтеграції додаткового сенсорного обладнання;

– розробити методи та алгоритми обробки відеопотоку для розпізнавання об’єктів у режимі реального часу з урахуванням обмеженості обчислювальних ресурсів. Це передбачає вибір оптимального алгоритму розпізнавання об’єктів та його адаптацію до мобільного застосунку керування маніпулятором;

– створити програмне забезпечення для керування маніпулятором із мобільного пристрою. Це передбачає створення зрозумілого інтерфейсу керування рухомими механізмами маніпулятору, передачу команд управління цим механізмам, відображення відеопотоку та розпізнавання об’єктів з подальшим виведенням інформації про них;

– провести експериментальні дослідження ефективності системи розпізнавання об’єктів. Це передбачає тестування точності розпізнавання, в різних умовах освітлення та на різній відстані.

Результатом роботи мають підтвердити ефективність інтеграції КФС з технологія комп’ютерного зору.

2 ТЕОРЕТИЧНИЙ РОЗДІЛ

2.1 Загальна характеристика об’єкта дослідження

КФС віддаленого керування маніпулятором з інтегрованим комп’ютерним зором призначена для розпізнавання об’єктів, що функціонує в реальному часі.

Архітектура системи складається з трьох взаємопов’язаних підсистем:

– підсистема захоплення відеопотоку, що відповідає за формування та передачу візуальної інформації у робочій зоні маніпулятору;

– обчислювальна підсистема, що відповідає за обробку отриманої візуальної інформації, та формування і передачу команд для управління ланками маніпулятору;

– підсистема керування виконавчими механізмами, що відповідає за стабільне одночасне керування ланками маніпулятору.

За функціональними призначення система відноситься до автоматизованих КФС маніпулювання об’єктів, що використовується у начальних чи дослідницьких цілях. Основною сферою застосування є навчальні лабораторії університетів для підготовки фахівців з робототехніки.

За архітектурою керування система відноситься як напівавтоматична, оскільки частина операцій (вибір об’єкту, визначення траєкторії руху), виконуються оператором, а решта (розпізнавання об’єктів, стабілізація руху ланок маніпулятору) – виконується автоматично на основі алгоритмів комп’ютерного зору та програмного коду.

За архітектурою побудови системи є розподіленою, оскільки процес захоплення та обробки візуальної інформації здійснюється на різних пристроях.

До основних функціональних характеристик системи відносять:

– обмін інформацією між маніпулятором та мобільним пристроєм за допомогою технології Wi-Fi;

– формування та передача команд керування роботою ланок маніпулятору;

– прийом та обробка команд керування роботою ланок маніпулятору;

– керування роботою ланок маніпулятору (переміщення захоплювачу вліво/вправо, вгору/вниз, вперед/назад та захват/утримання об’єктів);

– захоплення та передача відеопотоку в реальному часі;

– розпізнавання об’єктів за попередньо навченої нейронної мережі.

2.2 Структура об’єкта дослідження

КФС має багаторівневу ієрархічну архітектуру, в межах якої апаратні та програмні складові функціонують як єдиний взаємопов’язаний комплекс. Така структура забезпечує чіткий розподіл між обчислювальними ресурсами, виконавчими механізмами та елементами керування.

Структура КФС умовно поділяється на чотири функціональні рівні:

– презентаційний рівень, який представлений мобільним застосунком, що забезпечує відображення відеопотоку в режимі реального часу, передачу команд керування ланками маніпулятору;

– комунікаційний рівень, що забезпечує обмін інформації між функціональними підсистемами, зокрема мобільними застосунком, бездротовим модулем та платою керування маніпулятором;

– рівень керування, що забезпечує логіку функціонування системи, зокрема алгоритми розпізнавання об’єктів завдяки КЗ та формування команд для виконавчих механізмів маніпулятору;

– фізичний рівень, що забезпечує роботу механічних та електронних компоненти маніпулятору, які взаємодіють із навколишнім середовищем.

На фізичному рівні конструкція маніпулятору складається з трьох серводвигунів, кожен з яких забезпечує окрему ступінь свободи. Серводвигун бази відповідає за горизонтальне обертання по осям x та z, правий серводвигун відповідає за рух вертикальний рух по осі y, лівий серводвигун відповідає за горизонтальний рух по осі x.

За геометрією робочої зони маніпулятор відноситься до маніпуляторів ангулярного типу (див. рисунок 1.11).

2.3 Математична модель руху маніпулятору

Маніпулятор можна розглядати як ланцюг, який складається з декількох твердих тіл (ланок), які послідовно з’єднані та приводяться в рух виконавчими пристроями. Один кінець такого ланцюгу з’єднується з основою, а інший кінець вільний та з’єднується з виконавчим органом, що надає змогу впливати на об’єкти, здійснювати їх переміщення, захоплення тощо [10].

Кінематика маніпулятору вивчає геометрію руху щодо попередньо визначеної абсолютної системи координат, не беручи до уваги сили та моменти, які породжують цей рух.

У кінематика руху маніпулятору виділяють дві фундаментальні задачі:

− пряма задача кінематики полягає у тому, що за заданими кутовими положеннями з’єднань $q(t)=(q1(t), ..., qn(t))T$ та геометричними параметрами ланок визначається кінцева позиція виконавчого органу маніпулятору;

− зворотна задача кінематики полягає у тому, що за цільовим координатами виконавчого органу маніпулятору та відомою геометрією конструкції розраховуються необхідні значення узагальнених координат для кожного ступеня свободи.

2.3.1 Рівняння прямої задачі кінематики

Для планування рухів маніпулятору, необхідно зрозуміти взаємозв'язок між виконавчими пристроями та кінцевим положенням робота в робочому середовищі. Для стаціонарних маніпуляторів знаючи положення або кут кожного суглоба, можна обчислити положення кінцевих виконавчих пристроїв (кінцівок) за допомогою тригонометричних розрахунків

Отже основне завдання рівняння прямої кінематики це визначення поточного положення виконавчого пристрою (захоплювача).

Для спрощення математичного аналізу та наочності презентації кінематичних рівнянь доцільно застосовувати спрощену модель дволанкового

маніпулятора з двома обертовими парами у площині xy (див. рисунок 2.1) та серводвигун бази буде розглядатись як окрема ступінь свободи.

Кожна з ланок є абсолютно жорсткою, довжиною L. Перша ланка L_1 закріплена до основи та розташовується під кутом Q_1 до неї. Друга ланка L_2 закріплена до кінця першої ланки та розташовується під кутом Q_2 до неї. На кінці другої ланки розташований виконавчий пристрій (захоплювач).

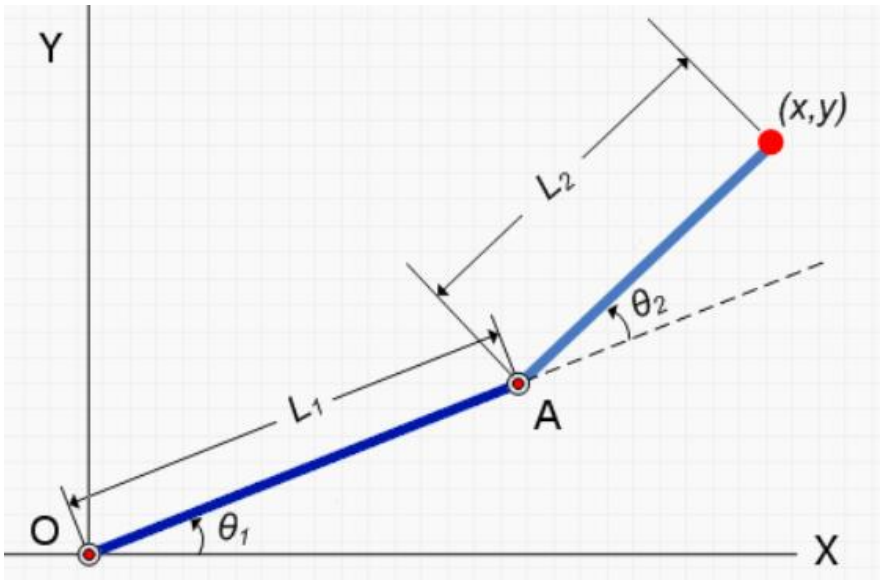


Рисунок 2.1 – Кінематична модель дволанкового маніпулятору

Виходячи з цього маємо першу систему відліку, яка пов’язана з початком координат в точці кріплення ланки першої ланки, в точці O.

Для початку необхідно визначити зміщення другої системи відліку відносно (координати точки A) першої (2.1):

$$x_A = l * \cos(q_1);\ y_A = l * sin(q_1) \tag{2.1}$$

Розрахунок координат (x,y) в системі відліку першої ланки (2.2):

$$x'' = l * \cos(q_2);\ y'' = l * sin(q_2) \tag{2.2}$$

З рисунку 2.5 видно, що стосовно точки O, друга ланка повернута відносно першої на (Q_1+Q_2) (2.3):

$$x' = l * \cos(q_1 + q_2);\ y' = l * sin(q_1 + q_2) \quad (2.3)$$

Формула для рішення, задачі прямої кінематики (2.4):

$$x = x_A + x' = l * \cos(q_1) + l * \cos(q_1 + q_2)$$
$$y = y_A + y' = l * \sin(q_1) + l * sin(q_1 + q_2) \quad (2.4)$$

2.3.2 Рівняння зворотної задачі кінематики

Для досягненні заданого положення виконавчого пристрою (захоплювачу) необхідно визначити положення кожного суглобу та інвертувати співвідношення між суглобами та виконавчим пристроєм.

Отже основне завдання рівняння зворотної кінематики це розрахунок кута та положення суглобів, необхідного для досягнення виконавчим пристроєм (захоплювачем) заданої точки.

Для вивчення оберненої задачі кінематики доцільно застосовувати спрощену модель дволанкового маніпулятора з двома обертовими парами (див. рисунок 2.2).

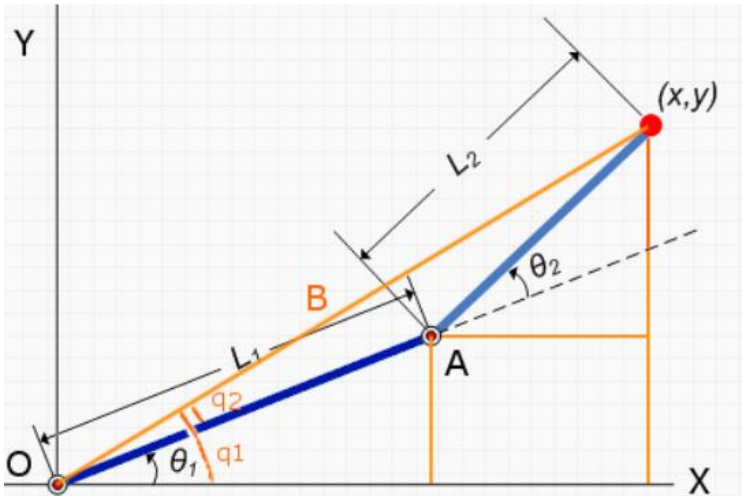


Рисунок 2.2 – Кінематична модель дволанкового маніпулятору

З відомими декартовими координатами (x, y) виконавчого пристрою за допомогою зворотної задачі кінематики можна знайти узагальнені координати Q_1 та Q_2. Проведемо додаткові побудови та позначимо кут q_2, для якого за теоремою косинусів отримуємо рівняння (2.5):

$$B^2 = l^2 + (x^2 + y^2) - 2 * \ l * \sqrt{x^2 + y^2} * \cos\,(\,q_2) \tag{2.5}$$

Виразимо q_2 та отримаємо вираз (2.6):

$$q_2 = arccos\frac{l^2+(x^2+y^2)-B^2}{2*l*\sqrt{x^2+y^2}} \tag{2.6}$$

З малюнка 4.13 видно: $tg(Q_1 + q_2) = y/x$. Отримуємо формулу (2.7):

$$Q_1 = arctg\frac{y}{x} - q_2 \tag{2.7}$$

Розпишемо отриманий вираз через тангенс (2.8):

$$tg(Q_1 + Q_2) = \frac{y-l*\sin\,(Q_1)}{x-l*\cos\,(Q_1)} \tag{2.8}$$

Виразимо Q_2 та отримаємо вираз (2.9):

$$Q_2 = arctg\frac{y-l*\sin\,(Q_1)}{x-l*\cos\,(Q_1)} - Q_1 \tag{2.9}$$

Таким чином, рішення зворотного завдання кінематики матиме вигляд (2.10):

$$Q_1 = arctg\frac{y}{x} - arccos\frac{l^2 + (x^2 + y^2) - B^2}{2 * l * \sqrt{x^2 + y^2}}$$

$$Q_2 = arctg\frac{y-l*\sin\,(Q_1)}{x-l*\cos\,(Q_1)} - Q_1 \tag{2.10}$$

Отримані рівняння дозволяють визначити необхідне положення ланок маніпулятора виходячи базуючись на його поточних координат. Перше рівняння дозволяє нам знайти координату Q_1 за відомими координатами (x, y), а друге кут Q_2.

Важливою особливістю зворотної кінематики є специфіка розрахунку тригонометричних функцій. В результаті таких обчислень не виключене виникнення інтервалів, для яких не буде розв’язків (особливо при роз розрахунку функції арккосинусу).

2.4 Обґрунтування і вибір методів синтезу системи

Розробка КФС з алгоритмами КЗ вимагає комплексного підходу, який об’єднує методи аналізу існуючих рішень, синтезу системи та експериментального дослідження.

Вибір методів дослідження залежить від специфіки об’єкту (обчислювальні ресурси, умови використання) та поставленими функціональними задачами (розпізнавання об’єктів, планування траєкторій, прийняття рішень).

Основними методами синтезу є: багатокритеріальний порівняльний аналіз апаратних компонентів, порівняння алгоритмів розпізнавання об’єктів, математичне моделювання кінематики маніпулятору.

2.4.1 Метод порівняльного аналізу для вибору апаратних компонентів

Вибір оптимальних апаратних компонентів системи можливий завдяки використанню методу багатокритерного порівняльного аналізу. Критерії вибору мікроконтролерів та обчислювальних платформ:

– обчислювальна продуктивність;

– наявність або підтримка модулю камери;

– сумісність з виконавчими механізмами;

– підтримка бездротових технологій;

– енергоефективність;

– обсяг оперативної пам’яті;

– вартість та доступність компонентів.

2.4.2 Метод порівняльного аналізу для вибору програмних компонентів

Для розробки програмного забезпечення необхідний аналіз середовищ та бібліотек за наступними критеріями:

– можливість реалізації алгоритмів КЗ;

– сумісність з апаратними платформами;

– зручність розробки;

– спільнота та документація.

2.4.3 Метод математичного моделювання руху маніпулятору

Для опису руху маніпулятору необхідно побудувати математичну модель, яка базується на основі методів прямої та зворотної кінематики.

Побудована модель враховує геометричні параметри конструкції, довжини ланок та кінематичні обмеження, що дозволять визначити кінцеве положення виконавчого органу у просторі за заданими кутами повороту ланок та обчислити необхідні кути для досягнення заданої точки.

2.5 Обґрунтування і вибір методів експериментальних досліджень

Експериментальна частина роботи спрямована на підтвердження працездатності розробленої системи та визначення ефективності її ключових характеристик: точність розпізнавання об’єктів та затримки в системі керування.

2.5.1 Метод вимірювання затримок

Мета дослідження: визначити часові затримки системи в проміжку між подачею команди оператора та фактичним виконанням дії маніпулятором.

Обґрунтування: затримка (latency) є одним з ключових параметрів ефективності роботи систем віддаленого керування, яка визначає спроможність оператора ефективно взаємодіяти з маніпулятором. Затримки виникають під час передачі інформації по бездротовому каналу Wi-Fi, обробкою команди мікроконтролером плати керування та рухом серводвигунів.

Для оцінки цього параметра доцільно використати метод вимірювання затримок з моменту передачі команди мобільним пристроєм до фактичного виконання кожної дії маніпулятору (вліво/вправо, вгору/вниз, вперед/назад).

Методика дослідження:

– фіксування часових міток в момент передачі команд з мобільного застосунку (ts);

– фіксування часових міток в момент фактичного виконання дії маніпулятору (te);

– розрахунок загальної затримки: $\Delta t = te – ts$;

Очікувані результати: загальна затримка не повинна перевищувати 300-500 мс для комфортного керування оператором [11].

2.5.2 Метод оцінки точності розпізнавання об’єктів

Мета дослідження: оцінити якість роботи алгоритму КЗ з розпізнавання об’єктів з використанням стандартних метрик машинного навчання.

Теоретичне обґрунтування: точність розпізнавання об’єктів оцінюється за допомогою трьох ключових метрик машинного навчання, що дозволяє не лише виявити загальну точність алгоритму КЗ, але й проаналізувати його поведінку та визначити, чи придатна модель для роботи в реальних умовах.

Для оцінки доцільно використати стандартну методика обчислення показників Precision, Recall, що дозволяє комплексно характеризувати ефективність алгоритму.

Методика проведення:

– використати стандартного датасету COCO, який містить 128 анотованих зображень з еталонними мітками;

– пропустити кожне зображення датасету через алгоритм КЗ на базі нейронної мережі;

– отримати усереднені метрики по всьому датасету;

Очікувані результати: прийнятним результатом точності розпізнавання вважається точність в межах 70-85%.

2.6 Висновки до розділу

У цьому розділі було здійснений теоретичний аналіз побудови КФС віддаленого керування маніпулятором з інтегрованим КЗ. Наведено загальну характеристику, визначено основні принципи функціонування та логічну структуру системи, що включає презентаційний, комунікаційний, фізичний рівні та рівень керування.

Сформовано теоретичну базу для вирішення рівнянь прямої та оберненої кінематики, які в подальшому будуть використані при розробці алгоритму керування в мобільному застосунку системи.

Окреслені методи до вибору апаратних та програмних компонентів системи, на основі порівняння за критеріями, що дозволить обрати найбільш оптимальну конфігурацію для синтезу системи.

Окреслені методи експериментальних досліджень, необхідних для оцінки ключових характеристик системи: точність розпізнавання об’єктів та величина затримки.

3 СИНТЕЗ СИСТЕМИ

3.1 Цілі впровадження системи

Головна мета розробки та впровадження КФС є створення автоматизованого робототехнічного комплексу, здатного розпізнавати об’єкти та виконувати маніпуляційні дії над об’єктами у режимі реального часу.

Система використовує принципи кіберфізичної реалізації, при якій фізичні процеси поєднуються з цифровими через комунікаційні засоби, що дозволяє досягти гнучке та адаптивне масштабування.

Основні цілі впровадження системи:

– інтеграція алгоритмів КЗ для розпізнавання об’єктів у процеси керування маніпулятором;

– розробка зручного інтерфейсу взаємодії оператора та маніпулятору;

– розробка модульної та масштабованої системи.

Практичним значенням впровадження КФС є створення модульної та масштабованої системи, яку можна використовувати в навчальних цілях, що демонструє принципи взаємодії фізичних процесів з цифровими.

Очікуваним результатом впровадження КФС є підвищення ефективності управління маніпуляторами, покращення якості та точності виконання операцій, а також створення інтелектуально-автоматизованих комплексів.

3.2 Формування технічних вимог до системи

Цей розділ визначає сукупність технічних вимог, які є основою для проектування та розробки системи. Базуючись на цілях впровадження (розділ 3.1), ці вимоги деталізують ключові аспекти реалізації, показники призначення, надійності та обмеження, що пред'являються до програмних та апаратних компонентів.

3.2.1 Вимоги до реалізації системи

Вимоги до середовища розробки системи:

– програмування мікроконтролера та бездротового модулю повинно бути реалізовано в Arduino IDE (версії 2.x) чи в VS Code з додатковим розширенням PlatformIO;

– програмування мобільного застосунку повинно бути реалізовано в Android Studio (версії Jellyfish або новіше) чи в VS Code з додатковим розширення Java;

– моделювання та тренування нейронної мережі повинно бути реалізовано в VS Code з додатковим розширенням Python чи в Google Colab.

Вимоги до архітектури системи:

– ПЗ для мобільного застосунку повинне бути модульним з дотримання архітектурного патерну MVVM (Model View ViewModel);

– ПЗ для бездротового модулю повинне забезпечувати ініціалізацію та підтримку Wi-Fi, ініціалізацію камери, обробку запитів від мобільного застосунку та передачу до мікроконтролера команд.

Вимоги до моделі КЗ:

– модель нейронної мережі повинна бути розроблена у форматі TFLite (.tflite);

– файл .tflite повиннеу бути інтегрований з мобільним застосунком;

– аналіз зображень має виконуватися з використанням SDK ML Kit від Google або бібліотеки TensorFlow Lite Interpreter.

Система повинна бути реалізована на базі робототехнічного набору Keyestudio.

3.2.2 Вимоги до показників призначення

Вимоги до показників призначення системи повинні включати:

– вантажопідйомність маніпулятора повинна бути не меншою за 10 г;

– похибка у позиціюванні виконавчого органу не повинна перевищувати ±10 мм;

– час відгуку системи на команди не повинен перевищувати 900 мс;

– затримка при передачі відеопотоку в режимі реального часу не повинна перевищувати 4 секунди;

– якість відеопотоку повинна бути більшою за 320x240 з частотою оновлення не менше 25 кадрів на секунду;

– середня точність розпізнавання об’єктів з набору COCO повинна становити не менше 72%.

3.2.3 Вимог до функцій (задач), виконуваним Системою

КФС призначена для виконання комплексу функцій направлених на: дистанційне керування маніпулятором, розпізнавання об’єктів в полі зору модуля камери та відображення результату розпізнавання в інтерфейсі мобільного застосунку. До основних функцій системи можна віднести: керування, інформаційна та допоміжна.

Вимоги до функції керування виконуваної системою:

– повинна забезпечувати дистанційне керування рухами маніпулятору за допомогою мобільного застосунку;

– повинна забезпечувати керування окремими серводвигунами: бази для обертання ліворуч/праворуч, плеча для руху вперед/назад, ліктя для руху вгору/вниз, захоплювача для відкриття/закриття, модулю камери для нахилу вперед/назад;

– повинна забезпечувати програмну реалізацію обмеження кутів обертання серводвигунів.

Вимоги до інформаційної функції виконуваної системою:

– повинна забезпечувати захоплення відеопотоку у режимі реального часу;

– повинна забезпечувати передачу відеопотоку на мобільний пристрій оператора використовуючи технологію Wi-Fi;

– повинна забезпечувати інтеграцію алгоритмів КЗ для розпізнавання об’єктів на основі набору даних COCO;

– повинна забезпечувати виявлення об’єктів на зображені з побудовою координатної рамки навколо та ідентифікацію класу об’єкта;

– повинна забезпечувати відображення результатів розпізнавання про клас об’єкта та його ідентифікатор в мобільному застосунку.

Вимоги до допоміжної функції виконуваної системою:

– повинна забезпечувати генерацію, прийом та обробку команди оператора, надісланих через мобільний застосунок;

– повинна забезпечувати можливість до масштабування системи шляхом додавання нових апаратних модулів;

– повинна забезпечувати синхронний обмін інформацією між компонентами системи (модулем камери, платою керування, серводвигунами);

– повинна забезпечувати стійку роботу без затримок у передачі команд і відеопотоку, які б перевищували б 900 мс.

3.2.4 Вимоги до видів забезпечення

3.2.4.1 Вимоги до математичного забезпечення

Вимоги до математичного забезпечення при розробці системи повинні включати:

− розрахунок рівняння прямої кінематики необхідної для визначення кінцевого положення та орієнтації на основі відомих параметрів положення ланок та з'єднань маніпулятору;

− розрахунок рівняння зворотної кінематики необхідної для визначення необхідних параметрів (кутів повороту, положення з'єднань) з метою досягнення заданого положення та орієнтації у просторі;

− реалізацію методів оптимізації обчислювальних процесів, які зменшують час затримки між виявленням об’єкта та безпосереднім виконанням команд керування.

3.2.4.2 Вимоги до інформаційного забезпечення

Вимоги до інформаційного забезпечення при розробці системи повинні включати:

− забезпечення структурного подання команд керування у буквено-цифровому вигляді;

− забезпечення передачі команд з мобільного застосунку до бездротового модулю з використанням захищеного протоколу бездротової взаємодії;

− забезпечення обміну даними між бездротовим модулем та платою керування серводвигунами з використанням протоколу прийому/передачі;

− забезпечення єдиної організованої передачі даних між компонентами системи.

3.2.4.3 Вимоги до лінгвістичного забезпечення

Вимоги до лінгвістичного забезпечення застосування в системі мов програмування високого рівня повинні включати:

− розробку графічного інтерфейсу оператора мовою Kotlin з додатковим використанням фреймворку Jetpack Compose;

− розробку програмного забезпечення мікроконтролеру та бездротового модулю мовою C++;

− навчання нейронної мережі для розпізнавання об’єктів мовою Python з додатковим використанням бібліотек програмної оптимізації розпізнавання.

Основною мовою взаємодії оператора з системою повинна бути англійська. Весь графічний інтерфейс мобільного застосунку, пункти меню, кнопки керування та інформаційні повідомлення повинні бути англійською мовою.

3.2.4.4 Вимоги до технічного забезпечення

Вимоги до програмного-технічного комплексу маніпулятору повинні включати:

− можливість інтеграції плати розширення для керування серводвигунами;

− можливість живлення від зовнішнього блоку живлення на 12В/5А або двома літій-іонними акумуляторами типу 18650, ємністю по 3200 мА кожний.

− можливість інтеграції модулю бездротового зв’язку;

Вимоги до програмно-технічного комплексу оператора повинні включати:

− наявність вбудованого модулю бездротового зв’язку Wi-Fi для взаємодії з маніпулятором;

− мінімальна кількість ядер процесору для обробки візуальної інформації – 6;

− наявність сенсорного дисплею для керування серводвигунами маніпулятору;

3.3 Синтез структурної схеми системи

На основі сформульованих технічних вимог розроблена розподілена структурна схема системи, що забезпечує модульність та масштабованість.

Структурна схема системи відображає взаємозв’язки компонентів з їх фізичним розташуванням, див. рисунок 3.1. Розробка структурної схема є фундаментальною у побудові та інтеграції системи як єдиної функціональної одиниці.

Основні компоненти структурної схеми та їх призначення:

– блок живлення: забезпечує електроенергією всі компоненти системи;

– плата керування: виконує функції центрального виконавчого елементу системи;

– плата розширення: встановлюється на плату керування та дозволяє стабільно керувати серводвигунами;

– серводвигуни: забезпечують рух ланок маніпулятору та модулю камери;

– модуль бездротового керування: забезпечує передачу відеопотоку та команд і взаємодію з модулем камери;

– модуль камери: OV2640 камера здійснює захоплення відеопотоку;

– маршрутизатор: забезпечує бездротове з’єднання між бездротовим модулем керування та мобільним пристроєм;

– мобільний пристрій: використовується як інтерфейс дистанційного керування маніпулятором.

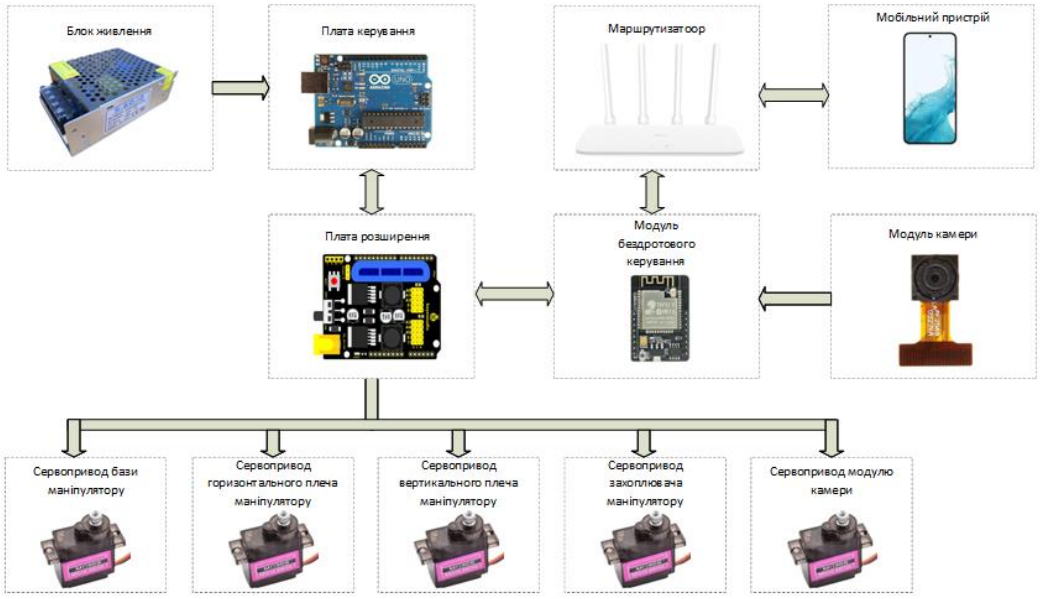


Рисунок 3.1 – Структурна схема системи

3.4 Вибір та обґрунтування застосування апаратних засобів

Вибір апаратних засобів системи базується з урахуванням технічних вимог, наведених у розділі 3.2, а також на основі порівняльного аналізу, наведеного в розділі 1.4.

Для реалізації підсистеми захоплення візуальної інформації був проведений порівняльний аналіз трьох мікроконтролерних платформ (див. таблицю 1.1, розділ 1.3.1): ESP32-CAM, Raspberry Pi Zero W та Arduino Nano 33 BLE Sense.

ESP32-CAM було обрано як найбільш оптимальний варіант, оскільки забезпечує співвідношення вартості, продуктивності обчислювальних ресурсів та функціональності необхідних для реалізації системи [12].

Для реалізації підсистеми обробки візуальної інформації та розпізнавання об’єктів був проведений порівняльний аналіз трьох обчислювальних платформ (див. таблицю 1.2, розділ 1.3.2): смартфони на базі Android, NVIDIA Jetson Orin Nano та Intel NUC.

Смартфон на базі Android було обрано як основну обчислювальну платформу для обробки візуальної інформації, оскільки використання смартфону дозволяє уникнути додаткових витрат на придбання обладнання, підтримує технології реалізації машинного навчання та має високу мобільність та автономність.

В якості бази робототехнічної система було обрано набір Keyestudio KS019x. Набір було обрано як оптимальний компроміс між функціональністю, доступністю та простотою інтеграції. Комплект містить готові матеріали, завдяки чому процес складання пришвидшується. Наявність плати розширення з драйвером PCA9685 дозволяє зручно підключати та керувати серводвигунами маніпулятору.

Для забезпечення рухливості ланок маніпулятора було обрано серводвигуни MG90S з металевим редуктором.

Обґрунтування використання серводвигунів MG90S:

– металевий редуктор дозволяє підвищити точність позиціювання виконавчого органу в просторі,

– металевий редуктор підвищує довговічність та надійність;

– можливість підіймати більшу вагу, за рахунок більшого крутного моменту;

Для забезпечення живлення всіх компонентів системи було обрано блок живлення MN-60-12-5A. Зовнішній блок живлення 12В/5А забезпечує стабільну вихідну напругу, має достатній запас потужності та виключає необхідність у періодичній зарядці.

На рисунку 3.2 представлено загальний вигляд апаратної частини системи, яка включає робототехнічний набір Keyestudio KS019x та блок живлення MN-60-12-5A.

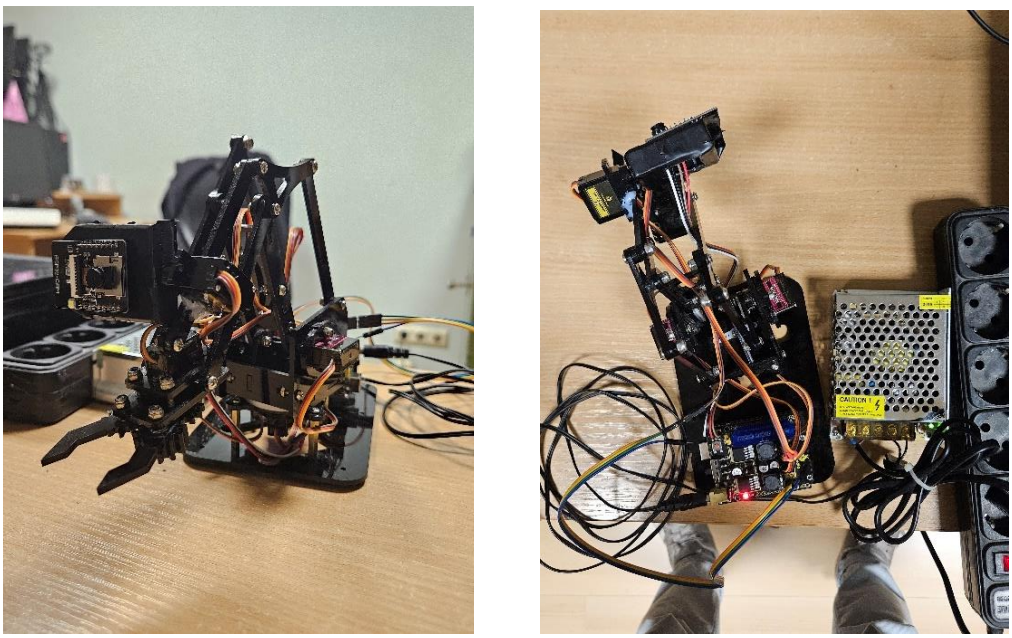


Рисунок 3.2 – Маніпулятор на базі набору Keyestudio KS019x

3.5 Синтез схеми підключення апаратних засобів

Схема підключення апаратних засобів демонструє електричні з’єднання між усіма компонентами системи.

Зовнішній блок живлення підключається до DC роз’єму на платі розширення. Серводвигуни підключаються до роз’ємів S (керуючий), V та G на платі розширення. Схема підключення серводвигунів маніпулятору до плати розширення наведена на рисунку 3.3.

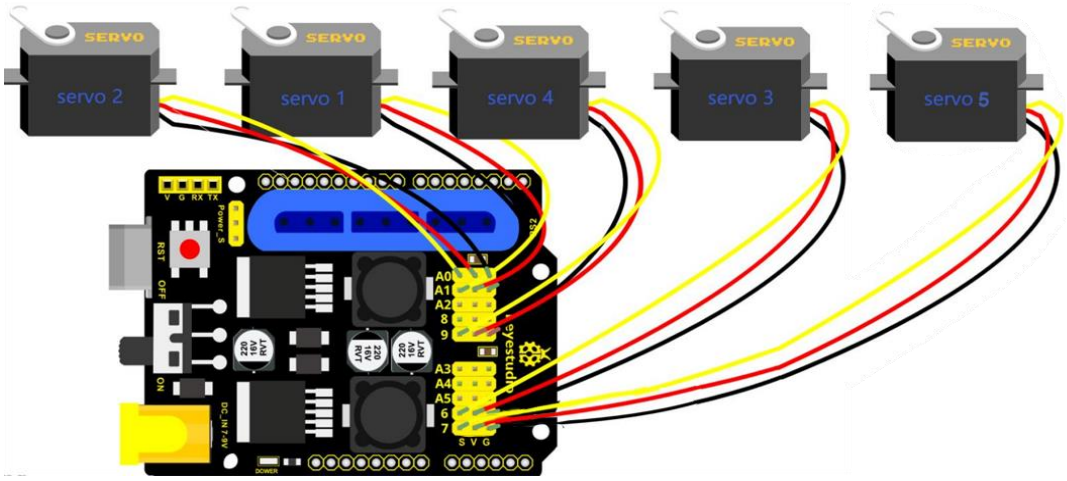


Рисунок 3.3 – Схема підключення серводвигунів до плати розширення

Відповідно до вимог (3.2.1), зв'язок між головним модулем та підлеглим контролером реалізовано через апаратний послідовний інтерфейс.

Pin TX (Transmit) модуля ESP32-CAM підключається до Pin RX (Receive) плати розширення.

Pin RX (Receive) модуля ESP32-CAM підключається до Pin TX (Transmit) плати розширення (необхідно для можливого двостороннього зв'язку).

Схема підключення бездротового модулю ESP32-CAM до плати розширення наведена на рисунку 3.4.

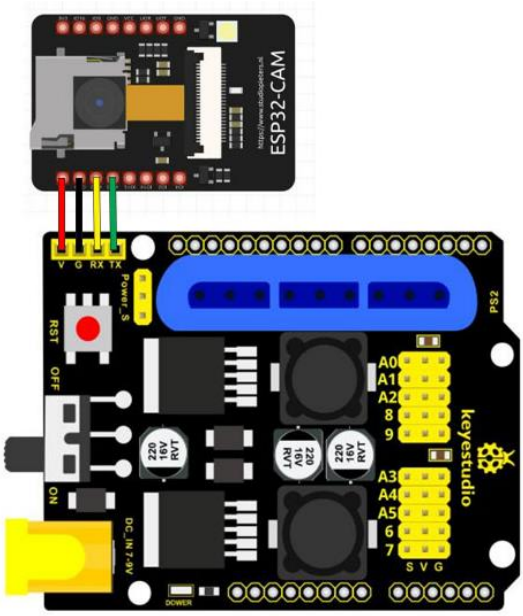


Рисунок 3.4 – Схема підключення бездротового модулю ESP32-CAM до плати розширення

3.6 Вибір та обґрунтування застосування програмних засобів

Програмна складова системи базується з урахуванням технічних вимог, наведених у розділі 3.2.

Згідно технічних вимог (розділ 3.2.1), для розробки мобільного застосунку обрано мову Kotlin з фреймворком Jetpack Compose.

Обґрунтування вибору:

– Kotlin з 2019 року є офіційною мовою розробки для Android, нові SDK та бібліотеки розповсюджуються з пріоритетом для Kotlin;

– використання декларативного UI фреймворку Jetpack Compose дозволяє значно скоротити обсяг коду;

– Jetpack Compose нативно інтегрується з Google ML Kit, що дозволяє скоротити час підключення моделі розпізнавання об’єктів;

– Kotlin разом з Jetpack Compose, ViewModel та LiveData автоматично реалізує MVVM, що спрощує розділення логіки та UI.

Бібліотекою для інтеграції моделі розпізнавання об’єктів в мобільний застосунок обрано Google ML Kit.

Обґрунтування вибору:

– ML Kit має велику кількість оптимізованих під мобільні пристрої інструментів для роботи з машинним навчанням;

– ML Kit містить попередньо натреновані моделі для розпізнавання об’єктів, тексту, облич, що використовувати готові рішення, без необхідності тренування власних мереж;

– алгоритми SDK оптимізовані під специфіку мобільних пристроїв, що забезпечує стабільну роботу навіть на апаратно слабших моделях;

– Google регулярно оновлює та оптимізує наявні продукти, що гарантує довгострокову підтримку.

В якості моделі розпізнавання об’єктів було обрано EfficientDet-Lite0 з використанням фреймворку TFLite [13].

Обґрунтування вибору:

– EfficientDet-Lite0 забезпечує точність розпізнавання датасету COCO на рівні більшому за 76% (вимога 3.2.2);

– формат моделі .tflite, дозволяє прямо інтегрувати її в мобільний застосунок.

Для програмування плати керування та бездротового модулю обрано середовище Arduino IDE.

Обґрунтування вибору:

– наявність великої кількості вбудованих бібліотек, що значно спрощує написання програмного коду;

– інтуїтивно зрозумілий інтерфейс, який скорочує час на налаштування та запуск проєкту;

– спільнота Arduino одна з найбільших, завдяки чому легко знайти приклади або готові рішення;

– Arduino IDE підтримує велику кількість апаратних платформ, що забезпечує гнучкість під час роботи з різними платформами.

3.7 Висновки до розділу

В цьому розділі виконаний синтез апаратної та програмної складової КФС дистанційного керування маніпулятором з використанням технологій КЗ.

Сформований перелік технічних вимог до розробки системи, які включають вимоги до показників призначення, реалізації та функції (задачі) виконуваними системою. Визначено ключові параметри ефективності системи: час відгуку на команди оператора, затримка передачі відеопотоку в режимі реального часу, точність розпізнавання об’єктів.

Розроблена структурна схема системи з основними компонентами та описана область їх використання на основі сформованих технічних вимог.

На основі порівняльного аналізу, наведеного в розділі 1, для реалізації підсистеми захоплення відеопотоку було обрано платформу ESP32-CAM. Вибір цієї платформи є оптимальним рішення з точки зору співвідношення ціна/ефективність. В якості обчислювальної платформи системи обрано смартфон на базі ОС Android, оскільки це дозволяє уникнути використання дороговартісних стаціонарних обчислювальних модулів. Виконавча частина системи базується на робототехнічному наборі Keyestudio KS019x із серводвигунами MG90S.

Для розробки мобільного застосунку обрано мову Kotlin та декларативний фреймворк Jetpack Compose, які забезпечують високу швидкість розробки, модульність інтерфейсу та зручність підтримки програмного забезпечення. Як архітектуру нейронної мережі використано EfficientDet-Lite0, що має інтегровані інструменти експорту моделі у формат .tflite, необхідний для ефективного виконання на мобільних пристроях. Програмування плати керування та бездротового модуля виконано у середовищі Arduino IDE, яке забезпечує широкі можливості апаратної інтеграції та доступ до численних бібліотек.

Розроблені в цьому розділі розроблені рішення дозволяють створити функціональну економічно ефективну та масштабовану системи. Обрані апаратні та програмні елементи системи відповідають сформованим технічним вимогам.

4 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

4.1 Призначення та область застосування ПЗ

Програмне забезпечення КФС віддаленого керування маніпулятором призначене для реалізації інтелектуального управління роботехнічним комплексом з додатковою можливістю розпізнавання об’єктів в режимі реального часу. Система забезпечує взаємодію фізичних процесів (рух серводвигунів) з цифровими (алгоритми КЗ, бездротова комунікація, програмний інтерфейс).

Згідно вимог технічного завдання (розділ 3.2.1) архітектура ПЗ системи реалізована за модульним принципом та складається з трьох рівнів: прошивка мікроконтролерних пристроїв (фізичний рівень), логіка обробки команд та візуальної інформації (рівень керування) та інтерфейс користувача (презентаційний рівень).

Область застосування ПЗ:

– дослідницькі проєкти з тестування алгоритмів КЗ для розпізнавання та відслідковування об’єктів, розпізнавання облич, тексту, штрих-кодів;

– прототипування систем керування виробничими маніпуляторами;

– навчальні лабораторії закладів вищої та професійної освіти галузі знань F «Інформаційні технології»;

– демонстраційні STEM стенди на виставка та конференціях.

4.2 Обґрунтування технічних характеристик програми

Розробка ПЗ направлена на створення розподіленої системи керування маніпулятором з інтегрованим алгоритмом КЗ.

4.2.1 Постановка завдання на розробку програми

Формування завдання на розробку програмних модулів включають:

– генерацію команд керування серводвигунами маніпулятором;

– прийом та обробку команд керування серводвигунами маніпулятора;

– захоплення та передачу відеопотоку через бездротову мережу в режимі реального часу з мінімальними затримками;

– розпізнавання об’єктів у відеопотоці з використаннааям нейромережевих модулів;

– візуалізація результатів розпізнавання;

– відображення зручного інтерфейсу керування.

Для реалізації алгоритмів керування серводвигунами маніпулятору в ПЗ забезпечені мобільного застосунку використовується спрощена кінематична модель (розділ 2.4). Не зважаючи на те, що фізично маніпулятор має 3 ступені свободи, для основних рахунків кінцевого положення виконавчого органу використовується дволанкова модель у площині XY з окремим розглядом повороту бази.

Використання спрощеної модель дозволяє краще зрозуміти та передбачити поведінку маніпулятора, оскільки рух у площині легше контролювати та візуалізувати, на відміну від складних просторових траєкторій.

Функціональні обмеження та умови застосування:

– система може ідентифікувати тільки 80 класів датасету COCO, розпізнавання спеціалізованих об’єктів потребує додаткового навчання моделі;

– похибка позиціонування виконавчого органу становить ±10 мм (вимога 3.2.2), що обумовлено кутовою точністю серводвигунів ±2-3°;

– для стабільного керування необхідне надійне Wi-Fi з'єднання, при наявності перешкод дальність керування зменшується;

– при підключенні ESP32-CAM до нової Wi-Fi мережі необхідна перепрошивка бездротового модуля з оновленими SSID та паролем у коді програми;

– мобільний застосунок розповсюджується у вигляді APK файлу для ручного встановлення через Android Studio.

4.2.2 Опис алгоритму та функціонування програми

ПЗ системи розподіляє функції між окремими програмними модулями, забезпечуючи дотримання розподіленої архітектури.

Загальний алгоритм взаємодії програмних модулів:

1. Мобільний застосунок генерує команди керування на основі дій оператора;

2. Бездротовий модуль ESP32-CAM приймає команди керування та передає їх на плату керування Arduino Uno;

3. Плата керування Arduino Uno декодує команди та керує серводвигунами;

4. Модуль камери OV2640 бездротового модулю ESP32-CAM захоплює та передає відеопотік;

5. ML Kit мобільного застосунку аналізує кадри та розпізнає об’єкти.

Блок схема алгоритму роботи ПЗ плати керування Arduino Uno наведена на рисунку 4.1.

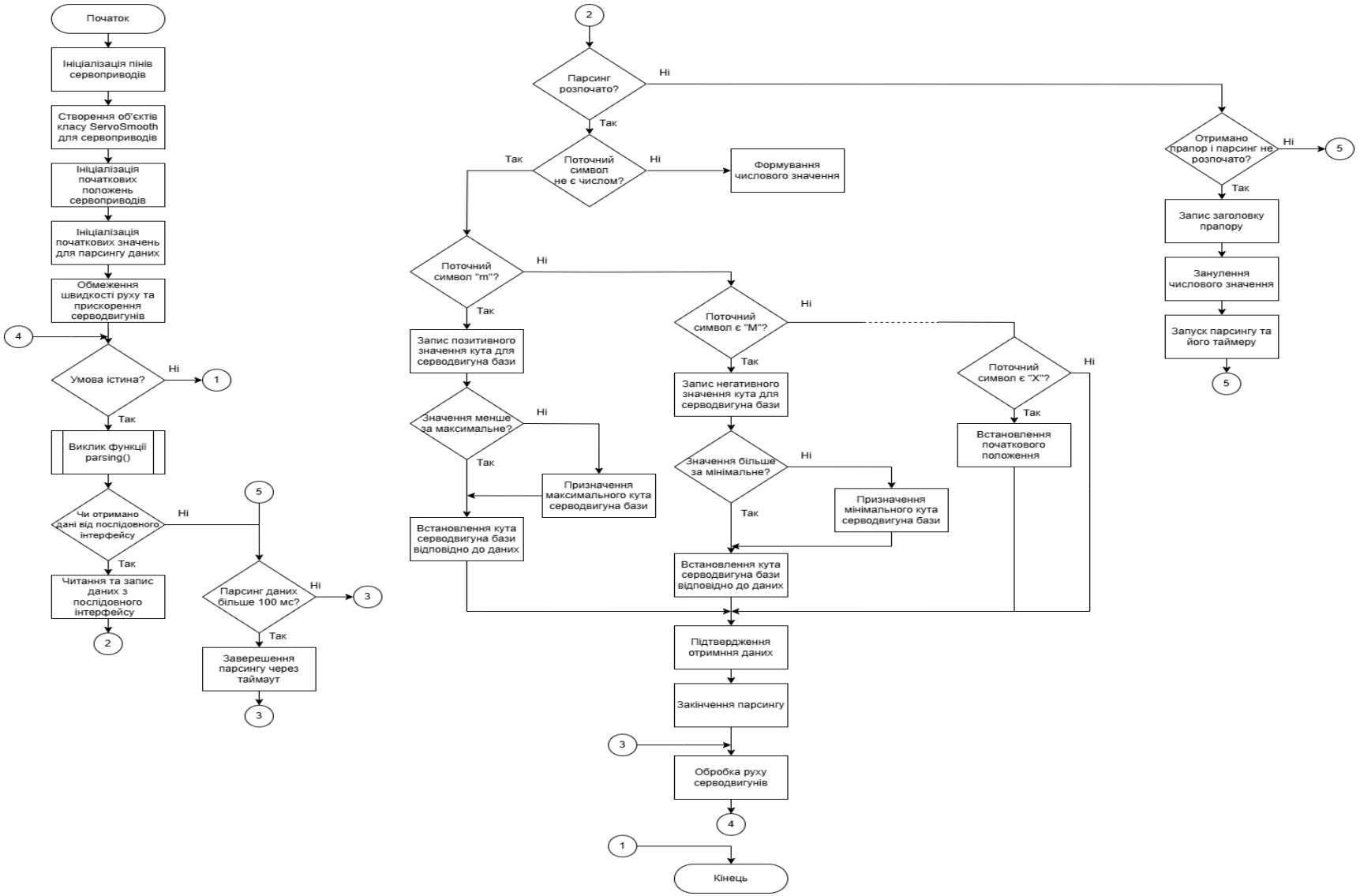


Рисунок 4.1 – Блок схема алгоритму роботи ПЗ плати керування

Блок схема алгоритму роботи ПЗ бездротового модулю ESP32-CAM наведена на рисунку 4.2.

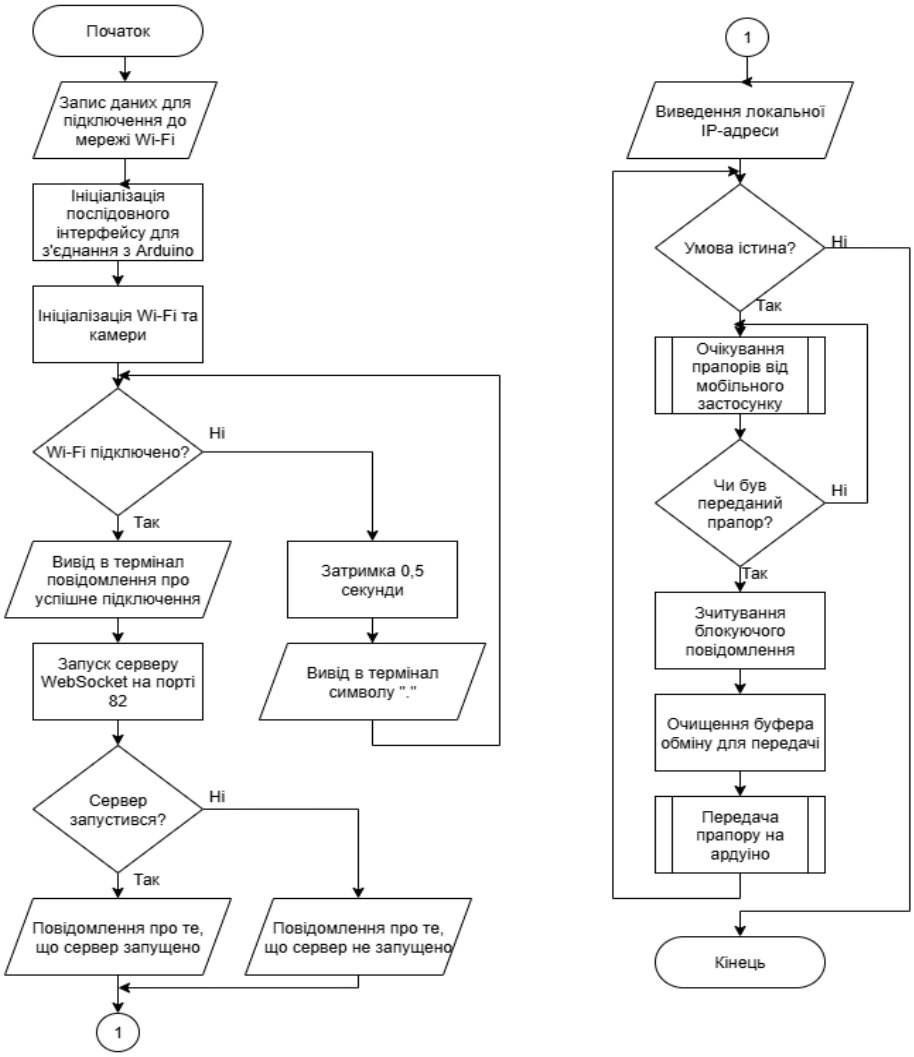


Рисунок 4.2 – Блок схема алгоритму роботи ПЗ бездротового модулю

4.2.3 Опис та обґрунтування вибору методів організації вхідних та вихідних даних

Формат команд керування згенерованих мобільним застосунком – текстовий, через протокол WebSocket.

Синтаксис: “{команда}{значення}”, де {команда} відповідає за серводвигун та напрямок руху, а {значення} відповідає за величину зміни кута.

Допустимі команди керування:

– m, поворот серводвигуна бази на +n градусів;

– M, поворот серводвигуна бази на -n градусів;

– l, поворот лівого серводвигуна на +n градусів;

– L, поворот лівого серводвигуна на -n градусів;

– r, поворот правого серводвигуна на +n градусів;

– R, поворот правого серводвигуна на -n градусів;

– g, поворот серводвигуна захоплювача на +n градусів;

– G, поворот серводвигуна захоплювача на -n градусів;

– c, поворот серводвигуна камери на +n градусів;

– C, поворот серводвигуна камери на -n градусів;

– X, скидання всіх серводвигунів у початкове положення.

Приклад:

– “m15”, рух маніпулятору бази на +15°;

– “M10”, рух маніпулятору бази на -10°.

Використання команд у текстовому форматі дозволяє забезпечити мінімальний розмір повідомлення, що важливо у системах керування в режимі реального часу. Літери команд робить протокол інтуїтивно зрозумілим, що не потребує додаткових знать та полегшує налагодження.

Формат передачі візуальної інформації – бінарний потік зображень JPEG (MJPEG) через протокол WebSocket.

Конфігурація технічних характеристик відеопотоку обрано з урахуванням обмежень пропускної здатності каналу:

– роздільна здатність відеопотоку 320x240 пікскелів (QVGA);

– формат стиснення JPEG з обмеженим параметром якості 80%.

Використання стиснення JPEG безпосередньо на ESP32-CAM дозволяє зменшити розмір переданих даних. Роздільна здатність 320x240 є оптимальним вибором для коректної роботи алгоритму розпізнавання об’єктів та плавності передачі відеопотоку в режимі реального часу з мінімальним навантаженням на мережу.

Для забезпечення оптимальної продуктивності на мобільних пристроях було проведено навчання моделі EfficientDet-Lite0. Процес підготовки моделі включав:

– вибір датасету з готовим набором класів для демонстрації можливостей розпізнавання;

– оптимізація архітектури базової моделі, що включала квантизацію ваг для зменшення розміру та відкидання незначущих зв’язків;

– навчання моделі з використанням датасету COCO для 128 зображень, впродовж 100 епох.

Після проведення навчання моделі був проведений аналіз динаміки навчання моделі (див. рисунок 4.3).

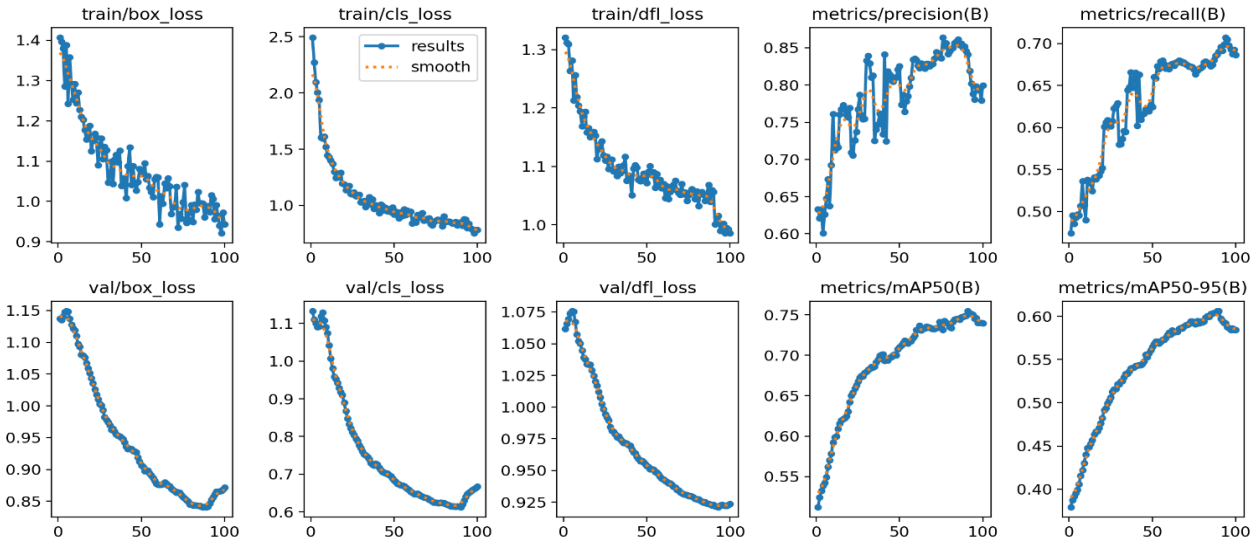


Рисунок 4.3 – Динаміка навчання моделі EfficientDet-Lite0

4.3 Опис розробленої програми

4.3.1 ПЗ плати керування Arduino Uno

4.3.1.1 Загальні відомості

Позначення програми: arduino_serial.

Найменування: програма керування серводвигунами маніпулятора.

Необхідне ПЗ для функціонування: бібліотека SmoothServo версії 3.9, Arduino Core.

ПЗ розроблено в середовищі Arduino IDE версії 2.3.4 з використанням мови C++. Плата керування відповідає за безпосереднє керування серводвигунами маніпулятору.

4.3.1.2 Функціональне призначення

Основне призначення ПЗ забезпечення позиціювання серводвигунів відповідно до команд керування, отриманих від бездротового модулю ESP32-CAM через послідовних інтерфейс UART.

Основні функції ПЗ:

– програмна прив’язка серводвигунів до виходів плати керування;

– декодування команд керування;

– контроль діапазону допустимих кутів руху для серводвигунів;

– генерація ШІМ-сигналів відповідно до отриманих команд.

4.3.1.3 Опис логічної структури

Структура програми складається з наступних функціональних блоків:

– функція void setup() відповідає за налаштування послідовного порту встановлення програмних параметрів (швидкість, прискорення, початкове та кінцеве положення) для серводвигунів та прив’язка їх до фізичних пінів мікроконтролера;

– функція void parsing() відповідає за декодування команд керування надісланих через послідовний порт та виконує вибір дії через використання оператора вибору switch-case;

– функція void loop() відповідає за безпосереднє виконання програмного коду у нескінченному циклі. Функції, які викликаються в середині цієї функції: void parsing та servo.tick(), що відповідає за оновлення поточного положення кожного з 5 серводвигунів.

4.3.1.4 Використовувані технічні засоби

При роботі програми використовуються наступні технічні засоби:

– мікроконтролер ATmega328P (Arduino Uno);

– драйвер серводвигунів PCA9685;

– бездротовий модуль ESP32-CAM;

– п’ять серводвигунів MG90S.

4.3.1.5 Виклик і завантаження програми

Програма завантажується в flash пам’ять мікроконтролера ATmega328P через USB інтерфейс. Точкою входу програми є функція loop(), що автоматично генерується в Arduino IDE. Програмний код використовує 7918 байт flash пам’яті (24%).

4.3.1.6 Вхідні та вихідні дані

В таблиця 4.1 наведений перелік вхідних та вихідних сигналів ПЗ плати керування.

Таблиця 4.1 – Вхідні та вихідні сигнали ПЗ плати керування

Інтерфейс	Джерело	Приймач	Призначення	Тип сигналу	Призначення сигналу
1	2	3	4	5	6
RX	ESP32-S	ATmega328P	Прийом команд керування	Дискретний	Вхідний
D6	ATmega328P	MG90S	Керування серводвигуном бази	Імпульсний	Вихідний
D7	ATmega328P	MG90S	Керування серводвигуном захоплювача	Імпульсний	Вихідний
D8	ATmega328P	MG90S	Керування лівим серводвигуном	Імпульсний	Вихідний
D9	ATmega328P	MG90S	Керування правим серводвигуном	Імпульсний	Вихідний
A5	ATmega328P	MG90S	Керування серводвигуном камери	Імпульсний	Вихідний

4.3.2 ПЗ бездротового модулю ESP32-CAM

4.3.2.1 Загальні відомості

Позначення програми: main_esp.

Найменування: мережевий шлюз з можливістю захоплення відео.

Необхідне ПЗ: бібліотеки ArduinoWebsockets версії 0.5.3, WiFi та esp_camera також менеджер плат ESP32 від Espressif Systems, версії 2.0.17.

ПЗ розроблено в середовищі Arduino IDE версії 2.3.4 з використанням мови C++. ПЗ адаптовано під потреби системи на базі прикладу «CameWebServer».

4.3.2.2 Функціональне призначення

ПЗ бездротового модулю ESP32-CAM забезпечує зв’язок між мобільним застосунком та платою керування, виступаючи у ролі мережевого шлюзу системи.

Основні функції ПЗ:

– ініціалізація та підтримка Wi-Fi з’єднання;

– налаштування оптимальних параметрів камери OV2640;

– ініціалізація веб-серверу для отримання запитів від мобільного застосунку;

– передача команд керування від мобільного застосунку до плати керування через послідовний інтерфейс UART;

– стиснення та передача відеопотоку з використанням протоколу WebSocket;

4.3.2.3 Опис логічної структури

Структура програми складається з наступних функціональних блоків:

– функція void setup() відповідає за налаштування послідовного інтерфейсу для передачі команд керування, модулю камери OV2640 (роздільна здатність, стиснення), ініціалізація Wi-Fi клієнта та запуск відео-

серверу. Функції, які викликаються в середині цієї функції: startCameraServer() запускає відео сервер, server.listen() встановлює порт для відео-серверу.

– функція void loop() відповідає за з'єднання по протоколу WebSocket у нескінченному циклі для отримання команд керування. Функції, які викликаються в середині цієї функції: WebsocketsClient.available() перевіряє чи доступний сервер, Serial1.println(msg.data()) надсилання команд керування через послідовний інтерфейс.

4.3.2.4 Використовувані технічні засоби

При роботі програми використовуються наступні технічні засоби:

– мікроконтролер ESP32-S;

– модуль камери OV2640;

– плата керування Arduino Uno.

4.3.2.5 Виклик і завантаження програми

Програма завантажується в flash пам'ять мікроконтролера ESP32-S через UART інтерфейс за допомогою USB-TTL перетворювача. Точкою входу програми є функція loop(), що автоматично генерується в Arduino IDE. Програмний код використовує 1713457 байт flash пам'яті (54%).

4.3.2.6 Вхідні та вихідні дані

В таблиця 4.2 наведений перелік вхідних та вихідних сигналів ПЗ бездротового модуля.

Таблиця 4.2 – Вхідні та вихідні сигнали ПЗ бездротового модуля

Інтерфейс	Джерело	Приймач	Призначення	Тип сигналу	Призначення сигналу
1	2	3	4	5	6
RX	Мобільний застосунок	ESP32-S	Прийом команд керування оператора	Радіочастотний	Вхідний
DVP	OV2640	ESP32-S	Отримання відепотоку	Цифровий	Вхідний

Закінчення таблиці 4.2

1	2	3	4	5	6
GPIO 14	ESP32-S	ATmega328 P	Передача декодованих команд	Дискретний	Вихідний
TX	ESP32-S	Мобільний застосунок	Трансляція відеопотоку	Радіочастотний	Вихідний

4.3.3 ПЗ мобільного застосунку

4.3.3.1 Загальні відомості

Позначення програми: manipulator_app.

Найменування: мобільний додаток керування маніпулятором.

Необхідне ПЗ для функціонування: фреймворк Jetpack Compose, модуль ML Kit, OkHttp, Android SDK (API level 24+).

ПЗ розроблено в середовищі Android Studio версії 2025.1.1 з використанням мови Kotlin.

4.3.3.2 Функціональне призначення

ПЗ мобільного застосунку являє собою інтерфейс взаємодії оператора з системою. Основне призначення ПЗ забезпечення зручного керування роботою маніпулятора з можливістю відображення результатів розпізнавання об’єктів у режимі реального часу.

Основні функції ПЗ:

– генерація команд керування маніпулятором;

– візуалізація інформації про розпізнані об’єкти;

– відображення відеопотоку в режимі реального часу.

Для інтеграції алгоритмів КЗ з мобільним застосунком був використаний модулю ML Kit, що попередньо навчену модель, дозволяє інтегрувати в мобільний застосунок.

Основні функції модуля ML Kit:

– розпізнавання декількох об’єктів у відеопотоці;

– визначення координат об’єкта та обмеження об’єкта рамками;

– класифікація з рівнем впевненості;

– фільтрація результатів з порогом точності більше 50%;

– оптимізація роботи на мобільних пристроях.

4.3.3.3 Опис логічної структури

Клас WsManipulator виконує наступні функції:

– функція connect() відповідає за встановлення з’єднання з сервером ESP32 з використанням протоколу WebSocket;

– функція calcServosDegreesForPoint() відповідає за тригонометричні розрахунки кінематики руху маніпулятора;

– функції moveForwardBackwardDelta(), moveUpDownDelta() відповідають за виклик кінематичного розрахунку і оновлення стану відповідних серводвигунів.

Клас WsServo виконує наступні функції:

– функція set() відповідає за перевірку, нового значення за допустимі межі, та визначає, чи перевищила зміна порогове значення для відправки команди;

– функція get() відповідає за отримання значення дельти (різниці) між поточним та попереднім станом для формування команд керування.

4.3.3.4 Використовувані технічні засоби

При роботі програми використовуються наступні технічні засоби:

– смартфон Android;

– GPU для прискорення ML операція;

– модуль Wi-Fi.

4.3.3.5 Виклик і завантаження програми

Програма завантажується в пам’ять смартфону через USB інтерфейс з Android Studio або через встановлення APK файлу безпосередньо на

смартфоні. Точкою входу програми є функція onCreate(), що автоматично генерується в Android Studio.

4.3.3.6 Діаграма класів

На рисунку 4.4 наведена взаємодія класів WsManipulator та WsServo мобільного застосунку.

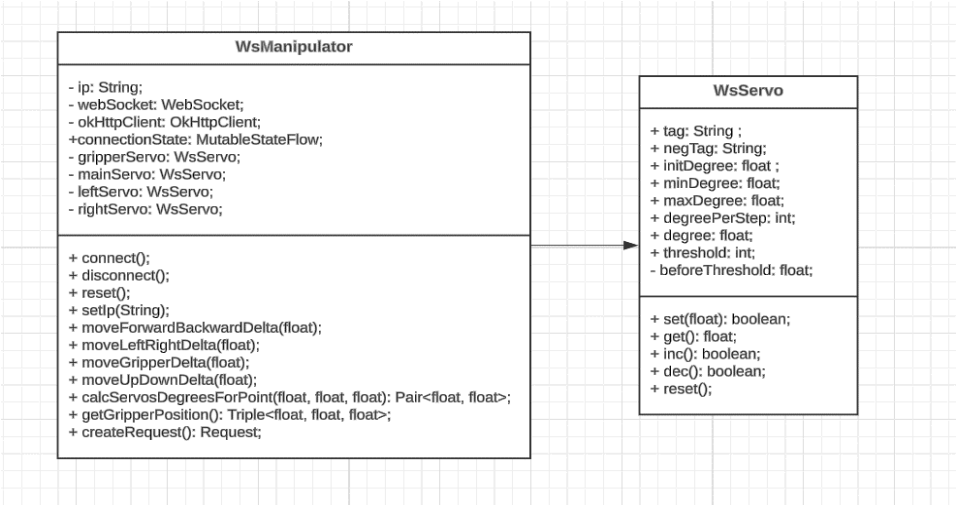


Рисунок 4.4 – Діаграма класів мобільного застосунку

4.4 Вікна мобільного застосунку

Початкове вікно мобільного застосунку програми містити два пункти меню: «Manual Control», що відповідає за безпосереднє керування маніпулятором з передачею відеопотоку в реальному часі та «Object Detection», що відповідає за виявлення об’єктів з виведення інформації про ці об’єкти.

Для з’єднання з маніпулятором, необхідно ввести IP-адресу модулю ESP32-CAM та натиснути кнопку Save (див. рисунок 4.5).

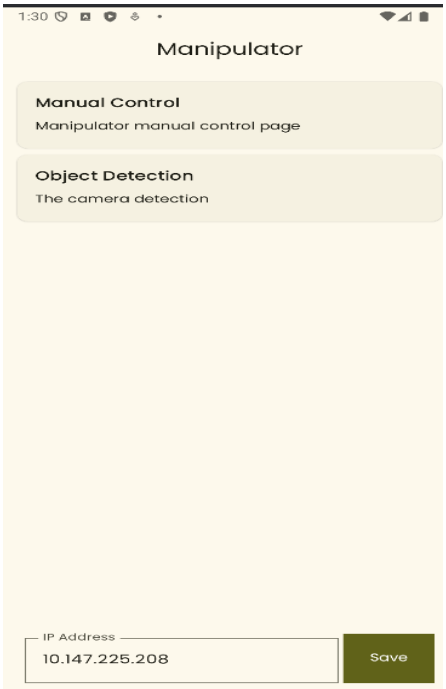


Рисунок 4.5 – Початкове вікно мобільного застосунку

Вікно пункту меню «Manual Control» зображене на рисунку 4.6. При невдалому підключені до маніпулятору, кнопки керування будуть не активні. Для повторного підключення до маніпулятору використовується іконка маніпулятору.

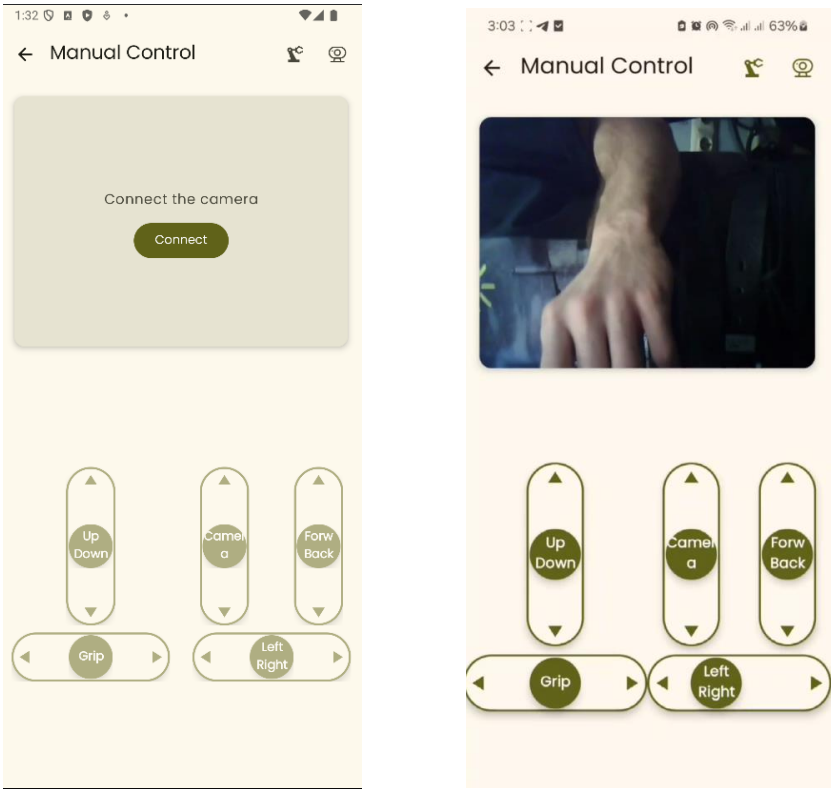


Рисунок 4.6 – Вікно «Manual Control»

Вікно «Manual Screen» складається з:

– джойстик (Up\Down) здійснює переміщення захоплювачу вгору та вниз;

– джойстик (Forw\Back) здійснює переміщення захоплювачу вперед та назад;

– джойстик (Grip) здійснює захоплення та утримання захоплювачу;

– джойстик (Left\Right) здійснює переміщення захоплювачу вліво та вправо;

– джойстик (Camera) здійснює нахил камери вгору та вниз;

– віджет камери здійснює передачу відеопотоку, захопленого з модулю камери;

– іконка повторного підключення до маніпулятору;

– іконка повторного підключення до бездротового модулю;

– кнопка підключення до бездротового модулю (коли не підключено);

– кнопка повернення назад.

Вікно пункту меню «Object Detection» зображене на рисунку 4.7. При невдалому підключені до камери, віджет камери буде порожній. Для повторного підключення до камери використовується іконка камери.

Якщо було виявлено об’єкт, його буде обведено в рамку, а під віджетом камери буде виведена інформація про клас об’єкту та піктограма відповідно до класу. Натискаючи на піктограми, можна виділяти об’єкт (виділений об’єкт буде поміченим червоним).

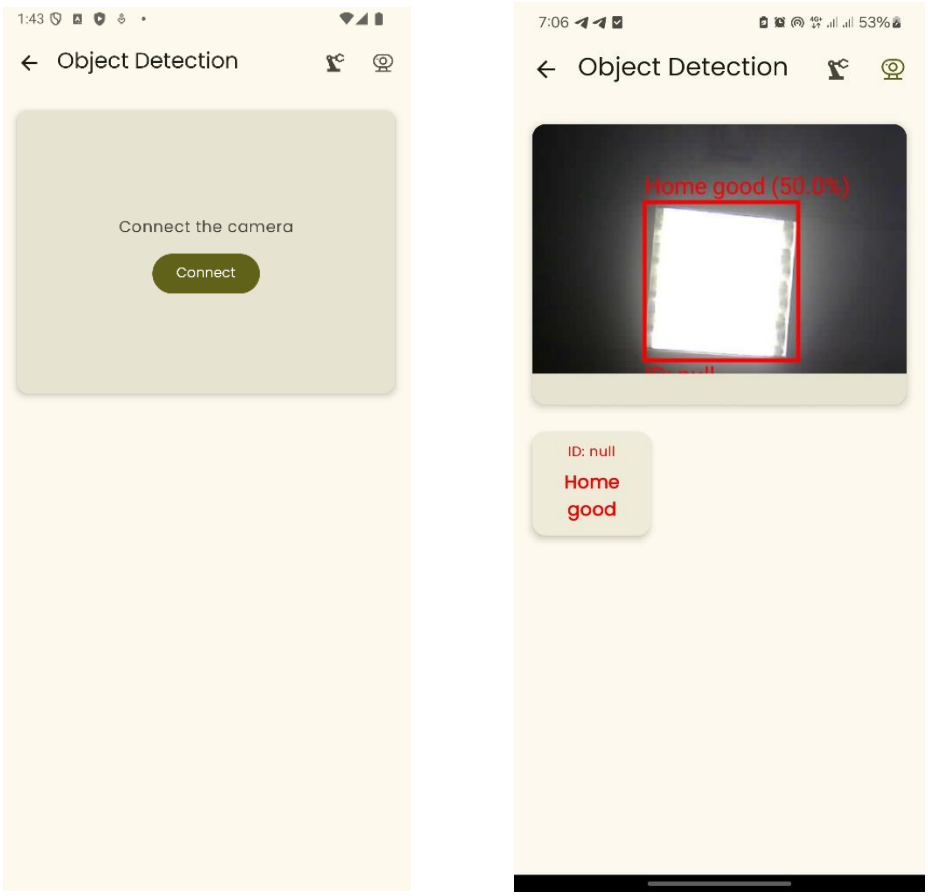


Рисунок 4.7 – Вікно «Object Detection»

4.5 Очікувані техніко-економічні показники

Розроблене програмне забезпечення КФС віддаленого керування маніпулятором демонструє значні переваги порівняно з існуючими комерційними аналогами, забезпечуючи оптимальне співвідношення функціональності, продуктивності та економічної ефективності.

Вартість впровадження системи станом на 01.12.2025 року на маркетплейсі AliExpress:

– набір Keyestudio KS019x – 2172,99 грн;

– бездротовий модуль ESP32-CAM – 298,47 грн;

– серводвигуни MG90S (5 шт.) – 460,10 грн;

– блок живлення MN-60-12V-5A – 235,32 грн.

Загальна вартість апаратних компонентів системи – 3166,88 грн.

4.6 Висновки до розділу

У даному розділі виконана розробка ПЗ для КФС віддаленого керування маніпулятором. Розроблене ПЗ складається з трьох взаємопов’язаних модулів, в межах розподіленої архітектури системи: ПЗ плати керування, ПЗ бездротового модулю та ПЗ мобільного застосунку.

ПЗ плати керування Arduino Uno виконує безпосереднє керування серводвигунами. Алгоритм для парсингу текстових команд отриманих від бездротового модулю дозволяє ефективно декодувати отримані команди з незначними часовими затримками.

ПЗ бездротового модулю ESP32-CAM виконує функції мережевого шлюзу між платою керування маніпулятора та мобільним пристроєм оператора, забезпечуючи передачу відеопотоку через WebSocket та передачу команд керування.

ПЗ мобільного застосунку виконує функції інтерфейсу керування оператора. Інтеграція навченої моделі EfficientDet-Lite0 через модуль ML Kit забезпечує розпізнавання об’єктів безпосередньо на мобільному пристрої без необхідності використання хмарних обчислень.

Розроблене ПЗ відповідає сформульованим технічним вимогам та готове до проведення експериментальних досліджень для підтвердження ефективності системи.

5 ЕКСПЕРИМЕНТАЛЬНИЙ РОЗДІЛ

5.1 Мета і завдання експерименту

Мета: експериментальна перевірка працездатності розробленої системи та перевірка відповідності фактичних показників, технічним вимогам сформованих у розділі 3.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– дослідити вплив рівня освітленості робочої зони на точність розпізнавання об’єктів алгоритмами КЗ;

– дослідити точність розпізнавання об’єктів з використанням метрик машинного навчання для кожного рівня освітлення;

– встановити мінімальний рівень освітлення, за якого система забезпечує прийнятну точність розпізнавання;

– проаналізувати результати експериментальних досліджень з теоретичними очікуваннями (розділ 2.6) та технічними вимогами (розділ 3.2.2).

Об’єкт дослідження: точність розпізнавання об’єктів мобільного застосунку з інтегрованою моделлю EfficientDet-Lite0.

Предмет дослідження: вплив рівня освітленості на якість розпізнавання об’єктів з датасету COCO.

5.2 Методика проведення експерименту

Експериментальні дослідження проводились на базі розробленої системи та з використанням наступного апаратно-програмного забезпечення:

– світлодіодної лампи з можливістю регулювання діапазону освітлення;

– бездротового модулю ESP32-CAM з модулем камери OV2640;

– емулятору телефону Google Pixel 4 в Android Studio;

– мобільного застосунку manipulator_app з фреймворком Google ML Kit 17.0.2;

– системи логування Android Logcat;

– нейронної мережі EfficientDet-Lite0;

– мобільної точки доступу стандарту 802.11n (2,4 ГГц) на телефоні Samsung S23 та мобільним оператором Lifecell;

– мобільним застосунком Lux Light Meter (версія 036.2024) для вимірювання освітлення.

Послідовність виконання експерименту складається з чотирьох етапів.

Етап 1. Підготовка стенду:

1. Закріпити ESP32-CAM на штативі на відстані 30 см від робочої зони;

2. Розмістити лампу на штативі на висоті 50 см та 30 см від робочої зони;

3. Підключити ESP32-CAM до живлення та мобільної точки доступу;

4. Запустити мобільний застосунок на емуляторі в Android Studio;

5. Підключитись до ESP32-CAM з мобільного застосунку.

Етап 2. Визначення рівнів освітлення:

1. Встановити мобільний пристрій із застосунком Lux Light Meter безпосередньо в робочій зоні;

2. Ввімкнути лампу на перший рівень роботи;

3. Зафіксувати показник рівня освітленості.

Етап 3. Проведення вимірювань:

1. Відкрити систему логування Android Logcat та поставити її на паузу;

2. Розмістити об’єкт в робочій зоні;

3. Відновити логування протягом 5 секунд;

4. Зупинити логування та скопіювати результати розпізнавання;

5. Змінити об’єкт та повторити кроки 1-4;

6. Змінити рівень освітлення та повторити кроки 1-5.

Етап 4. Аналіз даних:

1. Структурувати дані логування для кожного рівня освітлення та предмету;

2. Розрахувати метрики (Precision, Recall, F1-score) для кожного рівня;

3. Побудувати графіки залежності рівня освітленості та точності розпізнавання;

4. Провести аналіз фактичних та теоретичних даних.

5.3 Вимоги до експерименту

5.3.1 Вимоги до вимірювальної апаратури

Програма вимірювання освітленості повинна покривати діапазон від 0 до 1000 люкс.

Похибка вимірювання освітленості не повинна перевищувати ±10%.

Модуль камери бездротового модулю повинен передавати відеопотік з роздільною здатністю не менше 320x240 пікселів.

Частота передачі кадрів відеопотоку повинна бути більшою за 20.

5.3.2 Вимоги до об’єктів розпізнавання

Тестові об’єкти для розпізнавання повинні:

– належати до різних класів з датасету COCO;

– забезпечувати різний рівень складності розпізнавання;

– мати різну геометрію та оптичні властивості;

– включати один об’єкт, що не входить до датасету, для перевірки роботи класифікатора на невідомих об’єктах.

5.3.3 Вимоги до експериментального стенду

Конструкція експериментального стенду повинна:

– бути стабільно зафіксованою виключаючи фактор вібрації;

– бути розміщена на відстані 30 см від робочої зони;

– передбачати фіксацію бездротового модулю ESP32-CAM паралельно до робочої зони.

Освітлення експериментального стенду повинно:

– виключати наявність сторонніх джерел штучного та природного освітлення;

– забезпечувати наявність декількох режимів освітлення.

Робоча зона для розпізнавання об’єктів повинна бути світлою та з матовим покриттям, для мінімізації відблисків та контрасту.

5.3.4 Вимоги до фіксації результатів дослідження

Параметри виявленого об’єкта повинні складатись з:

– класу об’єкта;

– рівня впевненості (%);

– координатної рамки (x, y, width, height);

– результату класифікації (TP/FP/FN/TN);

Результати експерименту повинні фіксуватись у таблиці, структура якої наведена у таблиці 5.1.

Таблиця 5.1 – Фіксація даних експериментальних досліджень

№	Рівень освітленості	Об’єкт	Виявлений клас	Об’єкт з датасету	Впевненість, %	Результат
1	2	3	4	5	6	7
1						

5.3.5 Критерії оцінювання результатів дослідження

Критерії оцінювання результаті дослідження повинні включати:

– TP (True Positive), об'єкт правильно розпізнано (виявлений клас співпадає з еталонним об’єктом);

– FP (False Positive), помилкове спрацювання (виявлений клас не співпадає з еталонним об’єктом або відсутній у датасеті);

– FN (False Negative), пропуск об'єкта (об’єкт є у датасеті, але не розпізнаний);

– TN (True Negative), правильно не розпізнано (об’єкта немає в датасеті).

Кількість правильних розпізнавань серед усіх об’єктів знаходиться за формулою (5.1):

$$Precision = \frac{TP}{(TP+FP)} \quad (5.1)$$

Кількість знайдених об’єктів серед усіх об’єктів знаходиться за формулою (5.2):

$$Recall = \frac{TP}{(TP+FN)} \quad (5.2)$$

Точність розпізнавання моделі (F1-score) знаходиться за формулою(5.3):

$$F1 = 2 * \frac{(P*R)}{(P+R)} \quad (5.3)$$

Відповідно до вимоги пункту розділу 3.2.2 F1-score повинен бути ≥72%.

5.4 Результати експерименту

5.4.1 Сутність експерименту

Дослідження проводились у офісному приміщенні площею 18 м^2 у період з 7 по 9 грудня 2025 року об 18:00 та до 19:30. Природнє освітлення – відсутнє, штучне освітлення – вимкнено. Експеримент виконувався одноосібно.

В таблиці 5.2 наведений перелік тестових об’єктів відібраних для перевірки розпізнавання.

Таблиця 5.2 – Перелік тестових об’єктів

№	Об’єкт	Клас датасету	Складність розпізнавання
1	2	3	4
1	Чашка	cup	Середня
2	Банан	banana	Низька
3	Телефон	cell phone	Середня
4	Монета	coin	Висока
5	Флешка	–	–

Параметри рівня освітлення робочої зони діодною лампою, наведено в таблиці 5.3.

Таблиця 5.3 – Рівень освітлення робочої зони

Рівень	Освітленість (люкс)	Режим лампи	Примітка
1	2	3	4
1	30	Димер 10%	Напівтемрява, високий рівень шумів
2	185	Димер 40%	Помірне освітлення
3	325	Димер 70%	Нормальне робоче освітлення
4	445	Димер 100%	Високий рівень освітлення

Для логування інформації про об’єкт використаний наступний фрагмент програмного коду в класі ObjectDetectionScreen:

```
  private val TAG = "ML_Kit_Detection"

    fun processDetectionResults(
        detectedObjects: List<DetectedObject>,
        timestamp: Long
    ) {
        detectedObjects.forEach { obj ->
            val boundingBox = obj.boundingBox
            val labels = obj.labels

if (objects.isEmpty()) {
                                // Об'єкти НЕ знайдено
                                Log.w(TAG_EXPERIMENT, """
                                    ═══════════════════════════════════════
                                    Frame #$frameCounter
                                    Timestamp: ${formatTime(startTime)}
                                    Processing Time: ${processingTime}ms

                                    ⚠️ NO OBJECTS DETECTED
                                    ═══════════════════════════════════════
                                """.trimIndent())
                                // CSV формат для Excel
                                Log.i(TAG_CSV,
"$startTime,$frameCounter,NONE,0.00,0,0,0,0,$processingTime")
                            } else {
                                // Об'єкти ЗНАЙДЕНО
                                objects.forEachIndexed { index, obj ->
                                    val label = obj.labels.firstOrNull()
                                    val bounds = obj.boundingBox

                                    if (label != null) {
                                        Log.d(TAG_EXPERIMENT, """
                                            ═══════════════════════════════════════
                                            Frame #$frameCounter | Object ${index +
1}/${objects.size}

                                            Timestamp: ${formatTime(startTime)}
                                            Processing Time: ${processingTime}ms

                                            📦 Object Info:
                                               - Class: ${label.text}
                                               - Confidence: ${String.format("%.2f",
label.confidence * 100)}%

                                               - Tracking ID: ${obj.trackingId ?: "N/A"}

                                            📐 Bounding Box:
                                               - Left: ${bounds.left}
                                               - Top: ${bounds.top}
                                               - Width: ${bounds.width()}
                                               - Height: ${bounds.height()}
                                               - Center: (${bounds.centerX()},
${bounds.centerY()})
                                            ═══════════════════════════════════════
```

```
                                    """.trimIndent())            } else {
                // Об'єкт не розпізнано
                Log.w(TAG, "No objects detected at ${formatTime(timestamp)}")
            }
        }
    }

    private fun formatTime(millis: Long): String {
        val sdf = SimpleDateFormat("yyyy-MM-dd HH:mm:ss.SSS", Locale.getDefault())
        return sdf.format(Date(millis))
    }
```

На рисунку 5.1 наведена фотографія стенду, який було використано під час експериментальних досліджень.

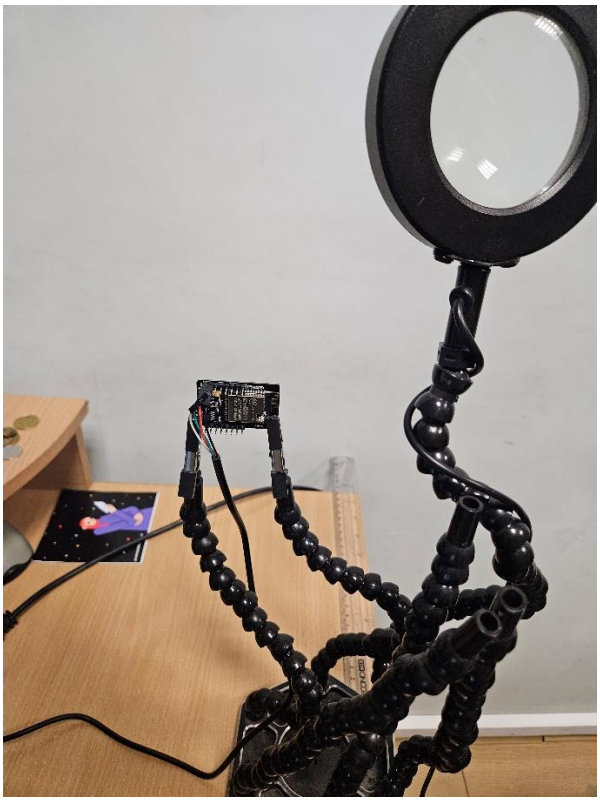


Рисунок 5.1 – Експериментальний стенд

На рисунку 5.2 наведена фотографія під час фіксування трьох рівнів освітленості робочої зони: без освітлення, перший режим та четвертий режим.

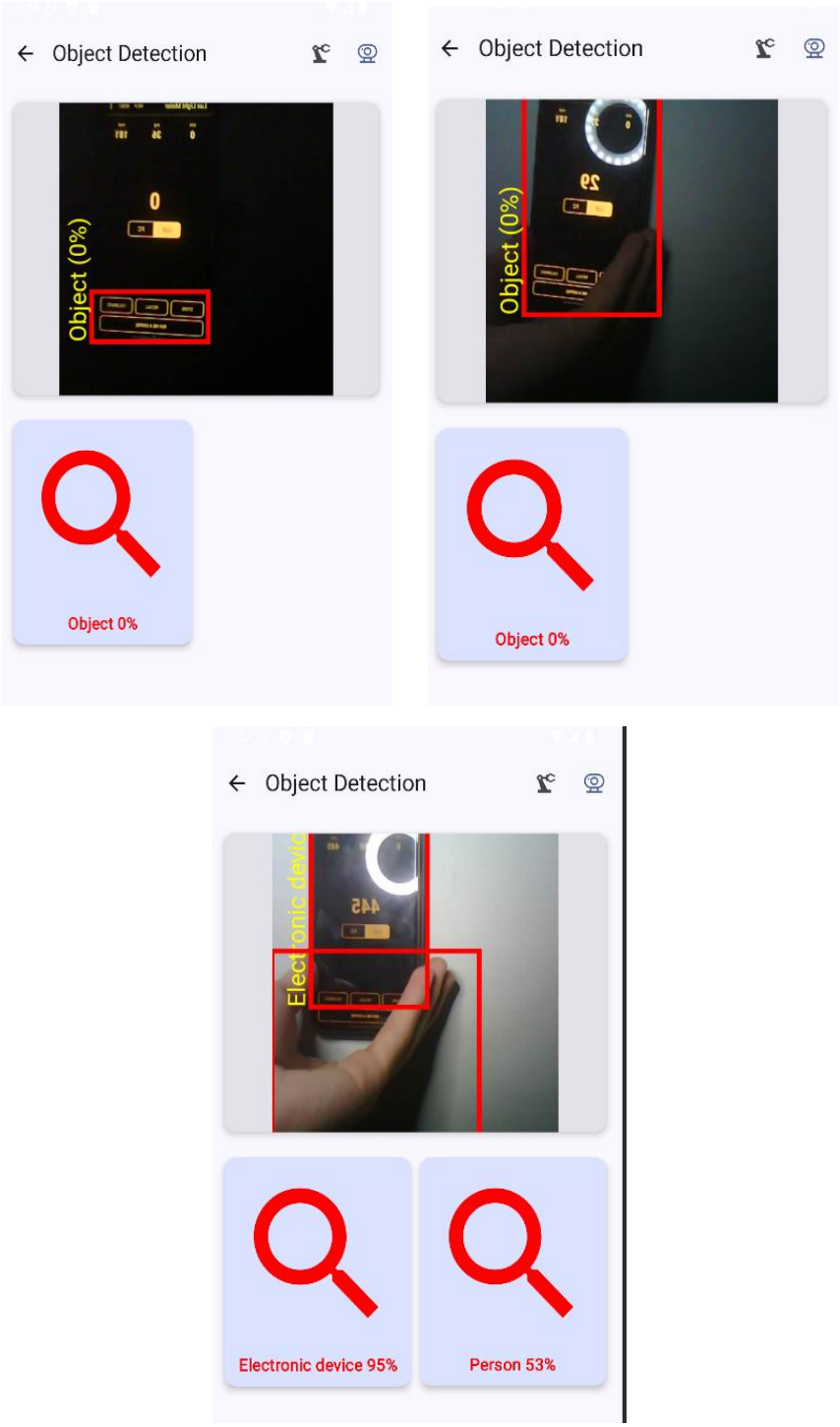


Рисунок 5.2 – Заміри рівня освітленості робочої зони

На рисунку 5.3 наведена фотографія під час розпізнавання чашки з впевненістю 94% в мобільному застосунку, при третьому рівні освітлення.

Рисунок 5.3 – Розпізнавання об’єкта чашка з впевненістю 94%

На рисунку 5.4 наведена фотографія логування об’єкту чашка з впевненістю 70%, при третьому рівні освітленні.

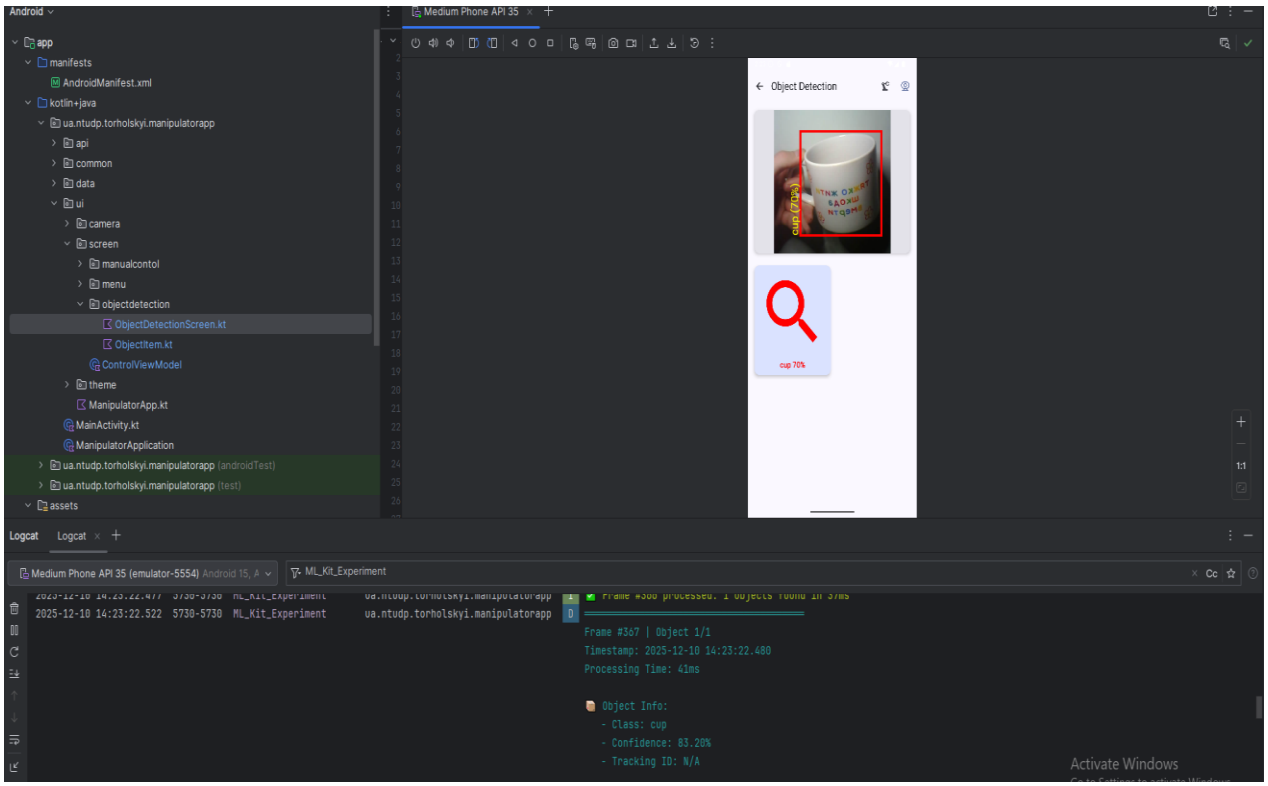


Рисунок 5.4 – Розпізнавання об’єкта чашка

Хід експерименту:

1. Виконано калібрування люксметра Lux Light Meter;

2. Налаштовано світлодіодну лампу для забезпечення першого рівня освітленості;

3. Послідовно розміщено 5 тестових об'єктів (чашка, банан, телефон, монета, флешка) на відстані 30 см від ESP32-CAM;

4. Для кожного об'єкта виконано розпізнавання протягом 5 секунд з фіксацією результатів через систему логування Android Logcat;

5. Змінено рівень освітленості на наступний та повторено пункт 3-4;

6. Після завершення 20 вимірювань дані занесено у таблицю.

5.4.2 Результати експерименту в цифрах і фактах

Отримані результати зведено в таблицю 5.4. Середній показник впевненості розпізнавання наведено у відсотках.

Таблиця 5.4 – Залежність точності розпізнавання від рівня освітленості

№	Рівень освітленості	Об’єкт	Виявлений клас	Об’єкт з датасету	Впевненість, %	Результат
1	2	3	4	5	6	7
1	1	Чашка	tableware	cup	77	FP
2	1	Банан	banana	banana	82	TP
3	1	Телефон	(не розпізнано)	cell phone	0	FN
4	1	Монета	(не розпізнано)	coin	0	FN
5	1	Флешка	(не розпізнано)	(відсутній)	0	TN
6	2	Чашка	tableware	cup	69	FP
7	2	Банан	banana	banana	100	TP
8	2	Телефон	electrical device	cell phone	85	FP
9	2	Монета	(не розпізнано)	coin	0	FN
10	2	Флешка	toy	(відсутній)	60	FP

Закінчення таблиці 5.4

1	2	3	4	5	6	7
11	3	Чашка	cup	cup	81	TP
12	3	Банан	banana	banana	100	TP
13	3	Телефон	electrical device	cell phone	89	FP
14	3	Монета	coin	coin	75	TP
15	3	Флешка	electrical device	(відсутній)	57	FP
16	4	Чашка	cup	cup	92	TP
17	4	Банан	banana	banana	100	TP
18	4	Телефон	cell phone	cell phone	82	TP
19	4	Монета	coin	coin	78	TP
20	4	Флешка	electrical device	(відсутній)	72	FP

Отримані дані дають змогу розрахувати метрики машинного навчання. Для першого режиму освітлення:

– кількість успішно розпізнаних об’єктів (TP) з датесету дорівнє одному (банан);

– кількість не правильно розпізнаних об’єктів (FP) з датасету дорівнює одному (чашка, як tableware);

– кількість не розпізнаних об’єктів (FN) з датасету дорівнює двом (телефон та монета);

– кількість розпізнаних об’єктів, які не входять до датасету (TN) дорівнює одному (флешка).

Кількість правильних розпізнавань серед всіх об’єктів знаходиться за формулою 5.1:

$$Precision = \frac{TP}{(TP + FP)} = \frac{1}{(1 + 1)} = 50\%$$

Кількість знайдених об’єктів серед всіх об’єктів знаходиться за формулою 5.2:

$$Recall = \frac{TP}{(TP + FN)} = \frac{1}{(1 + 2)} = 33{,}3\%$$

Загальна точність розпізнавання моделі знаходиться за формулою 5.3:

$$F1 = 2 * \frac{(P * R)}{(P + R)} = 2 * \frac{(0{,}50 * 0{,}33)}{(0{,}50 + 0{,}33)} = 40\%$$

Зведені показники метрик точності розпізнавання для кожного рівня освітлення наведено в таблиці 5.5.

Таблиця 5.5 – Зведені метрики точності розпізнавання

Рівень освітленості	TP	FP	FN	Precision, %	Recall, %	F1-score, %
1	2	3	4	5	6	7
1	1	1	2	50,0	33,3	40,0
2	1	3	1	25,0	50,0	33,3
3	3	2	0	60,0	100,00	75,0
4	4	1	0	80,0	100,00	88,9

На рисунку 5.5 наведено графік залежності F1-score від рівня освітлення.

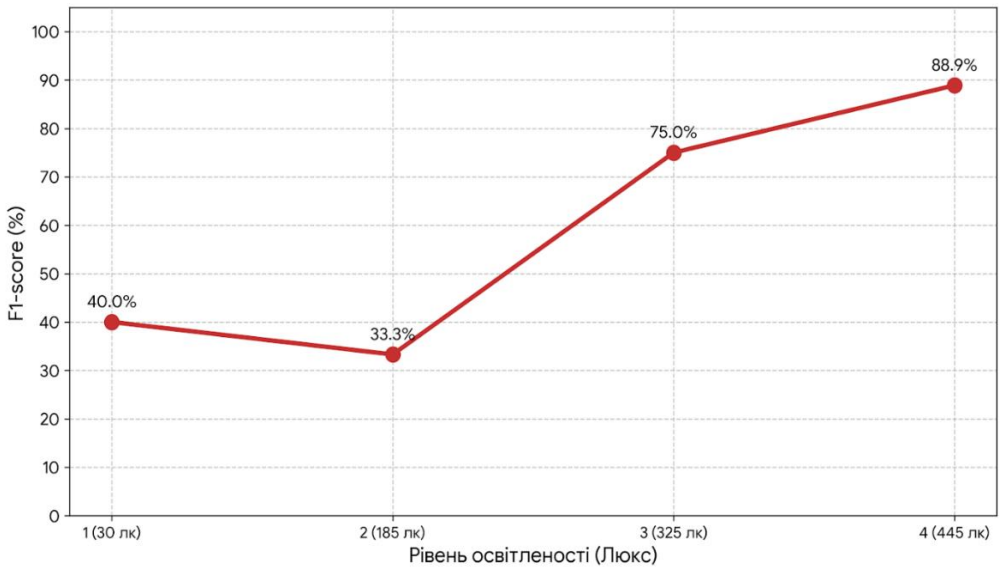


Рисунок 5.5 – Залежність F1-score від рівня освітлення

Графік залежності демонструє, що:

– при першому рівні освітлення система непрацездатна (нижче вимоги 72%);

– при другому рівні освітлення точність розпізнавання погіршилась через збільшення FP;

– при третьому рівні освітлення система демонструє допустимий рівень прийнятності (>72%);

– при четвертому рівні освітлення система демонструє оптимальні умови (+23% вище вимоги).

5.4.3 Аналіз відповідності теоретичних та експериментальних досліджень

В таблиці 5.6 наведене порівняння теоретичних вимог та експериментальних результатів.

Таблиця 5.6 – Порівняння теоретичних вимог та експериментальних результатів

Параметр	Теоретичне значення	Експериментальний результат
1	2	3
Точність розпізнавання при оптимальному рівня освітлення (F1-score)	≥72% (вимога 3.2.2)	75-89% (3 та 4 режими освітлення)
Мінімальне допустиме освітлення для роботи системи	300 люкс	350 люкс

5.4.4 Характеристика новизни результатів

Науково-практична новизна дослідження:

– експериментально встановлено пороговий рівень освітленості для стабільної роботи мобільної системи розпізнавання об'єктів на базі EfficientDet-Lite0 + ML Kit;

– кількісна залежність освітленості від площі об'єкта для EfficientDet раніше не досліджувалась (для об’єктів площа яких <10 см^2 потрібне додаткове освітлення);

– підтверджено працездатність гібридної архітектури (ESP32-CAM + мобільний пристрій + EfficientDet-Lite0) для задач розпізнавання у режимі реального часу з точністю >88% при стандартних умовах.

5.5 Висновки до розділу

Експериментально підтверджено працездатність розробленої КФС віддаленого керування маніпулятора з алгоритмом КЗ на базі нейронної моделі EfficientDet-Lite0 та фреймворку Google ML Kit 17.0.2.

Середня точність розпізнавання реальних об’єктів при 3 та 4 режимах освітлення становить F1-score = 75.0-88.9%, що відповідає та перевищує технічні вимоги (≥72%, розділ 3.2.2) на 3-17%.

Експериментально встановлено пороговий рівень освітленості ~300-350 люкс, нижче якого точність падає до неприйнятних значень (33.3-40.0%). При режимах 1 та 2 (30-185 люкс) система демонструє F1-score нижче 72%, що робить її непридатною для практичного застосування за умов недостатнього освітлення.

Система придатна для практичного застосування у навчальних лабораторіях та дослідницьких проектах за умови забезпечення освітлення робочої зони не менше 325-350 люкс (режим 3-4 за таблицею 5.3).

ВИСНОВКИ

Кваліфікаційна робота є завершеною науковою роботою, в якій вирішена науково-практична задача розробки та дослідження кіберфізичної системи віддаленого керування маніпулятором із використанням технології комп'ютерного зору. За результатами проведених досліджень та розробок можна зробити наступні висновки:

1. Проведено аналіз стану розвитку кіберфізичних систем та методів інтеграції із технологіями комп'ютерного зору. Систематизовано напрямки досліджень у галузі інтеграції комп'ютерного зору в роботизовані системи.

2. Ідентифіковано ключові проблеми сучасних систем інтеграції комп'ютерного зору в кіберфізичні системи:

– високі вимоги до обчислювальних ресурсів для обробки відеопотоку у реальному часі;

– зниження точності розпізнавання за умов нестабільного освітлення;

– мережеві затримки при передачі відеопотоку;

– обмежені можливості мобільного керування.

3. Досліджено апаратні та програмні платформи для створення системи керування маніпулятором.

4. Розроблено архітектуру кіберфізичної системи з передачею відеопотоку та розпізнаванням об'єктів.

5. Створено методи та алгоритми обробки відеопотоку для розпізнавання об'єктів у режимі реального часу.

6. Розроблено програмне забезпечення для керування маніпулятором із мобільного пристрою. Застосунок забезпечує:

– перегляд відеопотоку з камери маніпулятора у реальному часі з затримкою 80-120 мс;

– візуалізацію результатів розпізнавання об'єктів з обмежуючими рамками та класифікацією;

– інтуїтивне керування маніпулятором через зручний інтерфейс;

7. Проведено експериментальні дослідження ефективності системи розпізнавання об'єктів.

8. Визначено перспективи подальшого розвитку системи:

– впровадження 3D комп'ютерного зору з використанням стереокамер або структурованого світла для точнішого визначення положення об'єктів у просторі;

– інтеграція технологій reinforcement learning для адаптивного навчання маніпулятора оптимальним траєкторіям захоплення;

– розширення функціоналу мобільного застосунку.

У роботі вирішено поставлені завдання та досягнута мета дослідження. Розроблено функціональну кіберфізичну систему віддаленого керування маніпулятором з інтегрованим комп'ютерним зором, яка демонструє прийнятну точність розпізнавання (75-89%) за прийнятної вартості реалізації (3167 грн). Система придатна для практичного впровадження у виробничих умовах та може слугувати основою для подальших досліджень у напрямку інтелектуалізації промислових роботизованих систем.

Результати роботи підтверджують доцільність використання гібридних підходів до інтеграції комп'ютерного зору в кіберфізичні системи та демонструють можливість створення ефективних рішень на базі доступних апаратних платформ та відкритих програмних інструментів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Рубан, О. А. Використання роботизованих систем у логістиці та навантажувально-розвантажувальних процесах / О. А. Рубан, І. О. Монат, В. В. Зражевська // Scientific trends: history, development and existing problems : матеріали XVI Міжнар. наук.-практ. конф. (Краків, Польща, 21–23 квіт. 2025 р.). Краків, 2025. 195 с.

2. Зарубін, І. С. Ефективність використання роботизованих систем у виробництві / І. С. Зарубін, С. В. Сотник // Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки 2024 : матеріали І Всеукр. конф. (Харків, 16–17 трав. 2024 р.). Харків, 2024. С. 45–48.

3. Robotics Statistics and Trends 2024 [Електронний ресурс] // AIPRM. URL: https://www.aiprm.com/en-gb/robotics-statistics/ (дата звернення: 11.11.2025).

4. Kocsi, A. Deep Learning-Based Visual Servoing of Industrial Robots for Object Manipulation [Електронний ресурс] / A. Kocsi, P. Koren, T. Szalay // Applied Sciences. 2025. Vol. 15, No. 14. Article 7905. URL: https://www.mdpi.com/2076-3417/15/14/7905 (дата звернення: 06.11.2025).

5. Устимчук, Д. О. Конструкторсько-технологічне забезпечення виготовлення основи маніпулятора [Електронний ресурс] : дипломний проєкт бакалавра / Д. О. Устимчук ; Нац. техн. ун-т України «КПІ ім. Ігоря Сікорського». К. URL: https://ela.kpi.ua/server/api/core/bitstreams/4ec930c0-a5ad-49d7-8008-3768890e20cd/content (дата звернення: 07.10.2025).

6. Adam, G. K. Embedded Microcontroller with a CCD Camera as a Digital Lighting Control System [Електронний ресурс] / G. K. Adam, P. A. Kontaxis, L. T. Doulos [та ін.] // Electronics. 2019. Vol. 8, No. 1. Article 33. URL: https://www.mdpi.com/2079-9292/8/1/33 (дата звернення: 01.12.2025).

7. Brownlee, J. A Gentle Introduction to Object Recognition With Deep Learning [Електронний ресурс] / J. Brownlee // Machine Learning Mastery. URL: https://machinelearningmastery.com/object-recognition-with-deep-learning/ (дата звернення: 22.11.2025).

8. Model Comparisons: Choose the Best Object Detection Model for Your Project [Електронний ресурс] // Ultralytics Documentation. URL: https://docs.ultralytics.com/compare/ (дата звернення: 22.11.2025).

9. Торгольський, А. О. Проблеми впровадження кіберфізичних робототехнічних систем з інтеграцією комп'ютерного зору / А. О. Торгольський // Молодь: наука та інновації 2025 : матеріали XIII Міжнар. наук.-техн. конф. студ., аспірантів та молодих вчених (Дніпро, 12–14 листоп. 2025 р.) : у 3 т. / Нац. техн. ун-т «Дніпровська політехніка». Дніпро : НТУ «ДП», 2025. Т. 2. 378 с.

10. Кінематика руху маніпулятору. Передмова [Електронний ресурс]. URL: https://vlp.com.ua/files/201092_peredmova.pdf (дата звернення: 15.11.2025).

11. The Effects of Communication Delay on Human Performance and Neurocognitive Responses in Mobile Robot Teleoperation [Електронний ресурс] // arXiv. 2025. URL: https://arxiv.org/html/2508.18074v1 (дата звернення: 01.12.2025).

12. A Benchmark Reference for ESP32-CAM Module [Електронний ресурс] // arXiv. 2025. URL: https://arxiv.org/html/2505.24081v1 (дата звернення: 28.12.2025).

13. Object Detection with TensorFlow Lite Model Maker [Електронний ресурс] // Google AI Edge. URL: https://ai.google.dev/edge/litert/libraries/modify/object_detection (дата звернення: 29.12.2025).

14. Цвіркун, Л. І. Атестація здобувачів вищої освіти. Методичні рекомендації до виконання кваліфікаційної роботи магістра здобувачами галузі знань 12 Інформаційні технології спеціальності 123 Комп'ютерна інженерія / Л. І. Цвіркун, В. В. Гнатушенко, С. М. Ткаченко. Дніпро : НТУ «ДП», 2024. 54 с.

ДОДАТОК А

ТЕКСТ ПРОГРАМИ МОБІЛЬНОГО ЗАСТОСУНКУ

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
“ДНІПРОВСЬКА ПОЛІТЕХНІКА”

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ МОБІЛЬНОГО ЗАСТОСУНКУ КЕРУВАННЯ МАНІПУЛЯТОРОМ З РОЗПІЗНАВАННЯМ ОБ’ЄКТІВ

Текст програми

804.02070743.25018-01 12 01

Листів 8

2025

АНОТАЦІЯ

Дана програма містить в собі частини програмного коду мобільного застосунку кіберфізичної системи керування маніпулятором з функцією виявлення об'єктів у режимі реальному часі.

Програма призначена для:

– здійснення дистанційного керування окремих ланок маніпулятору;

– обробку та трансляцію відеопотоку в режимі реального часу;

– візуалізацію кордонів виявлених об’єктів;

– автоматичне розпізнавання об’єктів за допомогою алгоритмів КЗ.

Програма написана мовою Kotlin з використанням фреймворку Jetpack Compose для побудови інтерфейсу оператора та бібліотекою Google ML Kit для інтеграції машинного навчання у середовищі розробки Android Studio та призначена для застосування на мобільних пристроях з операційною системою не нижче версії 7.1.

ЗМІСТ

1 Екран виявлення об’єктів…….....4

1.1 Завантаження бібліотек……..4

1.2 Ініціалізація детектора об’єктів…….4

1.3 Обробка відеопотоку……...4

1.4 Інтерфейс користувача……5

2 Компонент виявленого об’єкта……6

3 Компонент камери……7

1 ЕКРАН ВИЯВЛЕННЯ ОБ’ЄКТІВ

1.1 Завантаження бібліотек

```
// Core Android and Compose libraries import android.graphics.Bitmap import
android.graphics.Canvas import android.graphics.Paint import
androidx.compose.foundation.layout.* import
androidx.compose.foundation.lazy.staggeredgrid.* import androidx.compose.material3.* import
androidx.compose.runtime.*
// ML Kit libraries for object detection import com.google.mlkit.vision.common.InputImage
import com.google.mlkit.vision.objects.DetectedObject import
com.google.mlkit.vision.objects.ObjectDetection import
com.google.mlkit.vision.objects.custom.CustomObjectDetectorOptions
```

1.2 Ініціалізація детектора об’єктів

```
LaunchedEffect(Unit) {
    // Load custom TensorFlow Lite model from assets
    val localModel = LocalModel.Builder()
        .setAssetFilePath("object_labeler.tflite")
        .build()
    // Configure object detector with custom model
    val options = CustomObjectDetectorOptions.Builder(localModel)
        .setDetectorMode(CustomObjectDetectorOptions.SINGLE_IMAGE_MODE)
        .enableMultipleObjects()  // Detect multiple objects simultaneously
        .enableClassification()   // Enable object classification
        .setClassificationConfidenceThreshold(0.5f)  // Minimum confidence 50%
        .setMaxPerObjectLabelCount(3)  // Up to 3 labels per object
        .build()
    val objectDetector = ObjectDetection.getClient(options)
```

1.3 Обробка відеопотоку

```
// Process frames from camera stream
    viewModel.cameraState
        .filter { it is CameraState.Connected }
        .collect { state ->
            // Prevent frame processing overload
            if (state is CameraState.Connected &&
                state.bitmap != null &&
                !isProcessingFrame) {

                isProcessingFrame = true

                // Create horizontally flipped bitmap for correct orientation
                val processingBitmap = try {
                    createFlippedBitmap(state.bitmap!!, xFlip = true)
                } catch (e: Exception) {
                    isProcessingFrame = false
                    return@collect
                }
                val image = InputImage.fromBitmap(processingBitmap, 0)
                // Run object detection on the frame
                objectDetector.process(image)
                    .addOnSuccessListener { objects ->
                        // Sort detected objects by tracking ID
                        detectedObjects = objects.sortedBy { it.trackingId }

                        // Auto-select first detected object if none selected
                        if (trackingObjectId == null && objects.isNotEmpty()) {
                            trackingObjectId = objects.first().trackingId
                        }
                        // Update selected object if it's no longer detected
                        if (objects.none { it.trackingId == trackingObjectId }) {
                            trackingObjectId = if (objects.isNotEmpty())
```

```
                        objects.first().trackingId else null
                }
```

1.4 Інтерфейс користувача

```
Scaffold(
    topBar = {
        TopAppBar(
            title = { Text("Object Detection") },
            navigationIcon = {
                IconButton(onClick = onBackClick) {
                    Icon(Icons.Filled.ArrowBack, contentDescription = null)
                }
            },
            actions = {
                // Manipulator connection toggle button
                val manipulatorConnected = manipulatorState is ManipulatorState.Connected
                IconToggleButton(
                    checked = manipulatorConnected,
                    onCheckedChange = {
                        if (it) viewModel.connectManipulator()
                        else viewModel.disconnectManipulator()
                    }
                ) {
                    Icon(
                        painter = painterResource(R.drawable.precision_manufacturing),
                        contentDescription = null
                    )
                }
                // Camera connection toggle button
                val cameraConnected = cameraState is CameraState.Connected
                IconToggleButton(
                    checked = cameraConnected,
                    onCheckedChange = {
                        if (it) viewModel.connectCamera()
                        else viewModel.disconnectCamera()
                    }
                ) {
                    Icon(
                        painter = painterResource(R.drawable.camera_video),
                        contentDescription = null
                    )
                }
            }
        )
    },
    content = { innerPadding ->
        Column(
            modifier = modifier
                .padding(innerPadding)
                .padding(16.dp)
                .fillMaxSize()
        ) {
            // Camera preview with annotated objects
            CameraCard(
                bitmapProvider = analyzedBitmap?.let { { analyzedBitmap } },
                cameraState = cameraState,
                onConnect = viewModel::connectCamera
            )
            // Grid of detected objects for selection
            LazyVerticalStaggeredGrid(
                columns = StaggeredGridCells.Fixed(2),
                horizontalArrangement = Arrangement.spacedBy(8.dp)
            ) {
                itemsIndexed(detectedObjects) { _, obj ->
                    ObjectItem(
                        detectedObject = obj,
                        isActive = trackingObjectId == obj.trackingId,
                        onClick = { trackingObjectId = obj.trackingId })}}}})
```

2 КОМПОНЕНТ ВИЯВЛЕНОГО ОБ’ЄКТА

```
package ua.ntudp.torholskyi.manipulatorapp.ui.screen.objectdetection

/**
 * Composable card displaying detected object information
 * Changes appearance when selected as tracking target
 */
@Composable
fun ObjectItem(
    modifier: Modifier = Modifier,
    detectedObject: DetectedObject? = null,
    isActive: Boolean = false,
    onClick: () -> Unit = {}
) {
    Card(
        modifier = modifier,
        elevation = CardDefaults.cardElevation(defaultElevation = 4.dp),
        colors = CardDefaults.cardColors(
            // Highlight active object with primary color
            containerColor = if (isActive)
                MaterialTheme.colorScheme.primaryContainer
            else
                MaterialTheme.colorScheme.surface
        )
    ) {
        Column(
            modifier = Modifier
                .clickable(onClick = onClick)
                .padding(8.dp)
                .fillMaxWidth()
        ) {
            // Object icon (red if active, blue if inactive)
            Icon(
                modifier = Modifier
                    .fillMaxWidth()
                    .aspectRatio(1f),
                painter = painterResource(id = R.drawable.baseline_search_24),
                contentDescription = null,
                tint = if (isActive) Color.Red else Color.Blue
            )

            // Extract label and confidence from detected object
            val label = detectedObject?.labels?.firstOrNull()?.text ?: "Object"
            val confidence = detectedObject?.labels?.firstOrNull()?.confidence ?: 0f
            val percent = (confidence * 100).toInt()

            // Display label with confidence percentage
            Text(
                text = "$label $percent%",
                style = MaterialTheme.typography.titleMedium.copy(
                    fontWeight = FontWeight.Bold
                ),
                color = if (isActive) Color.Red else Color.Blue,
                maxLines = 1,
                overflow = TextOverflow.Ellipsis
            )
        }
    }
}
```

3 КОМПОНЕНТ КАМЕРИ

```
package ua.ntudp.torholskyi.manipulatorapp.ui.camera

/**
 * Card component displaying camera stream with different states
 * Handles connecting, disconnected, and active streaming states
 */
@Composable
fun CameraCard(
    modifier: Modifier = Modifier,
    cameraState: CameraState,
    bitmapProvider: (() -> Bitmap?)? = null,
    onConnect: () -> Unit = {}
) {
    Card(
        modifier = modifier.aspectRatio(4 / 3F),
        elevation = CardDefaults.cardElevation(defaultElevation = 4.dp)
    ) {
        when (cameraState) {
            // Show loading indicator while connecting
            CameraState.Connecting -> {
                CameraConnecting()
            }

            // Show disconnection reason and connect button
            is CameraState.Disconnected -> {
                CameraDisconnected(
                    reason = cameraState.reason,
                    onConnect = onConnect
                )
            }

            // Display camera stream or processed bitmap
            is CameraState.Connected -> {
                (bitmapProvider?.invoke() ?: cameraState.bitmap)?.let { bitmap ->
                    CameraStream(bitmapProvider = { bitmap })
                } ?: CameraConnecting()
            }

            // Initial state with connect prompt
            CameraState.Initialized -> {
                CameraDisconnected(
                    reason = "Connect the camera",
                    onConnect = onConnect
                )
            }
        }
    }
}

/**
 * Display camera bitmap stream with rotation correction
 */
@Composable
fun CameraStream(
    bitmapProvider: () -> Bitmap,
    modifier: Modifier = Modifier
) {
    Image(
        bitmap = bitmapProvider().asImageBitmap(),
        contentDescription = null,
        modifier = modifier.fillMaxSize().rotate(-90f),  // Correct ESP32-CAM orientation
        contentScale = ContentScale.FillBounds
    )
}

/**
```

```
 * Loading indicator during camera connection
 */
@Composable
fun CameraConnecting(
    modifier: Modifier = Modifier
) {
    Box(modifier = modifier.fillMaxSize(), contentAlignment = Alignment.Center) {
        CircularProgressIndicator(strokeCap = StrokeCap.Round)
    }
}

/**
 * Disconnected state with error message and connect button
 */
@Composable
fun CameraDisconnected(
    modifier: Modifier = Modifier,
    reason: String,
    onConnect: () -> Unit = {},
    showConnectBtn: Boolean = true
) {
    Column(
        modifier = modifier.fillMaxSize(),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Center
    ) {
        Text(
            text = reason,
            modifier = Modifier.padding(8.dp),
            textAlign = TextAlign.Center,
            maxLines = 2,
            overflow = TextOverflow.Ellipsis
        )
        if (showConnectBtn) {
            Button(onClick = onConnect) {
                Text(text = "Connect")
            }
        }
    }
}
```