МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів рухової активності людини
за допомогою дронів
(тема)

Виконав:
здобувач 2 року навчання
групи ІПЗм-23-1
Юрій УТКІН
(Власне ім'я, ПРІЗВИЩЕ)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)

Тип програми освітньо-наукова

Керівник Проф. Білоус Н. В.
(посада, прізвище)

Допускається до захисту
Зав. кафедри

_____ _____
(підпис) (Власне ім'я, ПРІЗВИЩЕ)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук

Кафедра програмної інженерії
Рівень вищої освіти другий (магістерський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми освітньо-наукова програма
Освітня програма Інженерія програмного забезпечення
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Уткіну Юрію Євгеновичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Дослідження методів рухової активності людини за допомогою дронів»

Затверджена наказом по університету від 15.04. 2024р. № 290 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 19.06.2025

3. Вихідні дані до роботи Безпілотні літальні апарати, алгоритми детекції та класифікації об'єктів (YOLO, SSD, BlazePose), обчислювальні платформи для дронів (Orange Pi 5+, Raspberry Pi, RockPro), методи аналізу руху на основі якірних точок, алгоритми порівняння траєкторій і поведінки об'єктів (DTW, LSTM), програмне забезпечення для обробки даних у реальному часі та інтеграція сенсорів (LiDAR, GPS) для роботи в умовах реального часу.

4. Перелік питань, що потрібно опрацювати в роботі

Огляд теоретичних основ аналізу руху об'єктів за допомогою дронів, дослідження алгоритмів детекції та класифікації руху об'єктів у реальному часі, аналіз методів

порівняння траєкторій і поведінки об’єктів із еталонними значеннями, вибір апаратної платформи та програмного забезпечення для інтеграції алгоритмів, практична розробка системи, яка поєднує механізми детекції, класифікації та аналізу поведінки об’єктів.

КАЛЕНДАРНИЙ ПЛАН

Номер	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	15.04.2025	*виконано*
2	Аналіз предметної галузі і постановка задачі	20.04.2025-04.05.2025	*виконано*
3	Аналіз існуючих методів	05.05.2025-10.05.2025	*виконано*
4	Теоретичне дослідження	10.05.2025-20.05.2025	*виконано*
5	Практичне дослідження	21.05.2025-05.06.2025	*виконано*
6	Підготовка пояснювальної записки	05.06.2026-14.06.2025	*виконано*
7	Підготовка презентації та доповіді	15.06.2025	*виконано*
8	Перевірка на плагіат	17.06.2025	*виконано*
9	Нормоконтроль	19.06.2025	*виконано*
10	Рецензування	19.06.2025	*виконано*
11	Попередній захист	19.06.2025	*виконано*
12	Занесення диплома в електронний архів	19.06.2025	*виконано*
13	Допуск до захисту у зав. кафедри	19.06.2025	*виконано*
Номер	Назва етапів роботи	Термін виконання етапів роботи	Примітка

Дата видачі завдання 15 січня 2025р.

Студент (ка) _____ Уткін Ю. Є.
(підпис)

Керівник роботи _____ проф. Білоус Н. В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Квалифікаційна робота магістра містить: 94 с., 14 рис., 24 джерело.

КОМП'ЮТЕРНИЙ ЗІР, ГЛИБИННЕ НАВЧАННЯ, БЕЗПІЛОТНІ ЛІТАЛЬНІ АПАРАТИ, ОБРОБКА ВІДЕОПОТОКУ, АЛГОРИТМИ ДЕТЕКЦІЇ ОБ'ЄКТІВ, АНАЛІЗ ТРАЄКТОРІЇ, КЛАСИФІКАЦІЯ РУХУ, ОПТИМІЗАЦІЯ АЛГОРИТМІВ, НАВІГАЦІЙНІ СИСТЕМИ, GPS ТА LIDAR, СИСТЕМИ ПОЗИЦІОНУВАННЯ, АВТОНОМНІ ТЕХНОЛОГІЇ, РОЗПІЗНАВАННЯ ПОВЕДІНКИ.

Об'єкт дослідження – процес аналізу та класифікації руху об'єктів за допомогою дронів.

Предмет дослідження – методи та алгоритми детекції, класифікації руху об'єктів, порівняння траєкторій і поведінки з еталонними значеннями.

Мета роботи – дослідження методів детекції та класифікації руху об'єктів у реальному часі, аналіз їх ефективності та практичне створення інтегрованої системи для роботи з даними дронів.

Методи досліджень – емпіричний аналіз, моделювання, порівняння, розробка та тестування.

З розвитком безпілотних технологій і збільшенням обсягу завдань, які виконують дрони, аналіз руху об'єктів у реальному часі стає все більш актуальним. Здатність дронів точно ідентифікувати, класифікувати рухи та порівнювати поведінку з еталонними значеннями відкриває широкі можливості для використання у сферах моніторингу, безпеки та автоматизації. Неточність або затримки в аналізі можуть призвести до неправильної ідентифікації, що може мати критичні наслідки, особливо в умовах швидко змінного середовища.

У роботі здійснено теоретичний аналіз методів детекції, класифікації та аналізу траєкторій руху, а також досліджено можливості оптимізації роботи алгоритмів на обмежених апаратних платформах, таких як Orange Pi або NVIDIA

Jetson. Підсумкова мета включає створення інтегрованої системи, яка поєднує ці алгоритми для роботи з відеопотоком із дронів у реальному часі.

COMPUTER VISION,DEEP LEARNING, UNMANNED AERIAL VEHICLES (UAV), VIDEO STREAM PROCESSING, OBJECT DETECTION ALGORITHMS, TRAJECTORY ANALYSIS, MOTION CLASSIFICATION, ALGORITHM OPTIMIZATION, NAVIGATION SYSTEMS, GPS AND LIDAR, POSITIONING SYSTEMS, AUTONOMOUS TECHNOLOGIES, BEHAVIOR RECOGNITION.

The object of the study is the process of analyzing and classifying the movement of objects using drones.

The subject of the study is methods and algorithms for detecting, classifying the movement of objects, comparing trajectories and behavior with reference values.

The purpose of the work is to study methods for detecting and classifying the movement of objects in real time, analyze their effectiveness, and practically create an integrated system for working with drone data.

Research methods are empirical analysis, modeling, comparison, development, and testing.

With the development of unmanned technologies and the increase in the volume of tasks performed by drones, the analysis of the movement of objects in real time is becoming increasingly relevant. The ability of drones to accurately identify, classify movements, and compare behavior to reference values opens up wide opportunities for use in the fields of monitoring, security, and automation. Inaccuracies or delays in analysis can lead to incorrect identification, which can have critical consequences, especially in rapidly changing environments.

The paper provides a theoretical analysis of methods for detecting, classifying, and analyzing movement trajectories, and explores the possibilities of optimizing the operation of algorithms on limited hardware platforms such as Orange Pi or NVIDIA Jetson. The ultimate goal includes the creation of an integrated system that combines these algorithms to work with a video stream from drones in real time.

Умови публікації звіту: заява щодо самостійного виконання кваліфікаційної роботи та можливості її публікації в електронному архіві відкритого доступу ElAr KhNURE.

Завідувачу кафедри
ПІ_____
(скорочена назва кафедри)
проф. Кирилу СМЕЛЯКОВУ_
(вчене звання, сласне ім'я, прізвище)

ЗАЯВА
щодо самостійності виконання кваліфікаційної роботи та можливості її публікації (та/або публікації анотації кваліфікаційної роботи) в електронному архіві відкритого доступу ElAr KhNURE

Я,
_____Уткін Юрій Євгенович_____
(прізвище, ім'я, по батькові)
здобувач вищої освіти на другому (магістерському) рівні вищої освіти академічної групи_____ІПЗм-23-1_____

кафедра_____програмної інженерії _____,
(повна назва кафедри)

заявляю: моя кваліфікаційна робота на тему
__Дослідження методів рухової активності людини за допомогою дронів_____,
(назва роботи)

що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в репозиторії "ElArKhNURE". Погоджуюся з авторським договором, відповідно до Положення про репозиторій ХНУРЕ "ElArKhNURE". Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений (а) з вимогами академічної доброчесності, згідно з якими виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

Дата Підпис

ЗМІСТ

Вступ 10
1 Аналіз предметної галузі 12
1.1 Аналіз предметної галузі дослідження 12
1.1.1 Ідентифікація та розпізнавання цілей 12
1.1.2 Аналіз існуючих підходів до розпізнавання цілей і трекінгу 14
1.1.3 Аналіз області машинного навчання 19
1.2 Актуальність проблеми 21
1.2.1 Аналіз безпілотних літальних апаратів 21
1.2.2 Військова сфера застосування 22
1.2.3 Проблеми у військовій сфері застосування 24
1.3 Огляд і аналіз наукових та літературних джерел 27
1.3.1 Вибір матеріалів предметної області 27
1.3.2 Дослідження розпізнавання рухів 29
1.3.3 Дослідження апаратного прискорення для дронів 33
1.3.4 Вітчизняні дослідження в області ШІ та комп’ютерного зору 34
1.3.5 Аналіз вибору джерел дослідження 36
1.4 Постановка задачі 37
2 Опис прийнятих проектних рішень 41
2.1 Проектування архітектури системи 41
2.2 Аналіз даних для навчання моделі 45
2.3 Вибір алгоритму детекції та класифікації руху 47
2.4 Класифікація руху об'єкту та наукова новизна 50
3 Опис програмної реалізації 52
3.1 Огляд технологічного стеку та вибір інструментів 52
3.2 Архітектура та компоненти системи 55
3.2.1 Компоненти системи 55
3.2.2 Файлова структура 58

3.2.3 Архітектура Embaded системи 59
3.3 Проектування тестового сцйенарію 60
3.4 Розроблені алгоримтми 61
3.4.1 Алгоритм пайплайну детекції 61
3.4.2 Алгоритм донаведення дрону 64
4 Проведення та аналіз результатів експеременту 67
4.1 Проведення експеременту 67
4.1.1 Вхідні дані 67
4.2 Результати детекції та трекінгу 71
4.3 Результат донаведення та моніторингу навантаження 74
Висновки 77
Перелік джерел посилання 78
Перелік джерел посилання за науковими напрямами керівника та науковців кафедри програмної інженерії 81
Додаток А Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ 82
Додаток Б Слайди презентації 83
Додаток В Апробація результатів роботи 92

ВСТУП

З розвитком сучасних технологій та зростаючою роллю безпілотних літальних апаратів (дронів) у військових, моніторингових та безпекових завданнях, аналіз руху об’єктів набуває надзвичайної актуальності. Особливо це стає важливим у контексті військових конфліктів, де своєчасна ідентифікація та класифікація рухів може відігравати вирішальну роль у забезпеченні безпеки та стратегічного управління. Використання дронів для задач розпізнавання й аналізу руху об’єктів відкриває нові перспективи для автоматизації процесів і підвищення точності прийняття рішень, проте водночас створює низку викликів, пов’язаних із обробкою великих обсягів даних у реальному часі.

Об’єктом цього дослідження є процес аналізу та класифікації руху об’єктів за допомогою безпілотних літальних апаратів. Предмет дослідження охоплює алгоритми детекції, класифікації та порівняння траєкторій руху об’єктів, а також методи їх реалізації на компактних обчислювальних платформах. Головною метою роботи є створення системи, яка об’єднає сучасні підходи до детекції й класифікації рухів із застосуванням алгоритму YOLO11 для задач розпізнавання та класифікації в реальному часі. Особливу увагу приділено інтеграції цих алгоритмів із одноплатними комп’ютерами, що дозволяє оптимізувати використання ресурсів дронів.

Актуальність дослідження визначається потребою у створенні високоефективних систем для аналізу динамічних об’єктів у складних умовах, які часто виникають під час військових операцій. Своєчасне й точне розпізнавання рухів об’єктів, таких як люди чи техніка, дозволяє підвищити ефективність управління та знизити ризики, пов’язані з помилковими оцінками. Військові конфлікти лише підкреслюють важливість розвитку таких технологій, які дозволяють дронам працювати в реальному часі, забезпечуючи стратегічну перевагу.

Новизна цього дослідження полягає в унікальному підході до задачі: алгоритм YOLO11, який є останнім поколінням у сімействі YOLO, вперше

адаптовано для використання в задачах детекції об’єктів із дронів. Додатково, інтеграція цього алгоритму з системами класифікації руху на базі одноплатних комп’ютерів, таких як Orange Pi чи Raspberry Pi, є інноваційним рішенням, яке раніше не було детально досліджене. Це поєднання дозволяє вирішувати задачі не лише розпізнавання, але й аналізу поведінки об’єктів із високою точністю та швидкістю.

Таким чином, робота спрямована на подолання технічних викликів і відкриття нових можливостей для використання безпілотних технологій у задачах детекції й класифікації руху, що робить її значущою як з наукової, так і з практичної точки зору.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Аналіз предметної галузі дослідження.

1.1.1 Ідентифікація та розпізнавання цілей

Ідентифікація цілей за допомогою дронів базується на використанні сучасних технологій, які дозволяють розпізнавати об’єкти, визначати їхні характеристики та аналізувати їх у реальному часі. Цей процес включає кілька ключових методів, які забезпечують точність і надійність, навіть у складних умовах. Завдяки комбінації різних підходів дрони можуть виконувати завдання в розвідці, військових операціях, моніторингу чи гуманітарних місіях.

Одним із найбільш поширених методів є використання відеокамер високої роздільної здатності. Камери, встановлені на дронах, забезпечують детальний візуальний огляд місцевості або об’єкта. Завдяки алгоритмам комп’ютерного зору дрони можуть аналізувати відео в реальному часі, виділяючи об’єкти, що відповідають заданим характеристикам. Наприклад, алгоритми розпізнавання можуть ідентифікувати транспортні засоби, споруди чи живу силу за характерними ознаками, такими як форма, розмір, рух або колір. Такий підхід часто використовується для моніторингу територій, виявлення підозрілих об’єктів або відстеження цілей, що рухаються[1].

Ще одним важливим методом є тепловізійне сканування. Тепловізори дозволяють виявляти цілі за їхнім тепловим випромінюванням, що особливо корисно в умовах обмеженої видимості, таких як нічний час, туман або густий ліс. Завдяки цьому дрони можуть розпізнавати живі організми чи об’єкти, які генерують тепло, навіть якщо вони приховані. Цей метод активно застосовується у військових і пошуково-рятувальних операціях, коли важливо швидко ідентифікувати людей або тварин у складних умовах.

Радіолокація є ще одним ефективним способом ідентифікації. Радіолокаційні системи дозволяють виявляти та ідентифікувати об’єкти, використовуючи відбиті радіохвилі. Цей підхід є надзвичайно корисним для роботи в умовах поганої видимості, а також для виявлення об’єктів, які не випромінюють тепла або мають

камуфляж. Радіолокаційні дрони здатні сканувати великі території та визначати розташування об’єктів із високою точністю.

Подібні системи також використовують акустичні сенсори для ідентифікації цілей за звуковими характеристиками. Ці сенсори дозволяють виявляти джерела звуку, аналізувати їхню природу та визначати розташування. Наприклад, вони можуть розпізнавати звуки двигунів транспортних засобів, вибухів чи стрілянини, що допомагає у військових операціях і моніторингу ситуацій у зонах конфліктів.

Інфрачервоні сенсори є ще одним потужним інструментом для ідентифікації. Вони дозволяють виявляти об’єкти за їхнім інфрачервоним випромінюванням, що є невидимим для людського ока. Цей метод схожий на тепловізійне сканування, але часто використовується для більш специфічних завдань, таких як виявлення прихованої техніки або обладнання, що працює на великих відстанях.

Дрони також оснащуються сенсорами для збору радіочастотних даних. Ці сенсори можуть виявляти й аналізувати електромагнітне випромінювання, яке походить від електронних пристроїв, таких як мобільні телефони, радіостанції або інші комунікаційні системи. Такий підхід дозволяє визначати розташування цілей, навіть якщо вони приховані або недоступні для прямого візуального спостереження.

Інтеграція штучного інтелекту (ШІ) є важливим компонентом процесу ідентифікації цілей. Завдяки алгоритмам машинного навчання дрони можуть аналізувати великі обсяги даних, виділяючи ключові характеристики цілей і порівнюючи їх із наявними шаблонами. ШІ допомагає дронам не лише розпізнавати об’єкти, але й передбачати їхню поведінку чи рух. Наприклад, дрони можуть визначати ймовірність того, що транспортний засіб змінить напрямок, або передбачати, як рухатиметься група людей.

Відповідно важливу роль у процесі ідентифікації відіграє обробка даних у реальному часі. Системи передають отриману інформацію до командних центрів, де вона аналізується за допомогою сучасного програмного забезпечення. Це дозволяє операторам швидко ухвалювати рішення, засновані на точних і актуальних даних. Отримана інформація може бути інтегрована з іншими

системами, такими як супутникові знімки чи дані від наземних спостережних пунктів, що забезпечує комплексний аналіз ситуації.

Використання мультиспектральних і гіперспектральних камер дозволяє дронам ідентифікувати об’єкти на основі їхніх спектральних характеристик. Ці технології використовуються для виявлення матеріалів, аналізу рослинності або розпізнавання прихованих об’єктів. Такий підхід є особливо корисним у ситуаціях, де візуальна інформація є недостатньою або неточною.

Сучасні методи ідентифікації також передбачають використання геопросторових даних. Дрони обладнані системами GPS і картографічним програмним забезпеченням, що дозволяє точно визначати координати цілей. Це важливо для виконання завдань, де точність розташування є критичною, наприклад, для нанесення точкових ударів чи планування рятувальних операцій.

Усі ці методи, працюючи разом, створюють комплексну систему ідентифікації, яка дозволяє дронам ефективно виконувати різноманітні завдання. Завдяки швидкому розвитку технологій дрони стають дедалі більш автономними та точними, забезпечуючи ефективність навіть у найскладніших умовах.

1.1.2 Аналіз існуючих підходів до розпізнавання цілей і трекінгу

У рамках реалізації системи виявлення людини та автоматичного керування дроном важливо здійснити огляд наукових підходів, які вже реалізовані в практиці комп’ютерного зору, зокрема в контексті розпізнавання об’єктів на відеопотоці та їх трекінгу. Задача, що досліджується у цій роботі, є міждисциплінарною, оскільки поєднує інтелектуальний відеоаналіз, роботу з глибокими нейронними мережами, обчислення на edge-пристроях і взаємодію з модулем керування польотом (PX4). Особливість предметної області полягає в необхідності забезпечення роботи в реальному часі при обмежених апаратних ресурсах і високих вимогах до точності та надійності.

Однією з ранніх ідей для аналізу людських рухів є використання методу опорних точок, який застосовувався для відстеження положення голови людини

[2]. Такий підхід дозволяє будувати відносні вектори руху за координатами частин обличчя, що забезпечує не лише фіксацію наявності людини, але й можливість інтерпретації її стану (наприклад, напрямок погляду). Ця технологія може бути корисною як допоміжна при уточненні позиції цілі або класифікації її поведінки, але вона є досить вузькоспеціалізованою і не придатна як основна система для FPV-дронів, де потрібна цільова обробка загального зображення сцени.

Зі зростанням обчислювальної потужності edge-пристроїв активного поширення набули архітектури глибокого навчання типу YOLO (You Only Look Once). У роботі [3] проведено детальний порівняльний аналіз CNN-архітектур для виявлення об'єктів з метою визначення класу об’єкта. Автори порівнюють різні моделі згорткових нейронних мереж, зокрема Faster R-CNN, SSD та YOLO, і роблять висновок, що архітектура YOLO показує кращий компроміс між точністю та продуктивністю. Це робить її найбільш привабливою для завдань з обмеженим доступом до GPU, як у випадку з вбудованими мікрокомп’ютерами на дронах.

Особливу увагу заслуговує публікація [4], в якій розглядається порівняння версій YOLO від v3 до v8 у контексті пошуку людей на складному фоні (наприклад, на воді). Це дозволяє зробити висновок, що із кожною новою версією точність виявлення значно зростає, а кількість помилкових спрацювань зменшується. YOLOv8 вже є повноцінним інструментом для застосування у реальному часі. Цей факт дає підстави розглядати YOLOv11 як логічне продовження розвитку лінійки, яке може забезпечити ще кращі результати при схожих апаратних умовах.

Дослідження [5] також акцентує увагу на важливості адаптивного підходу до багатомасштабного виявлення цілей в аерофотознімках. Запропонована модель AFEDet інтегрує три ключові модулі: адаптивне вилучення ознак (AFE), вдосконалену багаторівневу мережу злиття ознак (MeFPN) та функцію втрат, чутливу до масштабу (SAL). Вони забезпечують ефективну обробку як малих, так і великих об'єктів на складному фоні, підвищуючи точність розпізнавання більш ніж на 7% у порівнянні з базовими моделями.

У ще одній роботі [6] продемонстровано застосування YOLO-9 для виявлення людини на потоковому відео. Автори показали, що використання

попередньо натренованої моделі дозволяє досягти точності понад 90% навіть у ситуаціях з динамічним фоном, що характерно для відео з дронів. У результаті було підтверджено, що модель здатна обробляти відеопотік у реальному часі при достатній оптимізації, що є вкрай важливим у бойових сценаріях або при розвідці.

Для таких завдань важливо не лише виявити об'єкт, а й забезпечити його стійке супроводження, що вимагає надійного алгоритму трекінгу. Усі вищезгадані підходи можуть бути поєднані з алгоритмами трекінгу, які, на відміну від одноразового визначення об'єкта, дозволяють зберігати ідентичність цілі протягом часу та забезпечують стабільне її відстеження.

Найпоширенішими є KCF, SORT та DeepSORT. KCF демонструє високу швидкодію, проте не враховує зміну масштабу. SORT, як зазначається у багатьох оглядах, забезпечує хорошу продуктивність у реальному часі, проте не ідеально працює при зникненні цілі з кадру. DeepSORT, своєю чергою, використовує глибоке навчання для побудови дескрипторів, які дозволяють повертатися до раніше втрачених цілей, проте має високу вимогу до ресурсів. Детальніше це представлено на рисунку 1.3.1.

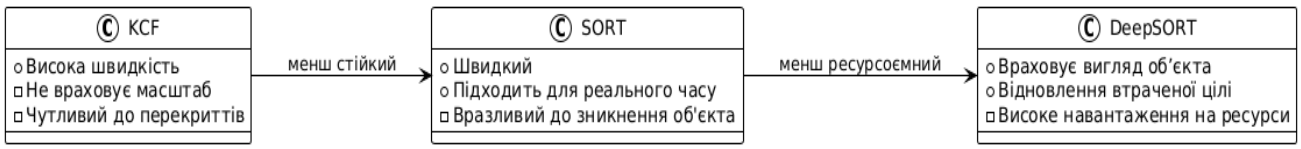

Рисунок 1.1 – Порівняльна характеристика алгоритмів трекінгу об’єктів у реальному часі (рисунок виконано самостійно)

Окрему категорію становлять роботи, що використовують аналіз скелетних точок тіла людини для класифікації рухів або емоцій. У [7] запропоновано методику визначення емоційного стану та положення людини шляхом аналізу поз тіла. Хоча задача емоційного аналізу менш релевантна для FPV-дронів, сам підхід до фіксації скелетної структури може використовуватись для уточнення пози тіла та контекстного розпізнавання (наприклад, якщо людина сидить, лежить або

рухається). Методи визначення положення тіла у відео були також розглянуті в [8], де автори порівнюють кілька підходів до виявлення поз за координатами частин тіла. Найбільш ефективними виявилися MediaPipe та OpenPose, які забезпечують точність понад 94%. Проте вони вимагають суттєвої обчислювальної потужності або попереднього зменшення роздільної здатності відео.

Робота [9] висвітлює підхід до класифікації фізичних дій людини на основі 3D-координат суглобів, що отримуються зі скелетної структури. У дослідженні досягнуто високої точності розпізнавання дій, таких як присідання, ходьба, нахили. Це може бути застосоване як розширення функціональності у військових FPV-дронах для автоматичного розпізнавання ситуацій на полі бою.

У сучасних системах обробки відео з дронів особливу увагу приділяють якості вилучення просторових ознак та збереженню контекстної інформації при роботі з дрібними об’єктами. Одним із найбільш перспективних рішень є використання архітектур, що поєднують локальні й глобальні особливості сцени через механізми уваги. Приклад такої архітектури продемонстровано на рисунку 1.3.2 – це SCASM-модуль із SCASNet, розроблений для точного виявлення об’єктів у складних аерофотознімках.

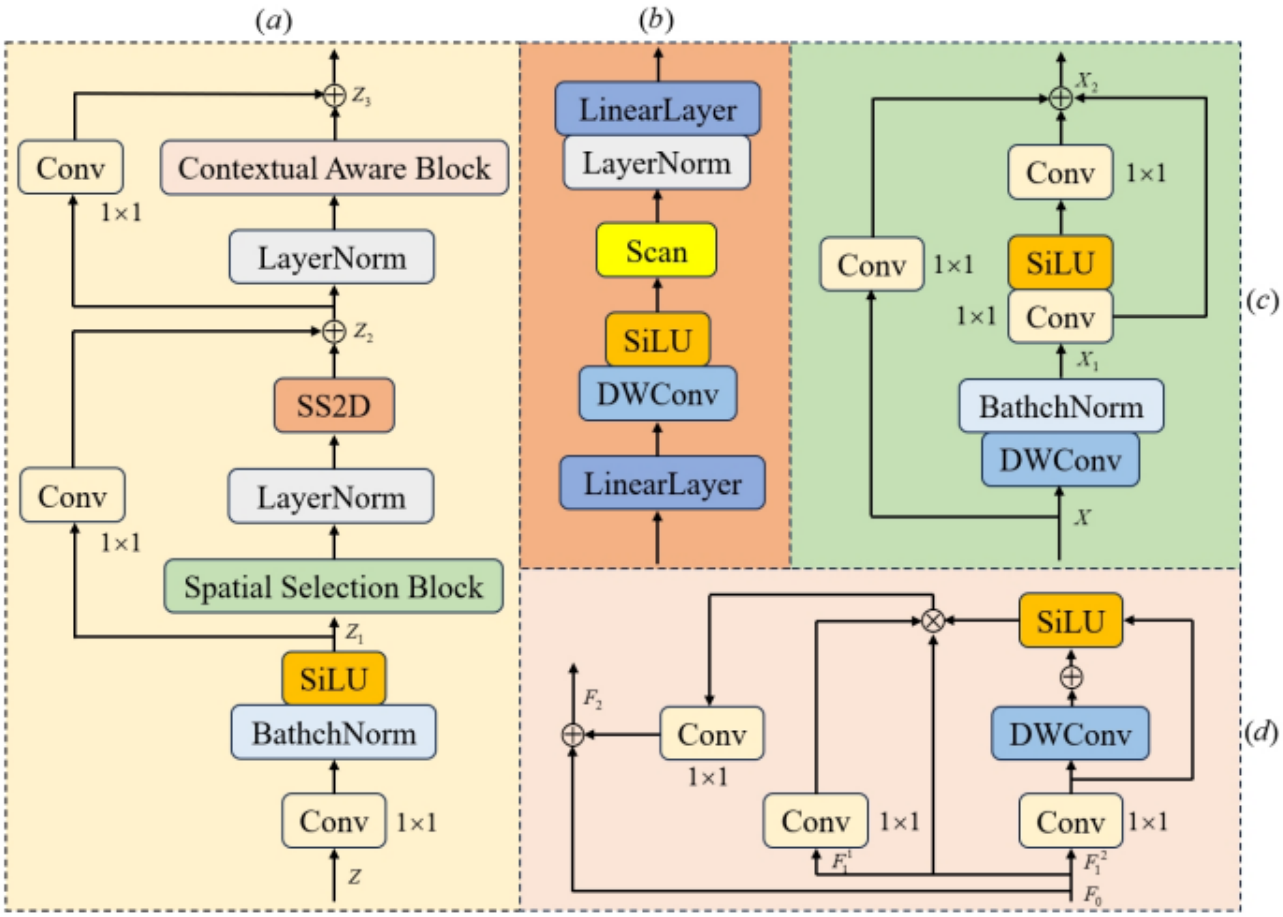

Рисунок 1.2 – Архітектура SCASM (Spatial Context-Aware Selection Module[10])

з роботи SCASNet, адаптована до задач виявлення об’єктів з дронів. Модуль поєднує блок просторового вибору та блок контекстної уваги для покращеної локалізації малих цілей в складних сценах[10].

Застосування такої архітектури у контексті FPV-дронів дозволяє реалізувати інтелектуальну обробку відео в реальному часі, з акцентом на виявлення малопомітних цілей та адаптацію до складного фону. У поєднанні з YOLOv11 або аналогічною детекторною мережею це відкриває можливість створення більш стійкої та точної системи супроводу об’єктів у польоті, навіть за умов обмежених апаратних ресурсів.

1.1.3 Аналіз області машинного навчання

Машинне навчання є однією з найперспективніших галузей сучасної науки та техніки, яка лежить в основі багатьох інноваційних рішень у різних сферах, зокрема у військовій, медицині, фінансах та транспорті. Ця область займається розробкою алгоритмів, здатних вчитися з даних та приймати рішення без явного програмування. Машинне навчання базується на математичному аналізі, статистиці та обчислювальних методах, що дозволяє моделям розпізнавати закономірності, прогнозувати події та адаптуватися до нових умов. Завдяки цьому технологія ефективно використовується для роботи з великими обсягами даних, аналізу складних систем і автоматизації процесів.

У контексті розпізнавання рухів дронами, машинне навчання є фундаментальною технологією, що дозволяє створювати високоточні системи обробки зображень та відео. Такі моделі здатні розпізнавати різні типи рухів, класифікувати об’єкти, передбачати їх траєкторії та аналізувати поведінку в реальному часі. Це можливо завдяки використанню різних підходів, таких як навчання з учителем, без учителя та підкріплення. Наприклад, нейронні мережі, що є ключовим інструментом машинного навчання, дозволяють моделювати складні взаємозв’язки в даних, імітуючи роботу людського мозку. Такі алгоритми забезпечують високу точність та адаптивність, що є критично важливим для військових застосувань, де швидкість і надійність обробки даних мають першочергове значення.

Надалі варто розібрати впроваджені рішення машинного навчання в предметну область дронів.

Дрони активно використовують технології детекції та розпізнавання об’єктів, і багато з вищезгаданих алгоритмів успішно адаптовані до їхніх потреб. Тепер варто розглянути ці технології, нижче наведено детальний огляд методів, які вже використовувалися на дронах:

– yolo (you only look once). це один із найпопулярніших алгоритмів для дронів завдяки високій швидкості та здатності працювати в реальному часі. він

застосовується для виявлення транспортних засобів, людей, тварин або інших об’єктів під час патрулювання чи моніторингу території. yolo ефективний для дронів через обмежені обчислювальні ресурси, особливо на моделях з низьким енергоспоживанням;

– ssd (single shot multibox detector). використовується у дронах для детекції об’єктів на великій висоті. ssd добре справляється з реальними умовами, такими як різні освітлення чи складний фон, що робить його ефективним у завданнях моніторингу сільськогосподарських полів або пошуково-рятувальних операціях;

– r-cnn та його модифікації (fast r-cnn, faster r-cnn). ці методи застосовувалися для більш точного виявлення та класифікації об’єктів, наприклад, для військових завдань, таких як розпізнавання техніки чи підозрілих об’єктів. проте їх використання на дронах часто обмежене через високу вимогу до обчислювальних ресурсів;

– mask r-cnn. використовувався в задачах сегментації об’єктів, наприклад, для оцінки пошкоджень інфраструктури чи аналізу стану лісів. цей алгоритм дозволяє не лише розпізнавати об’єкти, а й визначати їхню точну форму;

– hog (histogram of oriented gradients). цей метод застосовувався для простих завдань, таких як виявлення людей чи транспортних засобів. його популярність зменшилася з появою глибоких нейронних мереж, проте він досі використовується в деяких дронах завдяки простоті реалізації;

– vision transformers (vit). нещодавно почали тестувати на дронах для складніших задач, таких як аналіз великих територій. їх перевага — висока точність і здатність враховувати глобальні залежності в зображеннях, проте використання трансформерів обмежене через потребу в потужному обладнанні;

– навчання підкріплення (reinforcement learning). застосовувалося для автономного керування дронами, де дрон навчається визначати оптимальні маршрути та розпізнавати цільові об’єкти в складних умовах, наприклад, у густих лісах або міських зонах;

– lstm та інші рекурентні нейронні мережі. використовувалися для аналізу відеопотоку з камер дронів, особливо для визначення траєкторії рухомих об’єктів, таких як транспорт або пішоходи. це дозволяє дронам не лише бачити об’єкти, а й передбачати їх поведінку[11].

1.2 Актуальність проблеми

1.2.1 Аналіз безпілотних літальних апаратів

Дрони, або безпілотні літальні апарати (БПЛА) – це пристрої, які здатні літати без участі людини на борту. Вони управляються дистанційно або працюють автономно завдяки запрограмованим алгоритмам і системам навігації. Спочатку розроблені для військових цілей, дрони швидко знайшли застосування у багатьох інших сферах, зокрема в комерційних, військових, наукових, розважальних і громадських.

Конструкція дронів може варіюватися від компактних моделей, які поміщаються в долоні, до великих апаратів, здатних перевозити значні вантажі. Вони оснащені різноманітними сенсорами, камерами, GPS-модулями, а іноді навіть штучним інтелектом, що дозволяє їм виконувати складні завдання. Завдяки цьому вони стали надзвичайно корисними у виконанні таких завдань, як аерофотозйомка, моніторинг територій, доставка товарів, рятувальні операції та дослідження важкодоступних місць.

Однією з основних переваг дронів є їх здатність працювати в умовах, небезпечних або недоступних для людей. Наприклад, вони можуть використовуватися для спостереження за вулканічними виверженнями, дослідження океанів, перевірки інфраструктури, пошуку людей під час стихійних лих, або спостереження та безпосереднє застосування у бойових діях. Завдяки технологічному прогресу вони стали доступнішими, що сприяло їхньому широкому поширенню в повсякденному житті[12].

У цивільному секторі дрони часто використовуються для зйомок і створення контенту, адже вони дають змогу отримувати унікальні види з висоти пташиного польоту. Крім того, їх застосовують у сільському господарстві для моніторингу полів, у будівництві для огляду майданчиків і навіть у медицині для доставки ліків у віддалені райони.

Важливим аспектом розвитку дронів є регулювання їх використання. Зростання популярності цих пристроїв створює виклики, пов'язані з безпекою, приватністю та впливом на навколишнє середовище. Тому багато країн вводять правила, що регулюють польоти дронів, зокрема обмеження висоти, відстані від населених пунктів і необхідність реєстрації апаратів.

Загалом, дрони стали символом сучасних технологій і трансформацій у багатьох галузях. Вони об'єднують досягнення в інженерії, електроніці та програмуванні, створюючи нові можливості для людства. Незважаючи на виклики, пов'язані з їх використанням, дрони продовжують удосконалюватися, стаючи все більш ефективними та універсальними.

1.2.2 Військова сфера застосування

Військові дрони стали одним із найважливіших інструментів сучасних збройних сил, революціонізуючи підхід до ведення бойових дій, розвідки та логістики. Завдяки своїм технологічним можливостям і універсальності, вони забезпечують арміям значні переваги на полі бою. Їх використовують для розвідки, цілевказівки, атаки, моніторингу та навіть доставки вантажів у складних умовах, де традиційні засоби не завжди ефективні або безпечні.

Розвідка є однією з ключових функцій військових дронів. Завдяки оснащенню високоточними камерами, тепловізорами, радарами та іншими сенсорами вони забезпечують детальний огляд місцевості в реальному часі. Дрони дозволяють відстежувати пересування противника, оцінювати бойову ситуацію та збирати стратегічно важливу інформацію без ризику для життя військових. Висока

маневровість і малопомітність роблять їх особливо ефективними для виконання місій у ворожих зонах, навіть у складних погодних умовах чи вночі.

Дрони також активно використовуються для наведення і коригування артилерійського чи ракетного вогню. Вони забезпечують точне визначення координат цілей, підвищуючи ефективність ударів і знижуючи ризик пошкодження цивільних об'єктів. Їх здатність тривалий час перебувати у повітрі дозволяє спостерігати за змінами на полі бою та оперативно передавати інформацію на командні пункти.

Озброєні дрони, або ударні БПЛА (безпілотні літальні апарати), стали важливим компонентом сучасного арсеналу. Вони можуть нести різні види озброєння, включно з гранатами, снарядами, чи навіть кулеметами, що дозволяє виконувати точкові удари по цілях. Їх використовують для знищення бронетехніки, укріплень, живої сили противника чи важливих інфраструктурних об'єктів. Застосування таких дронів зменшує ризики для пілотів і забезпечує виконання завдань із високою точністю, навіть у важкодоступних районах.

Військові дрони стали інструментом асиметричних дій, дозволяючи невеликим державам або групам ефективно протидіяти значно більшим силам. Їх застосування змінює характер бойових дій, зменшуючи залежність від великих армійських підрозділів та відкриваючи нові стратегії ведення війни. Проте зростання їхньої популярності також створює виклики, зокрема розвиток засобів боротьби з безпілотними апаратами, таких як системи радіоелектронної боротьби чи протидронові комплекси.

Одним з викликів покращення функціональності дронів в будь-яких умовах є вирішення або компенсація проблеми втрати зв'язку з дроном, будь-то викликану засобами боротьби або виходом дрону за зону досяжну сигналом керування.

Отже очевидно що наразі дрони відіграють ключову роль у військовій сфері завдяки своїй універсальності, маневровості та здатності виконувати завдання з мінімальним ризиком для людських життів. Їх використовують для розвідки, спостереження, виявлення загроз, коригування артилерійського вогню та доставки вантажів у важкодоступні або небезпечні зони. Завдяки високотехнологічним

сенсорам, камерам та системам передачі даних, дрони здатні забезпечувати оперативну інформацію, яка є критично важливою для ухвалення тактичних рішень. Крім того, автоматизація та штучний інтелект, інтегровані у дрони, дають змогу виконувати завдання з високою точністю навіть у складних умовах, таких як ніч, погана погода чи густий ліс.

Розпізнавання рухів – це один із ключових напрямів застосування дронів. Завдяки цій функції можна не просто спостерігати за об’єктами, а й аналізувати їхню поведінку в реальному часі. Це допомагає вчасно виявляти загрози: пересування техніки, підготовку до атаки чи незвичні дії людей у зоні бойових дій. Такі системи точно фіксують навіть незначні зміни, знижуючи ризик хибних тривог та підвищуючи ефективність спостереження.

Використовуючи штучний інтелект, ці рішення стають ще потужнішими – вони дозволяють військовим оперативно реагувати на будь-яку небезпеку та забезпечують додаткову безпеку в складних умовах військових операцій.

Також варто зазначити що системи ідентифікації є однією з найбільш цікавих тем, оскільки вони лежать в основі багатьох сучасних технологій і процесів. Завдяки їхньому розвитку забезпечується точне розпізнавання осіб, об'єктів або даних, що відкриває нові можливості в безпеці, автоматизації та персоналізації. Тому далі, ми будемо працювати саме з цією функцією дронів.

1.2.3 Проблеми у військовій сфері застосування

У військовій сфері дрони з алгоритмами машинного навчання допомагають швидко й точно розпізнавати цілі. Це особливо важливо в умовах бою, де швидкість ухвалення рішень може врятувати життя. Сучасні моделі, як YOLO або SSD, дають змогу дронам працювати ефективно навіть у складних ситуаціях – у темряві, під дощем чи коли цілі добре замасковані. Конструкція дронів може варіюватися від компактних моделей, які поміщаються в долоні, до великих апаратів, здатних перевозити значні вантажі.

Завдяки цьому дрони не лише допомагають зменшити ризики для військових, а й значно підвищують якість розвідки та спостереження. Це створює серйозну перевагу на полі бою.

Також на сьогоднішній день немає універсальних підходів, які могли б успішно працювати в усіх сценаріях, адже різні умови (погодні явища, висота польоту, види руху) накладають свої обмеження. Крім того, більшість існуючих досліджень зосереджені на роботі з даними з фіксованих камер, тоді як дрони додають додаткові виклики, такі як нестабільність зображення через рух самого дрону чи змінну перспективу.

Далі варто розглянути проблему із технічної точки зору обмежень потужностей обчислювання через зростання обсягів даних, різноманітність застосувань (безпека, рятувальні операції, агротехніка, військова сфера), обмеження апаратних ресурсів дронів та динамічні умови середовища. Нові сенсори (LiDAR, інфрачервоні) та алгоритми (YOLO, SLAM) створюють можливості, але потребують адаптації до конкретних завдань. Важливими також є етичні та правові аспекти, особливо у спостереженні. Вибір методів визначає ефективність, точність та швидкість роботи систем, що робить проблему актуальною в умовах стрімкого розвитку автономних технологій.

З технічного боку виклики включають обмежену обчислювальну потужність дронів, що ускладнює використання складних алгоритмів, таких як глибокі нейронні мережі. Динамічні умови середовища, включаючи погане освітлення, рухомий фон чи перешкоди, вимагають адаптивних моделей. Інтеграція даних із різних сенсорів створює додаткове навантаження на системи обробки. Висока енергоємність алгоритмів може вагомо знижувати тривалість автономної роботи дрона. Крім того, забезпечення обробки даних у реальному часі є критичною вимогою.

Одним із критичних обмежень у використанні FPV-дронів у бойових умовах є ймовірність втрати зв’язку з оператором. Це може статися через електронні засоби боротьби, природні перешкоди або вихід дрона за межі зони радіопокриття. У таких ситуаціях навіть кількасекундна затримка або повна втрата сигналу може

зробити дрон неефективним або навіть небезпечним для власних сил. Саме тому виникає потреба у створенні автономних систем донаведення, які зможуть самостійно визначати положення цілі, відстежувати її та коригувати напрямок польоту дрона без зовнішнього управління.

Застосування таких систем потребує виконання складних обчислень безпосередньо на борту дрона, що, у свою чергу, ставить під сумнів здатність мікрокомп'ютерів впоратися з такими задачами у режимі реального часу. Втім, на сьогоднішній день бракує систематичних досліджень, які б точно оцінили, наскільки подібні алгоритми (YOLOv11, DeepSORT, інші трекери) можуть працювати на платформах з обмеженими ресурсами, як-от Orange Pi 5, та яку реальну продуктивність вони забезпечують у бойових умовах.

Таким чином, наше дослідження не лише пропонує концепцію автономної системи донаведення на базі комп'ютерного зору, але й має на меті вирішити нагальну практичну проблему – надати обґрунтовані дані про продуктивність і споживання ресурсів таких рішень. Це дозволить оптимізувати архітектуру майбутніх військових дронів, зменшити витрати енергії, уникнути перевантажень системи та забезпечити більшу надійність у найважчих умовах.

Для подолання цих викликів необхідно оптимізувати алгоритми для роботи на апаратних платформах із обмеженими ресурсами, розробляти енергоефективні моделі та впроваджувати обчислення на краю (edge computing). Важливою залишається інтеграція систем штучного інтелекту, здатних адаптуватися до змін середовища в реальному часі, забезпечуючи баланс між продуктивністю та автономністю.

Однією з найактуальніших розробок у сфері використання безпілотних літальних апаратів для виявлення та розпізнавання людей є система, представлена в роботі Unmanned aerial vehicles for human detection and recognition using neural-network model [13]. У ній реалізовано підхід, що поєднує обробку повітряного відеопотоку з RGB-камер з застосуванням моделей глибокого навчання. Детекція цілей здійснюється за допомогою моделі YOLOv5, а для подальшого розпізнавання використовуються згорткові нейронні мережі, натреновані на датасеті з

позначеними діями людей (наприклад, сидить, стоїть, йде). Система демонструє високу точність на контрольованих тестових наборах, при цьому не потребує додаткової глибинної інформації.

Попри свою ефективність, описана система має важливий недолік — відсутність функціоналу автономного реагування на зміну положення об’єкта. Вона виконує лише детекцію та класифікацію, не маючи зворотного зв’язку з керуванням БПЛА. У разі втрати об’єкта з поля зору або його виходу за межі кадру, система не вживає жодних дій для утримання цілі в центрі кадру або довороту апарата. Також відсутня модульна інтеграція з прошивкою PX4, що унеможливлює автономну адаптацію траєкторії польоту дрона до динамічної ситуації.

Запропонована в нашому дослідженні система усуває ці обмеження. Вона включає механізм трекінгу на основі SORT або DeepSORT, що дозволяє відстежувати ціль у кадрі навіть при часткових перекриттях або зникненні на декілька кадрів. Крім того, інтеграція з прошивкою PX4 дозволяє формувати MAVLink-команди для автоматичного коригування положення БПЛА, доводячи реалізацію до рівня автономного донаведення. Такий підхід не тільки підвищує надійність системи, а й забезпечує її придатність для реального бойового застосування, коли втрата керування оператором або сигналу є частим явищем.

Виходячи із вище зазначеного стає очевидним що тема класифікації та розпізнавання руху об’єктів є актуальною у зв’язку із викликами сьогодення та малого обсягу напрацювань у цій області.

1.3 Огляд і аналіз наукових та літературних джерел

1.3.1 Вибір матеріалів предметної області

Для проведення дослідження критично важливо підібрати відповідні джерела, які забезпечують як теоретичну, так і практичну базу для розробки ефективних алгоритмів класифікації руху об’єктів із дронів. У ході роботи треба визначені три основні напрями, кожен із яких має власне обґрунтування вибору.

Дослідження з розпізнавання рухів (вправ) із прив’язкою до якірних точок. Цей напрям дозволяє зрозуміти методологію побудови моделей розпізнавання рухів, заснованих на ключових точках тіла. Використання якірних точок є стандартним підходом у багатьох алгоритмах комп’ютерного зору (наприклад, BlazePose, OpenPose). Ці алгоритми надають можливість точно визначати положення різних частин тіла, що критично для класифікації рухів.

Важливим аспектом роботи є дослідження методів оптимізації моделей для їхньої ефективної роботи в реальному часі. Це дозволяє усувати помилки, що виникають через зміну кута зйомки або умов освітлення, а також адаптувати алгоритми до використання на дронах, які працюють у значно складніших умовах, ніж статичні камери.

Окрему увагу слід приділити дослідженню апаратного прискорення для компактних обчислювальних платформ. Оскільки безпілотні літальні апарати мають обмежені ресурси, необхідно вивчити способи оптимізації роботи алгоритмів на таких пристроях, як Orange Pi 5+, Raspberry Pi та RockPro.

Цей напрямок досліджень дозволяє оцінити продуктивність апаратного забезпечення під час обробки алгоритмів машинного навчання, зрозуміти роль апаратного прискорення (GPU, NPU) у збільшенні швидкості аналізу даних, а також дослідити обмеження енергоспоживання та можливості зменшення навантаження на систему.

Загалом, такі підходи дозволяють створювати ефективніші, швидші та більш адаптивні алгоритми, що можуть працювати в умовах обмежених ресурсів дронів і складних зовнішніх факторів.

Результати з цього напряму дозволять обґрунтувати вибір пристроїв і побудувати стратегію розподілу ресурсів для алгоритмів.

Дослідження, пов’язані з дронами, їхнім використанням для відстеження та класифікації рухомих об’єктів, відіграють ключову роль у розробці сучасних систем аналізу поведінки в динамічному середовищі. Ефективне спостереження за об’єктами з дронів потребує врахування багатьох факторів, зокрема погодних умов, висоти польоту та швидкості руху цілі.

Аналіз існуючих наукових напрацювань у цій сфері дає можливість оцінити актуальні методи відстеження об’єктів у відеопотоці з дронів, а також розглянути технології, що допомагають компенсувати нестабільність зображення, яка може виникати через рух самого літального апарата.

Окрім цього, вивчення сучасних алгоритмів дозволяє визначити перспективні підходи до класифікації рухомих об’єктів, зокрема моделі, які здатні розрізняти різні типи активності, прогнозувати загрози або застосовуватися у військових цілях для спостереження за потенційно небезпечними об’єктами.

Таким чином можна узагальнити що обрані джерела забезпечать необхідними знаннями для вирішення основних завдань нашого дослідження. Інтеграція напрацювань у галузях розпізнавання рухів, апаратного прискорення й аналізу відеопотоків із дронів дозволяє створити оптимізовану систему, здатну працювати в складних умовах та на обмежених ресурсах.

1.3.2 Дослідження розпізнавання рухів

Дослідження «DETERMINATION AND COMPARISON METHODS OF BODY POSITIONS ON STREAM VIDEO»[8], досліджує методи визначення позицій тіла на потоковому відео та їх порівняння, з метою розробки системи, яка дозволить визначати коректність виконання вправ у режимі реального часу. У дослідженні аналізуються наявні бібліотеки для визначення поз, методи їх порівняння, а також вимірюється ефективність їх роботи за показниками швидкості, точності та відповідності. Для аналізу використано бібліотеку BlazePose, яка підтримує налаштування різних моделей і дозволяє отримувати ключові точки тіла в реальному часі.

Дослідження ґрунтується на глибокому аналізі бібліотек OpenPose, PoseNet і BlazePose. BlazePose обрана через її швидкість і здатність працювати в режимі реального часу на сучасних пристроях завдяки оптимізації MediaPipe. Основними критеріями порівняння є швидкість роботи, точність визначення поз та вплив

різних параметрів конфігурації, таких як роздільна здатність вхідного зображення, мінімальна довірча оцінка (confidence score), згладжування поз, тощо

Методологія дослідження включає побудову нормалізованих векторів на основі координат ключових точок тіла та використання трьох методів порівняння: косинусна відстань, зважена відстань і метод порівняння кутів між кінцівками. Особливу увагу приділено зваженій відстані, яка враховує довірчі оцінки визначення ключових точок. Вагома перевага цього методу полягає в тому, що більш точні точки отримують більшу вагу, забезпечуючи коректніше порівняння поз.

Значна частина роботи присвячена аналізу обмежень існуючих підходів. Наприклад, методи порівняння поз значною мірою залежать від кута зйомки, що може призводити до невідповідностей навіть у разі однакових поз. Для вирішення цієї проблеми пропонується використання тривимірного моделювання поз або камер із глибинними сенсорами, наприклад, Lidar, що дозволить знизити вплив кута зйомки та підвищити точність аналізу.

Результати експериментів демонструють, що BlazePose із використанням MediaPipe забезпечує високу швидкість і точність роботи, досягаючи понад 100 кадрів на секунду на сучасних графічних процесорах. У випадку порівняння поз метод зваженої відстані показав найкращі результати, забезпечуючи більш точне порівняння завдяки врахуванню довірчих оцінок. Однак використання цього методу все ще обмежене потребою в оптимальних умовах зйомки.

Загалом, робота демонструє значний прогрес у вирішенні завдання визначення та порівняння поз тіла на потоковому відео. Застосування BlazePose у поєднанні з методами порівняння, зокрема зваженою відстанню, відкриває нові можливості для створення ефективних і швидких систем аналізу поз у реальному часі[14].

BlazePose із MediaPipe – це передова система для розпізнавання поз людини, розроблена Google, яка забезпечує високу точність та швидкість роботи навіть на мобільних пристроях. Вона побудована на основі глибоких нейронних мереж і здатна відстежувати до 33 ключових точок тіла, включаючи голову, корпус, руки

та ноги, що значно перевищує можливості попередніх рішень, таких як OpenPose. BlazePose використовує кілька етапів обробки: спочатку модель визначає положення людського тіла в кадрі (детекція поз), потім визначає точні координати ключових точок. Цей підхід дозволяє досягати високої точності навіть у складних умовах, таких як низьке освітлення або часткове перекриття об’єктів. BlazePose також оптимізований для роботи в реальному часі, що робить його ідеальним для інтеграції у мобільні додатки, фітнес-трекери, ігри з доповненою реальністю та медичні системи для реабілітації. Завдяки високій продуктивності та відкритому коду в рамках MediaPipe, ця технологія знайшла застосування у багатьох галузях, де потрібне точне розпізнавання рухів.

Але попри значні досягнення, дослідження має свої недоліки. Перш за все, обмеження, пов’язані з залежністю точності від кута зйомки, не повністю вирішені, і навіть запропоновані рішення, як-от тривимірне моделювання або використання глибинних сенсорів, мають свої виклики. Крім того, результати значною мірою залежать від апаратних можливостей пристроїв. Використання BlazePose із MediaPipe на графічних процесорах показало високу продуктивність, але цей підхід не завжди застосовний для мобільних пристроїв таких як дрони, зі слабшою апаратною базою[15].

Алгоритми детекції та класифікації руху, такі як BlazePose або R-CNN, є дуже ресурсоємними на одноплатних комп'ютерах Raspberry Pi та Rock Pro, особливо у реальних умовах експлуатації на дронах. Основною причиною є висока обчислювальна складність обробки відеопотоку в режимі реального часу, яка включає розпізнавання до 33 ключових точок тіла, обчислення траєкторій руху та аналіз просторово-часових характеристик. Ці процеси вимагають значного навантаження на CPU, яке може використовувати до 80-100% одного ядра навіть на Rock Pro, а також потребують великого обсягу RAM (від 1.5 до 3 ГБ) для зберігання проміжних результатів. GPU, наприклад, VideoCore VI у Raspberry Pi або Mali-T860 у Rock Pro, забезпечують базове апаратне прискорення для обробки графіки, проте їхня продуктивність значно обмежена – до 50 GFLOPS для Raspberry

Pi та до 100 GFLOPS для Rock Pro, що недостатньо для складних алгоритмів, таких як R-CNN, які вимагають продуктивності на рівні 300+ GFLOPS.

Додатково, окрім роботи основного алгоритму, на одноплатних комп'ютерах одночасно функціонуватимуть інші критично важливі програми, зокрема системи управління польотом, алгоритми стабілізації, аналіз навігаційних даних та підтримка зв'язку. Це створює конкуренцію за апаратні ресурси, що призводить до зниження продуктивності та збільшення затримок. Для забезпечення стабільної роботи всієї системи може знадобитися інтеграція зовнішніх прискорювачів, таких як Google Coral TPU, які додають до 4 TOPS обчислювальної потужності, або NVIDIA Jetson, що пропонує до 21 TOPS, значно підвищуючи ефективність обробки нейронних мереж у реальному часі. Без таких прискорювачів навіть на Rock Pro підтримка обробки відеопотоку з роздільною здатністю 1080p при частоті 30 FPS буде викликом, особливо у сценаріях з високою динамікою та складними умовами, як-от погана освітленість чи задимленість.

Підводячи підсумок – дослідження «DETERMINATION AND COMPARISON METHODS OF BODY POSITIONS ON STREAM VIDEO» надає цінний інструментарій для аналізу алгоритмів визначення поз і порівняння їх точності, що може бути корисним у роботі з класифікації рухів об'єктів з дронів. Використання BlazePose, який підтримує оптимізацію через MediaPipe, дозволяє працювати в режимі реального часу і забезпечувати високу точність визначення ключових точок тіла. Методологія, заснована на порівнянні поз за допомогою косинусної відстані, зваженої відстані та кутового аналізу, допоможе адаптувати підходи для класифікації рухів на основі геометричних параметрів. Зокрема, зважена відстань, яка враховує довірчі оцінки, є перспективною для підвищення точності в умовах обмеженої видимості або складних умов зйомки, таких як рух дронів. Цей підхід може бути застосований для аналізу руху об’єктів, визначення їх стану або ідентифікації потенційних загроз. Крім того, детальний аналіз впливу параметрів конфігурації та використання 3D-моделювання поз може сприяти адаптації алгоритмів до особливостей роботи дронів.

З іншого боку, подібні алгоритми потребують значних апаратних ресурсів, що є серйозним викликом для мобільних платформ, таких як дрони. BlazePose, навіть з оптимізаціями MediaPipe, демонструє високу продуктивність (до 100 FPS) лише на потужних графічних процесорах, тоді як дрони зазвичай обладнані обмеженими за потужністю одноплатними комп'ютерами, такими як Raspberry Pi або Rock Pro. Робота алгоритму потребує значного навантаження на CPU, GPU та RAM, особливо при обробці відео високої роздільної здатності або за низької освітленості. Високі вимоги до апаратних ресурсів можуть спричинити зниження продуктивності, збільшення затримок або навіть неможливість роботи в реальному часі. Це означає, що для успішного застосування таких алгоритмів на дронах необхідно не тільки оптимізувати програмні рішення, але й забезпечити більш продуктивне апаратне забезпечення або інтегрувати додаткові апаратні прискорювачі, такі як TPU або більш потужні GPU.

Але надалі подібні проблеми з аптимізацією можуть бути нивільовані більш потужним апаратним забезпеченням.

1.3.3 Дослідження апаратного прискорення для дронів

Із зазначеного вище стає очевидним що також потрібно розглянути додаткові апаратні можливості із збільшеним ресурсом під більш складні алгоритми та зв’язки алгоритмів. Надалі буде розглянуто одноплатну систему orange pi5+.

Orange Pi 5 Plus – це високопродуктивний одноплатний комп'ютер, оснащений процесором Rockchip RK3588 з 8 ядрами (4 × Cortex-A76 та 4 × Cortex-A55) та графічним процесором Mali-G610 MP4. Ця конфігурація забезпечує значну обчислювальну потужність, що підтверджується результатами тестів.

Згідно з тестуванням, проведеним Джеффом Гірлінгом, Orange Pi 5 Plus демонструє продуктивність понад 50 GFLOPS у бенчмарку HPL, що вимірює швидкість розв'язання щільних лінійних рівнянь. Це перевищує показники Raspberry Pi 5, який досягає близько 30 GFLOPS. Ефективність Orange Pi 5 Plus

становить приблизно 4,64 GFLOPS на ват, що також перевищує показники Raspberry Pi 5 (2,75 GFLOPS/Вт) та Rock 5 Model B (4,28 GFLOPS/Вт)[16].

У тестах на компіляцію ядра Linux Orange Pi 5 Plus показав час менше 1500 секунд, тоді як Raspberry Pi 5 потребував близько 2000 секунд. При кодуванні відео 1080p Orange Pi 5 Plus досягає понад 20 кадрів за секунду, перевершуючи Raspberry Pi 5, який не досягає 18 кадрів за секунду[17].

Крім того, Orange Pi 5 Plus оснащений нейропроцесорним блоком з продуктивністю до 6 TOPS, що робить його привабливим для завдань, пов'язаних з машинним навчанням та штучним інтелектом[18].

Завдяки цим характеристикам, Orange Pi 5 Plus може розглядатися як потужна альтернатива для застосувань, що вимагають високої обчислювальної потужності, включаючи обробку відео та виконання складних алгоритмів у реальному часі.

1.3.4 Вітчизняні дослідження в області ШІ та комп’ютерного зору

Існує кілька проєктів, що спеціалізуються на класифікації руху об'єктів за допомогою камер дронів. Наприклад, у Національному технічному університеті України "Київський політехнічний інститут імені Ігоря Сікорського" було розроблено систему боротьби з дронами, яка використовує комп'ютерний зір для розпізнавання літаючих об'єктів[19].

Також у Харківському національному університеті радіоелектроніки досліджували методи та засоби керування безпілотними літальними апаратами для виявлення пожеж та визначення їхніх координат, що включає аналіз руху об'єктів на основі відеоданих з дронів.

Крім того, у Тернопільському національному технічному університеті імені Івана Пулюя було виконано магістерську роботу, присвячену розробці системи моніторингу з використанням дронів, яка включає аналіз відеопотоку для виявлення та класифікації рухомих об'єктів[20].

Ці дослідження підтверджують актуальність і перспективність використання дронів для аналізу та класифікації руху об'єктів у різних сферах, зокрема у військовій галузі. Завдяки сучасним алгоритмам і технологіям, дрони можуть не лише відстежувати рух окремих об'єктів, а й аналізувати складні динамічні процеси.

З проведених досліджень можна виділити кілька ключових напрямків. Перш за все, це адаптація методик розпізнавання об'єктів, спочатку створених для боротьби з дронами, до задач класифікації рухів у складних умовах. Це дозволяє використовувати вже напрацьовані алгоритми для нових завдань, таких як моніторинг рухомих цілей або виявлення підозрілої активності.

Ще одним важливим аспектом є застосування комп’ютерного зору для розпізнавання літаючих об’єктів. Такі технології можуть бути корисними не лише у військових операціях, а й у цивільних задачах, наприклад, для контролю повітряного простору чи моніторингу природних явищ.

Окрім цього, цікавим є досвід використання алгоритмів для аналізу складних патернів руху, наприклад, диму або поширення полум'я. Це може стати основою для розпізнавання нетипових або небезпечних рухів, що важливо як для безпеки, так і для екологічного моніторингу.

Також ведеться розробка методів для аналізу групових рухів, таких як переміщення натовпу або військової техніки, у режимі реального часу. Це дозволяє не лише відстежувати окремі об'єкти, а й аналізувати їхню взаємодію, що може бути корисним у прогнозуванні поведінки груп та прийнятті оперативних рішень.

Аналіз сучасних вітчизняних і міжнародних досліджень свідчить про високу актуальність розробки систем автономного виявлення та трекінгу об'єктів для безпілотних апаратів. Зокрема, у роботах останніх років акцент робиться на поєднанні алгоритмів глибокого навчання з методами оптимізації для роботи в реальному часі на енергозберігаючих пристроях.

Використання архітектур типу YOLO для швидкої детекції, доповнених трекінгом SORT або DeepSORT, є передовою практикою у галузі, однак потребує подальшої адаптації під обмеження військових FPV-дронів, де на перший план

виходять обмеження по вазі, енергоспоживанню та обчислювальним ресурсам. Наявні підходи ще недостатньо охоплюють інтеграцію вбудованого інтелекту безпосередньо на бортових системах керування польотом.

Оцінка новизни проведеного дослідження базується на тому, що запропонована архітектура передбачає глибоку інтеграцію між системою розпізнавання і трекінгу та автопілотом PX4 через адаптивну взаємодію за допомогою протоколу MAVLink. Такий підхід дозволяє здійснювати корекцію курсу дрона у реальному часі на основі результатів локальної обробки відеопотоку, що є суттєвим внеском у розвиток технологій автономної навігації дронів.

Таким чином, дослідження має не лише наукову цінність через використання сучасних методів комп’ютерного зору, але й практичну значущість завдяки орієнтації на реальні вимоги військової та розвідувальної практики.

1.3.5 Аналіз вибору джерел дослідження

Вибір джерел для формування теоретичного підґрунтя дослідження здійснювався з урахуванням актуальності тематики, спрямованості на вирішення задач комп’ютерного зору в реальному часі, застосування в обмежених апаратних умовах, а також на основі відповідності сучасним викликам розвитку безпілотних авіаційних систем. Для дослідження були відібрані наукові праці та статті провідних авторів і колективів, що висвітлюють питання розпізнавання об'єктів, трекінгу рухомих цілей, інтеграції інтелектуальних алгоритмів у системи керування дроном.

Одним із пріоритетів під час відбору літератури було використання джерел, що описують алгоритми, здатні працювати у режимі реального часу з обмеженим обчислювальним ресурсом, як це характерно для платформ типу Orange Pi 5 та Raspberry Pi 4. Особливої уваги заслуговують дослідження, присвячені використанню моделей YOLO різних поколінь (від v3 до v11) для задач швидкої ідентифікації об’єктів у складних сценах. Роботи з аналізу архітектур YOLO продемонстрували їхню високу ефективність та універсальність у завданнях

виявлення людей, що визначило вибір даної архітектури як базової для системи розпізнавання.

Важливим фактором було також залучення досліджень, що розглядають підходи до трекінгу об’єктів на відеопотоці. Використання трекерів типу SORT, DeepSORT дозволяє реалізувати постійне відстеження об'єктів із мінімальними обчислювальними затратами. У літературі чітко підкреслюється, що застосування трекінгових модулів на основі легких алгоритмів є необхідною умовою для підтримання стабільної роботи дронів в умовах обмежених ресурсів.

Окремо слід відзначити використання результатів новітніх досліджень SCASNet та AFEDet, які фокусуються на поліпшенні точності виявлення дрібних об’єктів у складних фонових умовах за рахунок введення просторово-контекстної уваги та адаптивного масштабного об'єднання ознак. Ці підходи є надзвичайно важливими в умовах розвідувальних операцій, де виявлення людини або об’єкта в складній обстановці є критичним для успіху завдання.

Важливо також відзначити, що обрані джерела містять не лише теоретичні викладки, але й результати експериментальних досліджень на реальних датасетах, таких як VisDrone, AI-TOD, SSDD та інші. Це дозволяє оцінити практичну ефективність розглянутих моделей і дає можливість екстраполювати результати на поставлену задачу автономного супроводження цілей FPV-дроном.

Таким чином, аналіз джерел свідчить про те, що вони охоплюють усі ключові аспекти поставленої задачі – від виявлення об’єктів і трекінгу до інтеграції системи в реальному часі та оптимізації для обмежених ресурсів. Сформована теоретична база дозволяє здійснити науково обґрунтовану розробку системи автономного розпізнавання і супроводження цілей для військових FPV-дронів.

1.4 постановка задачі

Робота на тему дослідження методів рухової активності людини за допомогою дронів спрямована на розробку підходів для автоматизованого виявлення, аналізу та класифікації рухів людини з використанням дронів. Це

дослідження охоплює застосування сучасних алгоритмів комп'ютерного зору та машинного навчання для аналізу потокового відео, отриманого з камер дронів, у динамічних та змінних умовах. Основна увага приділяється розробці систем, здатних працювати в реальному часі з обмеженими апаратними ресурсами, враховуючи специфіку польотів дронів, таких як нестабільність кадру, висота польоту та різноманітність рухів об'єктів.

Метою роботи є:

– аналіз існуючих методів виявлення та класифікації руху об'єктів;

– дослідження алгоритмів пошуку й відстеження об'єктів у відеопотоці;

– розробка прототипу інтегрованої системи для трекінгу об'єкту;

– тестування прототипу на основі апаратної платформи orange 5;

– аналіз результатів використання мінімальних ресурсів апаратного забезпечення;

– формулювання висновку.

У рамках дослідження планується досягти таких результатів:

– вибір алгоритму для пошуку об'єктів на відео: буде обрано найбільш підходящий метод для точного та швидкого виявлення об'єктів (людей) у потоковому відео, отриманому з камер дронів. оцінюватимуться алгоритми за критеріями точності, швидкості, стабільності роботи в реальному часі та можливості адаптації до складних умов;

– вибір алгоритму для класифікації поведінки об'єктів: буде досліджено й обрано алгоритм, який найкраще підходить для розпізнавання та класифікації рухів людини. увага приділятиметься методам, здатним працювати з різними типами поведінки, навіть у складних умовах, таких як нестабільна зйомка чи зміна освітлення;

– розробка інтегрованої системи: буде створено систему, яка поєднує алгоритми пошуку об'єктів і класифікації рухів у єдиний механізм. система матиме можливість працювати в реальному часі та відповідати умовам обмежених апаратних ресурсів дронів;

– вибір апаратної платформи та тестування прототипу: буде обрано відповідну апаратну платформу (наприклад, orange pi, raspberry pi, rockpro), що забезпечує оптимальний баланс між продуктивністю, енергоспоживанням і ціною. на обраній платформі буде протестовано роботу прототипу для оцінки його ефективності в реальних умовах;

– реалізація та тестування механізму донаведення дрона після втрати керування: система має забезпечити автоматичний доворот дрона в напрямку об'єкта, що був зафіксований до моменту втрати зв’язку з оператором;

– аналіз ефективності обробки відеопотоку й алгоритмів на платформі Orange Pi 5: буде визначено, чи є ресурси цієї платформи достатніми для стабільної та швидкої роботи всіх компонентів системи.

У сучасних умовах застосування FPV-дронів особливої уваги набуває питання автономності їхньої поведінки у разі втрати керування. Відомо, що під час бойових дій радіозв'язок між оператором і дроном часто може перериватися через радіоелектронні перешкоди або втрату сигналу. У таких випадках життєво важливо, щоб дрон був здатен самостійно скоригувати свій курс на вже визначену ціль без участі оператора. Це потребує реалізації функції донаведення, що включає автоматичне розпізнавання об'єкта, його трекінг та доворот платформи (дрона) у напрямку об’єкта. Одним із ключових завдань дослідження стає розробка та тестування такого механізму автоматичного коригування орієнтації дрона.

Ще одним критично важливим аспектом є технічна здійсненність реалізації повного комплексу комп’ютерного зору та трекінгу на борту дрона, зокрема на базі платформи Orange Pi 5. Через обмежені ресурси в обчислювальній потужності та енергоспоживанні постає питання, чи зможе така платформа забезпечити стабільну роботу алгоритмів виявлення, трекінгу та довороту в реальному часі. Тому окремою ціллю дослідження є експериментальне підтвердження або спростування достатності ресурсів Orange Pi 5 для виконання всіх програмних задач.

Для вирішення поставлених задач у дослідженні пропонується використати комплексний підхід, що включає кілька основних методів.

Першим етапом є збір та аналіз наукових статей, технічної документації, досліджень і доступного коду, що стосуються алгоритмів пошуку об'єктів, класифікації рухів та роботи з дронами. Зокрема, планується аналіз бібліотек BlazePose, OpenPose, MediaPipe та алгоритмів YOLO для пошуку об’єктів, що дозволить визначити найефективніші методи для вирішення поставленої задачі.

Наступним кроком стане вибір кількох перспективних алгоритмів, їхня реалізація або використання вже існуючих рішень, а також тестування на різних наборах даних. Це дасть змогу оцінити точність, швидкість та ефективність алгоритмів у реальних умовах, враховуючи обмеження дронів, такі як обмежена обчислювальна потужність і стабільність польоту. Наприклад, планується порівняння YOLOv8, YOLOv11, BlazePose і PoseNet на відеоданих із дронів.

Далі відбудеться інтеграція вибраних алгоритмів у єдину систему, яка дозволить одночасно виявляти об'єкти та класифікувати їхні рухи. Важливим аспектом дослідження стане перевірка роботи цієї системи на різних одноплатних комп’ютерах (наприклад, Orange Pi 5+, Raspberry Pi 4, RockPro), що допоможе оцінити ресурсоспоживання, продуктивність та енергозатрати. Також буде протестовано роботу BlazePose на Orange Pi 5+ із використанням апаратного прискорення NPU.

Окрему увагу приділять тестуванню готового прототипу в реальних умовах, зокрема на дроні. Це включатиме перевірку стабільності роботи, точності алгоритмів, швидкості реакції та коректності класифікації. Додатково буде оцінено точність класифікації рухів людини на відео, отриманому дроном у складних погодних умовах. У разі втрати зв’язку з оператором буде перевірено, чи здатен дрон самостійно скоригувати напрямок на виявлений об’єкт.

На завершальному етапі відбудеться аналіз і порівняння отриманих результатів з іншими відомими підходами або альтернативними реалізаціями.

Запропонований підхід дозволяє систематично дослідити всі аспекти теми, враховуючи технічні, теоретичні та практичні вимоги, і забезпечує якісне виконання поставлених завдань.

2 ОПИС ПРИЙНЯТИХ ПРОЕКТНИХ РІШЕНЬ

2.1 Проектування архітектури системи

У дослідженні детекції та класифікації руху, формування файлової структури системи є важливим етапом, який забезпечує систематизацію, узгодженість та прозорість усіх процесів розробки та аналізу. Щоб спростити роботу над проєктом та уникнути плутанини, ми створили зрозумілу файлову структуру системи. Вона допомагає впорядкувати всі елементи розробки — дані, скрипти, моделі та звіти — і легко орієнтуватися серед них.

Чітко визначена структура надає змогу систематизувати всі експериментальні дані, скрипти для обробки й тренування моделей, результати тестувань та звіти. Такий підхід забезпечує логічну організацію та дозволяє уникнути плутанини при роботі з великим обсягом інформації. Крім того, структурована система є основою для документування та аналізу проведеної роботи.

Організація проекту у вигляді модулів дозволяє виділити ключові аспекти, такі як тренування моделей, класифікація рухів, аналіз умов зйомки чи тестування системи. Це спрощує інтеграцію нових компонентів або покращення існуючих. До того ж, така структура забезпечує узгодженість у виконанні всіх етапів дослідження та полегшує оцінювання результатів. Вона дозволяє чітко описати, де зберігаються результати експериментів, як проводяться обчислення та які алгоритми використовуються.

Таким чином, створення продуманої файлової структури є невід’ємною частиною дослідження, яка забезпечує цілісність та якість виконаної роботи, а також надає змогу чітко описати всі етапи проекту.

Далі варто розробити структуру програмної системи, яка має на меті створення автоматичного виявлення, класифікації та аналізу рухів об'єктів у різних умовах зйомки. Основна функціональність яка буде розподілена по скриптах та функціональних елементах системи включає:

– детекція об'єктів: система виявляє об'єкти на зображеннях або відео за допомогою алгоритмів на основі yolo (you only look once). Детекція включає ідентифікацію положення та категорії об'єкта;

– класифікація рухів: після детекції система аналізує послідовність кадрів для визначення типу руху (наприклад, біг, їзда, стрибки). використовуються моделі машинного навчання для класифікації;

– аналіз умов зйомки: система враховує фактори, які можуть впливати на якість розпізнавання, такі як туман, дим, теплові камери або звичайне денне освітлення. відповідні моделі навчаються для корекції роботи в цих умовах;

– аугментація даних: автоматичне розширення датасетів через трансформацію зображень для покращення стійкості моделей;

– тестування та звітність: забезпечення функцій для оцінки точності та помилок системи. генерація звітів із результатами детекції та класифікації;

– обробка помилок: аналіз і корекція помилок моделей для покращення точності роботи системи[22].

Нижче наведено структуру проекту(див. рис. 2.1) та окремо папки класифікатору реалізовану в PyCharm(див. рис. 2.2).

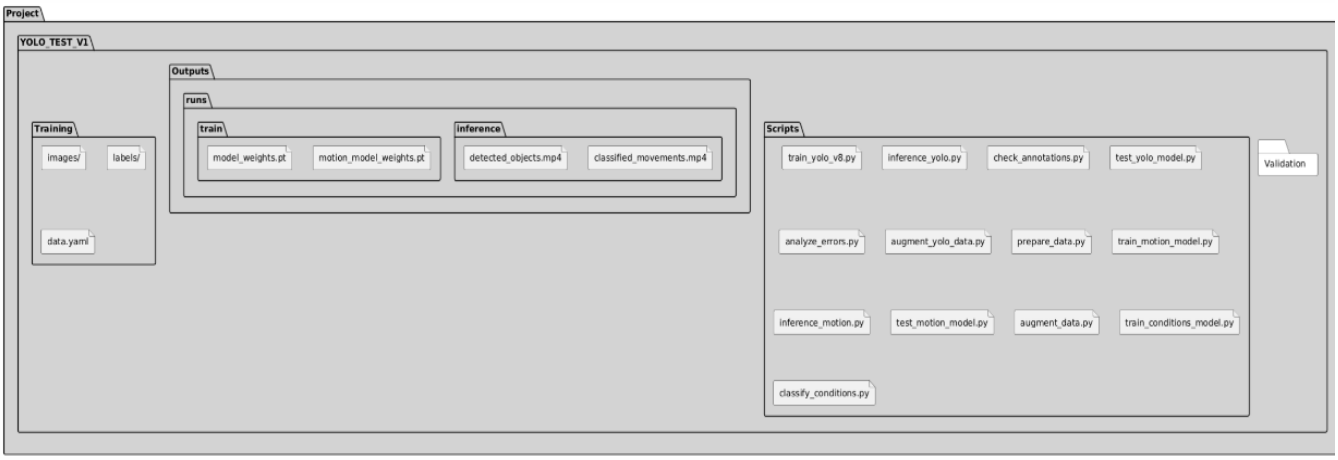

Рисунок 2.1 – Файлова структура проекту(рисунок виконано самостійно)

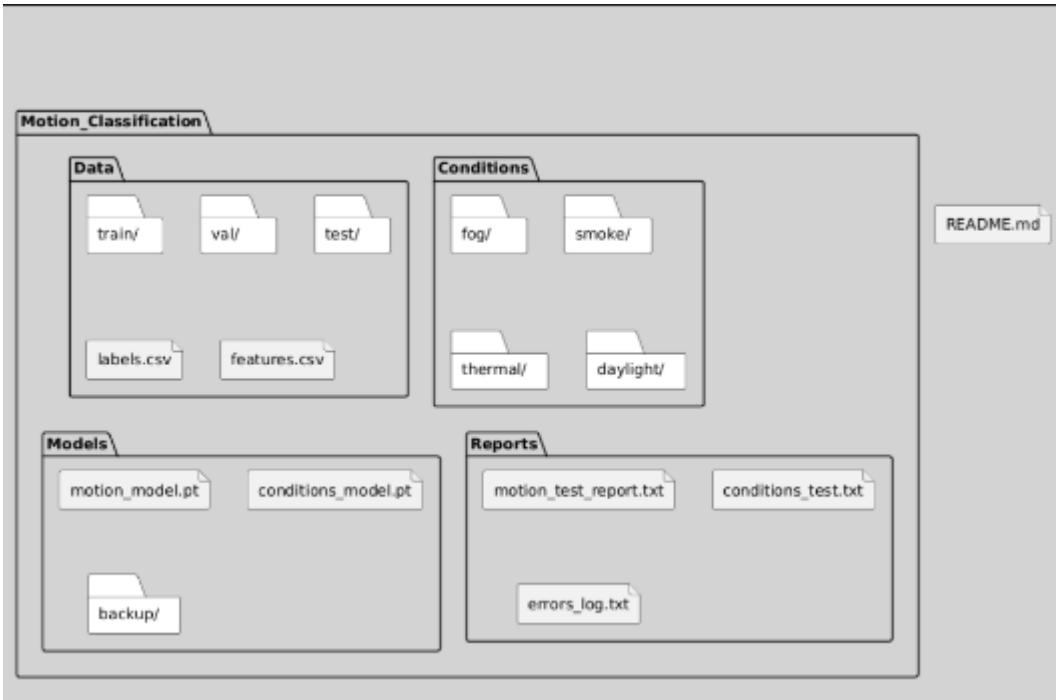

Рисунок 2.2 – Файлова структура класифікатору(рисунок виконано самостійно)

Ця частина проекту відповідає за детекцію об'єктів за допомогою YOLO-моделі. Далі варто розібрати кожен елемент окремо.

Training:

– images/: Папка зображень, на яких проводиться навчання моделі YOLO;

– labels/: Папка анотацій для навчальних зображень у форматі YOLO;

– data.yaml: Конфігураційний файл для навчання YOLO, що описує структуру даних.

Validation:

– images/: Зображення для валідації моделі;

– labels/: Анотації для валідаційних зображень.

Outputs:

– runs/train/;

– model_weights.pt: Збережені ваги моделі YOLO після тренування;

– motion_model_weights.pt: Ваги моделі для класифікації рухів;

– runs/inference/: detected_objects.mp4: Відео з результатами детекції об'єктів;

– classified_movements.mp4: Відео з результатами класифікації рухів.

Scripts:

– train_yolo_v8.py: Скрипт для навчання YOLO-моделі;

– inference_yolo.py: Скрипт для виконання предикцій YOLO на відео;

– check_annotations.py: Скрипт перевірки коректності анотацій;

– test_yolo_model.py: Тестування точності й продуктивності YOLO;

– analyze_errors.py: Аналіз помилок, допущених моделлю YOLO;

– augment_yolo_data.py: Аугментація даних для покращення навчання YOLO;

– prepare_data.py: Підготовка даних для класифікації рухів;

– train_motion_model.py: Навчання моделі для класифікації рухів;

– inference_motion.py: Скрипт для виконання предикцій класифікації рухів;

– classify_conditions.py: Класифікація умов зйомки (туман, дим тощо).

Motion_Classification – ця секція проекту призначена для класифікації рухів об'єктів.

Data:

– train/: Данні для навчання моделі класифікації;

– val/: Дані для валідації;

– test/: Дані для тестування;

– labels.csv: Файл з мітками для класифікації рухів;

– features.csv: Файл з виділеними ознаками для рухів.

Conditions:

– fog/: Дані зйомки в умовах туману;

– smoke/: Дані зйомки в умовах диму;

– thermal/: Дані з теплової камери;

– daylight/: Дані при звичайному денному освітленні.

Outputs:

– runs/train/: логи і ваги моделі для класифікації рухів;

– runs/inference/: результати класифікації рухів або умов зйомки.

Models:

– motion_model.pt: Модель класифікації рухів;

– conditions_model.pt: Модель для визначення умов зйомки;

– backup/: Резервні копії моделей.

Reports:

– motion_test_report.txt: Звіт про тестування моделі класифікації рухів;

– conditions_test.txt: Звіт про тестування моделі класифікації умов зйомки;

– errors_log.txt: Лог-файл з описом помилок моделей.

2.2 Аналіз даних для навчання моделі

Для вибору оптимального дата-сету для навчання – потрібно вибрати найбільш популярні сервіси яки надають подібні матеріали та обрати один або декілька для подальшої роботи.

VisDrone Dataset – це один із наймасштабніших і найвідоміших наборів даних, створених спеціально для задач комп’ютерного зору, орієнтованих на зображення та відео, отримані з дронів. Цей датасет був розроблений для тренування та оцінювання алгоритмів детекції, відстеження й класифікації об’єктів у складних і динамічних умовах. VisDrone містить великий обсяг даних, що охоплює різноманітні сцени, зокрема міські райони, дороги, парки, сільську місцевість і густо населені території, що робить його придатним для широкого спектра застосувань: від моніторингу транспорту до аналізу поведінки людей.

Вміст датасету включає як зображення, так і відеопотоки, з анотаціями об’єктів у кожному кадрі, які описують позиції, типи об’єктів і їхні траєкторії. Це дозволяє використовувати його для різних задач, таких як детекція об’єктів, трекінг

у реальному часі, оцінка щільності натовпу та аналіз поведінки транспортних засобів.

Одна з головних переваг VisDrone – це те, що датасет враховує реальні виклики, пов’язані з роботою дронів: різноманітні кути зйомки, зміни висоти польоту, складність освітлення та присутність перекриття об’єктів. Це робить його ідеальним для розробки та тестування алгоритмів, здатних працювати в умовах, що максимально наближені до реальних сценаріїв. Завдяки своїй популярності VisDrone став стандартом для оцінювання продуктивності алгоритмів у сфері комп’ютерного зору, пов’язаного з дронами, і надає унікальну можливість розробникам і дослідникам оптимізувати свої моделі для роботи у складних умовах(див. рис. 4.2).

Рисунок 4.2 – Приклад виділення обєктів декількох класів[14]

Це зображення демонструє приклад виявлення об'єктів на зображеннях де об'єкти анотовані за допомогою обмежувальних рамок. У наборі даних представлений широкий спектр зображень, отриманих із різних місць, оточень та з різною щільністю, що полегшує розробку моделей для цього завдання.

Цей приклад демонструє різноманітність та складність даних у наборі даних VisDrone та підкреслює важливість високоякісних сенсорних даних для завдань комп'ютерного зору на основі безпілотників.

Одним із найбільш відомих аналогів VisDrone Dataset є UAVDT (Unmanned Aerial Vehicle Detection Dataset). Це також великий набір даних, спеціально створений для задач комп’ютерного зору на основі відео й зображень, отриманих із дронів. UAVDT спрямований на вирішення задач детекції, трекінгу та класифікації об’єктів у складних умовах.

UAVDT включає дані, зібрані з дронів у різних середовищах, таких як автостради, міські райони та сільська місцевість, і містить анотації для більш ніж 23 000 кадрів. Усі кадри супроводжуються детальними мітками об’єктів, такими як автомобілі, вантажівки та автобуси, що дозволяє використовувати датасет для трекінгу рухомих транспортних засобів. UAVDT також враховує складнощі, які притаманні відеозйомці з дронів: зміни висоти польоту, кута огляду, погодних умов (сонце, дощ, хмари) та шуму від руху дронів.

Основною перевагою UAVDT є його багатофункціональність: датасет може використовуватися для навчання моделей як для простих задач детекції об’єктів, так і для складних задач трекінгу в реальному часі. Це робить його дуже схожим на VisDrone, але UAVDT краще адаптований для аналізу транспортного руху та роботи з трекінгом на великих автострадах.

Обидва датасети, VisDrone та UAVDT, доповнюють одне одного й можуть використовуватися залежно від потреб: VisDrone орієнтований на детекцію в ширшому спектрі середовищ, тоді як UAVDT більше підходить для трекінгу транспортних засобів у складних умовах.

2.3 Вибір алгоритму детекції та класифікації руху

Одним із головних завдань при виборі моделі комп’ютерного зору для FPV-дронів є забезпечення максимального балансу між продуктивністю та якістю розпізнавання. Серед сучасних алгоритмів об’єктної детекції особливої уваги

заслуговують три моделі: YOLO (You Only Look Once), SSD (Single Shot Multibox Detector) та Faster R-CNN. Кожна з них має власну архітектурну логіку, яка визначає її застосовність у конкретних сценаріях, особливо у випадку розміщення на обмежених за ресурсами edge-пристроях, таких як Orange Pi 5 або Raspberry Pi.

Faster R-CNN є класичним представником двоетапних моделей, що забезпечують найвищу точність детекції завдяки розділенню процесу виявлення об'єктів і класифікації. У першому етапі мережа формує регіони інтересу (ROI) за допомогою Region Proposal Network, після чого проводить уточнення меж і класифікацію. У складних умовах — наприклад, при слабкому освітленні або високій щільності об’єктів — ця модель забезпечує точність на рівні 95–97%, що є золотим стандартом у багатьох офлайн-сценаріях. Проте її найбільшим недоліком є надзвичайно високе навантаження на GPU, що унеможливлює її застосування на борту дрона в режимі реального часу.

Модель SSD також виконує об’єктну детекцію в один прохід, проте на відміну від YOLO, вона використовує багаторівневі карти ознак, кожна з яких відповідає за різний масштаб об’єктів. Її основна перевага полягає в тому, що вона працює швидше, ніж Faster R-CNN, з меншою втратою точності. Проте саме на малих об’єктах, які часто з’являються у відео з дронів (наприклад, люди з великої висоти), модель починає втрачати здатність до надійного розпізнавання, що підтверджується у дослідженнях [3,4].

У цьому контексті YOLO виявляється найбільш збалансованим рішенням. Модель демонструє оптимальний компроміс між швидкістю, точністю та стабільністю, що критично важливо в автономних системах, де необхідна обробка відео в реальному часі. Особливої уваги заслуговує YOLOv8 і вище, де архітектура була оптимізована з використанням модулів PANet, SPP та інших механізмів агрегації ознак, що дозволяє моделі утримувати точність у 88–93% при FPS понад 30 навіть на edge-пристроях без потужного GPU.

Перевага YOLO полягає саме в підході до обробки зображення: мережа не витрачає час на попереднє формування областей інтересу, а одразу прогнозує координати об’єктів та їхні класифікації. Це дозволяє зменшити обчислювальні

витрати майже вдвічі у порівнянні з Faster R-CNN та до 30% — у порівнянні з SSD. Такий підхід забезпечує реальну працездатність на мікрокомп'ютерах, де доступні лише CPU або NPU (Neural Processing Unit).

У статті [5] описано використання YOLOv9 у потоковому відео з дронів, де підтверджено її здатність обробляти нестабільні кадри зі змінною перспективою без втрати точності. Це вказує на високу робастність моделі до динамічних змін сцени, що є критично важливим у FPV-сценаріях, де дрон постійно змінює положення, а кадр схильний до тремтіння або розмиття.

Крім того, YOLO успішно масштабується. Завдяки широкій екосистемі та підтримці оптимізованих варіантів (наприклад, YOLO-Nano, YOLO-Fastest, YOLOv5-Lite) модель може бути адаптована до будь-якого пристрою — від NVIDIA Jetson до Orange Pi. Для нашої системи важливо, що версії YOLOv8/YOLOv11 уже мають попередню інтеграцію з TensorRT та OpenVINO, що дозволяє ефективно використовувати NPU на Orange Pi 5.

Таким чином, YOLO виступає не просто як компроміс, а як свідомий вибір на користь високої якості детекції при мінімальних вимогах до апаратного забезпечення. Це дає змогу забезпечити автономну роботу дрона навіть після втрати зв'язку, оскільки всі обчислення — включно з аналізом, виявленням і трекінгом — виконуються локально на борту. У рамках нашого дослідження це дозволяє реалізувати механізм донаведення без зовнішнього втручання.

На завершення слід зазначити, що порівняльний аналіз підтвердив вибір YOLO як основи для системи детекції у даному проєкті. Вона демонструє високу точність при обмеженому енергоспоживанні, короткий час відгуку та низькі вимоги до апаратної частини, що робить її придатною для використання на військових FPV-дронах.

Також варто відмітити що у проекті буде задіяний YOLO11, це нова версія цього сімейства алгоритмів. На рисунку 4.3 продемонстровано ефективність нової версії у порівняння із попередниками. Варто також зазначити що із цією версією не існує на даний час жодних досліджень пов'язаних із детекцією та трекінгом об'єктів із камери дрона[23].

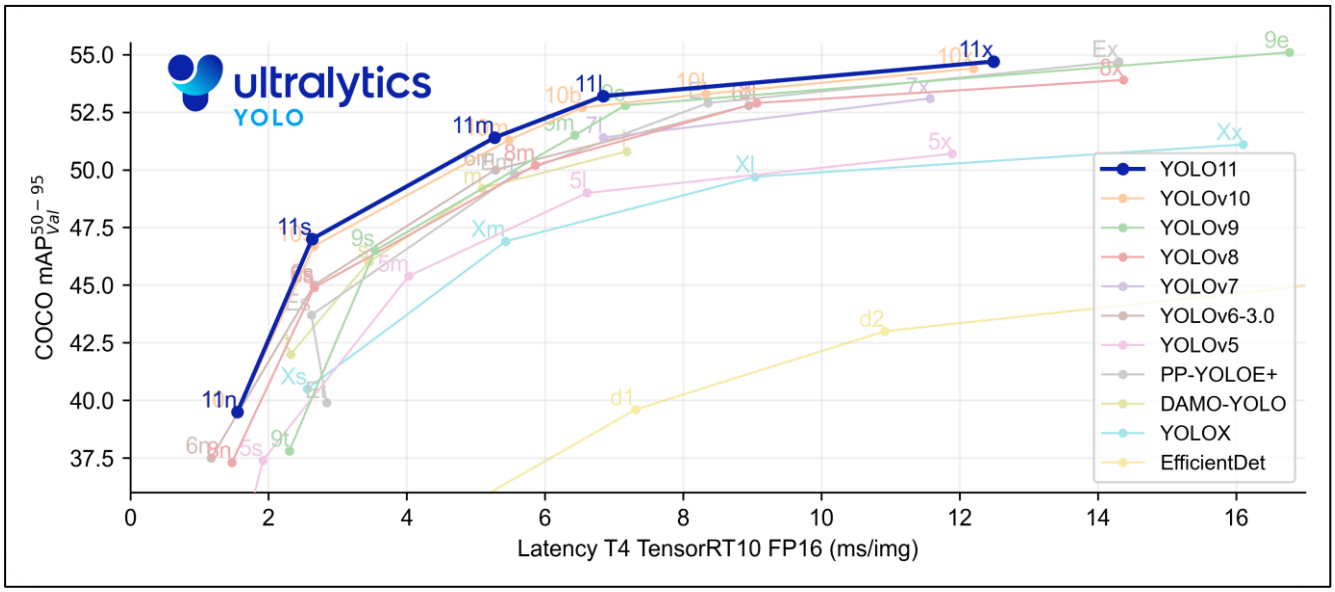

Рисунок 2.3 – Ефективність нової версії YOLO11[23]

YOLO11 демонструє виняткову ефективність серед алгоритмів детекції об'єктів завдяки значним покращенням як у швидкості, так і в точності. На графіку видно, що YOLO11 досягає найвищого рівня точності (понад 52% COCO mAP$_{50–95}$) серед усіх версій YOLO та конкурентів, таких як EfficientDet і YOLOX, зберігаючи низьку затримку (2-6 мс/зображення). Це означає, що модель здатна працювати в реальному часі без компромісу між швидкістю й точністю, що є критично важливим для задач, пов'язаних із відеопотоком, наприклад, з дронів.

Порівняно з попередніми версіями, такими як YOLOv8 чи YOLOv9, YOLO11 забезпечує кращу продуктивність завдяки оптимізації архітектури, вдосконаленим алгоритмам обчислення та більш точному врахуванню контексту об’єктів. Ці вдосконалення роблять YOLO11 ідеальним вибором для нашого дослідження, оскільки модель дозволяє працювати з високою адаптацією до різних сцен і ефективно використовувати обмежені апаратні ресурси дронів.

2.4 Класифікація руху об'єкту та наукова новизна

Для реалізації класифікації дій обєкту – необхідно буде вибрати алгоритм або потенційний алгоритм який буде створено згодом, чи поєднання яких би давало змогу після отримання класу обєкту, визначати місцевість, порівнювати траєкторію чи характер руху із еталонними значеннями та такі алгоритми які б з різних ракурсів аналізували якорні точки та усе це працювало за принципом розгалудження і виводи ло б інформацію на екран про гілку яка буде означати дію обєкту в поточний час відео спостереження.

Для реалізації системи, яка дозволяє аналізувати класи об’єктів, визначати місцевість, порівнювати траєкторію чи характер руху із еталонними значеннями, а також оцінювати якірні точки з різних ракурсів, оптимальним підходом буде багаторівневий алгоритм із розгалуженою архітектурою. Перший рівень базується на YOLO (наприклад, YOLOv8 або YOLOv11), який забезпечує швидке і точне визначення класу об'єкта та координат, а також дозволяє оцінити тип місцевості через класифікацію фону кадру.

Наступний рівень відповідає за аналіз траєкторії об'єкта за допомогою Dynamic Time Warping (DTW), який порівнює рух із еталонними шаблонами (наприклад, лінійний чи хаотичний рух), у поєднанні з Kalman Filter для прогнозування подальших дій[24].

Третій рівень здійснює аналіз якірних точок через BlazePose або OpenPose, визначаючи положення частин тіла або інших ключових точок, а також враховує зміни ракурсу за допомогою тривимірного моделювання. Усі отримані дані обробляються через блок логіки розгалужень, який на основі інформації про клас, траєкторію та якірні точки будує гілку дій об’єкта.

Наприклад, система може виводити інформацію: "Об’єкт – людина, рухається зигзагоподібно, виконує біг у нестабільному напрямку". Така архітектура дозволяє не лише оцінювати поточний стан об'єкта, але й робити висновки про його поведінку в реальному часі, забезпечуючи високу адаптивність до різних типів об’єктів, складних умов зйомки та обмежених апаратних ресурсів.

3 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

3.1 Огляд технологічного стеку та вибір інструментів

Розробка системи аналізу рухів людини за допомогою дронів потребує ретельного вибору технологічного стеку, що забезпечить високу точність розпізнавання, ефективну обробку відеоданих та оптимальне використання обчислювальних ресурсів. Основна вимога до обраних інструментів – здатність працювати в реальному часі на вбудованих пристроях із обмеженими обчислювальними можливостями.

Основою системи є використання сучасних алгоритмів комп’ютерного зору та глибокого навчання, які дозволяють ефективно визначати ключові точки людського тіла, аналізувати пози та класифікувати рухи. Для цього застосовуються BlazePose, OpenPose та MediaPipe, що відзначаються високою точністю та ефективністю в режимі реального часу. Ці технології забезпечують детальне розпізнавання положення тіла людини, навіть у складних умовах зйомки, таких як змінне освітлення або часткове перекриття об'єкта. Додатково для задач виявлення об’єктів використовується YOLOv11, що гарантує високу швидкість і точність ідентифікації об’єктів на зображенні, що є критично важливим при аналізі динамічних сцен.

Для реалізації проєкту використовуються одноплатні комп’ютери Orange Pi 5+, Raspberry Pi 4, RockPro, які є оптимальними для роботи вбудованих систем. Особливою перевагою Orange Pi 5+ є наявність Neural Processing Unit (NPU), що забезпечує прискорене виконання операцій машинного навчання та глибокого навчання. Це дозволяє досягти ефективної обробки відеопотоку без значного навантаження на центральний процесор, що є ключовим фактором для забезпечення стабільної роботи в умовах обмежених ресурсів. Raspberry Pi 4 використовується для тестування базових алгоритмів та відлагодження програмного забезпечення, тоді як RockPro застосовується для розширених сценаріїв, що потребують додаткової продуктивності.

Програмне забезпечення проєкту включає сучасні мови та фреймворки, що забезпечують гнучкість та ефективність реалізації. Основна розробка ведеться мовою Python, що має широку екосистему бібліотек для роботи з комп'ютерним зором та машинним навчанням. Додатково використовується C++ для оптимізації обчислювально інтенсивних операцій, що дозволяє зменшити час обробки відеопотоку та підвищити загальну продуктивність системи. Для реалізації алгоритмів комп'ютерного зору використовуються OpenCV, TensorFlow та PyTorch, що є провідними фреймворками у сфері глибокого навчання. Обробка відеопотоку здійснюється за допомогою GStreamer та FFmpeg, що дозволяє ефективно кодувати та декодувати відеодані в режимі реального часу.

Інтеграція компонентів системи виконується таким чином, щоб забезпечити модульність і можливість адаптації до різних сценаріїв використання. Алгоритми пошуку об'єктів і класифікації рухів поєднуються в єдину систему, здатну працювати в реальному часі навіть за обмежених ресурсів дронів. Архітектура рішення передбачає розподілену обробку даних, де попередній аналіз відеопотоку здійснюється безпосередньо на борту дрона, а глибший аналіз із залученням нейронних мереж може виконуватися на зовнішньому сервері або більш продуктивному пристрої. Розробка виконується з урахуванням принципів EdgeAI, які передбачають виконання нейромереж на краю мережі з мінімальним споживанням ресурсів [5].

Для проведення експериментального тестування та визначення ефективності роботи розробленого програмного рішення було використано тестову систему на базі процесора AMD Ryzen 5 7600X, відеокарти NVIDIA RTX 4070 Super та оперативної пам'яті обсягом 32 ГБ (DDR5). Дана конфігурація була обрана як продуктивна десктопна платформа, що забезпечує значні ресурси для роботи із сучасними методами комп'ютерного зору, зокрема моделей сімейства YOLO та супутніх алгоритмів обробки відеоданих у реальному часі.

Процесор Ryzen 5 7600X містить 6 фізичних ядер (12 логічних потоків) архітектури Zen 4 із номінальною частотою 4.7 ГГц, що забезпечує високу обчислювальну спроможність для паралельної обробки даних. Графічний адаптер

RTX 4070 Super був обраний через підтримку апаратного прискорення CUDA, необхідного для ефективної роботи нейромережевих моделей, та наявність 12 ГБ швидкої відеопам’яті типу GDDR6X, що суттєво впливає на продуктивність при роботі з великими тензорними структурами.

Оперативна пам’ять у 32 ГБ стандарту DDR5 дозволила комфортно виконувати ресурсомісткі завдання, зокрема одночасно утримувати модель YOLO, відео, проміжні обчислення та результати. Таким чином, запропонована тестова платформа повністю відповідає вимогам щодо продуктивності та дозволяє якісно оцінити ефективність розроблених алгоритмів у реальних сценаріях використання, створюючи необхідний запас потужності для коректного аналізу отриманих результатів експериментів.

Інструменти, технології та бібліотеки, які використовує наша тестова система, та їхнє призначення:

Python – головна мова програмування, що використовується для написання програмного рішення завдяки простоті синтаксису та широкому спектру доступних бібліотек для машинного навчання.

PyTorch (torch, torchvision) – фреймворк для машинного навчання та глибокого навчання, що дозволяє створювати, навчати та використовувати нейромережеві моделі, зокрема YOLO.

Ultralytics YOLO – сучасна реалізація нейромережі YOLO (You Only Look Once), яка використовується для детекції та розпізнавання об’єктів у реальному часі на відеопотоці.

OpenCV (opencv-python) – бібліотека для роботи з зображеннями та відео, яка використовується для читання відеоданих, попередньої обробки кадрів та візуалізації результатів роботи YOLO.

Deep SORT Realtime – бібліотека для трекінгу об’єктів у відеопотоці, що забезпечує стійкий до втрати кадрів супровід ідентифікованих об'єктів між послідовними кадрами.

FilterPy – бібліотека, що реалізує алгоритм фільтра Калмана, який використовується для точнішого трекінгу й прогнозування руху об'єктів на відео.

SciPy – бібліотека для наукових та інженерних розрахунків, яка підтримує допоміжні математичні операції, зокрема обчислення, які використовуються при обробці сигналів та відео.

NumPy – бібліотека для роботи з багатовимірними масивами та матрицями, яка використовується як базовий інструмент для обробки й передачі даних нейромережевим моделям.

psutil – бібліотека для моніторингу системних ресурсів, що застосовується для відстеження споживання процесорного часу, оперативної та відеопам'яті, необхідного для оцінки продуктивності програмного рішення.

CUDA (через NVIDIA RTX 4070 Super) – технологія для паралельних обчислень на графічних процесорах від NVIDIA, використовується для значного прискорення обчислень нейронних мереж, таких як YOLO. –

Зазначені технології забезпечують повний цикл роботи системи: від обробки вхідних відеоданих, ідентифікації та відстеження об’єктів, до моніторингу ресурсів і оцінки продуктивності.

Таким чином, вибір даного технологічного стеку дозволяє створити ефективну та адаптивну систему аналізу рухів людини, яка працює на сучасних вбудованих обчислювальних платформах та використовує передові алгоритми комп’ютерного зору. Комплексний підхід до вибору апаратних і програмних засобів забезпечує високу продуктивність, стабільність роботи в реальному часі та можливість подальшого розширення функціоналу системи відповідно до майбутніх дослідницьких та практичних задач.

3.2 Архітектура та компоненти системи

3.2.1 Компоненти системи

Наша система об’єднує декілька модульних блоків. Вона приймає кадри з камери або відео, детектує об’єкти, асоціює їх у часі, генерує поправки керування yaw та відстежує використання апаратних ресурсів.

Компоненти системи:

– детектор (yolo) – відповідає за швидку локалізацію об’єктів у кожному кадрі; повертає координати боксу, довіру та клас;

– трекер (deep sort) – отримує від детектора нові бокси або пустий список і за допомогою калман-фільтра зберігає та прогнозує траєкторію об’єкта через короткі втрати детекції;

– генератор px4-команд – перетворює обчислену помилку по осі yaw (і за потреби швидкість вперед) у текстові заглушки mavlink-повідомлень для керування дроном;

– головний модуль (main.py) – налаштовує параметри, запускає захват відео, ініціалізує детектор і трекер, реалізує стейт-машину;

– idle/tracking, візуалізує бокси й стрілки та викликає генерацію команд;

– монітор ресурсів (resource_monitor.py) – запускає процес інференсу як підпроцес, збирає і виводить час cpu, пік пам’яті ram і gpu для оцінки продуктивності.

Підхід із окремими модулями забезпечує гнучкість: можна замінити детектор, налаштувати трекер або змінити формат команд під різні протоколи. Основна концептуальна ідея – виклик update_tracks кожен кадр із можливістю передати пустий список, щоб трекінг не обривався під час короткочасних пропусків детекції. Це дозволяє досягти стійкого відстеження та коректної генерації команд керування дроном. Датальніше принцип взаємодії компонентів наведено на рисунку 3.1.

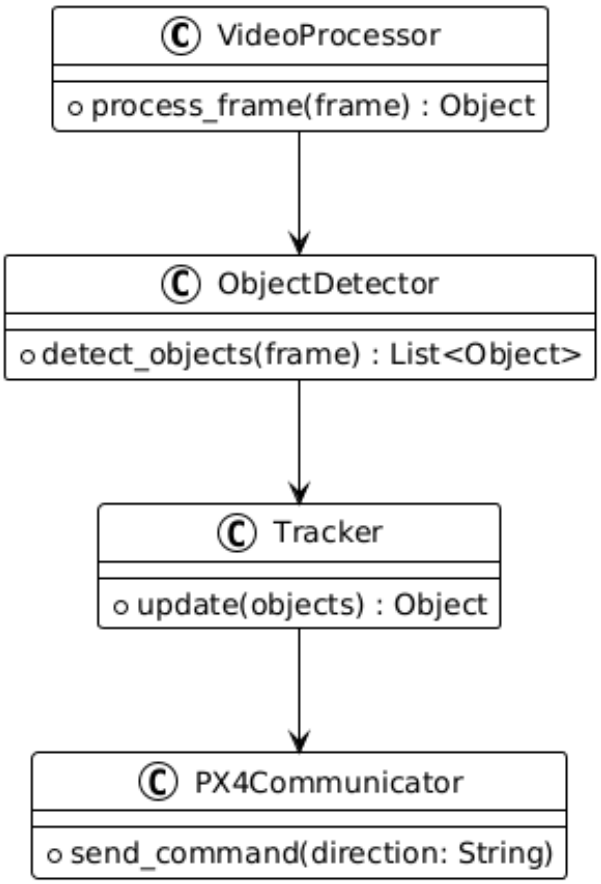

Рисунок 3.1 – Архітектура системи у контексті технологічного стеку(рисунок виконано самостійно)

Передача команд від мікрокомп’ютера до емулятора PX4 здійснюватиметься через протокол MAVLink, який забезпечує стандартизований обмін даними між керуючим модулем і бортовими системами дрона. Для зміни напряму руху використовуватиметься SET_POSITION_TARGET_LOCAL_NED. На основі даних від YOLO та алгоритмів трекінгу мікрокомп'ютер визначатиме корекцію траєкторії, що дозволить безперервно утримувати об'єкт у кадрі. Це дає змогу гнучко змінювати траєкторію відповідно до даних системи розпізнавання, а не лише коригувати курс у горизонтальній площині. Алгоритм буде розглянуто пізніше.

Процес взаємодії між компонентами відбуватиметься у кілька основних етапів:

– ObjectDetector аналізує відеопотік і виявляє цілі (людей або теплові сигнатури).

– Tracker відстежує рух виявлених об'єктів, визначаючи їх зміщення у просторі (вліво-вправо, вгору-вниз, вперед-назад).

– На основі отриманих даних PX4Communicator формує команду руху (turn_left, turn_right, move_up, move_down, move_forward, move_backward).

Команда передається через MAVLink у PX4, який виконує відповідну корекцію траєкторії у реальному часі, у тестовій системі буде вивід команд у консоль у форматі доступному для MAVLink у PX4.

3.2.2 Файлова структура

Нижче наведена файлова структура проекту(див.рис.3.2).

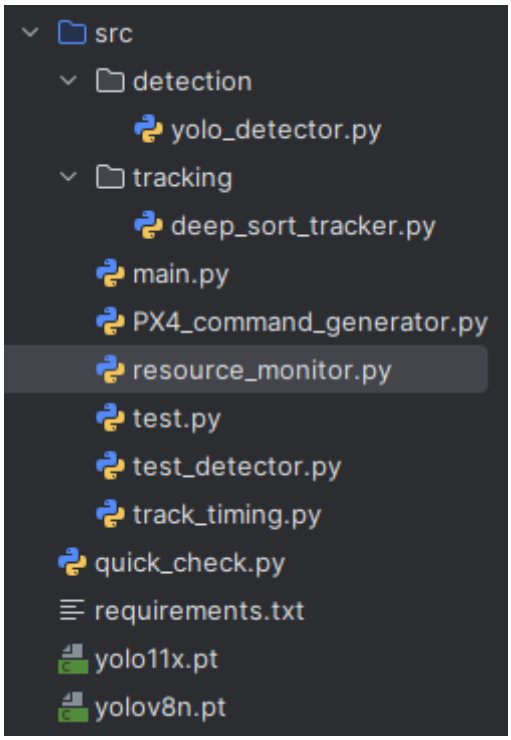

Рисунок 3.2 – Файлова структура додатку(рисунок виконано самостійно)

У корені проєкту розміщені дві теки віртуальних середовищ (.venv і .venv311), папка data з тестовими відеофайлами (Test1_tree_human_run.mp4, Test1_tree_human_run2.mp4) та теку src, в якій реалізована вся логіка. У src

перебувають дві підтеки: detection із файлом yolo_detector.py для запуску YOLO-детектора, та tracking із файлом deep_sort_tracker.py, який обгортає Deep SORT. Безпосередньо в src лежать скрипти main.py (головний модуль, що зв’язує детекцію, трекінг і генерацію команд), PX4_command_generator.py (модуль для формування “реальних” MAVLink-повідомлень у текстовому вигляді), resource_monitor.py (утиліта для моніторингу CPU/GPU/RAM під час запуску pipeline), test.py і test_detector.py (юніт-тести та демонстраційні приклади), track_timing.py (скрипт для заміру швидкості детекції й трекінгу) та quick_check.py (швидка перевірка встановлених пакетів). У корені також знаходиться файл requirements.txt зі списком залежностей.

3.2.3 Архітектура Embaded системи

Також варто розглянути структуру вбудованої системи.
Вбудована (embedded) версія цієї системи матиме такі відмінності та додаткові компоненти. По-перше, замість повноцінної Linux-машини підходить на вибір RTOS (наприклад NuttX для PX4 або FreeRTOS на апаратній платі) з жорстким планувальником, щоб гарантувати детерміністичний каденс обробки відео.

Друга велика різниця — inference-движок: замість «важкого» PyTorch на CPU/GPU десктопа використовують оптимізовані рушії (TensorRT на NVIDIA Jetson, OpenVINO на Intel-платах, або Coral-TPU API), що дають прискорення на апаратних NPU/TPU. Третій аспект — інтерфейси до датчиків і контролера польоту: замість OpenCV відеопотоку через файл і камеру на десктопі, вбудована система приймає RTSP/CSI-стрім з камери, може використовувати GStreamer-пайплайн із апаратним декодером, а об’єднує компоненти через middleware (наприклад ROS 2 DDS або нативний uORB/PX4-MAVLink).

Ще одна важлива відмінність — канал телеметрії: в тесті ми «відправляємо» лише текст у консоль, а у вбудованому рішенні формуються реальні MAVLink-повідомлення (COMMAND_LONG, SET_POSITION_TARGET_LOCAL_NED тощо) і надсилаються по UART або UDP до польотного контролера PX4. Нарешті,

щоб зменшити затримки, вбудована система об’єднує всі етапи в один C++/C-сервіс із мінімальною кількістю копіювань кадрів, використовує lock-free черги або shared memory для обміну даними між inference-модулем і трекером, а моніторинг ресурсів робиться через lightweight API (наприклад ring-buffer лічильник кадрів, NPU utilization counters) замість psutil і nvidia-smi.

У підсумку embedded-версія поєднує апаратно-прискорену детекцію, real-time трекінг і надійну комунікацію з автопілотом по MAVLink, зберігаючи всі наші бізнес-логіки по фільтрації об’єкта та розрахунку команд yaw у реальному часі.

3.3 Проектування тестового сценарію

Тестовий сценарій експеременту буде склададись із наступних етапів які описують суть та резултиаи кожного:

– відкриття відеопотоку – перевірити, що система може відкрити файл або камеру та зчитати перший кадр, результат: вивід діагностики (-- width, height, fps, cap.isopened()=true);

– детекція об’єктів – на кожному кадрі запуск yolo з заданими порогами, результат: повернення списку боксів [x1,y1,x2,y2,confidence,клас];

– підготовка для трекінгу – перетворити детекції в формат deep sort ([x,y,w,h], confidence, cls) або передати пустий список, результат: коректний список входів для трекера;

– безперервне оновлення трекера – виклик tracker.update_tracks() щокадру, результат: список підтверджених треків із постійними track_id навіть під час коротких втрат детекцій (до max_age кадрів);

– fsm-логіка idle/tracking – перевірити, що система переходить у режим трекінгу при появі першої детекції, а при втраті активного треку повертається в idle, результат: коректний стан машини (state_idle або state_tracking);

– генерація px4-команд – обчислити yaw_error і yaw_command на основі положення треку відносно центру, результат: у консолі друкуються stub-повідомлення у форматі mavlink-команд (command_long_encode…);

– візуалізація результату – накладати на відеокадр зелені бокси детекції, сині (та сірі) бокси треків та стрілку/yaw-підпис, результат: opencv-вікно з усіма накладками.

Моніторинг ресурсів (опціонально) – прогін через resource_monitor.py, результат: після завершення роботи вивід сумарного CPU-часу, пікової RAM і GPU-споживання

3.4 Розроблені алгоритми

3.4.1 Алгоритм пайплайну детекції

У нашій системі відеоаналітики обробка кадрів відбувається в єдиному циклі, де спочатку за допомогою OpenCV (cv2.VideoCapture) отримується зображення, а потім клас YoloDetector викликає метод detect(frame), який під патчем torch.load завантажує модель YOLO (наприклад, yolov8n.pt), виконує перед-обробку кадру, прогоняє його через мережу та повертає масив сирих детекцій формату [x1,y1,x2,y2,confidence,cls]. Після цього із отриманих детекцій відбираються ті, що перевищують поріг довіри та належать цікавому класу (наприклад, cls==0 для людини), і перетворюються в набір вхідних даних для трекера: список кортежів ([x, y, w, h], confidence, cls).

Цей список передається в метод update_tracks(ds_inputs, frame=frame) інстансу класу DeepSortTracker, який ініціалізовується один раз на початку з налаштованими параметрами (max_iou_distance, max_age, n_init, max_cosine_distance, nms_max_overlap, half) і застосовує Калманів фільтр та зовнішні ознаки для асоціації детекцій з існуючими траєкторіями.

Після виклику update_tracks ми отримуємо список об’єктів Track із унікальним track_id, координатами боксу (to_ltrb()), а також властивістю is_confirmed(), яка дозволяє відкинути непідтверджені траєкторії. Логіка FSM

перемикає стан між STATE_IDLE (очікування першої детекції) та STATE_TRACKING (тримання активного треку): при появі першої довірчої детекції створюється новий трек, а якщо протягом max_age кадрів немає жодної асоційованої детекції, трек вважається втраченим і система повертається в режим очікування. Нарешті, для візуалізації OpenCV малює зелені бокси детекцій, сині (або сірі) бокси треків та стрілку з cv2.arrowedLine, а також генерує yaw-команди через функцію-заглушку send_px4_command(), де на основі зсуву центру боксу відносно центру кадру та горизонтального поля огляду (hfov) обчислюється помилка та пропорційна команда повороту. На рисунку 3.3 наведено алгоритм пайплайну детекції та трекінгу.

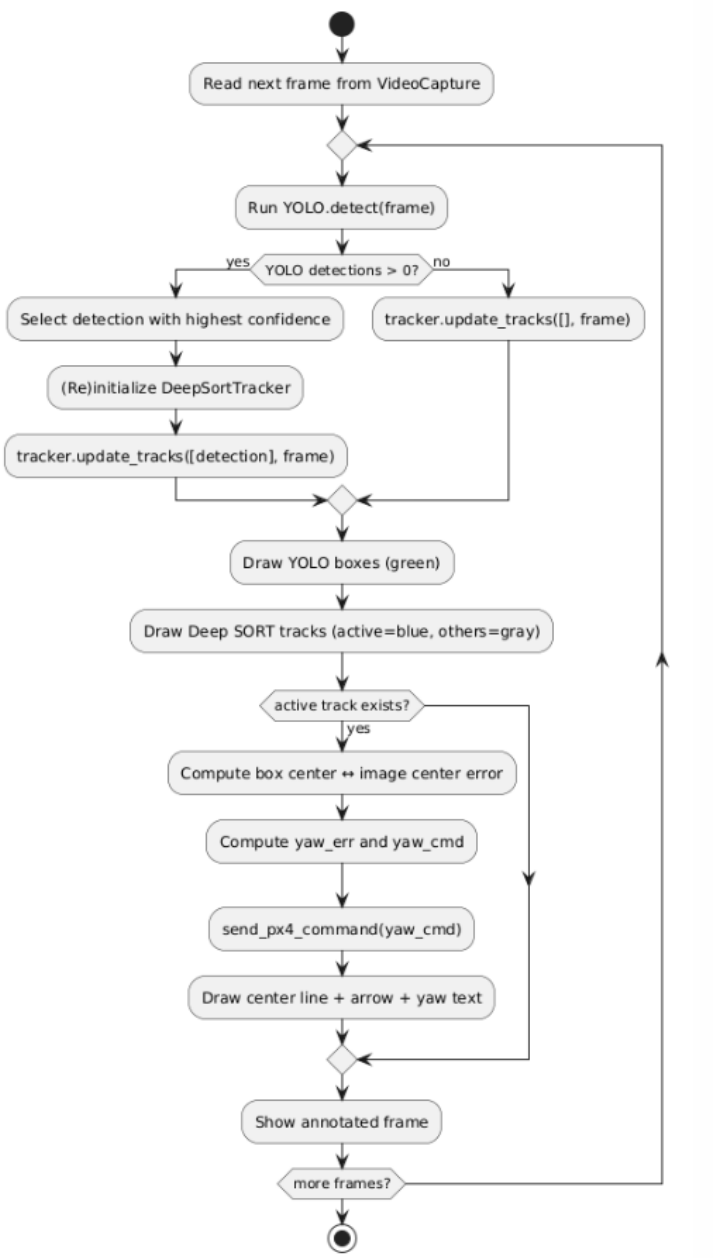

Рисунок 3.3 – Алгоритм пайплайну детекції та трекінгу(рисунок виконано самостійно)

На кожному кадрі відео ми запускаємо детекцію YOLO і переводимо результат у трекінг через Deep SORT всередині одного циклу «зчитування – обробка–відображення»: спочатку читаємо кадр, передаємо його в YoloDetector.detect(), і якщо YOLO повертає бодай один бокс із довірою ≥ порогу, ми кидаємо цей бокс (або всі бокси) у DeepSortTracker.update_tracks() – одразу видаляючи попередній активний трек і створюючи новий на основі найвпевненішої детекції; далі, у кожному наступному кадрі, поки YOLO не виявить новий об’єкт,

ми продовжуємо викликати update_tracks([], frame) із порожнім списком вхідних детекцій, дозволяючи Kalman-фільтру Deep SORT прогнозувати позицію поточного треку; щойно YOLO знову знаходить об’єкт, ми знову руйнуємо старий трек і переініціалізуємо трекер за координатами свіжої детекції – так цикл «детекція, ініціалізація треку, передбачення руху без нових детекцій, перезавантаження треку при новій детекції» повторюється для забезпечення безперервного та адаптивного трекінгу.

3.4.2 Алгоритм донаведення дрону

Алгоритм керування дрона на основі відеоаналітики складається з кількох послідовних етапів і працює лише тоді, коли YOLO-детектор повертає хоч одну детекцію.

При надходженні кадру з новою детекцією ми беремо координати центру боксу об’єкта та обчислюємо горизонтальну відстань від цього центру до центру зображення. Далі, аналізуючи різницю поточної помилки центрування з помилкою попереднього кадру (Δdx), оцінюємо динаміку руху об’єкта: якщо об’єкт рухається швидко й помилка зростає, ми підсилюємо команду повороту (збільшуємо yaw_rate), якщо рух повільний чи об’єкт майже стоїть – зменшуємо yaw_rate пропорційно. Формула керуючого сигналу виглядає як P-регулятор із врахуванням Δdx:

```
yaw_err = dx_px / frame_width * HFOV
delta_err = yaw_err − prev_yaw_err
yaw_cmd = kp * yaw_err + kd * delta_err
```

де HFOV – горизонтальне поле зору камери, kp і kd – налаштовані коефіцієнти, prev_yaw_err – попередня помилка.

Якщо детекція на кадрі відсутня, але в попередньому кадрі об’єкт лежав ближче до краю зображення (наприклад, |dx_px| > 0.4·frame_width), вважаємо, що він вийшов із поля зору, і віддаємо команду «зависнути та обертатися на місці на

360°». Якщо ж об’єкт зник, перебуваючи близько до центру (|dx_px| ≤ 0.2·frame_width), віддаємо команду «зависнути на місці» без обертання. У момент, коли YOLO знову знайде об’єкт, алгоритм повертається до стандартної обробки кадру з детекцією, обчислення помилки та видачі yaw-команди. Код наведено нижче:

```
    def compute_yaw_correction(x1: float, y1: float, x2: float, y2: float,
                               frame_width: int, hfov: float, kp: float) -
> tuple[float, float]:
        box_cx = (x1 + x2) / 2.0
        img_cx = frame_width / 2.0
        dx = box_cx - img_cx
        yaw_error = (dx / frame_width) * hfov
        yaw_rate = kp * yaw_error
        return yaw_error, yaw_rate
    def generate_px4_commands(yaw_rate: float, target_speed: float = 5.0)
-> tuple[str, str]:
        cmd_yaw = (
            "MAV_CMD_CONDITION_YAW \
    "
            f"param1=0 param2={yaw_rate:+.2f} param3=1 param4=1 param5=0
param6=0 param7=0"
        )
        cmd_speed = (
            "MAV_CMD_DO_CHANGE_SPEED \
    "
            f"param1=1   param2={target_speed:.2f}   param3=-1   param4=0
param5=0 param6=0 param7=0"
        )
        print(f"[PX4_CMD] {cmd_yaw}")
        print(f"[PX4_CMD] {cmd_speed}")
        return cmd_yaw, cmd_speed
    if __name__ == "__main__":
        import argparse

        parser = argparse.ArgumentParser(
            description='Генерація PX4-команд на основі детекційної рамки'
        )
        parser.add_argument('x1', type=float, help='ліва координата bbox
(пікселі)')
        parser.add_argument('y1', type=float, help='верхня координата bbox
(пікселі)')
        parser.add_argument('x2', type=float, help='права координата bbox
(пікселі)')
        parser.add_argument('y2', type=float, help='нижня координата bbox
(пікселі)')
        parser.add_argument('--width', type=int, default=640,
                            help='ширина кадру (пікселі)')
        parser.add_argument('--hfov', type=float, default=90.0,
                            help='горизонтальне поле зору (градуси)')
        parser.add_argument('--kp', type=float, default=0.5,
                            help='P-коефіцієнт для yaw')
        parser.add_argument('--speed', type=float, default=5.0,
```

```
                    help='цільова лінійна швидкість (м/с)')

args = parser.parse_args()
yaw_err, yaw_rate = compute_yaw_correction(
    args.x1, args.y1, args.x2, args.y2,
    args.width, args.hfov, args.kp
)
print(f"YawError = {yaw_err:+.2f}°")
```

generate_px4_commands(yaw_rate, args.speed)

Такий підхід дозволяє поєднати точність детекції з плавністю й стійкістю рухів дрона, уникнути надто різких маневрів при повільному переміщенні об’єкта та реагувати на повну втрату його в полі зору.

4 ПРОВЕДЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ ЕКСПЕРЕМЕНТУ

4.1 Проведення експеременту

Спочатку було обрано та завантажено відеоджерела для тестування (записи з камери дрону), після чого ,екі протестовано скрипт, який для кожного відео здійснює кадр за кадром детекцію, працює паралельно із Deep SORT-трекер та одночасно фіксує час інференсу, кадрову статистику (загальну кількість кадрів і кадрів із детекцією) і апаратне навантаження (CPU, GPU, пам’ять),також зібрано ключові метрики продуктивності (FPS, середній час детекції, відсоток кадрів із виявленими об’єктами, тривалість трекінгу при втраті об’єкта тощо), зібрані дані було оброблено та візуалізовано в графіках і таблицях, після чого проведено порівняльний аналіз результатів для тестової системи та

4.1.1 Вхідні дані

Загалом було протестовано декілька відеофайлів, специфіка вибору лягла на відео специфічні для майбутніх задач, а саме - відео поганої якості, періодична втрата відеопотоку, бегато перешкод, відео на якому присутні люди в русі зазвичай, відео ішли у якості 480р. але зі скейлінгом на розміри самого екрану, зазвичай Full HD.

Для тестування було обрано відео з лінії зіткнення, на якому відсутні жорстокі кадри, показано слідкування за людиною яка переміщується у складних для детекції умовах, поміж дерев та ракурс задіяний при зйомці спочатку знаходиться збоку а потім слідкує майже чітко за обєктом, це все дозволило відносно усередковано надати умови реальної роботи. Також було протестовано відео з розвідувального дрону де більше для наочності продемонстрований трекінг на більш прийнятній якості але також зі специфікою - на великій висоті та при снігопаді. Короткі кадри наведено на рисунку 4.1.

Рисунок 4.1 – Кадри з тестових відео(рисунок виконано самостійно)

Як ми описали раніше – надано 2 перспективи для трекінгу реальних умов та для відладки та налаштувань поведінки системи на більш чіткому відео.

Нижче розглянемо налаштування для алгоритмів детекціх та трекінгу, код наведено нижче відповідно:

```
      class YoloDetector:
          def   __init__(self,   model_path='yolov8n.pt',   conf_thres=0.25,
iou_thres=0.45, device=None):
              self.device = device or ('cuda' if torch.cuda.is_available()
else 'cpu')
              self.model = YOLO(model_path)
              self.model.to(self.device)
              self.model.fuse()
              self.conf_thres = conf_thres
              self.iou_thres  = iou_thres
```

Для відео з низькою якістю зображення ми свідомо обрали досить «м’які» налаштування YOLO,DeepSORT:

Нижчий поріг довіри (--conf=0.2). Через розмитість, шум і низький контраст на кадрах моделі зазвичай присвоюють об’єктам менші значення confidence. Щоб не пропускати справжні об’єкти (не підвищувати помилково кількість «промахів»), ми знижуємо поріг до 0.2. Це дає змогу підхоплювати слабкі сигнали і потім коригувати їх за допомогою трекера, навіть якщо в окремому кадрі детекція невпевнена.

Помірний поріг NMS (--iou=0.45). Коли кадри «шумні», правильні бокси іноді зміщуються чи дублюються. Ми не хочемо агресивно скидати дуже пересічні бокси (інакше можуть гаситися легкі або частково перекриті об’єкти), але й не допустити велике загущення накладених прямокутників. Значення IoU=0.45 забезпечує компроміс: воно дозволяє трохи більше перекриття перед застосуванням NMS, зберігаючи при цьому адекватну чіткість однієї детекції на об’єкт.

У результаті такого поєднання ми підвищуємо чутливість системи (щоб вловити якомога більше потенційних об’єктів в умовах поганого відео), але при цьому залишаємо достатній рівень селекції, щоб не «заповнити» кадр зайвими накладками, — і далі дозволяємо DeepSORT відновлювати та згладжувати ці дуже неоднорідні детекції в стійкі траєкторії:

```
class DeepSortTracker:
    def __init__(
        self,
        max_iou_distance: float = 0.7,     # зменшили, щоб більше прив’язувало по руху
        max_age: int = 30,    # тепер трек живе 30 кадрів без детекції
        n_init: int  = 1,     # підтверджуємо трек вже після 1 детекції
        max_cosine_distance: float = 0.5,      # трохи послабили appearance-поріг
        nms_max_overlap: float  = 1.0,   # майже без NMS
        half: bool  = True,  # FP16-режим для пришвидшення, якщо є GPU
        use_appearance: bool  = True  # якщо False — вмикаємо тільки IOU + Kalman
    ):
```

Більш детально розглянемо трекінг:

max_iou_distance = 0.7. Цей поріг відповідає за просторове зв’язування нових детекцій із існуючими треками на основі їхнього перетину (IoU). Ми знизили його з типових ~0.9 до 0.7, щоб сильніше орієнтуватися на узгоджений рух (якщо дві рамки суттєво не накладаються, ми їх уже не «прив’язуємо»), але при цьому не втрачати трек, якщо об’єкт трохи змістився;

– max_age = 30. Скільки кадрів трек вважається «живим» після останньої детекції. При значенні 30 трекер продовжить прогнозувати положення об’єкта до

півсекунди (при 60 FPS) без нових входів від YOLO, дозволяючи системі витримувати короткі провали детекції;

– n_init = 1. Мінімальна кількість підтверджених детекцій, необхідних для того, щоб трек вважався валідним і почав відображатися (та впливати на керування). Ми виставили 1, щоб моментально реагувати на першу детекцію, без додаткової затримки «розбігу»;

– max_cosine_distance = 0.5. Поріг схожості зовнішнього вигляду (appearance feature). Збільшивши його до 0.5 (з типового ~0.2–0.3), ми трохи послабили вимоги до візуального «збігу» між детекціями, віддаючи більше ваги руху й формі рамок, а не лише глибоким ознакам;

– nms_max_overlap = 1.0. Поріг NMS для попередньої обробки входів. Встановивши його в 1.0, ми фактично відключаємо NMS усередині трекера, щоб не скидати близькі детекції — вся селекція вже відбувається на боці YOLO за confidence та IoU;

– half = True. Дозволяє використовувати 16-бітні (FP16) обчислення для екстрактора appearance-фіч (якщо є GPU). Це прискорює побудову embedding-ів без помітної втрати якості;

– use_appearance = True. Включає комбіновану стратегію прив’язування: спершу IoU + Калман-фільтр, потім appearance-схожість. Якщо встановити False, трекер працюватиме лише за положенням (IOU + Калман), ігноруючи візуальні ознаки — це може бути корисно, коли зовнішній вигляд об’єкта змінюється або обчислювальні ресурси вичерпуються.

Завдяки такому налаштуванню ми отримуємо стійкий, «м’який» трекінг: трекер довго витримує провали в детекції, швидко реагує на перший вхід, коригує прив’язку за зовнішнім виглядом, але не глушить хороші рухові зв’язки.

4.2 Результати детекції та трекінгу

Для кращого розуміння візуальної наочності відпрацювання кожного елементу – наведемо опис елементів інтерфейсу що зображено на рисунку 4.2.

Рисунок 4.2 – Елементи інтерфейсу системи(рисунок виконано самостійно)

На нашому інтерфейсі кожен кадр містить кілька накладень: по-перше, зелений прямокутник – це рамка детекції YOLO, підписана значенням довіри (confidence) над лівим верхнім кутом; по-друге, синій прямокутник із написом «ID: N» – це активний трек DeepSORT, який асоціює об’єкт із його унікальним ідентифікатором (інші живі треки, якщо вони є, могли б малюватися сірим кольором); по-третє, тонка біла вертикальна лінія по центру кадру відображає ідеальну вісь курсової централізації; по-четверте, зелена стрілка від цієї осі до центру коробки треку показує напрямок і величину необхідного повороту дрона (yaw), а над стрілкою виводиться текст на кшталт “YawErr +21.6° Cmd +10.8°/s”,

який інформує про поточну курсову помилку й обчислену команду швидкості обертання.

Усі ці елементи разом дають операторові чітке уявлення про виявлення об’єкта, його стеження та необхідні керуючі корекції в реальному часі.

Далі послідовно наведено кадр з відпрацюванням і детекції (див.рис. 4.3), та кадр з утримання обєкуту в стані трекінгу – коли модель розпізнавання втратила або не може розпізнати об'єкт (див.рис. 4.4).

Рисунок 4.3 – Відпрацювання детекції + трекінг(рисунок виконано самостійно)

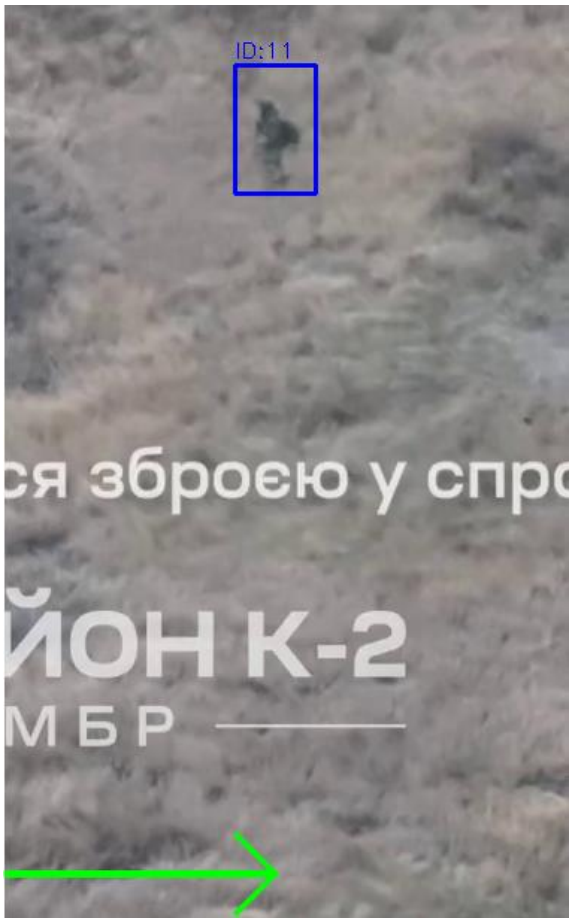

Рисунок 4.4 – Утримання трекінгом(рисунок виконано самостійно)

Як видно – навіть коли детекція втрачається – трекер підхоплює об'єкт для того щоб не втратити його з детекції та продовжувати надсилати команди на модуль дрону.

Також для наочності було написано модуль який збирає статистику по відео-потоку та результатам відпрацювання детекції, результат наведено на рисунку 4.5.

```
=== Підсумковий звіт (SIM) ===
Кількість кадрів              : 638
FPS (від CAP_PROP_FPS)         : 30.00
Тривалість відео (секунд)     : 22.20
Загальний час детекції (сек)  : 6.380
Середній час/кадр детекції    : 0.01000 сек
Кадрів із виявленими об'єктами: 88 (13.8% від усіх кадрів)
```

Рисунок 4.5 – Метричні показники YOLO11 (рисунок виконано самостійно)

Оброблено 638 кадрів за загалом 6.38 с чистого часу детекції – це в середньому по 0.010 с (10 мс) на кадр.

При такій швидкості детекція виконується приблизно на 100 кадрів/сек (1 000 мс / 10 мс), що значно перевищує заявлені 30 FPS відео. Це означає, що детектор легко працює в реальному часі із запасом по потужності.

Також видно що об’єкт було знайдено лише в 88 кадрах з 638, тобто в 13.8 % випадків. Враховуючи якість відео та не знаходження обєкуту в полі зору камери певний час – це середній показник.

4.3 Результат донаведення та моніторингу навантаження

Нижче наведено приклад відпрацювання модулю донаведення, від відпрацьовував тільки тоді – коли обєкт знаходився під детекцією або трекінгом. Вивід наведено на рисунку 4.6.

```
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.84 param3=1 param4=1 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.83 param3=1 param4=1 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.85 param3=1 param4=1 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.84 param3=1 param4=1 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.84 param3=1 param4=1 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.83 param3=1 param4=1 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.83 param3=1 param4=1 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.82 param3=1 param4=1 param5=0 paramб=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 paramб=0 param7=0
```

Рисунок 4.6 – Команди корегування донаведення (рисунок виконано самостійно)

Розглнутий вивід команд – це представлення MAVLink-повідомлення COMMAND_LONG з командою MAV_CMD_DO_CHANGE_SPEED, яке PX4 інтерпретує як керувати FM модулем дрону. Елементи означають наступне:

– MAV_CMD_DO_CHANGE_SPEED — ідентифікатор команди зміни швидкості;

– param1=1 — тип швидкості: 0 = airspeed, 1 = ground speed, 2 = climb speed, 3 = descent speed;

– param2=5.00 — значення швидкості в м/с (тут 5 м/с);

– param3=-1 — throttle (процент подачі палива): –1 залишає поточне значення без зміни;

param4=0, param5=0, param6=0, param7=0 – зарезервовані місця для додаткових параметрів (не використовуються, тому 0).

Аналізуючи логіку руху дрона, порівняно з відео – стає очевидним коректність поведінки дрону адже дрон рухався з постійною лінійною швидкістю вперед, здійснюючи плавні коливальні зміни кута повороту курсу: спочатку стабільно до права, потім корекція й переведення вліво, а потім знову легке повернення вправо.

Далі розглянемо результат моніторингу навантаження(див.рис. 4.7).

```
=== Resource Usage Summary ===
CPU time  – user: 116.52s, system: 24.83s, total: 141.34s
RAM peak  – 749.2 MiB
```

Рисунок 4.7 – Результати виводу апаратного навантаження (рисунок виконано самостійно)

З разультату видно що «user» і «system» CPU-тайми беруться з psutil.Process.cpu_times(): user-time показує, скільки секунд процес провів у користувацькому коді, system-time – в ядрі ОС; їхня сума дає «total» CPU-секунди, витрачені програмою. «RAM peak» (RSS) – максимальний обсяг оперативної пам’яті, зайнятий процесом, зчитується через psutil.Process.memory_info().rss. GPU-метрика (не показана тут) вимірюється викликом nvidia-smi та парсингом рядків «pid, used_memory».

Протягом 20 с обробки відео було витрачено 141.34 с чистого CPU-часу, отже в середньому програма використовувала 141.34 / 20 ≃ 7.07 CPU-ядер. На Ryzen 5 7600X (6 фізичних ядер, 12 потоків) це означає завантаження близько 118 % від фізичної ресурсної бази (або ∼59 % від загального числа потоків), тобто навіть із SMT лінійка охоплена більш ніж наполовину. На типовому Orange Pi 5 із 4-ма ядрами Cortex-A55 за такого навантаження потрібно було б ≃7 ядер, тобто близько

175 % від його можливостей – отже він не здатний обробити відео в реальному часі з еквівалентною швидкістю.

На Orange Pi 5 є два класи ядер: чотири ARM Cortex-A76@2.4 GHz і чотири ARM Cortex-A55@1.8 GHz. Щоб порівняти їх із ядрами Zen 4, використовуємо орієнтовні пікові DMIPS-навантаження:

Cortex-A76 ≈ 3 DMIPS/MHz → 2 400 MHz × 3 ≈ 7 200 DMIPS на ядро

Cortex-A55 ≈ 2 DMIPS/MHz → 1 800 MHz × 2 ≈ 3 600 DMIPS на ядро

Zen 4 на 4.7 GHz ≈ 3.11 DMIPS/MHz → 4 700 MHz × 3.11 ≈ 14 617 DMIPS на ядро. Тоді відносна одноядерна потужність A76 ≈ 7 200/14 617 ≈ 0.49, A55 ≈ 3 600/14 617 ≈ 0.25, у той час як ядро Ryzen = 1.00.

Отож для нашого алгоритму, який на Ryzen виконувався за 20 с, Orange Pi 5 із цією архітектурою потребуватиме близько 45 с.

ВИСНОВКИ

У ході дослідження було розроблено та протестовано систему автономного управління дроном, яка поєднує методи розпізнавання об’єктів та алгоритми трекінгу для автоматичної корекції польоту. Основна мета полягала в інтеграції YOLO11 для виявлення людей та використання трекінгових алгоритмів для стабільного відстеження цілі в русі. Отримані координати об’єкта використовуються для формування керуючих команд, що надсилаються на польотний контролер PX4 через MAVLink.

Дослідження включало порівняння різних алгоритмів трекінгу, таких як KCF, SORT та DeepSORT, що дозволило визначити їх ефективність у контексті реального часу. Виявлено, що SORT забезпечує хорошу продуктивність за низьких обчислювальних витрат, у той час як DeepSORT покращує точність, але потребує більше ресурсів. Оптимальним варіантом для експериментів став баланс між швидкістю та точністю, залежно від апаратної платформи.

Тестування проводилося на різних обчислювальних платформах (Orange Pi 5+, Raspberry Pi 4, RockPro), що дозволило оцінити їх продуктивність у режимі реального часу. Використання NPU на Orange Pi 5+ значно знизило навантаження на центральний процесор, забезпечуючи стабільну обробку відеопотоку. Аналіз продуктивності показав, що Raspberry Pi 4 має суттєві обмеження у швидкості обробки кадрів, що може призводити до затримок у реакції дрона.

Результати експериментів продемонстрували, що запропонована система дозволяє виконувати автоматичне донаведення дрона на ціль без втручання оператора. Це критично важливо для військових FPV-дронів, оскільки зменшує когнітивне навантаження на оператора та підвищує ефективність місій. Однак подальші дослідження можуть бути спрямовані на покращення стабільності трекінгу при змінних умовах освітлення, а також на інтеграцію додаткових сенсорів для підвищення точності розпізнавання. Загалом розроблена система підтвердила свою ефективність, а отримані результати можуть бути використані для подальшого вдосконалення автономних безпілотних платформ.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Elijah, T., Jamisola, R.S., Tjiparuro, Z. et al. A review on control and maneuvering of cooperative fixed-wing drones. Int. J. Dynam. Control 9, 1332–1349 (2021). https://doi.org/10.1007/s40435-020-00710-2 (дата звернення: 04.02.2025).

2. Rakova, A. O., & Bilous, N. V. (2020). Reference points method for human head movements tracking. Radio Electronics, Computer Science, Control, (3), 121–128. https://doi.org/10.15588/1607-3274-2020-3-11 (дата звернення: 30.02.2025).

3. Bilous, N., Malko, V., Frohme, M., & Nechiporenko, A. (2024) Comparison of CNN-based architectures for object detection to define what class of object is, pp1-15, AI (in printed). (дата звернення: 10.02.2025).

4. Bilous, N., Malko, V., Moshenskyi, N. (2024). Search and Detection of People in the Water Using YOLO Architectures: A Comparative Analysis from YOLOv3 to YOLOv8. In: Szewczyk, R., Zieliński, C., Kaliczyńska, M., Bučinskas, V. (eds) Automation 2024: Advances in Automation, Robotics and Measurement Techniques. AUTOMATION 2024. Lecture Notes in Networks and Systems, vol 1219. Springer, Cham. https://doi.org/10.1007/978-3-031-78266-4_21 (дата звернення: 05.02.2025).

5. Gao, P., Li, Z. (2025). YOLO-S3DT: A Small Target Detection Model for UAV Images Based on YOLOv8. *Computers, Materials & Continua*, 82(3), 4555. ISSN 1546-2218. https://doi.org/10.32604/cmc.2025.060873. (дата звернення: 30.01.2025).

6. Bilous, N., Ivanichev V, (2024). Human Recognition in Streaming Video Using YOLO-9 2024 IEEE International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T), 15–17 October, Kharkiv, Ukraine (in printed). (дата звернення: 30.01.2025).

7. N.V. Bilous, O.V. Rassokha, I.A. Ahekian, O.V. Hramm (2020). Research on methods for development of software system for emotions recognition and state of human health determination. Bionics of Intelligence, 1(94), 65–70. (In Ukrainian) doi: 10.30837/bi.2020.1(94).10. (дата звернення: 03.03.2025).

8. Bilous, N. V., Ahekian, I. A., & Kaluhin, V. V. (2023). Determination and comparison methods of body positions on stream video. Radio Electronics, Computer Science, Control, (2), 52. https://doi.org/10.15588/1607-3274-2023-2-6 (дата звернення: 30.01.2025).

9. Bilous, N., Svidin, O., Ahekian, I., & Malko, V. (2024). A Skeleton-Based Method for Exercise Recognition Based On 3D Coordinates of Human Joints. IAES

International Journal of Artificial Intelligence (IJ-AI), ISSN/e-ISSN 2089-4872/2252-8938 DOI: 10.11591/ijai.v13.i2.pp1805-1816. (дата звернення: 29.02.2025).

10. Wang Z., Dong X.-M., Xu Y. SCASNet: Spatial Context-Aware Selection Network for Small Object Detection in Aerial Imagery // IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing. – 2025. – P. 1–17. – DOI: 10.1109/JSTARS.2025.3555627. (дата звернення: 19.03.2025).

11. Daeil Hyun, Jaeyoung Han, Seokmoo Hong. Development of hybrid-powered, sustainable multi-purpose drone system: An analysis model. International Journal of Hydrogen Energy, Volume 61, 2024, Pages 762-773, ISSN 0360-3199. https://doi.org/10.1016/j.ijhydene.2024.02.251 (дата звернення: 15.02.2025).

12. C. Y. Yong, R. Sudirman and K. M. Chew, "Motion Detection and Analysis with Four Different Detectors," 2011 Third International Conference on Computational Intelligence, Modelling & Simulation, Langkawi, Malaysia, 2011, pp. 46-50. http://dx.doi.org/10.1109/CIMSim.2011.18 (дата звернення: 01.03.2025).

13. Adhikary, S., Mondal, T., Kumari, P., & Laskar, M. H. (2024). Unmanned aerial vehicles for human detection and recognition using neural-network model. Frontiers in Neurorobotics, 18, Article 1443678. https://doi.org/10.3389/fnbot.2024.1443678 (дата звернення: 16.05.2025).

14. Pierre Boussel. The Golden Age of Drones: Military UAV Strategic Issues and Tactical Developments. Michael Johnson. Ресурс: Trends Research. URL: https://trendsresearch.org/insight/the-golden-age-of-drones-military-uav-strategic-issues-and-tactical-developments/ (дата звернення: 18.03.2025).

15. Ferreira D, Basiri M. Dynamic Target Tracking and Following with UAVs Using Multi-Target Information: Leveraging YOLOv8 and MOT Algorithms. Drones. 2024; 8(9):488. https://doi.org/10.3390/drones8090488 (дата звернення: 29.02.2025).

16. Jeff Geerling. New 2GB Pi 5 Has 33% Smaller Die, 30% Idle Power Savings. Ресурс: Jeff Geerling Blog. URL: https://www.jeffgeerling.com/blog/2024/new-2gb-pi-5-has-33-smaller-die-30-idle-power-savings (дата звернення: 10.11.2025).

17. Orange Pi 5 Plus (4GB/8GB/16GB). Ресурс: Orange Pi Official. URL: https://web.archive.org/web/20240227101906/https://www.orangepi.org/html/hardWare/computerAndMicrocontrollers/details/Orange-Pi-5-plus.html (дата звернення: 12.02.2025).

18. David Kirichenko. The Rush for AI-Enabled Drones on Ukrainian Battlefields. Jane Smith. Ресурс: Lawfare Media. URL: https://www.lawfaremedia.org/article/the-rush-for-ai-enabled-drones-on-ukrainian-battlefields (дата звернення: 10.03.2025).

19. Система боротьби з дронами: для класифікації об’єктів. Гаханова І. О. Ресурс: Харківський національний університет радіоелектроніки. URL: https://nure.ua/wp-content/uploads/2024/disertacija_hahanova.pdf (дата звернення: 10.03.2025).

20. Огляд алгоритмів комп'ютерного зору для виявлення небезпечних об'єктів дронами / О.І. Лактіонов, Б.Р. Боряк, Н.М. Педченко [та ін.] // Системи управління, навігації та зв'язку. – 2023. – № 3 (73). – С. 120–122. – Doi: 10.26906/SUNZ.2022.3.120 https://reposit.nupp.edu.ua/handle/PoltNTU/13468 (дата звернення: 10.03.2025).

21. Raspberry Pi 5 8GB Vs Orange Pi 5 Plus 16GB.: YouTube. URL: https://www.youtube.com/watch?v=oWkgjbJYBNQ (дата звернення: 12.02.2025).

22. Особливості алгоритму YOLO V.7. Невідомий. Ресурс: Національний технічний університет "Харківський політехнічний інститут". URL: https://repository.kpi.kharkov.ua/server/api/core/bitstreams/adea16b3-5b22-4c33-9672-cdd6a6544d1a/content (дата звернення: 10.02.2025).

23. Ultralytics YOLO11. Ресурс: Офіційна документація Ultralytics. URL: https://docs.ultralytics.com/ru/models/yolo11/ (дата звернення: 10.03.2025).

24. Dynamic Time Warping. Mark Stent. Ресурс: Medium. URL: https://medium.com/@markstent/dynamic-time-warping-a8c5027defb6 (дата звернення: 10.03.2025).

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ ЗА НАУКОВИМИ НАПРЯМАМИ КЕРІВНИКА ТА НАУКОВЦІВ КАФЕДРИ ПРОГРАМНОЇ ІНЖЕНЕРІЇ

2. Rakova, A. O., & Bilous, N. V. (2020). Reference points method for human head movements tracking. Radio Electronics, Computer Science, Control, (3), 121–128. https://doi.org/10.15588/1607-3274-2020-3-11 (дата звернення: 30.02.2025).

6. Bilous, N., Ivanichev V, (2024). Human Recognition in Streaming Video Using YOLO-9 2024 IEEE International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T), 15–17 October, Kharkiv, Ukraine (in printed). (дата звернення: 30.01.2025).

7. N.V. Bilous, O.V. Rassokha, I.A. Ahekian, O.V. Hramm (2020). Research on methods for development of software system for emotions recognition and state of human health determination. Bionics of Intelligence, 1(94), 65–70. (In Ukrainian) doi: 10.30837/bi.2020.1(94).10. (дата звернення: 03.03.2025).

8. Bilous, N. V., Ahekian, I. A., & Kaluhin, V. V. (2023). Determination and comparison methods of body positions on stream video. Radio Electronics, Computer Science, Control, (2), 52. https://doi.org/10.15588/1607-3274-2023-2-6 (дата звернення: 30.01.2025).

9. Bilous, N., Svidin, O., Ahekian, I., & Malko, V. (2024). A Skeleton-Based Method for Exercise Recognition Based On 3D Coordinates of Human Joints. IAES International Journal of Artificial Intelligence (IJ-AI), ISSN/e-ISSN 2089-4872/2252-8938 DOI: 10.11591/ijai.v13.i2.pp1805-1816. (дата звернення: 29.02.2025).