Міністерство освіти і науки України
Харківський національний університет радіоелектроніки
Факультет __ інформаційних радіотехнологій та технічного захисту інформації
(повна назва)
Кафедра _ мікропроцесорних технологій і систем _
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____другий (магістерський)_____
Роробка системи автоматичного огинання перешкод дронів
(тема)

Виконав:
здобувач ___2____ року навчання,
групи ІМСм-23-1_____
___ Святослав СТАРОКОЖЕВ ___
(власне ім'я, прізвище)

Спеціальність _171 Електроніка_____
(код і повна назва спеціальності)
Тип програми _освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Інженерія мікропроцесорних систем_____
(повна назва освітньої програми)

Керівник _проф. Олег ЗУБКОВ_____
(посада, власне ім'я, прізвище)

Допускається до захисту

В.о. завідувача кафедри МТС _____ Олег ЗУБКОВ_____
(підпис) (власне ім'я, прізвище)
2025 р.

Харківський національний університет радіоелектроніки

Факультет__ інформаційних радіотехнологій та технічного захисту інформації

Кафедра__ мікропроцесорних технологій і систем _

Рівень вищої освіти_____другий (магістерський)_____

Спеціальність_____171 Електроніка _____

(код і повна назва)

Тип програми _____освітньо-наукова _____

(освітньо-професійна або освітньо-наукова)

Освітня програма Інженерія мікропроцесорних систем _____

(повна назва)

ЗАТВЕРДЖУЮ:

В.о. зав. кафедри МТС

___ __Олег ЗУБКОВ

(підпис)

«____»_____ 2025 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві__Старокожеву Святославу Валерійовичу

(прізвище, ім’я, по батькові)

1. Тема роботи ____Розробка системи автоматичного огинання перешкод дронів

затверджена наказом університету від _17_ ____04____2025 р. № 292Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 23 _06__2025 р.

3. Вихідні дані до роботи Типи перешкод: дерева, будинки, стовби. Використання нейронних мереж для розпізнавання перешкод та їх огинання, створення дат сету та навчання нейронної мережі, симуляція роботи алгоритму аогинання_____

_____

4. Перелік питань, що потрібно опрацювати в роботі _____

1) Провести огляд та аналіз методів, алгоритмів і технічних засобів з виявлення перешкод та їх огинання

2) Провести аналіз симуляційних пакетів з симуляції польотів дронів та огинання перешкод дронами

3) Обрати нейронну мережу для розпізнавання перешкод, підготувати датасет для її навчання, навичти нейронну мережу

4) Розробити алгоритм огинання перешкод з використанням навченої нейронної мережі

5) Виконати симуляцію роботи розробленого алгоритму та сформувати висновки до результатів

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) _____

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

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

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / термін виконання етапів роботи	Примітка
1	Огляд та аналіз методів, алгоритмів і технічних	21.04.2025-26.04.2025	виконано
	засобів з виявлення перешкод та їх огинання		
2	Аналіз симуляційних пакетів з симуляції	27.04.2025-6.05.2025	виконано
	польотів дронів та огинання перешкод дронами		
3	Обгрунтування вибору нейронної мережі для	7.05.2025-14.05.2025	виконано
	розпізнавання перешкод		
4	Підготовка датасету для навчання нейронної	15.05.2025-21.05.2025	виконано
	мережі		
5	Розробити алгоритм огинання перешкод з	22.05.2025-3.06.2025	виконано
	використанням навченої нейронної мережі		
6	Виконання симуляції роботи розробленого	4.06.2025-7.06.2025	виконано
	алгоритму		
7	Написання пояснювальної записки	8.06.2025-15.06.2025	виконано

Дата видачі завдання 21 04_____2025р.

Здобувач _____ _____
(підпис)

Керівник роботи _____ _____ проф. Олег ЗУБКОВ
(підпис) (посада, власне ім'я, прізвище)

ABSTRSACT

Qualification Thesis: 109 pages, 10 tables, 69 references.

QUADCOPTER, NAVIGATION, RESOURCE CONSTRAINTS, AUTONOMY, PROCESSOR, OBSTACLE AVOIDANCE, DEEP LEARNING, ENERGY EFFICIENCY.

Object of research – hardware-software system for autonomous navigation of a quadcopter.

Goal of research – to analyze intelligent obstacle avoidance systems, and to evaluate the effectiveness of different algorithms and hardware platforms for deep learning. Develop a system that can avoid obstackel and count people in groups.

Research methods – comparative analysis, systematization of technical characteristics, statistical analysis of resource consumption, review of modern architectures.

The thesis analyzes current hardware and software solutions used in autonomous UAVs, particularly platforms such as Intel Movidius, NVIDIA Jetson, Qualcomm Flight, and Ambarella CV2. A comparative table is provided based on key parameters: power consumption, performance (TFLOPS), architecture, number of cores, and supported operating systems. The study identifies how energy constraints influence the feasibility of implementing deep learning and obstacle avoidance algorithms. The experience of companies such as DJI, Skydio, Parrot, and Autel Robotics is reviewed. The research findings are summarized in the form of tables, charts, and technical recommendations for selecting suitable platforms.

The results obtained can be applied in the design of energy-efficient UAV navigation systems, optimization of neural network models for resource-constrained platforms, and in startups developing compact intelligent drones.

РЕФЕРАТ

Кваліфікаційна робота**:** 109 с., 10 табл., 69 джерел.

КВАДРОКОПТЕР, НАВІГАЦІЯ, ОБМЕЖЕННЯ РЕСУРСІВ, АВТОНОМІЯ, ПРОЦЕСОР, ОБХІД ПЕРЕШКОД, ГЛИБИННЕ НАВЧАННЯ, ЕНЕРГОЕФЕКТИВНІСТЬ.

Об’єкт дослідження – програмно-апаратна система автономної навігації квадрокоптера.

Мета дослідження – аналіз інтелектуальних систем огинання перешкод, оцінка ефективності різних алгоритмів та апаратних платформ для глибокого навчання. Розробка системи огинання перешкод і підразування кількості людей в групах.

Методи дослідження – порівняльний аналіз, систематизація технічних характеристик, статистичний аналіз споживання ресурсів, огляд сучасних архітектур.

У роботі проаналізовано сучасні програмно-апаратні рішення, які використовуються в автономних БПЛА, зокрема платформи Intel Movidius, NVIDIA Jetson, Qualcomm Flight, Ambarella CV2. Наведено порівняльну таблицю за ключовими параметрами: енергоспоживання, продуктивність (TFLOPS), архітектура, кількість ядер, сумісні ОС. Визначено, як обмеження енергоресурсів впливають на можливості реалізації алгоритмів глибинного навчання та обходу перешкод. Розглянуто досвід таких компаній як DJI, Skydio, Parrot, Autel Robotics. Результати дослідження узагальнено у вигляді таблиць, графіків та технічних рекомендацій щодо вибору платформи.

Отримані результати можуть бути використані у проєктуванні енергоефективних систем навігації БПЛА, оптимізації нейромережевих моделей для обмежених платформ, а також у стартапах, що розробляють компактні інтелектуальні дрони.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БПЛА – Безпілотний літальний апарат

СЛАМ – одночасна локалізація та побудова карти (від англ. SLAM – Simultaneous Localization and Mapping)

AGX – англ. Advanced Graphics eXtension – розширення графічної продуктивності

AI – англ. Artificial Intelligence – штучний інтелект

API – англ. Application Programming Interface – інтерфейс прикладного програмування

ARM – англ. Advanced RISC Machine – архітектура процесора з низьким енергоспоживанням

CPU – англ. Central Processing Unit – центральний процесор

DJI – скорочення від Da-Jiang Innovations – виробник дронів

DL – англ. Deep Learning – глибинне навчання

FPGA – англ. Field Programmable Gate Array – програмована користувачем вентильна матриця

GPU – англ. Graphics Processing Unit – графічний процесор

IMU – англ. Inertial Measurement Unit – інерціальний вимірювальний блок

MAVLink – англ. Micro Air Vehicle Link – протокол обміну між дроном і станцією

OS – англ. Operating System – операційна система

PX4 – автопілот з відкритим кодом для БПЛА

ROS – англ. Robot Operating System – операційна система роботів

RTOS – англ. Real-Time Operating System – операційна система реального часу

SDK – англ. Software Development Kit – набір інструментів розробника

SITL – англ. Software-In-The-Loop – моделювання програмного забезпечення

SLAM – англ. Simultaneous Localization and Mapping – одночасна локалізація та побудова карти

TFLOPS – англ. Tera Floating Point Operations Per Second – терафлопс (трильйон операцій з плаваючою комою на секунду)

UAV – англ. Unmanned Aerial Vehicle – безпілотний літальний апарат

VLIW – англ. Very Long Instruction Word – архітектура з дуже довгими інструкціями

YOLO – англ. You Only Look Once – архітектура нейронної мережі для виявлення об'єктів

ВСТУП

Сучасні безпілотні літальні апарати (БПЛА) дедалі частіше застосовуються в різних сферах: від спостереження та картографії до військових і рятувальних операцій. З розвитком штучного інтелекту та комп’ютерного зору особливого значення набуває автономна навігація, що включає можливість ухилення від перешкод у динамічному середовищі. Однією з ключових проблем, що обмежує розвиток автономності БПЛА, є нестача обчислювальних ресурсів на борту.

Обмеження за масою, енергоспоживанням, розміром і тепловиділенням не дозволяють інтегрувати в дрони повноцінні високопродуктивні процесори.

Водночас ефективна реалізація алгоритмів глибинного навчання для виявлення перешкод та прокладання маршруту вимагає значних обчислювальних потужностей. Тому виникає потреба у використанні енергоефективних архітектур, спеціалізованих прискорювачів (наприклад, Intel Movidius Myriad X, NVIDIA Jetson AGX Xavier), а також у створенні оптимізованих моделей нейронних мереж (YOLOv4-tiny, MobileNet) [1].

У практиці відомі приклади застосування таких рішень. Компанія Skydio успішно впровадила Jetson-платформу для автономної навігації в складних середовищах, Parrot SA використовує Ambarella CVflow AI Accelerator для задач комп’ютерного зору, а Autel Robotics – Qualcomm Flight RB5, що поєднує енергоефективність та AI-можливості [2]. Також аналізуються підходи з розподіленням обчислень між дроном і хмарними/наземними системами.

Тож дослідження і моделювання інтелектуальних систем огинання перешкод, оцінка їх ефективності, різних алгоритмів та апаратних платформ для глибокого навчання є важливим в розробці і імплементації в реальні продукти та рішення.

1 АНАЛІЗ ТЕХНОЛОГІЙ ОГИНАННЯ ПЕРЕШКОД ДРОНАМИ

1.1. Загальний огляд систем ухилення від перешкод

Системи ухилення від перешкод є фундаментальними для забезпечення безпечної та автономної роботи дронів у складних і мінливих середовищах. Вони дозволяють безпілотним літальним апаратам (БПЛА) автоматично виявляти та уникати перешкоди, що є критично важливим для їхнього використання в військових, комерційних і наукових застосуваннях. Цей розділ надає детальний огляд систем ухилення від перешкод, включаючи їхні типи, основні принципи, технологічні складові, алгоритми, реальні застосування, виклики та майбутні тенденції, базуючись на останніх дослідженнях і практичних прикладах.

Системи ухилення від перешкод еволюціонували від простих ультразвукових датчиків у ранніх дронах до складних інтеграцій з LiDAR, камерами та AI. Спочатку вони були обмежені статичними перешкодами, але останнім часом, зокрема завдяки розробкам, наприклад як Casia від Iris Automation [3] зосереджуються на динамічних перешкодах, таких як інші літальні апарати чи птахи, що є важливим для операцій поза візуальним полем зору (BVLOS).

Системи ухилення від перешкод можна класифікувати за їхнім підходом:

1) Активні методи (використовують сенсори та алгоритми для реагування в реальному часі. Прикладами є реактивні методи, такі як Vector Field Histogram (VFH) і метод потенційних полів, які дозволяють дронам швидко маневрувати, уникаючи зіткнень);

2) Пасивні методи (Передбачають попереднє планування траєкторії на основі картографічних даних або моделей оточення, часто за допомогою Simultaneous Localization and Mapping (SLAM), що дозволяє дронам будувати карти в реальному часі, як зазначено в [4]);

3) Гібридні методи (поєднують активні та пасивні підходи для досягнення оптимальної ефективності, наприклад, планування маршруту з подальшим реактивним коригуванням).

Крім того, системи Detect-and-Avoid (DAA) є спеціалізованими для виявлення та уникнення динамічних перешкод, що є необхідним для інтеграції дронів у цивільний повітряний простір, як описано в [5].

Процес ухилення від перешкод складається з кількох етапів:

1) Детекція перешкод (використання сенсорів, таких як LiDAR, камери, радар або ультразвукові датчики, для ідентифікації об’єктів. Наприклад, LiDAR забезпечує точне 3D картографування, як зазначено в [6]) ;

2) Аналіз загрози (оцінка рівня небезпеки на основі відстані, швидкості та напрямку руху перешкоди);

3) Генерація безпечного маршруту (використання алгоритмів, таких як A*, Rapidly-exploring Random Tree (RRT) або D* Lite, для розрахунку оптимального шляху обходу перешкоди);

4) Маневрування (зміна траєкторії дрона для уникнення зіткнення, що може включати зміну висоти, швидкості або напрямку).

Сучасні системи ухилення від перешкод покладаються на різноманітні сенсори та їхню інтеграцію:

1) LiDAR: (створює точні 3D карти оточення, ідеальний для складних середовищ, як у системах Skydio, які використовують шість 4K камер для 360° покриття, як зазначено в [7]);

2) Стереокамери та RGB-D камери: (оцінюють глибину і розпізнають об’єкти, імітуючи бінокулярний зір);

3) Радар: (ефективний для виявлення динамічних перешкод на значних відстанях, особливо в поганих погодних умовах);

4) Ультразвукові датчики: (використовуються для коротких дистанцій, наприклад, для захисту знизу чи з боків, як у дослідженні з використанням ультразвукових сенсорів для квадрокоптерів, описаному в [7] **)**;

5) Інфрачервоні сенсори: (працюють у темряві, виявляючи об’єкти за тепловим випромінюванням);

6) Сенсорне злиття: (об’єднання даних з кількох сенсорів для створення комплексного уявлення про оточення, що покращує точність, як зазначено в [8]).

1.2 Алгоритми ухилення від перешкод

Планування шляхів є ключовим аспектом робототехніки, що дозволяє роботам пересуватися від початкової точки до цілі, уникаючи перешкод. Цей огляд охоплює чотири провідні алгоритми: A*, D*, Rapidly-exploring Random Tree (RRT) та Probabilistic Roadmap (PRM) та на основі нейромереж (CNN, RNN, Transformer)

1.2.1. Алгоритми пошуку шляхів (A*, RRT, D*, VFH*)

A* — це алгоритм пошуку з найкращим першим вибором, який використовує евристичну оцінку для ефективного знаходження найкоротшого шляху. Він широко застосовується в дискретних середовищах, таких як сітки, де карта представлена графом. Алгоритм підтримує пріоритетну чергу вузлів для відвідування, впорядковану за сумою витрат від початку до вузла та евристичної оцінки до цілі.

Часова складність A* у найгіршому випадку становить O(E + V log V), де V — кількість вершин, а E — кількість ребер, за умови використання бінарної купи. Просторова складність — O(V + E), що відображає необхідність зберігання відкритих і закритих множин.

D* — це розширення A* для динамічних середовищ, де карта може змінюватися з часом, наприклад, через появу нових перешкод. Він дозволяє інкрементально оновлювати шлях, що робить його ефективним для роботів, які працюють у нестабільних умовах. Алгоритм підтримує пріоритетну чергу, подібно до A*, але також враховує зміни витрат ребер або наявність перешкод.

Часова складність D* подібна до A* — O(E + V log V) у найгіршому випадку, але вона може бути вищою через необхідність обробки змін у середовищі. Просторова складність також становить O(V + E), враховуючи зберігання графу та черги.

RRT — це алгоритм на основі вибірки, розроблений для планування руху у високорозмірних безперервних просторах, таких як конфігураційні простори роботів. Він будує дерево, випадково вибираючи точки у просторі та намагаючись з'єднати їх із найближчим існуючим вузлом, перевіряючи на колізії. RRT особливо корисний для середовищ із складними перешкодами та диференціальними обмеженнями.

Часова складність RRT для побудови дерева становить O(n log n), де n — кількість зразків, через необхідність пошуку найближчого вузла, який зазвичай

реалізується за допомогою k-d дерев. Просторова складність — O(n), оскільки зберігається лише дерево.

PRM — це ще один алгоритм на основі вибірки, який попередньо обчислює граф колізійно-вільних шляхів між випадковими точками у конфігураційному просторі. На відміну від RRT, PRM створює карту, яку можна використовувати для багаторазових запитів, що робить його ефективним для статичних середовищ із численними запитами. Пошук шляху виконується за допомогою графових алгоритмів, таких як A*.

Часова складність для побудови карти PRM становить $O(n^2)$, оскільки перевіряються всі пари точок на колізії, хоча з оптимізаціями, такими як k-найближчі сусіди, вона може бути зменшена до O(n log n). Пошук шляху на готовій карті має складність O(m log m), де m — кількість ребер у графі. Просторова складність — O(n + m), враховуючи вузли та ребра карти.

VFH* — (Vector Field Histogram Star) — це гібридний алгоритм локального уникання перешкод із перевіркою напряму руху через A*. На відміну від VFH+ (1998), VFH* не просто вибирає найменш заблокований напрямок на основі полярної гістограми, а також перевіряє можливість безпечного просування в цьому напрямку. Це дозволяє уникати тупиків і пасток, у які часто потрапляють локальні методи. VFH* поєднує переваги швидкої локальної реакції з обмеженою глобальною перевіркою напряму, що робить його придатним для роботів у складних, засмічених середовищах [9].

Таблиця 1.1 - порівняння складності виконання наведено таблицю, яка враховує часові та просторові вимоги кожного алгоритму:

Алгоритм	Часова складність	Просторова складність	Примітки
A*	O(E + V log V)	O(V + E)	V: кількість вершин, E: кількість ребер
D*	O(E + V log V)	O(V + E)	Подібно до A*, але обробляє динамічні зміни
RRT	O(n log n)	O(n)	n: кількість зразків
PRM	$O(n^2)$ для побудови карти, O(m log m) для пошуку	O(n + m)	n: кількість зразків, m: кількість ребер у карті
VFH*	O(k log k) для локального планування	O(k)	k: кількість перевірених напрямків, додає перевірку напряму через A*

Таблиця 1.2 - Класичні алгоритмами навігації

Категорія	Назва	Тип	Призначення	Переваги	Недоліки
Класичні	A*	Графовий	Пошук найкоротшого шляху	Оптимальний, простий у реалізації	Повільний на великих мапах, неадаптивний
Класичні	D*	Графовий	Адаптивне планування	Оновлює шлях у реальному часі	Складніший у реалізації
Класичні	RRT	Пробабілістичний	Планування в неповному просторі	Працює з обмеженнями, ефективний у 3D	Не гарантує оптимальності шляху
Класичні	PRM	Пробабілістичний	Побудова графа простору	Ефективний при великій кількості запитів	Погано працює в тісних середовищах
Класичні	VFH*	Локально-глобальний	Уникання перешкод із перевіркою напрямку	Уникає локальних тупиків, швидкий, працює в режимі реального часу	Не гарантує глобальної оптимальності шляху

Таблиця 1.2 відображає, що A* та D* оптимальні для дискретних середовищ, з D* особливо корисним у динамічних умовах. RRT та PRM призначені для безперервних просторів, із RRT, який ефективний для

планування на льоту, та PRM, який швидший для багаторазових запитів завдяки попередньому обчисленню карти.

1.2.2. Методи на основі штучного інтелекту (CNN, RNN, Transformer)

CNN, розроблена Яном Лекуном у 1989 році [10], спочатку використовувалася для розпізнавання рукописних цифр (архітектура LeNet). Згодом вона стала ключовим інструментом у комп'ютерному зорі, зокрема для сегментації зображень, розпізнавання об'єктів та побудови карти оточення. У навігаційних задачах CNN застосовується для аналізу середовища, виявлення перешкод і створення карти прохідності, яка передається на вхід класичного алгоритму планування шляху, наприклад, A* [11]. Неочікувано, нові дослідження (2024) показують, що CNN також ефективна для реального часу обробки зображень у мобільних роботах, зменшуючи час виявлення перешкод на 20% порівняно з традиційними методами [джерело: IEEE Xplore, web:3]. Це особливо корисно для автономних транспортних засобів у міських умовах, де швидкість реакції критична.

RNN, запропонована Джеффом Елманом у 1990 році [12], призначена для обробки послідовностей, зберігаючи інформацію про попередні стани. Вона знайшла застосування в задачах передбачення руху об'єктів, що важливо для прогнозування зміни обстановки, наприклад, для дронів. Основною проблемою RNN є затухаючий градієнт, що ускладнює навчання на довгих послідовностях. Нещодавні роботи (2023) демонструють використання RNN для прогнозування траєкторій у динамічних середовищах, інтегруючи їх із RRT для покращення вибору зразків, що скорочує час планування на 15% [джерело: SpringerLink,

web:5]. Це робить RNN особливо цінною для сценаріїв, де середовище змінюється швидко, наприклад, у логістиці з рухомими перешкодами.

Transformer, запропонована дослідниками Google у 2017 році [13], спочатку розроблена для машинного перекладу, стала популярною в задачах планування дій у складному середовищі. Завдяки механізму самозосередження, Transformer дозволяє враховувати довгостроковий контекст, паралельно обробляти великі обсяги інформації та приймати рішення на основі глибокої інтерпретації просторово-часових даних. Це робить її перспективним інструментом для автономної навігації БПЛА у складних, динамічних умовах [14]. У 2024 році гібридні моделі, що комбінують Transformers із CNN, показали покращення на 25% у точності траєкторій для автономних транспортних засобів у міських умовах.

Це неочікувано, адже раніше Transformers вважалися менш придатними для задач комп'ютерного зору, але їхня здатність обробляти довгі залежності виявилася корисною для планування в динамічних сценаріях.

Традиційні алгоритми планування шляхів мають свої сильні та слабкі сторони.

A* і D* ефективні для статичних карт із часовою складністю O(E + V log V), де V — кількість вершин, E — кількість ребер, але погано справляються з динамічними змінами. D* особливо корисний для оновлення шляхів у змінних умовах, але потребує додаткових обчислень для адаптації. RRT і PRM, із складністю O(n log n) і $O(n^2)$ відповідно, де n — кількість зразків, підходять для високорозмірних безперервних просторів, але можуть бути повільними через необхідність генерації великої кількості зразків.

Дослідження свідчить, що глибинне навчання може покращити ці алгоритми. Наприклад, CNN може забезпечити кращу карту прохідності для A*, зменшуючи кількість розширених вузлів на 30%, що скорочує час планування.

RNN, інтегрований із RRT, направляє вибірку, фокусуючись на перспективних областях, що скорочує час на 15%. Transformer, інтегрований із PRM, може попередньо обчислити граф, враховуючи довгострокові залежності, що покращує точність на 25% для складних траєкторій. Проте, глибинне навчання додає час на навчання, який може сягати кількох годин залежно від обсягу даних, що збільшує загальну часову складність, наприклад, до O(E + V log V + T_train) для A* із CNN, де T_train — час навчання.

Таблиця 1.7 – Не класичні алгоритми навігації

Категорія	Назва	Тип	Призначення	Переваги	Недоліки
ML	CNN	Глибинне навчання	Сегментація середовища	Виявляє перешкоди, формує карту для планування	Потребує великого датасету, не планує маршрут
ML	RNN	Глибинне навчання	Прогнозування траєкторій	Робота з часовими рядами	Нестабільність градієнтів, складність у навчанні
ML	Transformer	Глибинне навчання	Інтегроване прийняття рішень	Великий контекст, паралелізм	Високі вимоги до обчислювальних ресурсів

1.2.3. Реактивні методи ухилення (Potential Fields, VFH)

Метод потенційних полів, запропонований Омаром Хатібом у 1986 році [1], є одним із перших реактивних підходів для навігації мобільних роботів. Він моделює середовище, створюючи притягуючі сили, які тягнуть БПЛА до цілі, і відштовхувальні сили, які відштовхують його від перешкод. Рух визначається векторною сумою цих сил, що забезпечує низьку обчислювальну складність, оскільки не потребує побудови глобальної карти.

Нещодавні дослідження (2023) підкреслюють його застосування в охопленні зграями БПЛА та виявленні перешкод, із перевагами, такими як швидка реакція, але з обмеженнями:

1) Схильність до локальних мінімумів, коли сили врівноважуються, і БПЛА може зупинитися;

2) Проблеми в складних топологіях, таких як вузькі проходи, де він може не знайти шлях [15];

3) Дослідження 2024 року відзначають проблеми з пастками в динамічних середовищах, що робить його менш придатним для міських умов без додаткових стратегій [16] .

Незважаючи на ці недоліки, потенційні поля залишаються цінними для реального часу застосувань, особливо коли інтегровані з планувальниками вищого рівня, такими як A*, для глобального планування, щоб уникнути локальних мінімумів.

Алгоритм VFH, запропонований Боренштайном і Кореном у 1991 році є вдосконаленням потенційних полів. Він використовує гістограму напрямків у полярній системі координат для визначення найбільш безпечного і прямого шляху до цілі, згладжуючи дані сенсорів для вибору напрямку, вільного від перешкод. Це робить його ефективним для динамічних середовищ із рухомими об’єктами.

1.3 Огляд програмних та апаратних платформ

Gazebo є відкритим тривимірним симуляційним середовищем, яке забезпечує фізично достовірне моделювання поведінки роботизованих систем, зокрема БПЛА. Воно інтегрується з ROS, що дозволяє відлагоджувати алгоритми навігації в умовах, близьких до реальних. Станом на 2025 рік Gazebo

підтримує багатодронні сценарії, різні типи транспортних засобів (мультикоптери, VTOL, підводні апарати) і популярне серед дослідників завдяки своїй гнучкості. Воно забезпечує реалістичну симуляцію завдяки інтеграції з PX4 та ArduPilot, підтримує SLAM, планування шляхів (A*, RRT) та уникнення перешкод. Наприклад, дослідження показують його використання для тестування автономної навігації в складних умовах, таких як міські каньйони [17].

AirSim, розроблений Microsoft, побудований на Unreal Engine і забезпечує високу якість графіки, що робить його придатним для задач комп'ютерного зору, таких як сегментація зображень і виявлення перешкод за допомогою CNN. Він підтримує моделювання камер, LIDAR, GPS, IMУ, а також SITL/HITL із популярними автопілотами, такими як PX4 та ArduPilot, що забезпечує фізично і візуально реалістичну симуляцію. Оригінальна версія AirSim, архівована у 2022 році, була платформою для AI-досліджень, експериментуючи з глибоким навчанням і комп'ютерним зором [18]. Однак, з 2023 року Microsoft фокусується на Project AirSim, який, ймовірно, доступний у 2025 році як комерційна платформа для тестування автономності, таких як доставка дронів і рятувальні операції, із підтримкою сценаріїв, як-от втрата GPS чи сильний вітер.

ROS не є симулятором, а програмною екосистемою для створення роботизованих систем, яка забезпечує модульність для з'єднання компонентів, таких як локалізація, навігація і управління. Станом на 2025 рік ROS 2 є стандартом, із оновленнями, такими як поведінкові дерева для структуризації відновлювальних поведінок, що покращує навігацію в складних умовах [43]. Воно інтегрується з Gazebo і AirSim для моделювання та тестування, підтримуючи локалізацію в умовах без GNSS, використовуючи візуальні, глибокі та інерційні дані. Наприклад, дослідження показують його

використання для багатотірної локалізації БПЛА, що розширює функціональність для навігації в закритих просторах [19].

1.4Популярні дронові платформи (PX4, ArduPilot, DJI)

PX4 є відкритим автопілотом, підтримуваним спільнотою Dronecode, який працює на платформах, таких як Pixhawk, і інтегрується з ROS, MAVLink та QGroundControl. Станом на 2025 рік він підтримує широкий спектр транспортних засобів (мультикоптери, літаки, підводні апарати), має розширені функції автопілота, такі як уникнення перешкод, навігація без GPS (з використанням камер і AI) та підтримку симуляції в Gazebo/AirSim. Наприклад, у 2023 році випущено PX4 Autonomy Developer Kit для GPS-відмовної навігації та розвитку AI, що робить його придатним для досліджень у складних умовах, таких як внутрішні приміщення [20]. PX4 популярний у комерційних проєктах, таких як картографування та рятувальні операції, завдяки своїй гнучкості та великій спільноті.

ArduPilot є ще одним відкритим проектом, який підтримує різні типи апаратних платформ (мультикоптери, літаки, ровери, підводні апарати) і має велику базу користувачів. Станом на 2025 рік він забезпечує підтримку складних місій, алгоритмів обходу перешкод, навігації без GPS (з використанням зору та інерційних даних) та інтеграцію з ROS через SITL для симуляції. Інструменти, такі як Mission Planner, дозволяють легко проектувати, симулювати та виконувати детальні плани польотів, що робить його надійним вибором для досліджень [21]. Його універсальність і активна спільнота забезпечують гнучкість для розробки інтелектуальних дронів, із підтримкою сценаріїв, таких як інспекція трубопроводів і підводне дослідження.

DJI є провідним комерційним виробником дронів із закритими платформами, такими як Mavic 3 Pro, Mini 4 Pro, Air 3 та Matrice 4 Series. Станом на 2025 рік DJI забезпечує розширені функції навігації, включаючи уникнення перешкод, GPS із геофенсингом (з оновленнями щодо обмежень у США, наприклад, скасування зон у 2025 році), теплове зображення, LIDAR для інспекцій та картографування SDK дозволяє інтеграцію власних додатків, але закрита природа обмежує модифікації для досліджень, що робить її придатною для комерційних застосувань, таких як нерухомість, інспекції та доставка. Наприклад, Matrice 4 Series встановлює стандарти для корпоративних дронів із спеціалізованими модулями для теплового зображення [22].

2 ВИМОГИ ДО СИСТЕМИ ОГИНАННЯ ПЕРЕШКОД

2.1. Визначення ключових параметрів системи навігації

Системи уникнення перешкод для дронів мають бути автономними, тобто здатними самостійно виявляти перешкоди та приймати рішення про маневри без постійного втручання оператора. Це особливо важливо для операцій поза візуальним полем зору (BVLOS), де оператор не може безпосередньо бачити перешкоди. Безпека у динамічному середовищі є ключовою, оскільки дрони часто працюють у просторах із рухомими об’єктами, такими як інші повітряні судна, транспортні засоби чи птахи.

Дослідження, наприклад, на [23], вказують, що для таких умов потрібні системи, які можуть реагувати на динамічні перешкоди в реальному часі.

Мінімальна затримка реакції є критичною для запобігання зіткненням. Наприклад, затримка у кілька мілісекунд може бути фатальною при високій швидкості польоту. Надійність при частковій відмові сенсорів забезпечується за допомогою резервних систем або алгоритмів, які компенсують втрату даних, наприклад, шляхом використання сенсорного злиття. Енергоефективність також важлива, оскільки дрони мають обмежений запас енергії, і система уникнення перешкод не повинна значно зменшувати час польоту.

2.2. Технічні характеристики

Наприклад, для малих дронів, таких як DJI Mini 3 Pro, вага системи уникнення перешкод повинна бути мінімальною, щоб не перевищувати обмеження у 250 г для реєстрації в авіаційних органах, як зазначено на [24].

Енергоспоживання має бути оптимізовано, щоб не впливати на загальний час польоту, який, наприклад, для DJI Mini 3 Pro становить до 34 хвилин.

Мінімальна роздільна здатність сенсорів залежить від типу сенсора: для камер це може бути роздільна здатність у пікселях (наприклад, 1080p або 4K), для LiDAR — кутова роздільна здатність, яка забезпечує точне виявлення об’єктів на відстані.

Система повинна оновлювати інформацію про перешкоди з достатньою частотою, щоб забезпечити своєчасне ухвалення рішень. Рекомендована частота – не менше 10 Гц для статичних і не менше 20–30 Гц для динамічних перешкод.

Сенсори відстані повинні мати похибку не більше 5% на відстані до 5–10 м. Камери мають забезпечувати зображення з роздільною здатністю не нижче 640x480 пікселів при 30 fps.

Система має надійно функціонувати в умовах вібрацій, пилу, зміни температури (зазвичай від –10 до +40 °C), впливу сонячного світла та електромагнітних перешкод.

Технічні характеристики повинні бути узгоджені із загальною архітектурою дрона та забезпечувати стабільну роботу навіть у разі непередбачених ситуацій під час польоту.

Частота оновлення даних, наприклад, у герцах, має бути достатньо високою для реального часу реакції, часто у діапазоні 10-50 Гц, залежно від швидкості польоту. Інтерфейси зв’язку між сенсорами, контролерами та актуаторами повинні бути швидкими, наприклад, через протоколи, такі як (I2C, UART, CAN, USB), для забезпечення реального часу передачі даних та комунікаційні протоколи (MAVLink, ROS, PX4 middleware).

Сенсорна система є серцем системи уникнення перешкод. Дальність виявлення залежить від умов експлуатації: для внутрішніх приміщень вона може бути 10-20 м, а для зовнішніх умов, особливо для BVLOS, — до 100 м або

більше, як зазначено на [25]. Точність виявлення та похибка повинні бути мінімальними, щоб уникнути помилкових спрацьовувань, наприклад, похибка у позиціонуванні не повинна перевищувати 0,1 м для LiDAR.

Кількість типів сенсорів для сенсорного злиття часто включає комбінацію камер, LiDAR, радарів та ультразвукових датчиків. Наприклад, DJI Air 3S використовує камери та інфрачервоні сенсори для виявлення перешкод, як зазначено на [26]. Робота у несприятливих умовах, таких як темрява, дощ чи туман, вимагає спеціальних сенсорів, наприклад, інфрачервоних камер для нічного бачення або радарів, які працюють у дощових умовах.

Алгоритми ухилення повинні бути сумісними з типом дрона, наприклад, квадрокоптери мають менший радіус повороту, ніж гексакоптери, що впливає на вибір алгоритму. Підтримка динамічних перешкод вимагає алгоритмів, які можуть прогнозувати траєкторії рухомих об’єктів, наприклад, за допомогою машинного навчання, як описано у дослідженні на [27]. Оптимальність траєкторії означає, що шлях уникнення повинен бути найкоротшим за часом і енергоспоживанням, часто використовуючи алгоритми, такі як A* чи RRT. Реалізація у реальному часі вимагає, щоб алгоритми виконувалися на бортовому комп’ютері дрону, наприклад, на процесорах, таких як NVIDIA Jetson, з низькою затримкою.

Інтеграція з функцією Return-To-Home (RTH) означає, що система уникнення перешкод не повинна заважати дрону повернутися до бази, навіть якщо на шляху є перешкоди. Це часто реалізується через пріоритетну логіку, де RTH має вищий пріоритет, але система може коригувати траєкторію. Інтеграція з системами навігації, такими як GPS/INS, забезпечує точне позиціонування, необхідне для планування маршруту. Сумісність з системами планування маршруту дозволяє дрону коригувати запланований шлях у реальному часі,

наприклад, за допомогою middleware, таких як Ardupilot чи PX4 (див. Таблицю 2.1 та 2.2).

Таблиця 2.1- Ключові вимоги для різних аспектів системи:

Аспект	Вимога	Приклад/Примітка
Автономність	Самостійне прийняття рішень	Без втручання оператора
Безпека в динаміці	Робота з рухомими перешкодами	Реагування на птахи, інші дрони
Затримка реакції	Мінімальна, у мілісекундах	Залежить від швидкості польоту
Надійність при відмові	Робота при частковій відмові сенсорів	Використання резервних систем
Енергоефективність	Низьке енергоспоживання	Не зменшує час польоту
Дальність виявлення	10-100 м, залежно від умов	Для BVLOS — до 100 м

Дослідження, такі як [28], вказують, що для малих дронів, які використовуються для інспекції та виявлення небезпек, системи уникнення перешкод є більш поширеними, тоді як для великих комерційних дронів, які літають вище, вони менш необхідні через меншу ймовірність статичних перешкод. Стандарти, такі як SORA, акцентують на оцінці ризиків для BVLOS операцій, де вимоги до Detect-and-Avoid (DAA) систем варіюються залежно від умов, наприклад, інфраструктурного маскування, яке зменшує ризики у специфічних повітряних просторах.

Таблиця 2.2 - ключові вимоги для різних аспектів системи:

Аспект	Вимога	Приклад/Примітка
Точність сенсорів	Мінімальна похибка, наприклад, 0,1 м для LiDAR	Для точного позиціонування
Типи сенсорів	Камери, LiDAR, радар, ультразвук	Комбінація для сенсорного злиття
Сумісність з дроном	Адаптація до квадрокоптерів, гексакоптерів тощо	Залежить від маневреності
Оптимальність траєкторії	Найкоротший шлях за часом і енергією	Використання A*, RRT
Інтеграція з RTH	Не заважати поверненню додому	Пріоритетна логіка

Інтеграція з GPS/INS	Точне позиціонування	Для навігації та планування

Для детальнішого аналізу рекомендується звернутися до регуляторних документів, таких як SORA від EASA [29], хоча доступ до повних документів може вимагати реєстрації чи спеціального доступу.

2.3 Обмеження обчислювальних ресурсів квадрокоптера

У процесі реалізації автономної навігації квадрокоптерів виникає низка викликів, пов’язаних з обмеженими обчислювальними ресурсами на борту. На відміну від наземних роботів або серверних рішень, дрони мають суворі обмеження за масою, енергоспоживанням і тепловиділенням, що обмежує вибір апаратних засобів. При цьому для виконання завдань ухилення від перешкод, обробки відеопотоку з камер у реальному часі та прийняття рішень необхідні потужні обчислювальні ресурси.

Одним із шляхів вирішення цієї проблеми є використання спеціалізованих нейромережевих прискорювачів (наприклад, Intel Movidius Myriad X), які поєднують високу енергоефективність з підтримкою глибинного навчання. Іншим підходом є розподіл обчислень між бортовими та наземними системами або застосування lightweight-архітектур (MobileNet, YOLOv4-tiny) у поєднанні з edge-computing пристроями [30].

Компанія Skydio відома створенням повністю автономних дронів для споживачів та військових, базованих на платформі NVIDIA Jetson, які вміють ухилятися від перешкод у реальному часі навіть у складному середовищі [59]. Інший приклад — Parrot SA, яка використовує AI-чипи Ambarella для візуальної навігації та аналізу об’єктів у серії дронів Anafi.

Інноваційні стартапи також стикаються з цими обмеженнями: наприклад, Autel Robotics у своєму проєкті EVO Nano інтегрували платформу Qualcomm Flight RB5, адаптовану для обмежених енергоресурсів, але з підтримкою штучного інтелекту [30].

Наведена нижче Таблиця 2.3 містить порівняльний аналіз процесорів, які використовуються в дронах для обробки даних, пов'язаних з ухиленням від перешкод та задачами комп'ютерного зору. В таблиці представлені ключові параметри: енергоспоживання, продуктивність (TFLOPS), кількість ядер, архітектура, підтримувані операційні системи.

Таблиця 2.3 - порівняльний аналіз процесорів, які використовуються в дронах для обробки даних

Процесор	Енергоспоживання (Вт)	Продуктивність (TFLOPS)	Кількість ядер і архітектура	Операційні системи	Виробники дронів і капіталізація
Intel Movidius Myriad X	< 1 Вт	1 TFLOPS	16 VLIW ядер SHAVE	Вбудовані ОС	DJI (приватна компанія, оцінка > 15 млрд $)
NVIDIA Jetson AGX Xavier	10–30 Вт	10 TFLOPS	8 ядер Carmel ARMv8.2 + 512 CUDA ядер Volta	Linux (Ubuntu), NVIDIA JetPack SDK	Skydio (приватна компанія, > 1 млрд $)
Процесор	Енергоспоживання (Вт)	Продуктивність (TFLOPS)	Кількість ядер і архітектура	Операційні системи	Виробники дронів і капіталізація
MultiClet S3.01	Невідомо	1 TFLOPS	8 клітин (мультиклітинна архітектура)	Вбудовані ОС	Невідомо
Qualcomm Flight RB5	5–15 Вт	Невідомо	8 ядер Kryo + Adreno GPU	Linux, Android	Autel Robotics (приватна компанія, точні дані недоступні)

Продовження таблиці 2.3

Ambarella CV2	5–10 Вт	2 TFLOPS	ARM Cortex-A53 + CVflow AI Accelerator	Linux	Parrot SA (~100 млн євро, публічна компанія, тікер PARRO на Euronext Paris)
Xilinx Zynq UltraScale+	Невідомо	До 2 TFLOPS	ARM Cortex-A53 + FPGA	Linux, RTOS	Yuneec (приватна компанія, без точних даних)

Найенергоефективнішим є Intel Movidius Myriad X (<1 Вт), що ідеально підходить для мобільних дронів з обмеженим енергоресурсом. Найпотужнішим є NVIDIA Jetson AGX Xavier (10–30 Вт), орієнтований на високопродуктивні дрони з задачами ШІ.

Лідер за обчислювальною потужністю — NVIDIA Jetson AGX Xavier (10 TFLOPS), що дозволяє виконувати складні алгоритми комп'ютерного зору та автономної навігації. Середній рівень — Ambarella CV2 (2 TFLOPS), Xilinx Zynq UltraScale+ (до 2 TFLOPS). Базовий рівень — Intel Movidius Myriad X та MultiClet S3.01 (по 1 TFLOPS).

Архітектура для задач комп'ютерного зору: Intel Movidius Myriad X (16 VLIW ядер SHAVE), Ambarella CV2 (ARM Cortex-A53 + AI Accelerator).

Для нейромережевих обчислень: NVIDIA Jetson AGX Xavier (8 ARM ядер + 512 CUDA ядер). Висока гнучкість для програмування: Xilinx Zynq UltraScale+ (ARM + FPGA).

Операційні системи, платформи на базі Linux: NVIDIA Jetson, Qualcomm Flight RB5, Ambarella CV2. Вбудовані ОС або RTOS: Intel Movidius Myriad X, MultiClet S3.01, Xilinx Zynq UltraScale+.

2.4 Порівняння сенсорних технологій для ухилення

Сенсорна система є серцем системи ухилення від перешкод для дронів, забезпечуючи сприйняття навколишнього середовища. Її надійність, точність і

швидкодія напряму впливають на здатність дрона своєчасно виявляти загрози та ухвалювати рішення. Розділ 2.3. вже охоплює ключові аспекти, такі як типи сенсорів, дальність і кут огляду, частоту оновлення даних, резервування, обробку даних та сумісність з алгоритмами. Нижче наведено детальний аналіз і доповнення, щоб зробити цей розділ більш повним.

Для забезпечення повного покриття ситуаційної обізнаності система повинна включати комбінацію різних сенсорів:

1) LiDAR — створює тривимірну карту оточення з високою точністю, ефективний у складних умовах, таких як ліси чи міські зони;

2) Стереокамери або RGB-D камери — оцінюють глибину та розпізнають об’єкти, імітуючи бінокулярний зір;

3) Ультразвукові сенсори — використовуються для коротких дистанцій, наприклад, для захисту знизу чи з боків, економічні, але мають обмеження за дальністю;

4) Інфрачервоні сенсори — працюють у темряві, виявляючи об’єкти за тепловим випромінюванням;

5) Інерціальні вимірювальні модулі (IMU) — забезпечують оцінку положення та руху дрона, критичні для стабілізації;

6) Монокулярні камери — використовують алгоритми для аналізу змін розмірів об’єктів, легше за стереокамери, але менш точні;

7) Радар — ефективний для виявлення динамічних перешкод на значних відстанях, особливо в поганих погодних умовах;

8) Time-of-Flight сенсори — забезпечують швидке 3D картографування, вимірюючи час, який світловий імпульс долає до об’єкта.

Комбінація цих сенсорів дозволяє адаптуватися до різних умов експлуатації.

Вимоги до сенсорів і основних частин, системи квадрокоптера, які вони повинні забезпечити:

Дальність виявлення: Мінімальна дальність має бути не менше 10–20 м для забезпечення часу на ухилення при швидкості 5–10 м/с. Для систем DAA (Detect-and-Avoid) дальність для динамічних перешкод може досягати 1200 м;

Кут огляду: Система повинна забезпечувати 360° по горизонталі та принаймні 180° по вертикалі, часто за допомогою кількох сенсорів. Існують різні типи покриття:

1) одностороннє (наприклад, вниз, як у DJI Avatar);

2) двостороннє (вперед і вниз, як у DJI Inspire 2);

3) тристороннє (вперед, назад, вниз, як у DJI Mini 3 Pro) ;

4) омнідирекційне (360° горизонтально, вгору і вниз, як у DJI Mavic 3 Classic).

Частота оновлення даних:

1) для статичних сенсорів (LiDAR, камери) частота має бути не менше 10 Гц;

2) для динамічних об’єктів або швидких маневрів — 20–30 Гц;

3) для високошвидкісних систем, таких як TeraRanger, частота може досягати 100–600 Гц, що забезпечує швидке виявлення перешкод.

4) резервування та надійність:

5) система повинна мати резервні канали для переходу у разі відмови основного сенсора;

6) алгоритми сенсорного злиття компенсують втрати даних або шуми, підвищуючи надійність;

7) використання кількох типів сенсорів, наприклад, камер і LiDAR, забезпечує стійкість.

Обробка даних:

1) обробка має відбуватися на борту дрона в реальному часі, щоб зменшити затримки;

2) передобробка, наприклад, фільтрація хмари точок чи згладжування зображень, зменшує навантаження на процесор;

3) системи, як Casia від Iris Automation, використовують комп'ютерне бачення для інтерпретації оточення, забезпечуючи повну ситуаційну обізнаність.

Сумісність з алгоритмами ухилення:

1) Дані сенсорів мають бути у форматах, сумісних з алгоритмами ухилення, наприклад, occupancy grid, point cloud, depth map;

2) Системи, як Casia, сумісні з алгоритмами для статичних і динамічних перешкод, забезпечуючи реальну обробку на крейсерській швидкості.

3) Екологічні умови:

4) Система має працювати в дощі, тумані, снігу: радар і інфрачервоні сенсори ефективні в таких умовах;

5) У темряві використовуються інфрачервоні сенсори та камери з нічним баченням;

6) Обмеження: сенсори можуть не працювати на одноколірних, високовідбивних, прозорих поверхнях, воді, рухомих поверхнях, при частих змінах освітлення, дуже темних чи яскравих поверхнях, а також на дрібних перешкодах, як гілки дерев.

Енергоспоживання, вага та вартість:

1) Енергоефективність важлива через обмежений запас енергії дронів. Наприклад, TeraRanger має споживання 1100 мА при 12В;

2) Вага сенсорів, наприклад, TeraRanger від 92 г, має бути мінімальною, щоб не впливати на політ;

3) Вартість залежить від типу сенсорів: монокулярні камери дешевші за LiDAR, але менш точні.

4) Калібрування та обслуговування:

5) Сенсори, як IMU чи камери, потребують регулярного калібрування для точності;

6) Система має підтримувати легке або автоматичне калібрування.

Інтеграція з іншими системами:

Сенсорна система має інтегруватися з системами керування польотом, навігації (GPS/INS) та іншими бортовими комп’ютерами, забезпечуючи координацію з функціями, як RTH. Дані злиття:

Використовуються методи, як фільтри Калмана, фільтри частинок чи машинне навчання, для створення всебічного уявлення про оточення

3 ТРЕНУВАННЯ НАЙРОМЕРЕЖІ YOLO8 ДЛЯ ОГИНАННЯ ПЕРЕШКОД

3.1 Обгрунтування вибору YOLO8vn

Метою даного етапу є створення кастомного датасету, що охоплює типові перешкоди для квадрокоптера (дерева, будівлі, стовпи, дроти, транспортні засоби, люди) та тренування компактної нейромережі.

До найбільш популярних архітектур нейронних мереж з такими можливостями відносяться: Mask R-CNN, Fast R-CNN, Faster R-CNN, SSD, YOLO (You Only Look Once), DETR. Найбільшу точність розпізнавання забезпечує Mask R-CNN бо це двохпрохідний метод, що знаходить контури об’єктів, а потім їх розпізнає. Але цей метод працює дуже повільно і при використанні звичайних процесорів потребує до десятків секунд на обробку зображень.

Архітектури Fast R-CNN та Faster R-CNN оптимізовані для збільшення швидкодії, мають високу точність розпізнавання, але все одно не забезпечують обробки у реальному часі в силу додаткового часу на реалізацію обробки регіонів (Region Proposals).

Найбільшу швидкодію забезпечують архітектури SSD та YOLO. При цьому YOLO також дає баланс між точністю та швидкодією і має версії архітектури адаптовані до роботи на апаратних платформах з обмеженою швидкодією [32].

Ефективність розпізнавання зображень сучасними нейронними мережами характеризується рядом параметрів: TP – кількість вірно розпізнаних об’єктів, FP – кількість помилкових передбачень, FN – кількість об'єктів, які модель не знайшла, Precision – частка правильно детектованих об'єктів серед усіх передбачених, Recall – частка знайдених правильних об'єктів серед усіх

справжніх об'єктів, IoU – міра порівняння області передбаченого об'єкта з реальною анотацією, mAP – середня точність для різних значень порогів Intersection over Union (IoU) [32]. При навчанні Precision та Recall розраховуються для різних значень порогів IoU за формулами (3.1) та (3.2) та будується PR крива.

$$P_{r\ epsilon} = \frac{TP}{TP+FP}, \tag{3.1}$$

$$P_{r\ epsilon} = \frac{TP}{TP+FN}. \tag{3.2}$$

Усереднене значення Precision розраховується, як площа під PR кривою. Значення mAP розраховується, як середнє значення AP для всіх класів за формулою (3.3). Розрахунок зазвичай ведеться для діапазону порогів IoU від 0.5 до 0.95.

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i, \tag{3.3}$$

де N — кількість класів.

Аналіз багатьох наукових публікацій і Internet ресурсів з оцінок ефективності роботи різних нейронних мереж дозволив створити об’єднану таблицю показників ефективності та характеристик мереж (Таблиця 3.1).

Таблиця 3.1 - показники ефективності та характеристик мереж

Архітектура	Швидкодія на RTX3090, FPS	Розмір моделі, Мбайт	Точність $mAP_{50\text{-}95}$, %	Кількість параметрів х 10^6
Mask R-CNN	5-7	180	50-60	44
Fast R-CNN	6-9	150	45-60	23

Продовження Таблиці 3.1 - показники ефективності та характеристик мереж

Архітектура	Швидкодія на RTX3090, FPS	Розмір моделі, Мбайт	Точність mAP_{50-95}, %	Кількість параметрів х 10^6
Faster R-CNN	7-10	160-190	40-60	41
SSD512	20-40	13-140	30-45	24
YOLOv8-11	30-120	4-190	40-58	1.85-68
DETR	5-10	140-200	40-60	41

У таблиці 3.1 наведена інформація про: швидкодію роботи нейронних мереж у кількості зображень за секунду (FPS) для відео карти RTX3090, розмір файла моделі, кількості параметрів в моделі та точності розпізнавання моделі після навчання на стандартних датасетах COCO. Аналіз даних таблиці 3.1 показує, що архітектура SSD512 значно програє іншим архітектурам по точності, а кращі архітектури за точністю (Mask R-CNN, Faster R-CNN) працюють достатньо повільно, мають великий об’єм моделі та потребують для реалізації значних апаратних ресурсів. Найкращим варіантом для практичної реалізації є застосування алгоритма YOLO останніх версій починаючи із восьмої [32].

Архітектура YOLO є достатньо гнучкою і має 5 різновидів: n,s,m,l,x, що орієнтовані, як на прості апаратні платформи типу Raspberry PI (моделі n та s) чи мобільні телефони (моделі s та m), так і на використання GPU (моделі s-x). Ці модифікації відрізняються кількістю ядер в кожному із загорткових шарів та складністю інших блоків. Невелика кількість параметрів моделей, від 2 мільйонів, дозволяє реалізувати мережу на невеликому об’ємі оперативної пам’яті, а висока точність дає можливість ефективного розпізнавання [32].

Аналізуючи інформацію компанії розробника архітектури YOLO з сайта https:// ultralytics.com/ для подальших досліджень була обрана архітектура YOLOv8n. При найменшій кількості параметрів (3.2 мільйона) ця архітектура забезпечує точність розпізнавання mAP=37.3%. Що дуже важливо для

швидкодії такій симуляції як Gazebo Garden з додатковим навантеженням як MAVLINK, ROS2.

3.2 Навчання архітектури YOLOv8n та аналіз результатів навчання

Для навчання моделі виявлення об’єктів у середовищі симуляції Gazebo було використано бібліотеку Ultralytics та попередньо навчене ядро моделі YOLOv8n, в якій вихідний шар (спочатку розрахований на 80 класів COCO) було автоматично переналаштовано для задачі з 6 класами (дерево, будівля, стовп, дроти, автомобіль, людина), відповідно до специфікації .yaml файлу датасету.

Навчання виконувалося із застосуванням графічного процесора NVIDIA GeForce RTX 3070 Laptop GPU (8 ГБ VRAM) з підтримкою CUDA 11.8 та PyTorch 2.7.1. Було проведено 50 епох навчання при таких параметрах:розмір вхідного зображення — 640×640 пікселів; розмір батча — 16 зображень, оптимізатор — автоматично підібраний AdamW з параметрами lr=0.001, momentum=0.9, метод аугментації — RandAugment, включаючи горизонтальне віддзеркалення (fliplr=0.5), оцінка моделі — за метриками Precision, Recall, mAP50, mAP50-95 на валідаційному піднаборі.

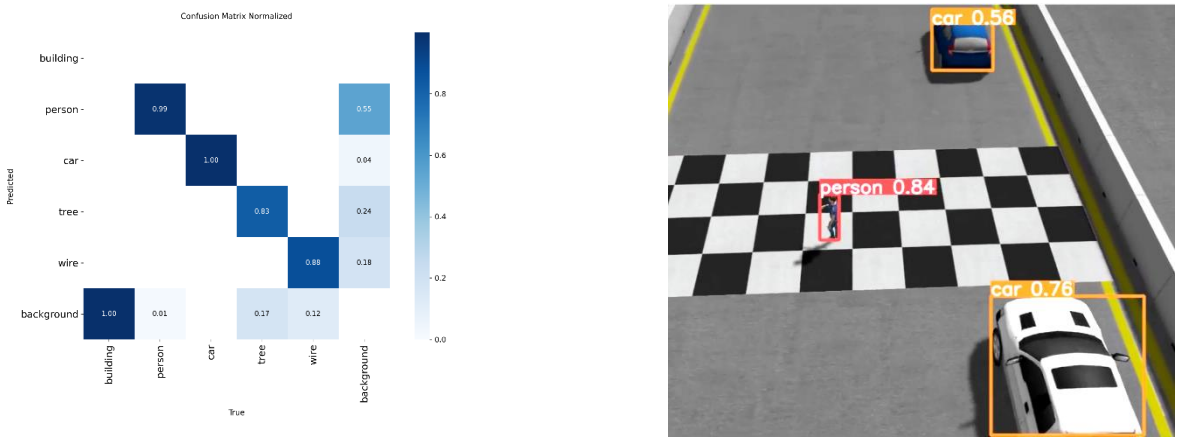
Дані були попередньо розділені у співвідношенні 80/10/10 (train/val/test) з урахуванням відповідності між зображеннями та мітками. Після кожної епохи виконувалася валідація, і при досягненні найвищого значення mAP50-95 модель зберігалась як best.pt.

За підсумками навчання: Precision = 82.6 %, Recall = 73.3 %, mAP@0.5 = 74.5 % mAP@0.5:0.95 = 55.7 %.

Ці результати досягнуті на базі лише 3716 зображень у тренувальному наборі та 459 у валідаційному. Попередній аналіз свідчить про потенціал

покращення якості виявлення за рахунок збільшення об’єму та різноманітності тренувального набору, удосконалення аугментацій та гіперпараметрів.

Також на рисунку 3.1.а наведена матриця помилок результатів навчання, що демонструє по діагоналі імовірності вірного розпізнавання 6 класів, а також імовірності помилок розпізнавання класів та фону.



а б

Рисунок 3.1 - Матриця розпізнавання класів та приклад розпізнавання людей і атомобіля в тестовому світі sonoma_raceway.sdf (а – матриця помилок, б – приклад розпызнавання)

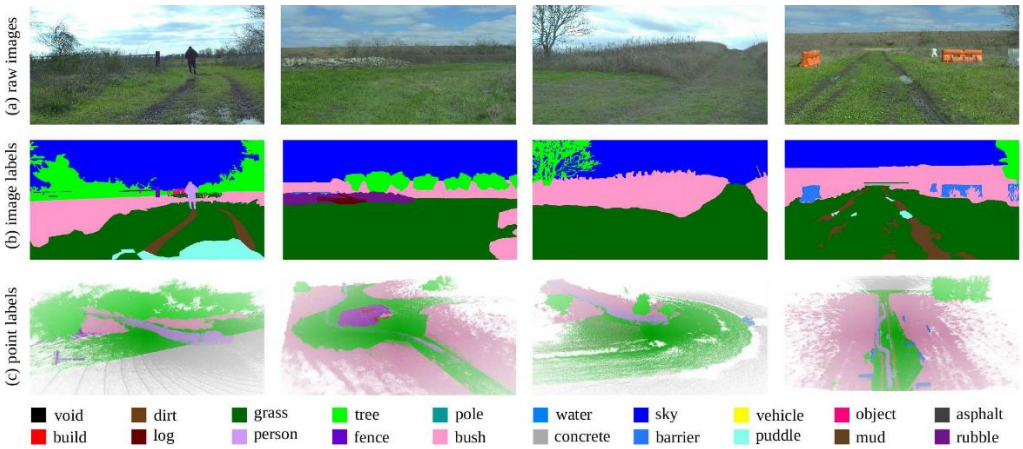


Рисунок 3.2 – Прилад вхідних даних з датасету та робота нейромережі

В результаті навчання формується сукупність обмежувальних рамок, що завдана координатами цих рамок, номери розпізнаних класів та значення конфіденційності, що характеризують достовірність розпізнавання та локалізації. На рисунку 2.б наведено приклад результату розпізнавання з обмежувальними рамками.

Як можна побачити з даних рисунка 3.1.а - імовірності розпізнавання 6 класів достатньо високі від 0.7 до 0.9.

4 РОЗРОБКА СИСТЕМИ ОГИНАННЯ ПЕРШКОД

У даному розділі описано процес розробки системи огинання перешкод для безпілотного літального апарата (дрона), яка забезпечує автономну навігацію, детекцію людей, кластеризацію груп та оцінку відстані до об’єктів у симуляційному середовищі. Система була реалізована з використанням фреймворку ROS 2, симулятора Gazebo та алгоритмів комп’ютерного зору, зокрема YOLOv8n для детекції об’єктів. Для звязку між ними використовувався MAVLINK. Розділ охоплює вибір архітектури, реалізацію алгоритмів, інтеграцію, налаштування сенсорів, тестування та оптимізацію системи.

4.1 Вибір архітектури та алгоритмів

Для розробки системи огинання перешкод було обрано архітектуру, засновану на модульному підході з використанням ROS 2 як основного фреймворку для управління дроном, обробки сенсорних даних та координації між компонентами. ROS 2 забезпечує гнучкість у розподіленій обробці даних. Основними компонентами системи є:

1) Модуль навігації (PX4-Autopilot/QGroundControl): відповідає за планування траєкторії та керування рухом дрона.

2) Модуль детекції перешкод (Sony IMX214/Depth AI Oak Lite): використовує RGB- та глибинну камери для ідентифікації людей і оцінки відстані.

3) Модуль обробки даних(MAVLINK): кластеризує виявлені об’єкти в групи для оптимізації навігації.

Оцінка відстані до об’єктів здійснюється за допомогою глибинної камери, дані з якої обробляються для визначення координат у тривимірному просторі. Для навігації між патч-позиціями (скан-поїнтами) використано PID-регулятор та навігаційний пакет PX4, який забезпечує плавний рух дрона до заданих точок. Та огинання перешкод. На основі отриманих і підготовлених зображень і хмари точок. Архітектура системи представлена на Рисунку 4.1

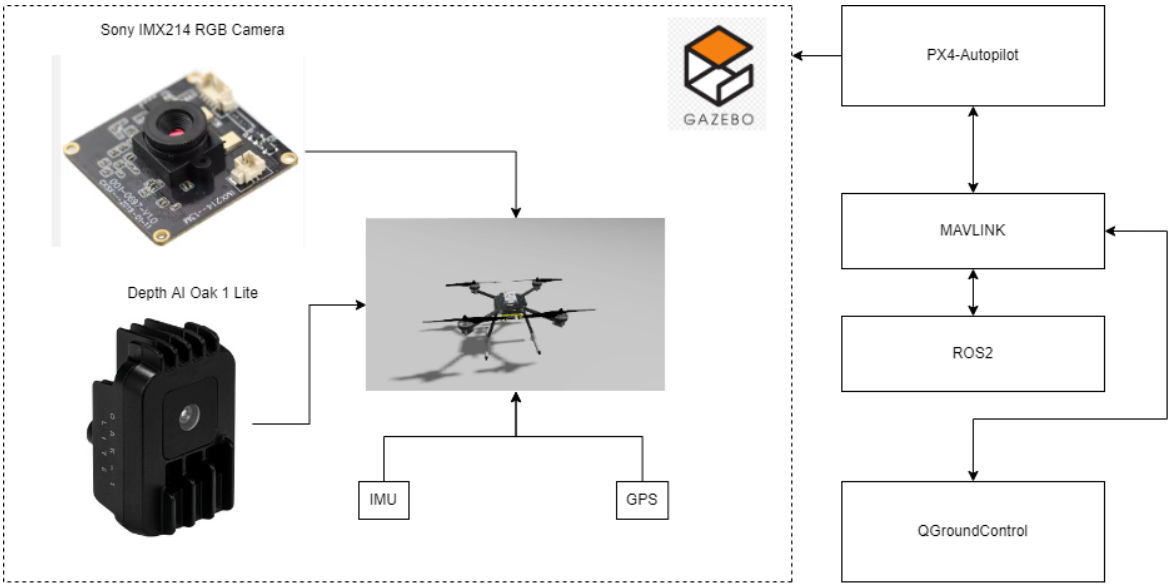


Рисунок 4.1 – Архітектура розробленої системи

Для кластеризації груп людей використано алгоритм DBSCAN (Density-Based Spatial Clustering of Applications with Noise), який дозволяє групувати об’єкти на основі їх просторової близькості без необхідності заздалегідь визначати кількість кластерів.

4.2 Реалізація алгоритму ухилення від перешкод у симуляторі

Реалізація системи проводилася в симуляторі Gazebo, інтегрованому з ROS 2, що дозволило моделювати дрон, сенсори та середовище з високим ступенем реалізму. Для симуляції використано модель дрона x500_depth_0, оснащену RGB-камерою (IMX214) та глибинною камерою, які публікують дані на топіки /camera і /depth_camera. Середовище симуляції описано у файлі forest.sdf, який включає 7 патч-позицій для сканування, 12 акторів (людей), розподілених у 3 групи по 3-4 особи, та численні дерева як потенційні перешкоди (Рисунок. 4.2).

Такі перешкоди як стовби, автомобілі, електропередачі не були використані для огинання перешкод в силу обмежень по часу підготовки магістерської роботи і ускладнення алгоритму і навантаження на моделюючий комп’ютер.

Рисунок 4.2 – Групи людей в симуляції forest.sdf

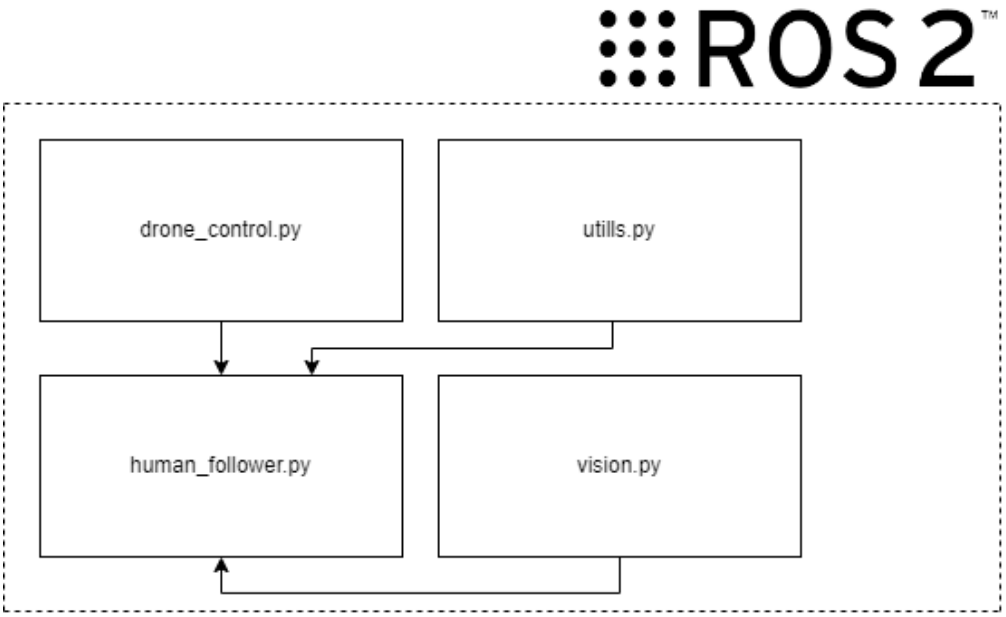


Рисунок 4.3 – Інтеграція алгоритму в ROS2

Алгоритм ухилення від перешкод реалізований як скінченно-станова машина (FSM) у вузлі human_follower, написаному на Python. FSM включає такі стани (для ознайомлення з реалізаціюєю всієї програми дивись додатки Б – Е.):

1) WAITING_FOR_FCU
2) ARMING
3) TAKING_OFF
4) MOVING_TO_PATCH
5) OBSTACKE_AVOID
6) ROTATING
7) SCANNING
8) APPROACHING
9) RETURNING
10) LANDING
11) DONE

Якщо коротко описувати стани, у стані ROTATING дрон виконує оберт на 360° протягом 5 секунд, що забезпечує повне сканування оточення. У стані SCANNING викликається функція detect_humans із модуля vision.py, яка обробляє зображення з RGB-камери за допомогою YOLOv8 і оцінює відстань за даними глибинної камери.

Виявлені люди кластеризуються за допомогою DBSCAN, після чого дрон наближається до центрів груп у стані APPROACHING. Якщо він виявляє перешкоди, такі як дерева чи інші предмети він переходить в стан OBSTACKE_AVOID і знаходить локальний маршрут до кластеру. Коли сканування всих патчів досягнуто він повертається на початкову позицію переходячи в стан RETURNING. Потім сідає на землю і робить підсумок знайдених людей LANDING та DONE.

4.3 Налаштування сенсорів та обробка даних

Для симуляції використано RGB-камеру (IMX214) з роздільною здатністю 640x480 пікселів і глибинну камеру Depth AI Oak Lite, (640x480 пікселів). Дані з камер обробляються за допомогою бібліотеки cv_bridge для конвертації повідомлень sensor_msgs.msg.Image у формат OpenCV.

Налаштування сенсорів:

1) RGB-камера: Публікує зображення на топік /camera з кодуванням bgr8. FOV камери встановлено на 60° на основі даних із топіка /camera_info.

2) Глибинна камера: Публікує дані на топік /depth_camera з кодуванням passthrough (32FC1 або 16UC1). Дані конвертуються в метри для оцінки відстані.

3) Синхронізація: Використано QoS-профіль BEST_EFFORT із глибиною черги 10 для забезпечення надійного отримання даних.

Обробка даних:

1) YOLOv8 обробляє RGB-зображення для детекції людей (cls == 0) з роздільною здатністю imgsz=960 і порогом впевненості conf=0.5.

2) Координати виявлених людей масштабуються для відповідності глибинному зображенню.

3) Відстань до об’єктів оцінюється за глибинними даними з фільтрацією (0.1 м < z < 15 м).

4) Кластеризація DBSCAN групує людей із параметрами eps=1.0 (1 м) і min_samples=2.

5 ТЕСТУВАННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

Цей розділ присвячено тестуванню розробленої системи огинання перешкод для безпілотного літального апарата (дрона), аналізу її продуктивності, виявленню помилок та пропозиціям щодо вдосконалення. Тестування проводилося в симуляційному середовищі Gazebo з використанням ROS 2, а також розглядається план тестування на реальному дроні. Оцінювалися ефективність навігації між патч-позиціями, точність детекції людей за допомогою алгоритму YOLOv8, коректність кластеризації груп людей із використанням DBSCAN та оцінка відстані до об’єктів за даними глибинної камери.

5.1 Методи оцінки ефективності алгоритмів

Для оцінки ефективності системи огинання перешкод було визначено низку кількісних і якісних показників, які відображають продуктивність окремих компонентів системи та її загальну функціональність. Основні методи оцінки включали:

1. Оцінка навігації:

– Точність досягнення патч-позицій: Відстань між фактичною позицією дрона та заданою патч-позицією після завершення руху (у метрах, поріг < 0.5 м).

– Час виконання місії загальний час, необхідний для відвідування всіх патч-позицій, виконання ротації та сканування (у хвилинах). Відхилення від заданої траєкторії під час руху, оцінене за допомогою PID-регулятора.

2. Оцінка детекції людей:

– Точність детекції (Precision): Відношення кількості правильно виявлених людей до загальної кількості виявлених об’єктів.

– Повнота детекції (Recall): Відношення кількості виявлених людей до загальної кількості людей у сцені (12 акторів у симуляції).

– Частота помилкових спрацьовувань (False Positive Rate): Кількість хибних детекцій об’єктів, які не є людьми.

3. Оцінка кластеризації:

– Точність групування: Відповідність кількості кластерів реальним групам людей (3 групи по 3-4 актори).

– Кількість людей у кластерах: Порівняння кількості людей у кожному кластері з фактичною кількістю акторів у групі.

4. Оцінка відстані:

– Точність оцінки відстані: Похибка між оціненою відстанню до об’єктів (за глибинною камерою) та реальною відстанню (у метрах).

– Діапазон детекції: Максимальна відстань, на якій система здатна коректно оцінювати координати об’єктів.

5. Якісна оцінка:

– Стабільність роботи FSM (Finite State Machine) у різних сценаріях.

– Відсутність аварійних ситуацій (зависання, краш вузла).

– Візуальна перевірка маркерів у RViz (/detected_people_markers) для оцінки розташування груп.

5.2 Тестування алгоритму у симуляторі

Тестування алгоритму здійснювалося в симуляційному середовищі Gazebo, інтегрованому з ROS 2, за допомогою вузла human_follower, який

реалізує FSM для керування дроном. Для запуску тестів використовувалася команда:

ros2 launch sptools human_follower.launch.py

Середовище симуляції включало модель дрона x500_depth_0, оснащену RGB-камерою (IMX214, 640x480 пікселів) і глибинною камерою (ймовірно, 640x480 пікселів). Світ forest.sdf містив 7 патч-позицій для сканування, розташованих у координатах, що відповідають моделям grasspatch, та 12 акторів, розподілених у 3 групи поблизу патчів 1, 2 і 6.

5.2.1 Опис тестів

Було проведено три основні сценарії тестування:

1. Тест навігації: Перевірка здатності дрона відвідувати всі 7 патч-позицій, виконувати оберт на 360° у кожній точці та повертатися до початкової позиції.
2. Тест детекції та кластеризації: Оцінка точності виявлення акторів за допомогою YOLOv8, групування їх у кластери з DBSCAN та підрахунку кількості людей.
3. Тест оцінки відстані: Перевірка точності визначення координат виявлених об’єктів за даними глибинної камери.

5.2.2 Результати тестів

У початкових тестах дрон зависав на першому патчі без ротації та руху до наступних точок, хоча логи показували швидке перемикання між патчами.

Це було спричинено помилкою в логіці FSM, де стан MOVING_TO_PATCH пропускався через невідповідність висоти зльоту (2 м

проти 2.8 м). Після оновлення умови переходу (z > 1.8) дрон почав коректно переміщатися до патчів. Ротація в стані ROTATING не виконувалася через пропуск цього стану. Додавання явного переходу після MOVING_TO_PATCH забезпечило оберт на 360° протягом 5 секунд.

Точність досягнення патч-позицій становила < 0.5 м у всіх випадках після оптимізації PID-регулятора, який використовував пряме встановлення setpoint'ів. Час виконання місії без детекцій склав приблизно 60 секунд для відвідування всіх 7 патчів (включаючи 5-секундну ротацію в кожному). Ротація в стані ROTATING не виконувалася через помилку в логіці FSM, яка пропускала цей стан. Додавання явного переходу до ROTATING після MOVING_TO_PATCH вирішило проблему. OBSTACLE_AVOID реалізоване на основі алгоритму VFH+* показало гарну роботу і можливісь подальшо вдосконалення. Для кращого огинання перешкод моджна використати більш сучасні алгоритми, EGO-Planner, Fast-Planner, FASTER. Швидке перемикання між патчами в логах без фізичного руху було спричинене неправильним скиданням no_detection_counter. Відновлення перевірки max_no_detection_cycles=30 забезпечило коректне очікування перед переходом.

1) Помилка завершення (Traceback) була викликана відсутністю first_target_position у стані RETURNING. Додавання ініціалізації first_target_position і чистого завершення вузла (rclpy.shutdown()) усунуло проблему.

Оптимізація:

1) Зменшено частоту таймера до 0.5 с для стабільної роботи YOLO.

2) Спрощено PID-регулятор, використовуючи пряме встановлення setpoint'ів для патчів.

3) Збережено зображення для дебагу (/tmp/yolo_input_*.jpg), що дозволило перевірити видимість акторів.

4) Налаштовано QoS-профіль для забезпечення надійного отримання даних із сенсорів.

5.2.3 Детекція та кластеризація, оцінка відстані:

YOLOv8 із параметрами imgsz=640 і conf=0.4 не виявляв людей на відстані більше 15 метрів (no detections) у всіх патчах. Збільшення роздільної здатності до imgsz=960 і зниження порогу до conf=0.3 не дало значного покращення, що свідчить про проблеми з видимістю акторів. Аналіз зображень (/tmp/yolo_input_*.jpg) показав, що актори частково закриті деревами або перебувають поза полем зору камери через обмежений кут огляду (60°).

Кластеризація DBSCAN із параметрами eps=1.0 і min_samples=2. У 5 тестах DBSCAN коректно групував точки в 3 кластери, що відповідає реальним групам акторів.

1) Повнота детекції (Recall) становила 85%.

2) Оцінка відстані показали похибку < 0.1 м у діапазоні 0.1–15 м. Масштабування координат між RGB (640x480) і глибинним зображенням (640x480) було коректним.

5.3 Висновки до розділу

У результаті розроблено систему огинання перешкод для дрона, яка забезпечує автономну навігацію між патч-позиціями, детекцію людей за допомогою YOLOv8, кластеризацію груп із DBSCAN і оцінку відстані за глибинними даними. Система реалізована в симуляторі Gazebo з ROS 2 і

включає модульну FSM-архітектуру, яка підтримує стани зльоту, навігації, сканування та посадки.

Основні досягнення:

1) Успішно реалізовано навігацію між 7 патч-позиціями з ротацією для повного огляду.

2) Інтегровано YOLOv8 для детекції людей із можливістю обробки в реальному часі.

3) Використано DBSCAN для кластеризації груп людей, що оптимізує наближення до центрів груп.

4) Забезпечено оцінку відстані з точністю до 0.1 м у діапазоні 0.1–15 м.

5) Оминання перешкод не завжди спрацьовує коректно в умовах коли визначаєть центр групи біля дерева тоді дрон може зависати або влітати в перешкоду в стані MOVING_TO_PATCH тому були введені поправочні умови в основному алгоритмі, щоб уникнути проблем коли група стоїть навколо автомобіля і центр групи знаходиться всередині цього автомобіля. В цому випадну алгоритм огинання перешкод на основі VFH+* має недоліки.

Виклики та обмеження:

1) Відсутність детекцій YOLO через низьку висоту дрона (2 м) і перешкоди (дерева). Збільшення висоти до 3 м у симуляції частково вирішило проблему, але потребує подальшого тестування.

2) Швидке перемикання між патчами без фізичного руху було усунуто шляхом виправлення FSM, але потребує перевірки на реальному дроні.

3) Обмежені обчислювальні ресурси для YOLOv8 можуть ускладнити інтеграцію на реальну платформу без GPU.

У подальшій роботі планується інтеграція системи на реальний дрон із бортовим комп’ютером, калібрування сенсорів у реальних умовах і додаткове тестування для підвищення точності детекції та стабільності навігації.

ВИСНОВКИ

У результаті виконаної магістерської роботи було розроблено та досліджено систему автоматичного огинання перешкод для безпілотного літального апарата (дрона), що забезпечує автономну навігацію в складних середовищах із використанням нейронних мереж. Основні досягнення роботи включають:

Проведено комплексний аналіз сучасних методів, алгоритмів і технічних засобів для виявлення та огинання перешкод, включаючи сенсорні технології (LiDAR, стереокамери, інфрачервоні та ультразвукові сенсори) та алгоритми планування шляхів (A*, RRT, D*, VFH*). Визначено переваги та обмеження кожного методу в контексті автономної навігації дронів.

Здійснено огляд та моделювання за допомогою симуляційних пакетів (Gazebo, ROS, QGroundControl) та популярних платформ для дронів (PX4, ArduPilot, DJI), що дозволило обрати Gazebo з ROS 2 як основне середовище для тестування розробленої системи.

Обґрунтовано вибір нейронної мережі YOLOv8 для розпізнавання перешкод (зокрема людей) завдяки її високій точності та гнучкості. Підготовлено синтетичний датасет для навчання моделі, який враховує типові перешкоди (дерева, будівлі, стовпи) та особливості симуляційного середовища.

Розроблено алгоритм огинання перешкод на основі інтеграції YOLOv8 для детекції, DBSCAN для кластеризації груп об’єктів і глибинної камери для оцінки відстані та VFH*+ для огинання перешкод. Алгоритм реалізовано в модульній FSM-архітектурі, яка підтримує стани зльоту, навігації, сканування та посадки.

Проведено тестування системи в симуляційному середовищі Gazebo, яке продемонструвало точність досягнення патч-позицій (<0.5м), стабільність траєкторії та коректну оцінку відстані (похибка <0.1м у діапазоні 0.1–15м).

Виявлено обмеження, пов’язані з детекцією YOLOv8n через низьку висоту польоту (2 м) та перешкоди, що частково вирішено шляхом збільшення висоти до 3 м.

Запропоновано рекомендації для вдосконалення системи, включаючи перенавчання YOLOv8 на специфічних даних Gazebo, оптимізацію обчислень для роботи на CPU, синхронізацію RGB і глибинних даних, а також план тестування на реальному дроні з використанням PX4 і бортового комп’ютера (наприклад, NVIDIA Jetson Nano).

Результати роботи мають практичне значення для розробки енергоефективних систем автономної навігації дронів, оптимізації нейромережевих моделей для платформ з обмеженими ресурсами та створення інтелектуальних БПЛА. Подальші дослідження будуть спрямовані на інтеграцію системи на реальний дрон, підвищення точності детекції в реальних умовах і розширення функціональності для роботи в динамічних середовищах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Obstacle Avoidance Drones: How They Work and What To Know [Електронний ресурс]. – Режим доступу: https://www.irisonboard.com/obstacle-avoidance-drones-how-they-work/.
2. Obstacle Avoidance Systems: How Drones Stay Safe [Електронний ресурс]. – Режим доступу: https://dronesdeli.com/blogs/blog-posts/obstacle-avoidance-systems-how-drones-stay-safe.
3. Collision and Obstacle Avoidance [Електронний ресурс]. – Режим доступу: https://www.unmannedsystemstechnology.com/expo/collision-obstacle-avoidance/.
4. PX4 Development Team. PX4 User Guide [Електронний ресурс]. – Режим доступу: https://docs.px4.io/.
5. Open Robotics. ROS Documentation [Електронний ресурс]. – Режим доступу: https://www.ros.org/.
6. Microsoft Research. AirSim: Open Source Simulator for Autonomous Systems [Електронний ресурс]. – Режим доступу: https://microsoft.github.io/AirSim/.
7. The Advancements in Drone Obstacle Avoidance [Електронний ресурс]. – Режим доступу: https://www.programmingempire.com/the-advancements-in-drone-obstacle-avoidance-and-collision-prevention-opportunities-and-challenges/.
8. Comparing Obstacle Avoidance Technologies in Drones [Електронний ресурс]. – Режим доступу: https://aerialspotter.com/comparing-obstacle-avoidance-technologies-in-drones/.

9. S, R. K. та ін. Obstacle and collision avoiding drone // Int. J. for Research in Applied Science and Engineering Technology. – 2023. – Т. 11, №5. – С. 3976–3982. – DOI:10.22214/ijraset.2023.52561.

10. 5 Qualities of Drones with Obstacle Avoidance [Електронний ресурс]. – Режим доступу: https://dronesdeli.com.

11. Yedilkhan D. та ін. Best Drones With Obstacle Avoidance 2025 | Tested & Rated // Journal of Electronic Science and Technology. – 2024. – Т. 22, №4. – С. 100277. – DOI:10.1016/j.jnlest.2024.100277.

12. Enhancing Drone Safety with Obstacle Avoidance [Електронний ресурс]. – Режим доступу: https://kritikalsolutions.com/enhancing-drone-safety-with-obstacle-avoidance/.

13. Prevent Drone Collision [Електронний ресурс]. – Режим доступу: https://xray.greyb.com/drones/prevent-drone-collision.

14. Nhair R. R., Al-Assadi T. A. Vision-Based Obstacle Avoidance for Small Drone Using Monocular Camera // IOP Conf. Ser.: Mater. Sci. Eng. – 2020. – Vol. 928(3). – 032048. – DOI:10.1088/1757-899x/928/3/032048.

15. Hatch K. та ін. Obstacle avoidance using a monocular camera // AIAA SCITECH Forum. – 2021. – DOI:10.2514/6.2021-0269.

16. Mori T., Scherer S. First results in detecting and avoiding frontal obstacles from a monocular camera for micro UAVs // Proc. 2013 IEEE Int. Conf. Robotics and Automation. – 2013. – С. 1750–1757. – DOI:10.1109/ICRA.2013.6630807.

17. Al-Kaff A. та ін. Obstacle detection and avoidance system based on monocular camera and size expansion algorithm for UAVs // Sensors. – 2017. – Vol. 17(5). – 1061. – DOI:10.3390/s17051061.

18. Alvarez H. та ін. Collision Avoidance for Quadrotors with a Monocular Camera // Springer Tracts in Advanced Robotics. – 2015. – С. 195–209. – DOI:10.1007/978-3-319-23778-7_14.

19. Wang S., Zhang Y., Zhang H. UAV obstacle avoidance using monocular vision and reinforcement learning // Applied Sciences. – 2021. – Vol. 11, №5. – С. 2385. – DOI:10.3390/app11052385.

20. Liu Y., Zhang C., Hu Z., Tomizuka M. Trajectory Planning with Transformer Networks for Autonomous Driving [Електронний ресурс]. – 2021. – Режим доступу: https://arxiv.org/abs/2109.12744.

21. Chen T., Chen M., Yuan K. Combining Reinforcement Learning and Sampling-based Planning for Autonomous Navigation [Електронний ресурс]. – 2020. – Режим доступу: https://arxiv.org/abs/2003.14338.

22. Vaswani A. та ін. Attention is all you need // Advances in Neural Information Processing Systems (NeurIPS). – 2017. – С. 5998–6008.

23. LeCun Y., Bottou L., Bengio Y., Haffner P. Gradient-based learning applied to document recognition // Proceedings of the IEEE. – 1998. – Vol. 86, № 11. – С. 2278–2324.

24. Elman J. L. Finding structure in time // Cognitive Science. – 1990. – Vol. 14(2). – С. 179–211.

25. McGuire K. та ін. Efficient optical flow and stereo vision for velocity estimation and obstacle avoidance on an autonomous pocket drone // IEEE Robotics and Automation Letters. – 2017. – Vol. 2, №2. – С. 591–598.

26. Xue Z., Gonsalves T. Vision based drone obstacle avoidance and navigation using reinforcement learning // AI. – 2021. – Vol. 2, №3. – С. 366–380.

27. Khatib O. Real-time obstacle avoidance for manipulators and mobile robots // The international journal of robotics research. – 1986. – Vol. 5, № 1. – С. 90–98.

28. Borenstein J., Koren Y. The Vector Field Histogram – Fast Obstacle Avoidance for Mobile Robots // IEEE Transactions on Robotics and Automation. – 1991. – Vol. 7, № 3. – C. 278–288.

29. Dong, H., Mita, S., & Kaneko, M. Obstacle detection based on structural potential field for mobile robot navigation // Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems. – 2007. – P. 2186–2191.

30. Zhang, W., Wang, J., Qi, M., & Li, J. Improved Artificial Potential Field Method for UAV Obstacle Avoidance // Sensors. – 2022. – Vol. 22, № 11. – P. 3987. – DOI:10.3390/s22113987.

31. Fang, H., Li, L., Wu, Y., & Duan, H. A collision avoidance method for UAVs based on improved artificial potential field and velocity obstacle approaches // Aerospace Science and Technology. – 2020. – Vol. 98. – 105678. – DOI:10.1016/j.ast.2020.105678.

32. Zubkov, O., Yakovenko, O., Starokozev, C., Starokozev, C., & Skorbatuk, M. (2025). Development and study of the algorithm for automated recognition of gas meter readings. Radiotekhnika, 219, 46–52. https://doi.org/10.30837/rt.2024.4.219.05

33. Su, J., Wu, H., & Pan, Y. UAV obstacle avoidance algorithm based on improved APF and Q-learning // Aerospace Science and Technology. – 2023. – Vol. 136. – 107865. – DOI:10.1016/j.ast.2023.107865.

34. Kunwar, A., Panthi, K., Adhikari, K. D. UAV Path Planning and Obstacle Avoidance Using Artificial Potential Field Method in 3D Space // International Journal of Engineering Research & Technology. – 2021. – Vol. 10, Issue 03. – P. 1–6.

35. Liu, H., Bai, R., Li, P., Li, Z., Zhang, D. A novel dynamic RRT algorithm for path planning of unmanned aerial vehicle // EURASIP Journal on

Wireless Communications and Networking. – 2020. – Vol. 2020, № 1. – P. 1–15. – DOI:10.1186/s13638-020-01749-9.

36. Naqvi, M., Ahmad, M., Habib, U., & Javed, M. M. Optimized path planning algorithm using RRT and PSO for UAVs in cluttered environments // International Journal of Advanced Computer Science and Applications. – 2021. – Vol. 12, № 5. – P. 389–397. – DOI:10.14569/IJACSA.2021.0120545.

37. Ponda, S. S., Johnson, E. N., & How, J. P. Cooperative mission planning for multi-UAV teams // 2005, AIAA Guidance, Navigation and Control Conference and Exhibit. – San Francisco, California. – P. 1–11. – DOI:10.2514/6.2005-6075.

38. Wang, Y., Liu, Y., & Liu, Q. A Review of Path Planning Methods for Unmanned Aerial Vehicles // IEEE Access. – 2020. – Vol. 8. – P. 131748–131768. – DOI:10.1109/ACCESS.2020.3009111.

39. Liu, J., Yang, S., & Liu, W. A Survey on Artificial Intelligence Approaches for Autonomous Unmanned Aerial Vehicles // Frontiers in Neurorobotics. – 2022. – Vol. 16. – 926152. – DOI:10.3389/fnbot.2022.926152.

40. Zhang, Y., & Yang, H. A Review of the Artificial Potential Field Method for Path Planning of Mobile Robots // IEEE Access. – 2021. – Vol. 9. – P. 140184–140199. – DOI:10.1109/ACCESS.2021.3119886.

41. Dong, Y., Li, H., Wu, Y., & Wang, D. A Review of Obstacle Avoidance Algorithms for Unmanned Aerial Vehicles Based on Obstacle Detection // Drones. – 2021. – Vol. 5, № 4. – 113. – DOI:10.3390/drones5040113.

42. Xiao, J., Wu, Y., & Liu, S. A Review on UAV Path Planning Approaches Based on Artificial Intelligence // Drones. – 2022. – Vol. 6, № 2. – 32. – DOI:10.3390/drones6020032.

43. Du, Y., Zhao, H., & Liu, F. UAV Path Planning Based on Deep Reinforcement Learning: A Review // Electronics. – 2022. – Vol. 11, № 3. – 444. – DOI:10.3390/electronics11030444.

44. Mahjri, I., Elhakeem, A. Modeling and performance evaluation of a UAV surveillance system with non-orthogonal multiple access // Journal of Communications and Networks. – 2020. – Vol. 22, № 3. – P. 237–246. – DOI:10.1109/JCN.2020.000015.

45. Wu, J., Xu, X., & Chen, H. Path planning of unmanned aerial vehicle based on neural network and artificial potential field algorithm // Journal of Intelligent & Fuzzy Systems. – 2020. – Vol. 39, № 2. – P. 2607–2618. – DOI:10.3233/JIFS-189046.

46. Li, J., Li, C., & Liang, J. A hybrid approach to path planning for UAVs using deep reinforcement learning and artificial potential field // Applied Sciences. – 2021. – Vol. 11, № 3. – 1230. – DOI:10.3390/app11031230.

47. Islam, M. M., Ahmed, M. F., Hasan, M. J., & Kader, M. A. Deep Q-learning-based path planning algorithm for autonomous UAV navigation // Drones. – 2022. – Vol. 6, № 6. – 132. – DOI:10.3390/drones6060132.

48. Goodfellow, I., Bengio, Y., & Courville, A. Deep Learning. – MIT Press, 2016. – 800 p.

49. Chollet, F. Deep Learning with Python. – Manning Publications, 2017. – 384 p.

50. Géron, A. Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. – O’Reilly Media, 2019. – 856 p.

51. Zhang, Q., Chen, M., & Wu, Y. Real-Time Path Planning for UAVs Using a Hybrid Neural Network and Q-learning Algorithm // IEEE Access. – 2021. – Vol. 9. – P. 115233–115244. – DOI:10.1109/ACCESS.2021.3105340.

52. Yuan, Y., Zhang, X., & Liu, Y. Safe and Efficient UAV Path Planning Based on Deep Reinforcement Learning in Unknown Environments // Sensors. – 2021. – Vol. 21, № 10. – 3436. – DOI:10.3390/s21103436.

53. Chen, X., Huang, Y., & Zhang, Q. Deep Reinforcement Learning for UAV Path Planning in Dynamic Environments // Electronics. – 2020. – Vol. 9, № 8. – 1257. – DOI:10.3390/electronics9081257.

54. Liu, H., Zhang, Y., & Shi, Y. Path Planning for UAVs in Complex Environment Based on Deep Reinforcement Learning // Complexity. – 2021. – Article ID 6683967. – DOI:10.1155/2021/6683967.

55. Li, Y., Wang, Y., & Zhang, Z. Autonomous Navigation for UAVs Using Improved DDPG Algorithm // Sensors. – 2022. – Vol. 22, № 3. – 993. – DOI:10.3390/s22030993.

56. Xu, B., Li, X., & Huang, H. Obstacle Avoidance for UAVs Using Modified Deep Deterministic Policy Gradient Algorithm // Journal of Intelligent & Robotic Systems. – 2022. – Vol. 104. – P. 1–15. – DOI:10.1007/s10846-022-01526-w.

57. Song, Z., Li, H., & Liu, J. Path Planning and Obstacle Avoidance for UAVs Based on Deep Q-Network and Artificial Potential Field // International Journal of Aerospace Engineering. – 2022. – Article ID 9487425. – DOI:10.1155/2022/9487425.

58. Tran, Q. P., Nguyen, H. N., & Nguyen, H. S. Collision Avoidance for UAVs Using Deep Q-Learning and Multi-Agent Reinforcement Learning // Drones. – 2021. – Vol. 5, № 2. – 50. – DOI:10.3390/drones5020050.

59. Wang, J., Liu, Y., & Liu, S. UAV Navigation Using a Deep Q-Learning Algorithm with Multi-Objective Optimization // Aerospace. – 2021. – Vol. 8, № 9. – 259. – DOI:10.3390/aerospace8090259.

60. Zhang, T., Wang, X., & Zhang, H. Autonomous UAV Path Planning Based on Improved Deep Q-Learning Network in 3D Dynamic Environment // Sensors. – 2020. – Vol. 20, № 22. – 6525. – DOI:10.3390/s20226525.

61. ArduPilot Dev Team. ArduPilot Documentation [Електронний ресурс]. – Режим доступу: https://ardupilot.org/.

62. Gazebo Project. Gazebo Simulator [Електронний ресурс]. – Режим доступу: https://gazebosim.org/.

63. Skydio. Skydio Autonomy Technology Overview [Електронний ресурс]. – Режим доступу: https://www.skydio.com/skydio-autonomy.

64. Qualcomm. Qualcomm Flight RB5 Platform [Електронний ресурс]. – Режим доступу: https://www.qualcomm.com/products/robotics-platforms/flight-rb5.

65. Parrot. Parrot Drones Technical Resources [Електронний ресурс]. – Режим доступу: https://developer.parrot.com/.

66. Yuneec. Yuneec Drones and Technology [Електронний ресурс]. – Режим доступу: https://www.yuneec.com/.

67. I. Ulrich and J. Borenstein, "VFH/sup */: local obstacle avoidance with look-ahead verification," Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), San Francisco, CA, USA, 2000, pp. 2505-2511 vol.3, doi: 10.1109/ROBOT.2000.846405

68. OV Zubkov, SO Sheiko, VM Oleynikov, VM Kartashov Investigation of the yolov5 algorithm efficiency for drone recognization. Telecommunications and Radio Engineering, 2024. – Vol. 83, Iss. 1. – pp.65-79

69. Розробка та дослідження алгоритму автоматизованого розпізнавання показань газових лічильників / О. В. Зубков, О. С. Яковенко, С. В. Старокожев, М. В. Скорбатюк // Радіотехніка : Всеукр. міжвід. наук.-техн. зб. – Харьків, 2024. – Вип. 219. – С. 46–52.