МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ім. Ю.М. ПОТЕБНІ

Кафедра електричної інженерії та кіберфізичних систем

Кваліфікаційна робота

другого (магістерського) рівня освіти
(рівень вищої освіти)

на тему Проміжний модуль автоматизації функцій керування безпілотним літальним апаратом

Виконав: здобувач 2 курсу, групи 8.1744 _____
спеціальності 174 «Автоматизація, комп'ютерно-інтегровані технології та робототехніка_____

освітньої програми Автоматизація та_____
комп'ютерно-інтегровані технології_____

_____Вакула Ю. О._____
я
Керівник доц., к.т.н. Міняйло Н.О._____
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
Рецензент директор ТОВ «Електротехмаш»____
_____Курілець І. А._____

Запоріжжя 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ім. Ю. М. ПОТЕБНІ

Кафедра електричної інженерії та кіберфізичних систем
Рівень вищої освіти другий (магістерський)
(другий (магістерський) рівень)
Спеціальність 174 «Автоматизація, комп'ютерно- інтегровані технології та робототехніка
(код та назва)
Освітня програма Автоматизація та комп'ютерно-інтегровані технології
(назва)
Спеціалізація
(код та назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри Коваленко В.Л.
"____" _____ 2025 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧЕВІ
Вакулі Юрію Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Проміжний модуль автоматизації функцій керування безпілотним літальним апаратом
Керівник роботи Міняйло Наталія Олександрівна, канд.техн.наук, доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом ЗНУ від від «19» вересня 2025 року № 1304-с
2. Строк подання студентом роботи 01 грудня 2025 р.
3. Вихідні дані до роботи: технічна документація, інструкції, дані, отримані під час проходження виробничої переддипломної практики, ДСТУ
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Аналіз технологій та засобів управління БПЛА; актуальність та переваги модуля автоматизації польотних функцій; математичне моделювання автоматичних режимів; вибір апаратної та сенсорної бази системи; опис алгоритму автоматизації зльоту, зависання та посадки; практична реалізація модуля та випробування.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Загальна характеристика роботи; аналіз технологій та засобів управління БПЛА; актуальність та переваги модуля автоматизації польотних функцій; математичне моделювання автоматичних режимів; опис алгоритму автоматизації зльоту, зависання та посадки; структурна схема автоматичної посадки; вибір апаратної та сенсорної бази системи; практична реалізація модуля

та випробування; 3D - моделювання та CAM технології; результати прототипування; програмна реалізація та тестування; налагодження та вибір закону регулювання; результати польових випробувань; висновоки.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Міняйло Н.О. , доцент	22.09.2025	03.10.2025
2	Міняйло Н.О. , доцент	06.10.2025	17.10.2025
3	Міняйло Н.О. , доцент	20.10.2025	31.10.2025
4	Міняйло Н.О. , доцент	03.11.2025	15.11.2025
5	Міняйло Н.О. , доцент	15.11.2025	20.11.2025
6	Міняйло Н.О. , доцент	17.11.2025	25.11.2025

7. Дата видачі завдання 22 вересня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Дослідження існуючих польотних контролерів	25.09.2025	
2	Досліджена кофігураторів	06.10.2025	
3	Дослідження етапів польотів	10.10.2025	
4	Визначення функціоналу для автоматизації	14.10.2025	
5	Початковий вибір апаратної бази для реалізації рішення	15.10.2025	
6	Випробування елементів	17.10.2025	
7	Розробка алгоритмів та коду	28.10.2025	
8	Тестування випробування та коригування рішень	14.11.2025	
9	Оформлення роботи	25.11.2025	
10	Створення презентації та підготовка доповіді	28.11.2025	

Студент _____ Ю.О. Вакула
(підпис) (ініціали та прізвище)

Керівник роботи _____ Н.О. Міняйло
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ С.В. Башлій

АНОТАЦІЯ

Вакула Ю. О. Проміжний модуль автоматизації функцій керування безпілотним літальним апаратом.

Кваліфікаційна робота магістра на здобуття вищої освіти за спеціальністю 174 «Автоматизація, комп'ютерно-інтегровані технології та робототехніка», науковий керівник Н. О. Міняйло, Запорізький національний університет, Інженерний науково-навчальний інститут ім. Ю.М. Потебні, кафедра електричної інженерії та кіберфізичних систем. Запоріжжя, 2025.

У кваліфікаційній роботі магістра виконано дослідження систем налаштування польотних контролерів БПЛА, визначені основні недоліки конфігураторів для FPV-систем, запропоновано проміжний модуль автоматизації функцій керування БПЛА – зльоту, зависання та безпечної посадки на базі мікроконтролера із застосуванням лазерного далекоміра. Модуль реалізовано за архітектурою транзитної ланки у лінії передачі керуючих сигналів між радіоприймачем і польотним контролером, що повністю виключає необхідність модифікації прошивки останнього.

Досягнуто зниження навантаження на оператора та підвищення експлуатаційної витривалості FPV-апарата. Експериментальні випробування підтвердили стійкість системи до інтенсивного зовнішнього освітлення, вібрацій та помірної аерозольної засміченості середовища – умов, критичних для більшості альтернативних сенсорних рішень.

Модуль характеризується високим рівнем модульності, plug-and-play інтеграцією та можливістю відтворення. Він готовий до практичного застосування в задачах логістики, аерофотозйомки, моніторингу територій, навчання пілотів та інших сценаріях з багаторазовими циклами зльоту-посадки.

Ключові слова: FPV-дрон, автоматизація посадки, автоматичний зліт, режим зависання, ПІД-регулятор, модульність, лазерний далекомір, аерофотозйомка, моніторинг територій, навчання пілотів FPV

ABSTRACT

Vakula Y. O. Intermediary Automation Module for UAVs Control Functions.

Master's thesis for the degree of Master in Specialty 174 "Automation, Computer-Integrated Technologies and Robotics", scientific supervisor N. O. Miniailo, Zaporizhzhia National University, Engineering Educational and Scientific Institute named after Y. M. Potebnia, Department of Electrical Engineering and Cyber-Physical Systems. Zaporizhzhia, 2025.

The master's thesis investigates flight controller configuration systems for UAV, identifies the main shortcomings of existing FPV configurators, and proposes an intermediary automation module for UAV control functions – takeoff, hovering and safe landing. The system is built around a microcontroller and a ToF rangefinder. The developed module operates as a pass-through link in the control signal line between the radio receiver and the flight controller, completely eliminating the need to modify the flight controller firmware.

The research has achieved a significant reduction in operator workload and increased operational durability of FPV drones. Experimental tests confirmed the system's robustness to intense ambient light, vibration and moderate aerosol contamination – conditions critical for most alternative ranging solutions.

The module is characterized by a high level of modularity, plug-and-play integration and reproducibility. It is ready for practical application in logistics missions, aerial mapping, territory monitoring, pilot training and other scenarios with multiple takeoff-landing cycles.

Keywords: FPV drone, automated landing, automated takeoff, hover mode, PID controller, modularity, ToF rangefinder, aerial mapping, safe landing, territory monitoring, FPV pilot training.

ЗМІСТ

ВСТУП……8

1. АНАЛІЗ ТЕХНОЛОГІЙ ТА ЗАСОБІВ УПРАВЛІННЯ БПЛА ……..12
   1.1. Класифікація та особливості систем управління FPV-дронами……..12
   1.2. Огляд комерційних польотних контролерів ……13
   1.3. Огляд програмних конфігураторів ……19
   1.4. Огляд радіоприймачів……..28
   1.5.Недоліки сучасних систем управління FPV-дронами ……..29
      1.5.1. Недоліки та переваги ручного керування……...30
      1.5.2. Відсутність автоматизації зльоту, зависання та посадки……..31
      1.5.3. Інтегрованість автоматизації та залежність від прошивки польотних контролерів ……31
      1.5.4. Проблеми інтеграції додаткових сенсорів у комерційних рішеннях ……32
2. АКТУАЛЬНІСТЬ ТА ПЕРЕВАГИ МОДУЛЯ АВТОМАТИЗАЦІЇ ПОЛЬОТНИХ ФУНКЦІЙ……...35
   2.1.Зниження навантаження на оператора через автоматизацію ключових етапів польоту……37
   2.2.Підвищення безпеки та збереження цілісності FPV-дронів ……37
   2.3.Модульність і сумісність через протокол Crossfire/CRSF…….38
   2.4.Зручність для навчання операторів……40
   2.5.Кастомні рішення та їх інтеграція з модулем автоматизації……40
3. МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ АВТОМАТИЧНИХ РЕЖИМІВ…42
   3.1.Розробка симулятора для моделювання польотних режимів ……..43
   3.2. Математична модель фізичних процесів……...45
   3.3.Математична модель сумарної тяги двигунів……47
   3.4.Математична модель ПІД-регулювання для стабілізації автоматичних режимів……52
   3.5.Розробка дешифратора логів для аналізу польотних даних……57

4. ВИБІР АПАРАТНОЇ ТА СЕНСОРНОЇ БАЗИ СИСТЕМИ…….64
4.1.Вибір апаратної платформи …….64
4.2.Визначення та аналіз контрольованих параметрів для автоматизації польоту……...68
5. ОПИС АЛГОРИТМУ АВТОМАТИЗАЦІЇ ЗЛЬОТУ, ЗАВИСАННЯ ТА ПОСАДКИ……...71
5.1.Алгоритм роботи системи автоматизації……73
5.2.Режими роботи: транзит або підміна сигналів керування…….74
5.3.Алгоритм розгону двигунів, зльоту та досягнення заданої висоти….75
5.4.Алгоритм посадки та уповільнення двигунів……75
5.5. Обробка даних датчика відстані та протоколу Crossfire/CRSF для управління……..76
6. ПРАКТИЧНА РЕАЛІЗАЦІЯ МОДУЛЯ ТА ВИПРОБУВАННЯ ……...78
6.1.Опис апаратної частини модуля автоматизації функцій……...78
6.2.Електрична схема підключення компонентів системи……..82
6.3.Програмна реалізація: код для ESP32…….85
6.4.Планування та результати польових випробувань ……...91
6.5.Економічна доцільність та гнучкість для кастомних збірок…….94
ВИСНОВКИ ……100
ПЕРЕЛІК ПОСИЛАНЬ……...102
ДОДАТКИ……104

ВСТУП

Актуальність теми. У сучасному світі безпілотні літальні апарати (БПЛА), зокрема FPV-дрони, стали невід'ємною частиною таких сфер, як логістика, моніторинг, відеозйомка та навчання операторів. Завдяки високій маневреності та здатності працювати на значних відстанях, FPV-дрони забезпечують виконання завдань, з оперативного збору інформації чи доставки вантажів у важкодоступні місця. Ці апарати цінуються за простоту конструкції, що дозволяє адаптувати їх до різноманітних потреб шляхом інтеграції нових сенсорів і систем. Проте швидкий розвиток технологій БПЛА супроводжується викликами. Серед них – високе психофізіологічне навантаження на оператора, особливо під час складних маневрів поблизу землі, де ризик аварій зростає через близькість до поверхні та непередбачувані фактори, такі як нерівності рельєфу чи перешкоди. Навіть досвідчені пілоти стикаються з труднощами при виконанні точних маневрів як наприклад, посадка на обмежені майданчики. Водночас на етапах зльоту та посадки оператори нерідко втрачають концентрацію через відчуття початку чи завершення місії, що може призводити до помилок і пошкодження апаратів. Ручне керування, хоча й забезпечує максимальний контроль, вимагає виняткових навичок і цілком доцільним є підвищення безпеки й ефективності у складних умовах. Тенденції розвитку БПЛА вказують на доцільність автоматизації ключових етапів польоту. Сучасні польотні контролери, через конфігуратори, як то Betaflight чи iNav, пропонують часткову стабілізацію, але не вирішують проблему комплексної автоматизації зльоту, зависання чи посадки без глибокого втручання в прошивку. Інтеграція додаткових сенсорів також часто ускладнена через обмеження апаратних ресурсів, що спонукає до пошуку модульних і універсальних рішень. Такі системи могли б не лише знизити навантаження на оператора, а й підвищити безпеку, фізичну збереженість дронів і економічну доцільність їх використання.

Мета дослідження: розробка модульного рішення на базі мікроконтролера і датчика ToF для автоматизації зльоту, зависання та посадки FPV-дронів.

Завдання дослідження: для досягнення поставленої мети у кваліфікаційній роботі сформульовано і вирішено такі завдання:

1. Аналіз сучасних технологій управління FPV-дронами та їх обмежень.
2. Створення математичного забезпечення, включаючи симулятор висоти та дешифратор логів.
3. Розробка алгоритмів автоматизації зльоту, зависання та посадки.
4. Вибір оптимальної апаратної платформи та сенсорів для автоматизації польотних функцій.
5. Реалізація та тестування системи

Об'єкт дослідження: система автоматизації польотних функцій FPV-дрона з режимами зльоту, зависання та посадки.

Предмет дослідження: модуль управління на базі мікроконтролера та датчика TOF для взаємодії з FPV-дронами через протокол Crossfire/CRSF.

Методи дослідження: для реалізації проєкту використано методи аналізу існуючих польотних контролерів, синтезу апаратних і програмних рішень, математичного моделювання для створення симулятора та дешифратора логів, 3D-моделювання для створення конструктивів, а також експериментальні методи тестування та методи спостереження й узагальнення результатів.

Наукова новизна отриманих результатів. Розробка математичного забезпечення для імітації польотних режимів, включаючи симулятор висоти та дешифратор телеметричних даних. Впровадження модульного підходу до автоматизації, який забезпечує сумісність із кастомними FPV-дронами через протокол Crossfire. Використання методу підміни керуючих сигналів для інтеграції без зміни прошивки дрона.

Практичне значення. Розроблена система автоматизації знижує навантаження на оператора, мінімізує ризик пошкодження дрона під час

виконання польотних функцій і підвищує ефективність у логістичних місіях, моніторингу та навчанні операторів. Модульність рішення забезпечує його універсальність і можливість інтеграції таких додаткових сенсорів, як барометри, акселерометри, модулі обробки оптичного потоку, виконавчі механізми, тощо, для розширення функціоналу.

Особистий внесок. Теоретичні та практичні аспекти дослідження виконано автором самостійно. Особистий внесок включає аналіз технологій, вибір апаратної платформи та датчиків, розробку алгоритмів і математичного забезпечення, а також проведення польових випробувань й аналіз отриманих результатів. Здобувачем також здійснено наукове обґрунтування теоретично-експериментальних досліджень

Апробація результатів роботи. Основні положення і результати магістерської роботи доповідалися і обговорювалися на: на конференції студентів, аспірантів, докторантів і молодих вчених «Молода наука-2025» 25 травня 2025 року, м. Запоріжжя); Всеукраїнському конкурсі студентських наукових робіт « IoT Challenge 2025: Drone Tech» (25- 26 червня.2025 року, м. Львів)

Публікації:

1. Вакула Ю., Міняйло Н. О. Виготовлення пристроїв для роботизованих систем із застосуванням 3D-технологій. Збірник наукових праць студентів, аспірантів, докторантів і молодих вчених «Молода наука-2025» Запоріжжя: Запорізький національний університет, 2025. Т. 5 С. 166-167.

2. Міняйло Н.О., Вакула Ю.О., Ендік Т.В. Модуль автоматичної посадки FPV-дрона. Інженерні інновації та розбудова національної економіки : матеріали II Міжнародної науково-практичної конференції Інженерного навчально-наукового інституту ім. Ю. М. Потебні (15–16 травня 2025 року, м. Запоріжжя) / наук. ред. Н. Г. Метеленко; Інженерний навчально-науковий інститут ім. Ю. М. Потебні Запорізького національного університету. Одеса: Видавничий дім «Гельветика», 2025. С. 930-933

Структура та обсяг магістерської роботи. Кваліфікаційна робота магістра складається з вступу, шести розділів, висновків, переліку посилань, викладена на 124 сторінках машинописного тексту, включаючи 35 рисунків, 8 таблиць та 3 додатків.

1 АНАЛІЗ ТЕХНОЛОГІЙ ТА ЗАСОБІВ УПРАВЛІННЯ FPV-ДРОНАМИ

Сучасні системи управління FPV-дронами базуються на комбінації апаратних та програмних технологій, що дозволяють обчислювати в реальному часі усі параметри польоту. Для комунікацій систем застосовуються ключові протоколи. CRSF для телеметрії між приймачем та польотним контроллером, DShot для регулювання моторів пропеллерів, MAVLink для реалізації автономних рішень. Автоматизація реалізується ПІД-регуляторами, фільтрами Калмана та кастомними системами прийняття рішень на базі ШІ.

1.1. Класифікація та особливості систем управління FPV-дронами

FPV-дрони (First Person View) – це безпілотні літальні апарати з передачею відео в реальному часі на окуляри або монітор оператора, що забезпечує ефект "пілотування з кабіни". Системи управління FPV-дронами при класифікації за рівнем автоматизації можна розподілити на: ручне, напівавтоматичне та автономне.

Ручне керування, з акцентом на акробатичний режим забезпечує повний контроль оператора через пульт; стабілізація проводиться тільки гіроскопом. Таким чином досягається висока маневреність, та застосовується мінімальна автоматизація, що вимагає добре напрацьованих навичок оператора, та дозволяє виконувати надскладні маневри з усією повнотою перспективного уявлення, що надає людська свідомість, звісно, за наявності досвіду оператора.

Напівавтоматичне включає додаткові режими котрі забезпечує польотний контролер. Вони впроваджують автоматичне вирівнювання горизонту; утримання висоти або кута нахилу до горизонту, обмеження впливу ротаційної складової задля запобігання занадто агресивному

керуванню, компенсацію дії гравітації при маневрах. Це знижує навантаження на оператората та широко застосовується для фрістайлу.

Автономне керування що зазвичай реалізовується з підтримкою систем GPS – це режим, в якому дрон виконує завдання без постійного втручання оператора, використовуючи попередньо запрограмовані функції.

Ключові компоненти системи. Польотний контролер: забезпечує обробку даних з IMU, барометра, GPS та ПІД-регуляцію. Приймач: приймає команди від передатчика пульта керування. ESC та мотори: система керування гвинтомоторними парами що керується польотним контролером за протоколом DShot для забезпечення швидкого відгуку виконавчих механізмів. Сенсори автоматизації: не є стандартними рішеннями для FPV систем та можуть застосовуватися опціонально як елементи кастомізації.

1.2. Огляд комерційних польотних контролерів

В розділі, як елементна база, розглянуті польотні контролери масового виробництва що охоплюють сегменти від бюджетного до професійного, з фокусом на мультикоптери, та програмні конфігуратори, як платформи, що забезпечують налаштування регуляторів та режимів роботи для польотних контролерів.

Speedy Bee F405 є одним з найпопулярніших польотних контролерів (рис. 1.1) у спільноті FPV-дронів, особливо для 5-10 дюймових квадрокоптерів [1]. Він базується на мікроконтролері STM32F405 (168 МГц, ARM Cortex-M4 з FPU), що забезпечує високу обчислювальну потужність для обробки даних з сенсорів у реальному часі [2]. Контролер оптимізований для прошивок конфігураторами Betaflight [3] та iNav [4], з акцентом на стабільність, простоту налаштування та інтеграцію з периферією. Виробник – SpeedyBee Technology Co., Ltd. (Китай), орієнтований на бюджетний сегмент з високою функціональністю.

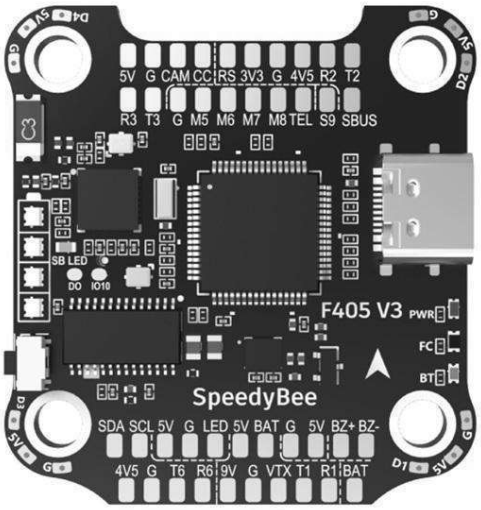


Рисунок 1.1 – Польотний контролер Speedy Bee F405

Широко застосовується у акробатичних польотах та перегонах, що передбачає агресивні фігури пілотажу. Із застосуванням GPS та ToF-сенсорів використовується для дальніх польотів та логістики на фіксованих маршрутах. Конфігурація через iNav [5] дозволяє проводити автономні місії, утримання висоти, доставка вантажів, моніторинг місцевості та для точної посадки. Придатний до завдань з Відеозйомки та кінематографії. Зручний для навчання операторів. На базі такого польотного контролера виконуються експериментальні збірки FPV з кастомними модулями та налаштуваннями.

Цей контролер має такі переваги як – високий ступінь надійності, захист від перегріву, висока якість, розширена кількість терміналів UART – 6 для зовнішніх пристроїв, підтримує підключення до 8 моторів, придатний для польотів на дальні відстані. Надійність програмного забезпечення, досягнена завдяки популярності та великій спільноті ентузиастів. Детальні характеристики представлені в таблиці 1.1.

Його недоліки включають обмежену підтримку ArduPilot [11], чутливість барометра до завад, та при тривалому зависанні більше 5 хв можливий перегрів плати ESC.

Таблиця 1.1 – Структура польотного контролера Speedy Bee F405

Мікроконтролер	STM32F405RGT6 (168 МГц, 1 МБ Flash, 192 КБ RAM)
Сенсори	MPU6000 містить гіроскоп та акселерометр
	барометр BMP280
ESC та живлення	PDB з підтримкою 3-6S LiPo (10-26V)
	9V для живлення периферії
	5V та фільтри для чистого живлення VTX з камерою
Інтерфейси та порти	6 UART для підключення GPS, радіоприймача, відеопередавача, тощо
	I2C для зовнішніх сенсорів
	SPI для MPU6000 та SD-карти
	8 моторних виходів
	SBUS/PPM/IBUS/DSM2/DSMX вхід для приймача
	USB Type-C для налаштування
	Bluetooth модуль
	Wi-Fi для OTA оновлень
	Вбудований Betaflight OSD чіп (AT7456E) для накладання телеметрії на відео
Розміри та монтаж	30.5x30.5 мм, товщина 7 мм; вага ~10 г; монтажні отвори M4
Інші функції	Підтримка SmartAudio та Tramp протоколів для відеопередавача
	RGB вихід для світлодіодної підсвітки
	Вихід для зумера
	Датчик струму до 100A

Цей контролер слугує далі еталоном для порівняння в аналізі інших моделей, оскільки демонструє баланс між вартістю, функціоналом.

JHEMCU GHF405 є одним з найдешевших польотних контролерів для сучасних FPV-дронів (рис. 1.2), орієнтованим на бюджетні збірки для новачків

та кастомних проектів [6]. Є дешевшим за провідні марки, фокусується на мінімалістичних, легких польотних контролерах з акцентом на сумісність з конфігураторами Betaflight та iNav, з мінімізацією елементної бази. Багато користувачів відзначають його як "кращій для початківців". Він побудований на мікроконтролері STM32F405 [2], що забезпечує достатню обчислювальну потужність для реального часу обробки сенсорів, але з меншою кількістю UART-терміналів, ніж у провідних моделей. Перевагу в низькій ціні JHEMCU GHF405 відносно SpeedyBee F405 V4 отримує за рахунок того, що має на 2 UART менше, 4 моторні виходи – проти 8, відсутні Bluetooth та Wi-Fi та вбудований стек ESC та датчик току.

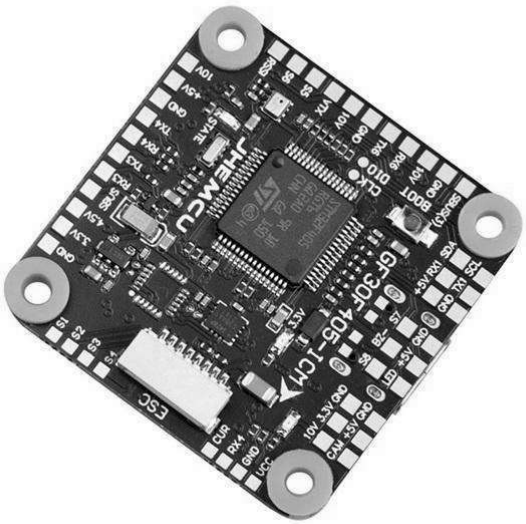


Рисунок. 1.2 – Польотний контролер JHE MCU GHF405

У спільноті це кращий вибір для бюджетних FPV наприклад, у проєктах на 3D-друкованих рамах. Підтримує свою популярніс за рахунок співвідношення ціна-якість, особливо коли, немає потреби бездротового налаштування.

В підсумку маємо такі ключові переваги як економія вартості, простота збирання та налаштувань та легкість при застосуванні на малих моделях.

До недоліків умовно можна віднести обмежену периферію, відсутність бездротового доступу та моніторингу батареї та чутливість до шумів.

Цей польотний контролер слугує прикладом мінімалізму.

Pixhawk 6X [7], як представник серії Pixhawk – це професійний польотний контролер. Є одним з надійніших та універсальних польотних контролерів у екосистемі безпілотних систем. Він кардинально відрізняється від SpeedyBee F405 своєю орієнтацією на автономію та надійність, а не на FPV-фрістайл. Pixhawk 6X має потрійний резерв IMU, два барометри, 16 виходів ШИМ-регулюваня, 2 CAN-шини, Ethernet для комп'ютерів-супутників, типу Raspberry Pi або Jetson, і повну підтримку таких платформ як PX4 та ArduPilot з MAVLink. Pixhawk важчий, з платою розміщення вага досягає до 51 г, більший за розмірами і не використовує протокол CRSF. Він є кращим для логістики та моніторингу з технологією RTK GPS. Pixhawk створений для місій з високою автономією та безпекою, де ціна виправдана додатковим запасом надійності та модульністю.



Рисунок 1.3 – Польотний контролер Pixhawk 6X

Він базується на мікроконтролері STM32H753, що є більш продуктивним ніж STM32F405 з бюджетного сегменту, та забезпечує високу обчислювальну потужність для реального часу обробки даних з чисельних сенсорів та складних алгоритмів.

TykhoLit S4 є компактним та надійним польотним контролером (рис.1.4) від української компанії Tykho Electronics [8,9], орієнтованим на FPV-

дрони для фрістайлу, відеозйомки та логістичних завдань. TykhoLit S4 це український компроміс між SpeedyBee F405 V4 та JHEMCU GHF405 V4. Має усереднену ціну, 5 UART-терміналів. Це більше, ніж JHEMCU, але менше за SpeedyBee. Кількість моторних виходів – 4, та вбудований OSD-чіп, датчик току до 80 А та вбудовані LC-фільтри, чого немає у JHEMCU. На відміну від максимально облегшеного JHEMCU, TykhoLit вже готовий до польоту без додаткового паяння фільтрів і OSD, але не має можливості вбудованого бездротового налаштування, як SpeedyBee. По суті це збалансований компроміс функціоналу і надійності SpeedyBee проти ціни JHEMCU за рахунок локального контролю якості та можливості техпідтримки українською.

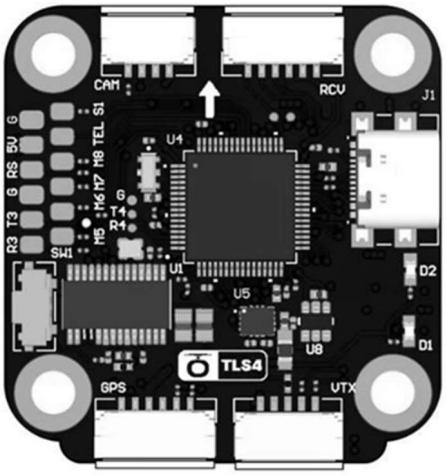


Рисунок 1.4 – Польотний контролер TykhoLit S4

Цей польотний контролер підкреслює переваги локального виробництва.

Matek F405-STD є компактним та універсальним польотним контролером від китайського виробника Matek Systems, спеціально адаптованим для мультикоптерів у FPV-дронах [16]. Ця модель входить до топ-10 польотних контролерів за популярністю на глобальному ринку, з акцентом на баланс між автономією, легкістю та сумісністю з іншими компонентами та програмним забезпеченням (рис. 1.5). Вона популярна серед

ентузіастів фрістайлу та розробників логістичних рішень, де потрібна інтеграція GPS без перевантаження процесора. За відгуками на форумах IntoFPV та Reddit, Matek F405-STD хвалять за повну сумісність з конфігураторами Betaflight та iNav. Контролер базується на мікроконтролері STM32F405 – найбільш поширений варіант для бюджетного сегменту, що забезпечує високу обчислювальну потужність для реального часу обробки сенсорів, з оптимізацією для мультикоптерних конфігурацій від 4 до 8 моторів.

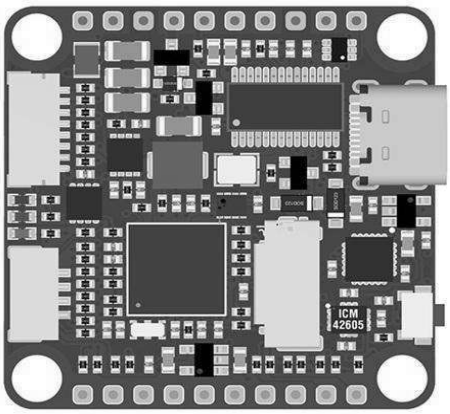


Рисунок 1.5 – Польотний контролер Matek F405-STD

Узагальнена характеристика польотних контролерів, які були проаналізовані у роботі, представлена у порівняльній таблиці додатка А.

1.3. Огляд програмних конфігураторів

Програмні конфігуратори є ключовими інструментами для налаштування польотних контролерів FPV-дронів, дозволяючи користувачам адаптувати поведінку апарату під конкретні завдання, апаратну конфігурацію та стиль пілотування. Вони працюють на основі графічного інтерфейсу, що взаємодіє з прошивкою контролера через USB-з'єднання або бездротові протоколи, наприклад, Bluetooth. Основні принципи роботи включають читання та запис параметрів у флеш-пам'ять контролера, симуляцію поведінки, моніторинг телеметрії в реальному часі та валідацію налаштувань

для запобігання помилкам. Конфігуратори базуються на відкритих або частково відкритих проектах, що забезпечує гнучкість для спільноти FPV-ентузіастів. Нижче розглянуто три основні конфігуратори: Betaflight, iNav та платформу ArduPilot, з акцентом на їх архітектуру, ключові функції та відмінності.

Betaflight [3] оптимізований для акробатичного пілотування FPV-.6дронів саме в ручному режимі (рис.1.6). Він фокусується на швидкості реакції, стабільності в агресивних маневрах та мінімізації затримок. Конфігуратор Betaflight доступний для Windows, macOS та Linux. Він побудований на фреймворку Electron з використанням веб-технологій JavaScript, HTML та CSS, що забезпечує крос-платформенність.

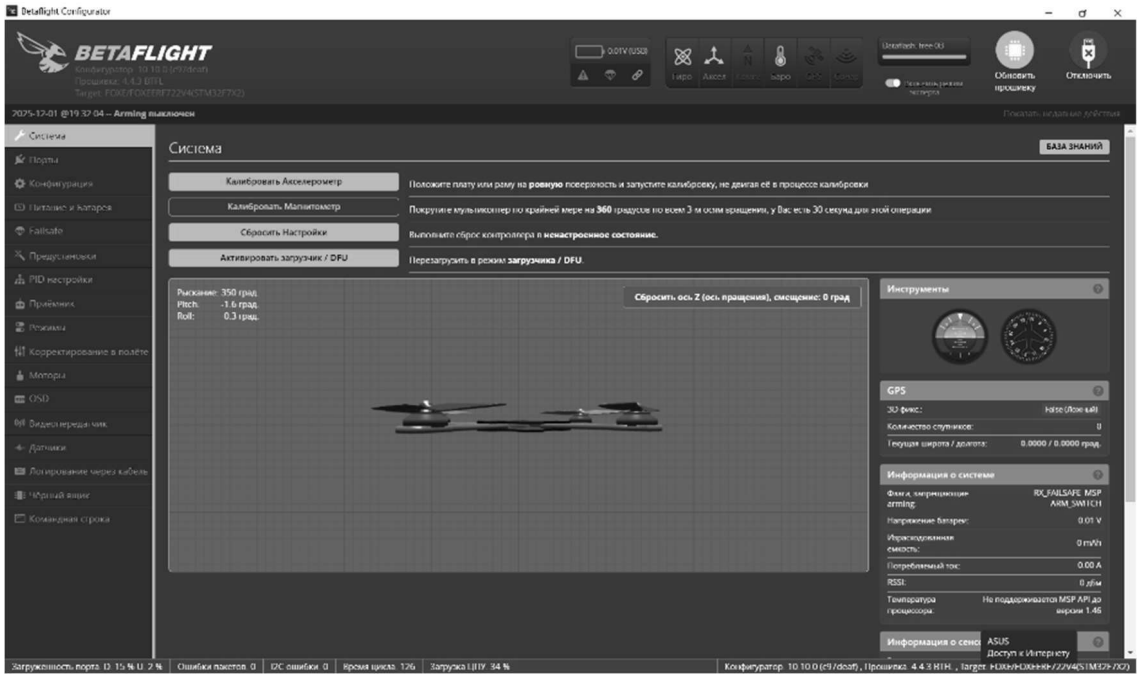


Рисунок 1.6 – Кнфігуратор Betaflight

Використовує протокол MSP (Multiwii Serial Protocol) для обміну даними між контролером та персональним комп’ютером. Після підключення через USB конфігуратор завантажує поточну конфігурацію (dump) у форматі CLI-команд.

Забезпечує налаштування та коригування ПІД-регуляторів. Центральний елемент конфігуратора – ПІД-контролер для стабілізації усіх осей обертання. Користувач налаштовує коефіцієнти P, I, D окремо для кожної осі. При цьому підтримується динамічне регулювання котре включає фільтрацію похідної складової, та компенсаційний режим антігравітації, що дозволяє компенсувати падіння при нахилі вперед та зміні кута й ефективності тяги пропелерів. В окремому додатку доступна симуляція в реальному часі через логи чорної скриньки.

Дозволяє застосовувати режими польоту Horizon та Angle з частковою автоматизацією, на відміну від акробатичного режиму, котрий забезпечує повну свободу маневрів (рис.1.7). Всі ці режими визначаються через налаштування допоміжних каналів, AUX-channels, що їх закріплено за тумблерами та кнопками пультів керування. Конфігуратор дозволяє пов’язувати перемикачі пульта не тільки з режимами та ще й з допоміжними функціями.

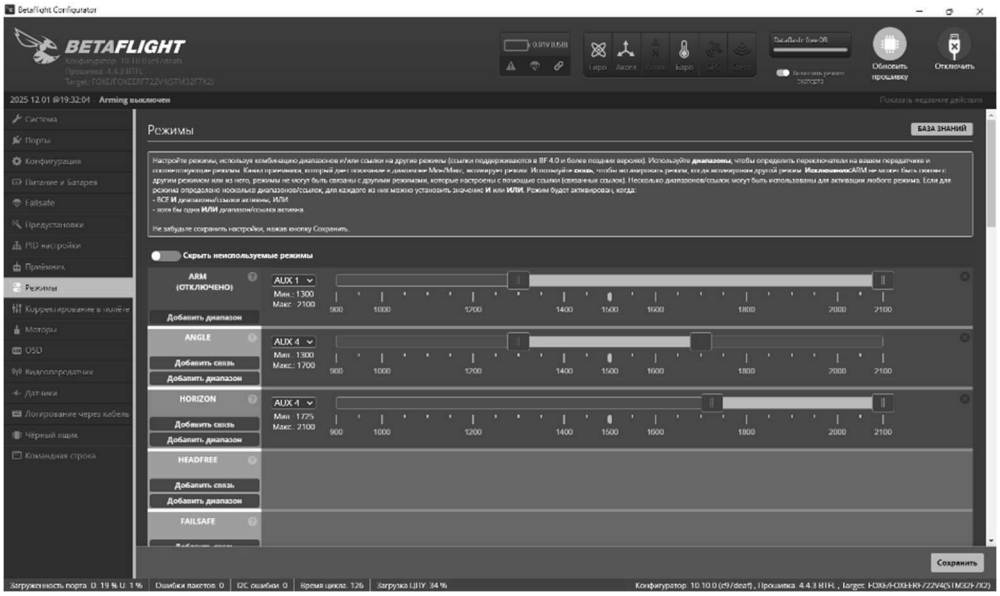


Рисунок 1.7 – Режими Betaflight з частковою автоматизацією

Розділ фільтрів надає інструменти для придушення шуму від моторів. можна легко підібрати набір Notch-фільтрів, та низькочастотних фільтрів LPF

та фільтрів першого порядку. Інструментарій дозволяє очистити гіроскоп від вібрацій і досягти плавної роботи навіть на потужних моторах. Вкладка OSD формує індикацію на екрані. За бажанням можна відобразити рівень сигналу RSSI, напругу батареї, поточну висоту, споживаний струм, таймер польоту та багато іншого. Підтримка протоколів SmartAudio і Tramp дозволяє керувати потужністю та каналом відеопередавача прямо з пульта, не торкаючись самого VTX передавача.

Особливою перевагою є наявність повноцінного інтерфейсу командного рядка CLI. Через нього досвідчений оператор може переналаштовувати будь-який параметр, навіть той, що не відображено графічному інтерфейсі. У пам’яті контролера зберігається до трьох незалежних профілів налаштувань, між якими можна перемикатися перемикачем на пульті прямо в повітрі. Наприклад, один для спокійного круїзу, другий для агресивного фрістайлу, третій для зйомки.

Для аналізу польоту доступні графіки в реальному часі (гіроскоп, помилки PID, команди стіків) та функція Blackbox: усі дані записуються на microSD-карту і потім аналізуються в Betaflight Blackbox Explorer або Gyroflow. Це дозволяє точно виявити джерело вібрацій, нестабільності чи помилок пілотування.

Betaflight ідеальний для швидкісних та фрістайл дронів завдяки низькій латентності робочого циклу ~1-2 kHz. Та при цьому має свої обмеження, як то слабка підтримка автономних функцій.

iNav [4] орієнтований на навігацію та стабільний політ, з акцентом на для позиціонування за даними GPS та барометру (рис.1.8). iNav займає проміжне місце між чисто акробатичним Betaflight та повністю автономним ArduPilot і найчастіше використовується на FPV-дронах середньої та великої дальності.

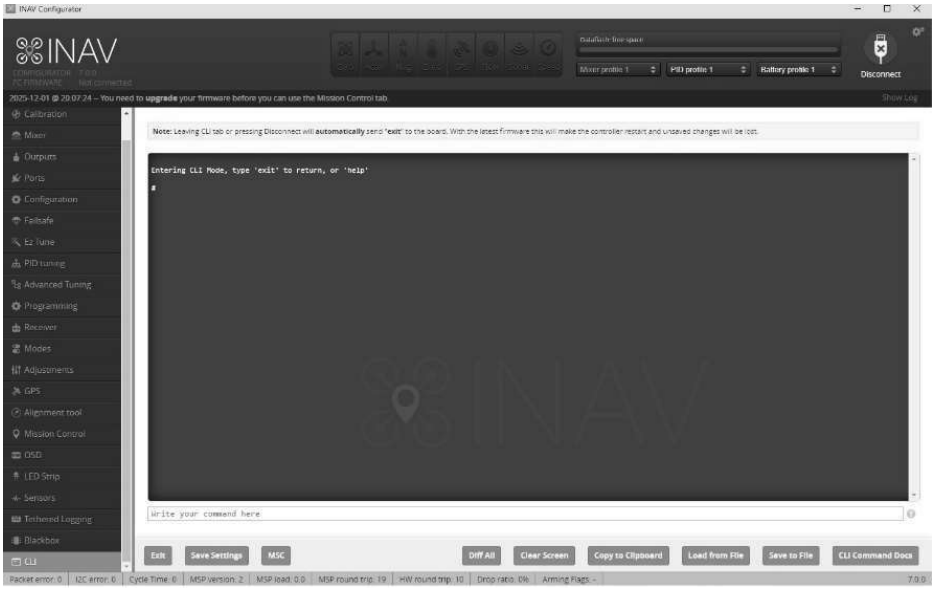


Рисунок 1.8 – Кнфігуратор iNav

Середовище iNav чітко поділяє два принципово різні стилі польоту.

Перший – ручне керування та напівавтоматичні режими. У цих режимах iNav працює як стабілізатор: пілот повністю відповідає за всі етапи польоту, включаючи зліт, маневри, зависання та посадку. Прошивка забезпечує лише стабілізацію кутів і можливість утримання горизонту, але не втручається в керування висотою чи швидкістю зниження. Оцінка відстані до землі, плавне гальмування та м’яка посадка залишаються виключно на операторі. Динамічні фільтри та добре відкалібрований IMU роблять дрон надзвичайно слухняним, але жодної автоматизації саме зльоту й посадки в цих режимах не передбачено.

Другий – повністю автономний режим з плануванням географічного положення. Тут iNav переходить у роль повноцінного автопілота, орієнтованого на геопозицію (рис.1.9). Дрон самостійно утримує координати та висоту за допомогою GPS, барометра й магнітометра. Зліт і посадка виконуються за заданими в місії параметрами, однак точність вертикального позиціонування на останніх метрах залишається обмеженою. Барометр надає дані з доволі великою похибкою ±30 см – 50 см, GPS на низькій висоті неефективний, а жорсткість посадки часто перевищує 0,5 м/с – 0,7 м/с, що

неприйнятно для FPV-конструкцій. У ручних і напівавтоматичних режимах, а саме вони становлять переважаючу більшість типового FPV-польоту, пілот залишається єдиним джерелом команд. Навіть наявність напівавтоматичних функцій iNav не закриває всіх ризиків на перших і останніх метрах польоту. Зліт з маленької площадки, зависання 0,8м – 3 м над нерівною поверхнею, посадка в траву чи пил. Це створює природну нішу для зовнішніх рішень, які могли б взяти на себе локальні, але критично важливі задачі – плавний старт, стабільне зависання на малій висоті до кількох метрів і, головне, безпечну посадку – не зачіпаючи при цьому ні навігацію, ні планування маршруту.

В ручних режимах iNav поводиться максимально «прозоро»: будь-яка зміна значень радіоканалів сприймається як остаточне рішення пілота і виконується миттєво, без додаткових перевірок чи компенсацій з боку прошивки.

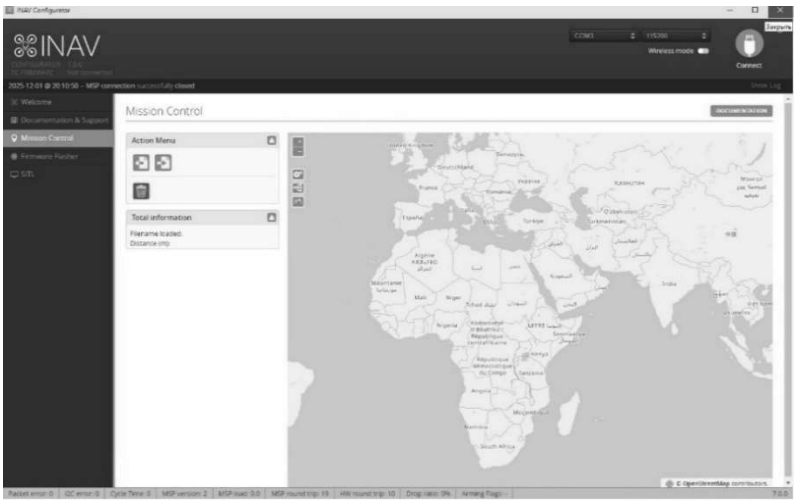


Рисунок 1.9 – Географічне планування iNav

ArduPilot [11] є повноцінною автопілотною платформою (рис.1.10), розрахованою насамперед на автономні місії мультироторів, літаків, роверів та інших транспортних засобів. Вона спирається на потужні польотні контролери (Omnibus F4/F7/H7, Pixhawk-серії, Cube) або навіть моноблочні комп’ютери типу Raspberry Pi та Jetson Nano як компаньйони. Сенсорна база також суттєво відрізняється від типового FPV-дрона: обов’язковий

високоточний GPS/RTK, зовнішні магнітометри, кілька барометрів, оптичний потік, далекоміри LiDAR, іноді навіть стереокамери. Прошивка SpeedyBee F405 чи Matek H743 на бюджетному FPV-дроні теоретично може бути перепрошита під ArduPilot, але без значного апгрейду сенсорів і живлення така збірка втратить більшість своїх переваг і перетвориться на «обрізаний» варіант з суттєво обмеженою функціональністю.

Усі ключові функції – від стабілізації до зльоту, зависання й посадки – реалізовані через протокол MAVLink і розподілені між внутрішніми та зовнішніми циклами керування з використанням розширеного фільтра Калмана. Ручне керування існує, але завжди має нижчий пріоритет порівняно з автономними режимами й розглядається як допоміжний або аварійний інструмент. Основний потік команд проходить не через RC-пакети, а через MAVLink-повідомлення, тому будь-які зовнішні зміни значень RC-каналів у більшості сценаріїв просто ігноруються або компенсуються вищими циклами керування.

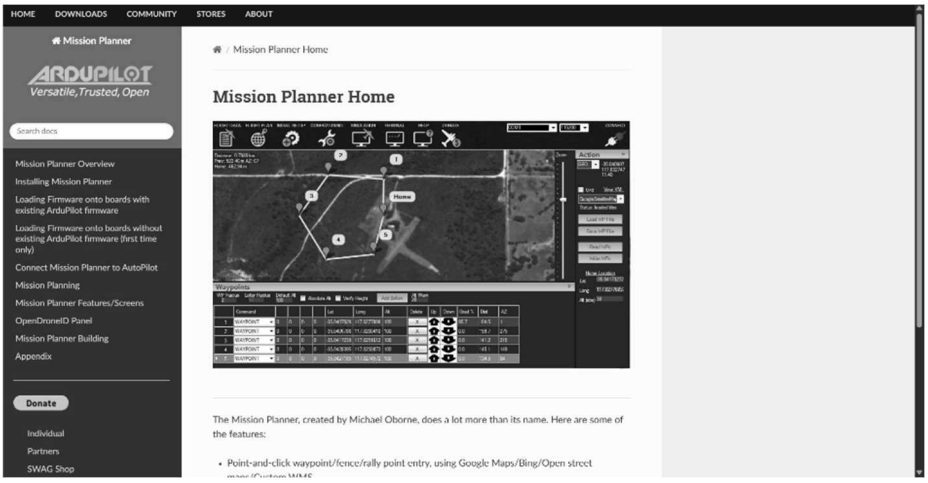


Рисунок 1.10 – Автопілотна платформа ArduPilot

ArduPilot з самого початку проектувався як самодостатня, замкнена екосистема, у якій усі елементи – від сенсорів до виконавчих команд – тісно інтегровані між собою. Це забезпечує надзвичайно високу надійність у професійних задачах, але водночас робить платформу цілком замкненою та

самодостатньою. Будь-яка спроба додати сторонній пристрій вимагає глибокої інтеграції на рівні драйверів, LUA-скриптів або компаньйон-комп'ютера.

Таким чином, у контексті типових FPV-дронів і ручного керування польоту від першої особи ArduPilot залишається спеціалізованою платформою, орієнтованою на глобальну автономію, а не на гнучке доповнення пілота на окремих етапах польоту.

Таблиця 1.2 дозволяє підсумувати, розглянуті комерційні польотні контролери, в комплексі із можливостями, що надають їм різні програмні конфігуратори, демонструють широке розмаїття апаратних можливостей і підходів до організації керування: від мінімалістичних FPV-платформ з пріоритетним керуванням по радіоканалу, до потужних автономних систем із MAVLink-центричною архітектурою. Ці відмінності суворо обумовлюють ступінь участі оператора на всіх етапах польоту від першої особи.

Таблиця 1.2 – Порівняльна таблиця конфігураторів

Критерій	Betaflight	iNav	ArduPilot
1	2	3	4
Основна спеціалізація та типові сценарії використання	FPV-пілотування. короткі динамічні польоти з максимальним контролем пілота	FPV польоти середньої та великої дальності з елементами навігації та частковою автономією	Повністю автономні місії за сценарієм. З можливістю корекції траєкторії, висоти або ракурсу. активація наступної точки місії
Рівень автономності польоту	ручне керування, з елементами автоматичної стабілізації	Ручне, напівавтоматичне, або повна автономія	Переважно повна автономія; ручне втручання лише для коригування

Продовження таблиці 1.2

1	2	3	4
Джерела команд керування та їх пріоритет	Радіосигнал з пульта, за його відсутності політ унеможливлений	радіосигнал з пульта; перемикання або втрата сигналу переводить у автономне керування	MAVLink-команди, компаньйон-комп'ютер, радіоканал від пульта. З пріоритетом відповідно до поточного режим польоту.
Протокол сигналів керування	Crossfire – основний. Допоміжні: SBUS/IBUS/FPort /PPM - застарілі	Crossfire – основний. Допоміжні: SBUS/IBUS/FPort /PPM - застарілі	MAVLink (основний) та CRSF (резервний)
Типові апаратні платформи (контролери)	F4; F7; H7 SpeedyBee, Matek, JHEMCU тощо	F4; F7; H7 з GPS Matek, SpeedyBee, Omnibus F4 Nano)	Pixhawk 6C/6X, Cube Orange/Black, Durandal, тощо. часто з компаньйон-комп'ютером Raspberry Pi, Jetson
Мінімальна необхідна сенсорна база	гіроскоп та акселерометр	Гіроскоп, акселерометр, барометр та GPS	Гіроскоп, акселерометр, барометр, GPS/RTK, магнітометр, далекомір
Кількість та типи доступних режимів польоту	3 основні режими польоту: Acro, Angle, Horizon	11 режимів: Acro, Altitude Hold, Position Hold, Heading Hold, Course Hold,	Понад 25 режимів: Stabilize, Loiter, RTL, Auto, Guided, Takeoff, Land, Survey тощо
складність базового налаштування	Зручна: обмежений набір параметрів.	Ускладнена: калібрування приборів, та налаштування навігації	Висока: тисячі параметрів, «Розширений фільтр Кальмана», тощо

1.4. Огляд радіоприймачів

Сучасні системи радіозв'язку для FPV-дронів забезпечують передачу команд керування та телеметрії між пультом оператора та польотним контролером на доволі суттєві відстані. Основні робочі діапазони – 2.4 ГГц (ISM-діапазон, доступний у всьому світі) та субгігагерцові: 433 МГц, 868/915 МГц залежно від регіону. 868 МГц притаманне для Європи, в той час як США, Азія, Україна використовують 915 МГц.

Діапазон 2.4 ГГц використовується в ExpressLRS (ELRS) 2.4 GHz – найпопулярнішому наразі рішенні для FPV. Переваги: висока частота оновлення пакетів (до 500–1000 Гц), низька затримка, компактні антени, добра стійкість до перешкод на відкритих просторах. Недоліки: сигнал гірше проникає через перешкоди як то дерева, будівлі, тощо. При однаковій потужності, дальність менша, ніж у субгігагерцових системах. ELRS 2.4 GHz ідеальний для фрістайлу, перегонів та динамічних польотів у лінії прямої видимості.

Діапазон 868/915 МГц використовується під умовним маркуванням ELRS 900 MHz, в радіомодулях TBS Crossfire та деяких LoRa-модулях. Нижча частота забезпечує кращу дифракцію і більшу дальність. Прівняно до діапазону 1.2 ГГц має помітні переваги, як то відмінна проникність сигналу, стійкість у складних ландшафтних умовах. При цьому вимагає дещо більші антени, та зменшується максимальна частота пакетів, регіональні обмеження застосовуються терироріально, як до потужності, так і до певних каналів діапазону. З боку виробників, TBS Crossfire пропонує класичне рішення з високою надійністю, ELRS 900 MHz, в свою чергу, надає дешевшу та гнучкішу альтернативу в технічних реалізаціях.

Діапазон 433 МГц цей субгігагерцовий ISM-діапазон (433.05–434.79 МГц) застосовується в ELRS-модулях, які не так поширені, бо на багатьох територіях підпадають під законодавчі обмеження для щастосування у цивільній сфері. Переваги цього діапазону відносно попередньо розглянутих,

це ще більша проникність сигналу через перешкоди, більша дальність та низьке енергоспоживання. З’являються і певні недоліки, такі як більші антени, нижча частота пакетів, суворі регуляторні обмеження на потужність, що робить його менш поширеним для динамічних застосувань. В Україні цей діапазон широко використовується для логістичних місій та моніторингу, де потрібна стабільність на відстані.

Більшість сучасних систем базуються на технології LoRa – розширення спектру модуляції, забезпечуючи таким чином високу чутливість приймача та стійкість до завад. Протокол Crossfire Serial Protocol, спільний для систем радіозв'язку як для виробників спільноти ExpressLRS так і від виробника TBS, передає не лише канали керування, а й телеметрію з високою швидкістю та низькою затримкою [17].

Вибір діапазону системи радіозв’язку залежить від завдань, для яких планується щастосовувати апарат. 2.4 ГГц зазвичай пропонуються для динаміки та компактності, 433 МГц або 868/915 МГц, як більш специфічні резервуються для дальності та проникності в різних сферах та умовах застосування, часто надскладних.

1.5. Недоліки сучасних систем управління FPV-дронами

Розгляд сучасних систем управління FPV-дронами в комплексі апаратної та програмної складових, попри їхню гнучкість і поширеність у різних сферах застосування, дозволяє виділити загальні тенденції, котрі висвітлюють низку суттєвих спільних і специфічних обмежень та недоліків польотних систем вцілому. Ці обмеження проявляються насамперед у відсутності вбудованої автоматизації етапів польоту в критичній близькості до поверхні землі, високій залежності від навичок оператора та суттєвих труднощах інтеграції додаткових сенсорів без глибокого втручання в прошивку чи апаратну частину. Нижче наведено детальний аналіз цих

проблем з урахуванням особливостей легких FPV-конструкцій, які свідомо мінімізують сенсорну базу заради балансу маневреності та автономності.

1.5.1. Недоліки та переваги ручного керування.

Ручне керування залишається основним і часто єдиним режимом роботи більшості FPV-дронів, особливо в кастомних збірках на найпоширеніших польотних контролерах середнього та бюджетного сегменту під прошивками Betaflight та iNav. Саме цей стиль пілотування від першої особи забезпечує оператору повний контроль над апаратом і дає унікальні переваги, яких неможливо досягти в повністю автономних системах.

До них належать: максимальна гнучкість, миттєва реакція на зміну обстановки, відсутність залежності від додаткових сенсорів, що дозволяє зберігати мінімальну вагу, вартість і високу маневреність, а також формування стійких рефлексів у досвідчених пілотів, що дозволяє виконувати складні траєкторії з високою швидкістю та точністю.

Проте в реальних умовах тривалої експлуатації та високого психофізіологічного навантаження недоліки ручного керування висвітлюють його зворотний бік і стають основною причиною аварій та пошкоджень дронів саме поблизу землі.

Після 5–10 хвилин інтенсивного польоту концентрація уваги різко падає, з’являється тунельне бачення через FPV-окуляри, координація погіршується, а стрес від відповідальності за апарат і місію лише посилює напруження. Більшість зафіксованих у логах Betaflight та iNav інцидентів спричинена саме помилками пілотування, а не технічними відмовами. Особливо високий ризик виникає на останніх 2–3 метрах перед посадкою: обмежений кут огляду камери, інерція реакцій після агресивних маневрів, втрата візуальної орієнтації на тлі трави, пилу чи нерівностей роблять цей етап найнебезпечнішим у всьому польоті. Навіть досвідчені пілоти не здатні протягом 10–20 хвилин утримувати постійну концентрацію без періодів «просідання» уваги, що

робить ручне керування непридатним для довготривалих логістичних чи моніторингових місій.

Саме ці недоліки змушують шукати рішення, які могли б зняти з пілота критичне навантаження саме на етапах зльоту, зависання на малих висотах та посадки, зберігши при цьому повний контроль у решті фаз польоту.

1.5.2. Відсутність автоматизації зльоту, зависання та посадки

Більшість комерційних і кастомних FPV-дронів базуються на Betaflight та iNav і не мають вбудованої автоматизації для ключових етапів польоту – зльоту, зависання та посадки. Це обмеження успадковане від фокусу на маневреності та швидкості, а не на автономності.

Зліт, що виконується вручну дуже часто має надмірний керуючий вплив, що призводить до різкого, не надто контрольованого набору висоти.

Зависання в стандартних конфігураціях відсутнє. Оператор утримує вертикальну позицію вручну, витрачаючи на це додатковий ресурс уваги, та за вдсутності близьких та зручних орієнтирів, призводить до дрейфа.

Посадка це найкритичніший етап. Ручне зниження виконується з суб’єктивною оцінкою відстані "на око". Похибка в 20–50 см може призводити до ударів, пошкодження пропелерів чи корпусу. Відсутність автоматизації збільшує на певний час окремі етапи місії через вимушену підвищену обачність оператора.

Відсутність автоматизації цих функцій знижує ефективність у задачах доставки вантажу та моніторингу. У порівнянні з автоматизованими системами, FPV-дрони поступаються за показником надійності та безпеки, але виграють у вартості, бо автоматизація вимагає додаткових витрат.

1.5.3. Інтегрованість автоматизації та залежність від прошивки польотних контролерів

Автоматизація в FPV-дронах (наприклад, утримання висоти в iNav) жорстко інтегрована в прошивку польотного контролера.

Залежність від прошивки: Функції типу "Altitude Hold" доступні лише в певних конфігураторах iNav, ArduPilot, але відсутні в Betaflight. Зміна прошивки вимагає перекомпіляції, тестування та несе ризик нестабільності.

Обмежена модульність: Додавання автоматизації як то, через Lua-скрипти в Betaflight, залежить від версії програмного забезпечення і може конфліктувати з іншими налаштуваннями. У випадку керування через протокол передаються лише керуючі впливи по каналах, без вбудованої логіки автоматизації, що убезпечує наявну прошивку від конфліктів.

Ризики оновлень: Оновлення прошивки може зламати інтегровані функції, вимагаючи повного переналаштування.

Це ускладнює автоматизацію через конфігуратори в кастомних збірках, де спільнота FPV віддає перевагу гнучкості, але страждає від відсутності універсальної автоматизації без втручання безпосередньо в програмне забезпечення.

1.5.4. Проблеми інтеграції додаткових сенсорів у комерційних рішеннях

Інтеграція сенсорів таких як TOF, барометр або LIDAR у комерційні контролери обмежена, в першу чергу через обмеження кількості портів.

Також кількість доданих компонентів впливає на завантаженість процесору. Польотні контролери що базуються на F4/F7 чіпах, обробляють ПІД-цикли на 8–32 кГц. Додавання сенсора наприклад, ToF підключеного за протоколом I2C/UART, збільшує навантаження на 10–20%, призводячи до збільшення часу обробки кожного окремого циклу, що критично для маневрів.

Споживання ресурсів: Фільтрація даних із застосуванням фільтрів Кальмана для сенсорів споживає оперативну та флеш-пам’ять, обмежуючи ресурс польотного контролера.

Сумісність: Не всі прошивки підтримують обробку поєднаних даних сенсорів. Наприклад, Betaflight оброблює тільки барометр для показників висоти, без датчиків відстані.

Ці недоліки підкреслюють потребу в зовнішніх модулях, як ESP32, для ізоляції автоматизації без навантаження основного контролера.

Таким чином, сучасні прошивки для FPV-дронів чітко поділяються за рівнем автономності польотів та ступенем участі пілота в режимі керування. Однак при цьому зростають вимоги до апаратної бази і, відповідно, збільшується ціна.

Betaflight впроваджує виключно ручне пілотування. Радіоканал є єдиним і остаточним джерелом команд.

iNav демонструє гібридний підхід дозволяючи як повноцінне ручне керування, так і розвинені автономні функції, за наявності необхідних датчиків.

ArduPilot базується на принципі макссимальної автономії. Сигнали ручного керування розглядаються як тимчасове втручання в автоматичний план місії, що є приорітетним.

Висновок до розділу. У контексті класичного FPV-пілотування від першої особи, саме Betaflight та iNav залишаються платформами, де пілот несе повну відповідальність за висоту і траєкторію на всьому маршруті та на його найвідповідальніших ділянках. ArduPilot же принципово відходить від концепції FPV-пілотування.

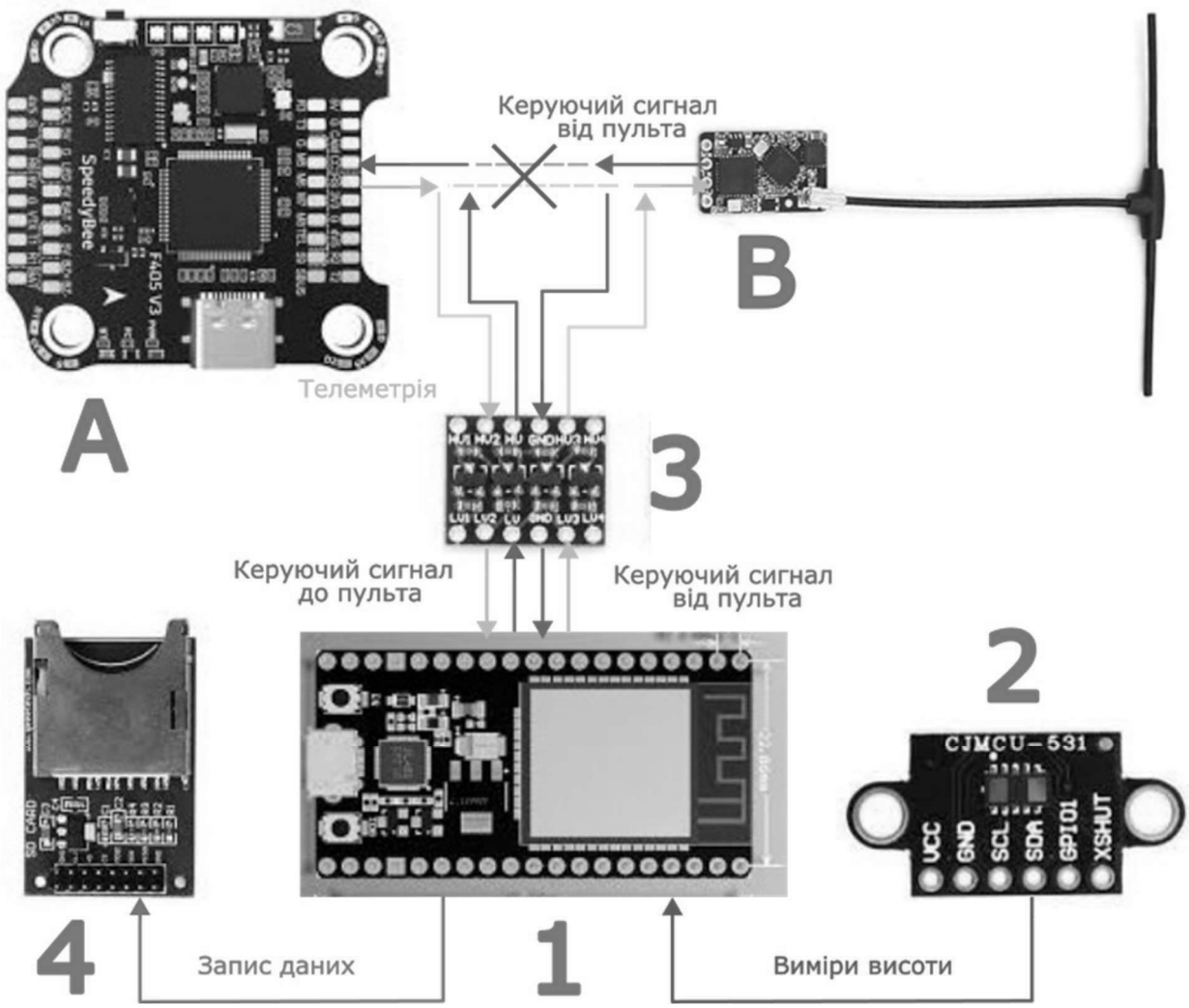
Саме тому, незважаючи на всі досягнення сучасних стабілізаційних алгоритмів такі ключові локальні задачі як плавний зліт, стабільне зависання на малій висоті до 3 м, безпечна посадка і досі залишаються виключно на відповідальності оператора. Ці операції виконуються вручну навіть на повністю налаштованих дронах і продовжують залишатися основною причиною пошкоджень апаратів та їх компонентів поблизу землі.

Така ситуація однозначно вказує на назрілу потребу в додаткових технічних рішеннях будь то доопрацювання існуючих прошивок, чи то введення окремих автоматичних функцій, чи то розробка спеціалізованого зовнішнього пристрою, який міг би взяти на себе саме ці критичні, локальні етапи польоту, не порушуючи при цьому загальних принципів ручного FPV-

пілотування та забезпечуючи легкість інтеграції з усім різномаїттям аппаратних компонентів та їх програмного забезпечення.

2 АКТУАЛЬНІСТЬ ТА ПЕРЕВАГИ МОДУЛЯ АВТОМАТИЗАЦІЇ ПОЛЬОТНИХ ФУНКЦІЙ

З метою вирішення задач безпеки в безпосередній близькості до поверхні землі, пропонується проміжний модуль (рис. 2.1) на базі мікроконтролера та датчика ToF котрий є зовнішнім пристроєм, що встановлюється в розріз лінії сполучення між радіоприймачем та польотним контролером і діє як проміжна ланка.



Елементи FPV: A – Польотний контролер; B – Радіоприймач. Елементи модуля: 1 – Мікроконтролер; 2 – датчик висоти; 3 – конвертер логічних рівнів; 4 – модуль SD-карти.

Рисунок 2.1 – Схема підключення модуля

У ручному режимі він повністю пропускає сигнали пульта без втручання, а в автоматичному – перехоплює та коригує задані канали в залежності від режиму, забезпечуючи зліт до заданої висоти та зависання, або безпечну посадку. Такий підхід принципово відрізняється від монолітних рішень, що пропонують програмні конфігуратори, де всі функції виконуються єдиним пристроєм та є неподільними за своїм функціоналом та впливом.

Заради чіткого відмежування сфери застосування, варто одразу окреслити коло самих пристроїв та їх конфігураторів, що розглядаються як доцільні для застосування такого модуля. На підставі розгляду, що було проведено в Розділі 1 цієї роботи, доцільним вважається застосування такого модуля з конфігураторами Betaflight та iNav. Ardupilot, з огляду її комплексності, багатозадачності та чисельності закладеного функціоналу як і аналоги накшталт PX4 виключають можливість та необхідність застосування модуля. Те саме стосується й польотних контролерів. Використання модуля доцільно з тими продуктами, що орієнтовані на FPV-нішу, як то: SpeedyBee, Matek, JHE MCU, TYKHO Lit, тощо. Так само польотні контролери таких класів як: Pixhawk 6C та 6X, Cube Orange та Black, Durandal, виключають потребу застосування модуля.

Основними елементами схеми є мікроконтролер та датчик відстані. Саме вони виконують отримання даних, розрахунок та за необхідності втручання в роботу двигунів. Допоміжні елементи – SD-карта та конвертер логічних рівнів. В процесі побудови та налагодженя заплановано розподілення ресурсів відповідно до можливостей двоядерного процесора. Отримання даних відстані та розрахунки проводяться на першому ядрі, в той час, як друге ядро має взяти на себе більш громіздкі та витратні з точки зору часу процеси запису логів на SD-карту. Конвертер рівнів, опціонально в залежності від типу контролера, має узгоджувати рівень логічних сигналів для запобігання конфліктів між модулем, польотним контролером та радіоприймачем.

2.1. Зниження навантаження на оператора через автоматизацію ключових етапів польоту

Високе навантаження на оператора виникає через інерційні психофізіологічні фактори: втома, стрес, нестача концентрації, бажання розслабитися та відчуття, що основна частина місії вже завершена. Особливо після тривалого польоту або в складних умовах ці фактори призводять до помилок саме на етапі приземлення, коли нервова система ще не встигла перейти від агресивних маневрів до плавного режиму. Модуль повністю автоматизує вертикальні етапи: оператор активує режим одним перемикачем, після чого модуль самостійно розганяє двигуни до обертів ефективної тяги, та виводить апарат на задану висоту, утримує її ПІД-регулятором по висоті або виконує посадку зі швидкістю зниження 0,1 м/с – 0,3 м/с. Пілот зберігає частковий контроль над нахилом по осях, але звільняється від необхідності одночасно стежити за газом і висотою. Це знижує когнітивне навантаження на 80 % – 90 % і робить польоти тривалістю 15 хв – 20 хв фізіологічно комфортними навіть для початківців.

2.2. Підвищення безпеки та збереження цілісності FPV-дронів

Ручна посадка залишається основною причиною пошкоджень FPV-дронів: зігнуті пропелери, пошкодження двигунів, тріщини в карбоновому корпусі, поломки камер та відеопередавачів (рис.2.2). Обмежена видимість через кут камери, наближений до горизонталі, нерівна поверхня та сторонні об’єкти поза полем зору роблять оцінку відстані емпіричною та неточною. Модуль усуває ці ризики завдяки датчику ToF, який вимірює висоту перпендикулярно землі з точністю ±5 см навіть при яскравому сонці та помірній запиленості. ПІД-регулятор змінює закон управління по швидкості зниження а не по висоті, і тим гарантує постійні 0,1м/с – 0,3 м/с. При висоті максимально наближеній до положення дрона на землі, газ плавно знижується

до безпечного рівня, залишаючи при цьому апарат в стані готовності до наступних команд. При втраті сигналу датчика газ фіксується на рівні керуючого впливу трохи нижче за значення відповідне утриманню висоти.

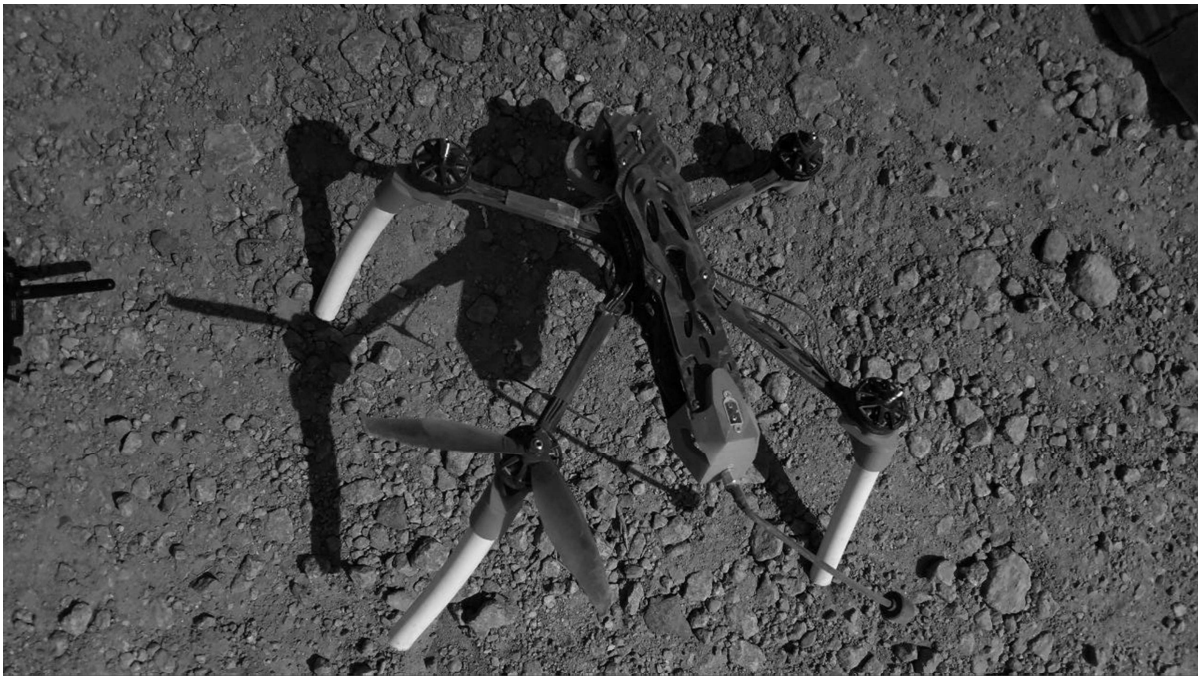


Рисунок 2.2 – Ушкодження дрона внаслідок посадки

Така стратегія дозволяє забезпечити мінімальне зниження та вхід в зону видимості датчика з мінімальним прискоренням. Очікуються стабільні плавні приземлення, за умови що швидкість на момент ввімкнення не перевищує критичну. Розрахункова тяга апарату має бути в межах максимальної, що здатні розвинути гвинтомоторні пари. За цієї умови керуючий вплив зможе створити силу достатньою для гальмування та посадки, тим самим значно збільшивши фізичну збереженість дрона.

2.3.Модульність і сумісність через протокол Crossfire

Модуль підключається між радіоприймачем та польотним контролером і працює виключно з протоколом CRSF, не торкаючись прошивки Betaflight чи iNav. Це дозволяє інтегрувати його в різноманітні кастомні збірки на будь

якому польотному контролері, що користується зазначеним протоколом, та будь яеим сумісним радіоприймачем, незалежно від робочих частот без перекомпіляції. Налаштування зводиться до узгодження відповідності каналів керування режимами. Оновлення прошивки модуля виконується через USB, дрон при цьому не розбирається. Така «прозора» архітектура робить модуль універсальним для FPV-спільноти, де кастомні збірки змінюються щотижня.

Протокол Crossfire – Crossfire Serial Protocol, являє собою відкритий, високошвидкісний послідовний протокол передачі даних, що був розроблений командою TBS спеціально для низьколатентного радіокерування FPV-дронами та телеметрії. На фізичному рівні він працює через один UART-провід чи то односпрямований або дуплексний, зі швидкістю до 420 000 бод, що забезпечує оновлення всіх 16 каналів з періодом ~4 мс (250 Гц) і реальну затримку від пульта до польотного контролера менше 10–12 мс навіть на відстані 20–40 км.

Ключові особливості CRSF роблять його ідеальним для введення проміжного модуля. Протокол формує пакетну структуру фіксованої довжини 26 байт для каналів радіокерування типу 0x16, з 11-бітним кодуванням кожного каналу 172 мкс – 1811 мкс і захищений вбудованою контрольною сумою CRC8. Містить повний дуплекс. команди керування передаються одночасно від пульта і телеметричні параметри від дрона такі як стан заряду акумулятору, індикатор рівня прийнятого сигналу, позиціонування за GPS, висота тощо, у зворотному напрямку.

Стосовно протоколу сформована відкрита специфікація. Документація та приклади коду TBS публічні, що дозволяє будь-якому мікроконтролеру (ESP32, STM32, Teensy) повністю розібрати за складовими, модифікувати і переслати пакет за мікросекунди. Реалізована підтримка до 16 каналів та додатково двох 8-бітних Link Quality пакетів, що дає достатньо місця для допоміжних перемикачів типу активація режимів, регулювання ПІД, тріммінг тощо, без конфлікту з основними каналами керування по осях.

2.4.Зручність для навчання операторів

Початківці втрачають або суттєво ушкоджують 3 – 5 дронів саме на етапі освоєння посадки. Модуль дозволяє тренувати як окремо вільний політ вище зони вимірювання датчика, так і забезпечує страховку при тренуванні посадок завдяки тому, що вертикаль повністю автоматизована. Оператор в будь який момент має можливість увімкнути автоматичне приземлення та уникнути падіння. Логування на SD-карту параметрів висоти, вертикальної швидкості, та значень керуючого впливу кожні 250 мс, дає детальний графік, що можна проаналізувати завдяки дешифратору логів, для подальшого аналізу та виявлення похибок учня. Досвідчений інструктор на підставі цих записів зможе надати ретельний аналіз тренування. Якщо паралельно ведеться відеозапис, учень може побачити наочно результати свого керування без поспіху та скоріше дійти розуміння як покращити свої навички. Час до першої безпечної посадки скорочується, кількість розбитих дронів значно зменшується.

2.5.Кастомні рішення та їх інтеграція з модулем автоматизації

Модуль не дублює функції польотного контролера. Схема умовно надбудовується поверх нього, зберігаючи всі ПІД-налаштування, що їх було сформовано Betaflight чи iNav. Власний ПІД-регулятор працює тільки для корекції по вертикальній осі, розвантажуючи основний процесор польотного контролера від додаткових обчислень коригуючого впливу, відносно показників, що надходять з пульта оператора, та необхідності обробки даних датчика висоти. Ізоляція помилок гарантує, що збій у модулі не виводить дрон з ладу. Та при тривалій втраті сигналу, забезпечує плавну посадку. Логування на окремій SD-карті також не навантажує польотний контролер. При майбутньому підключенні комп’ютера-компаньона, наприклад, Raspberry Pi, модуль вже має готовий набір базових функцій: зліт, зависання, посадка, що

можуть бути передані під керування компаньйона, тим самим зменшуючи обсяг його задач та навантаження. Таким чином, отримуємо економію ресурсів і для задач вищого рівня. Своєю архітектурою модуль створює проміжний “рефлекторний” тип керування, між ручним та повною автономією, залишаючись при цьому простим, дешевим і сумісним з будь-якою кастомною збіркою в межах протоколу.

Висновок до розділу. Проміжний модуль автоматизації польотних функцій вирішує три головні проблеми сучасних FPV-дронів: дозволяє компенсувати психофізіологічне виснаження оператора, помітно скоротити високий відсоток пошкоджень при посадці та нівелювати складність інтеграції нових функцій у кастомні збірки за принципом plug-and-play. При прогнозованій низкій собівартості, що значно менше ніж вартість елементів що він убезпечує, модуль суттєво підвищує фізичну збереженість дрона, помітно скорочує час навчання новачків та відкриває шлях до надійної автоматизації зльоту, зависання та посадки в зоні безпосередньої близькості до поверхні землі без зміни прошивки польотного контролера.

3 МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ АВТОМАТИЧНИХ РЕЖИМІВ

Математичне моделювання є ключовим етапом розробки проміжного модуля автоматизації польотних функцій FPV-дрона, оскільки дозволяє відпрацювати алгоритми зльоту, зависання та посадки ще до проведення реальних польотів, значно мінімізувати ризики пошкодження апаратури, а також, у контрольованих умовах, порівняти різні закони регулювання та оптимізувати параметри регулятора з метою отримання оптимальних показників. Використання математичного моделювання дає змогу провести низку віртуальних експериментів, виявити критичні режими роботи системи та підібрати оптимальні коефіцієнти регулятора без жодного реального пошкодження дрона.

Усі моделі та інструменти симулятора польотних режимів і дешифратора логів реалізовано мовою програмування Python версії 3.10.11. Вибір Python зумовлений високою швидкістю прототипування, читабельністю коду та зручністю вираження математичних залежностей. Бібліотека NumPy для роботи з масивами та векторними обчисленнями забезпечує виконання інтегрування методом Ейлера, імітацію зашумлення сигналу та розрахунок швидкості й прискорення. Бібліотека Matplotlib для візуалізації в Python застосована для побудови графіків у реальному часі.. Вона дозволяє реалізувати динамічний інтерфейс симулятора з частотою оновлення 50–100 кадрів/с, створення слайдеру, регулювання параметрів та кольорового позначення швидкості дотику. Keyboard, дає можливість обробки натискань клавіш у фоновому режимі без блокування основного потоку. Забезпечила миттєве перемикання режимів між ручним та автоматичним. Бібліотека Tkinter використана при побудові дешифратора логів для створення діалогового вікна вибору файлу, та відображення легенд з кольоровим маркуванням режимів польоту.

Поєднання цих бібліотек з мінімальною кількістю залежностей дозволило створити два цілком функціональні науково-дослідні інструменти – симулятор приземлення та дешифратор логів.

3.1.Розробка симулятора польотних режимів FPV-дрона

Розробка спеціалізованого симулятора (рис. 3.1) дозволяє моделювати динаміку дрона в контрольованому віртуальному середовищі, в умовах реального часу.

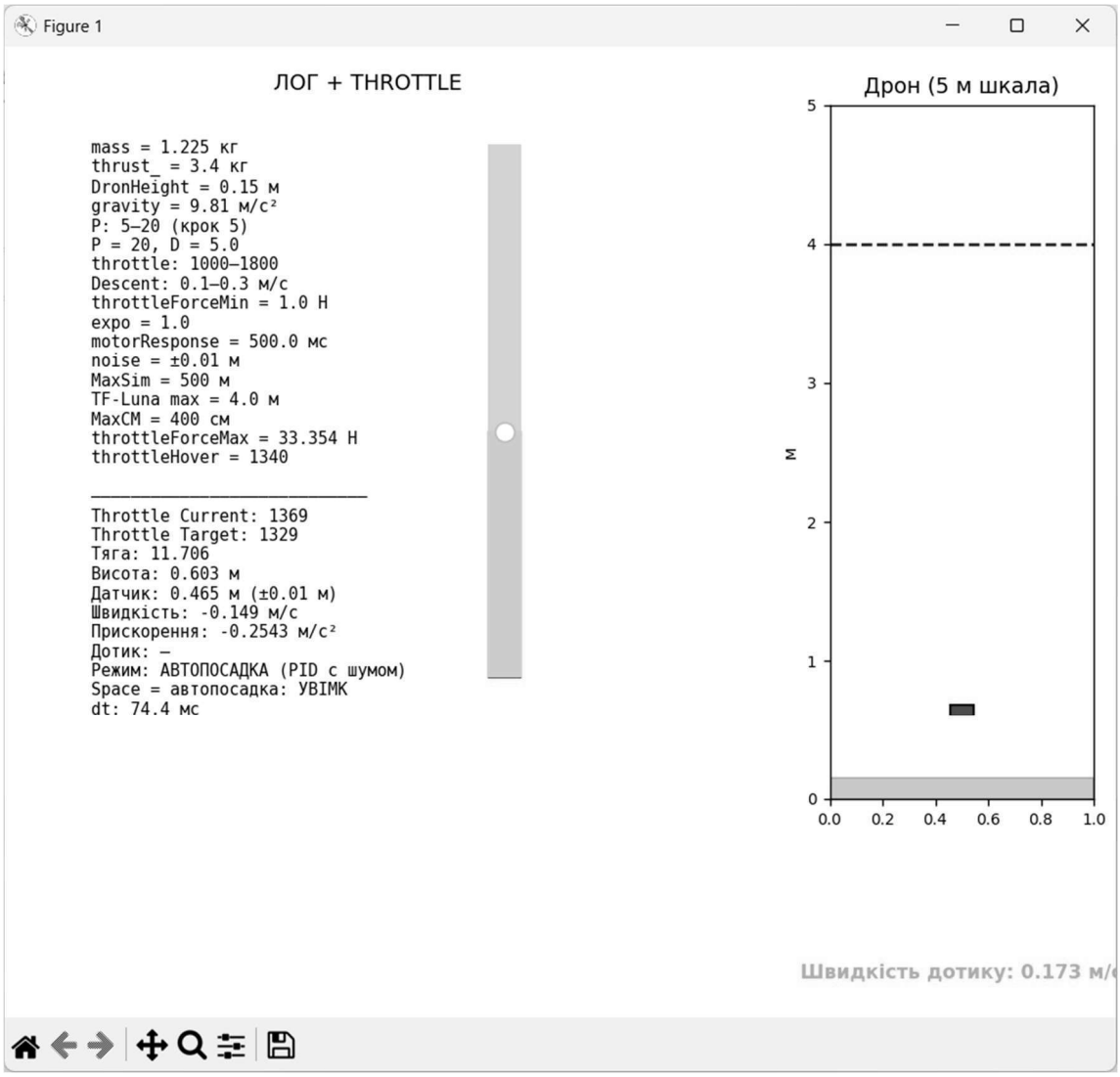


Рисунок 3.1 – Інтерфейс симулятора приземлення дрона

Традиційні методи відпрацювання алгоритмів на фізичних прототипах пов'язані з високими ризиками пошкодження обладнання, значними витратами часу та ресурсів, а також обмеженнями в ітеративнім налагодженні параметрів.

Попереднє моделювання в симуляторі забезпечує відсутність фізичних ризиків, дозволяє тестувати екстремальні режими як то: різке падіння, удари об землю або перевантаження двигунів без шкоди для апарату. З’являється можливість швидкої зміни констант (маса, тяга, експонента керування) та параметрів регуляторів (P, PD, PID) з швидкою візуалізацією результату.

Симулятор підтримує перемикання режимів ручного керування та автоматичного приземлення, утримання висоти та автоматизовану посадку з ПІД-регулятором, що імітує реальні сценарії FPV-пілотування.

Імітація шуму датчиків передбачає включення адитивного шуму, в межах ±0.01 м, до вимірів висоти. Моделювання реальної поведінки ToF сенсора дозволяє оцінити стійкість алгоритмів до реальних похибок сенсорів.

Інтерактивна анімація з логом параметрів (висота, швидкість, значення керуючого впливу) полегшує діагностику поведінки системи, фіксація швидкості дотику в момент приземлення дозволяє аналізувати ефективність роботи регуляторів.

Симулятор, як проміжний етап між теоретичним моделюванням та польовими випробуваннями, приймає на себе всі ризики та помилки прискорюючи етап розробки алгоритмів автоматизації.

Код симулятора польотних режимів наведено в додатку Б.

В основу симулятора покладено математичну модель фізичних процесів. Саме вона забезпечує максимальне наближення умов симуляції до реальних умов польоту.

3.2.Математична модель фізичних процесів

Динаміка вертикального руху дрона моделюється за висотою, на основі класичної фізики другого закону Ньютона вздовж вертикальної осі. Основні фізичні величини:

m = 1,225 кг – *маса дрона;*

g = 9,81 м/с² – *прискорення вільного падіння;*

F_t – *миттєва тяга двигунів, Н;*

N – *сила нормальної реакції опори;*

h – *висота центру мас дрона над рівнем землі, м;*

$v = \dot{h}$ – *вертикальна швидкість, м/с;*

$a = \ddot{h}$ – *прискорення, м/с².*

Рівняння руху:

$$m\,a = F_t - m\,g + N \tag{3.1}$$

$$a = \frac{F_t - m\,g + N}{m} \tag{3.2}$$

Інтегрування виконується чисельно методом Ейлера з кроком Δt:

$$v_{k+1} = v_k + a_k\,\Delta t, \tag{3.3}$$

$$h_{k+1} = h_k + v_k\,\Delta t. \tag{3.4}$$

Умови обмеження висоти та контакту з поверхнею визначені наступним чином. Положення дрона по висоті обмежено мінімальним значенням h_{min} = 0,15м (висота спокою на посадковому майданчику).

При досягнення параметром висоти значення $h \leq h_{min}$:, фіксується швидкість дотику до поверхні грунту. За умови, якщо вшидкість дотику превищує порогове значення, моделюється відскок з коефіцієнтом відбиття від

землі 0,25%. Таким чином, за спрощенною схемою моделюється удар з відскоком. Оскільки для розрахунків має значення лише швидкість першого дотику, як основний показник, це дозволяє значно спростити задачу, та позначити відскок від поверхні та швидкість підйому в наслідок удару умовно. Подальші стрибки та касання вже не розглядаються, оскільки сам факт сильного удару, що провокує відскок вже є неприпустимим для поточного завдання.

Розрахунок порогового значення та швидкості відскоку:

$$v < -0{,}5 \text{ м/с},$$
$$v = -0{,}25 * v.$$

В іншому випадку – удар вважається незначущім, а сила дотику – такою що повністю поглинається. При цьому посадка вважається успішною. Поверхня грунту розглядається як статична опора:

$$N = max(0,\ mg - F_t),\ v = 0,$$

де N – сила реакції поверхні землі

Фізична модель, що використана для симуляції, завжди свідомо починається з початкових умов падіння дрона з висоти 5 м, така стратегія реалізована для контролю коректності розрахунку дії сили тяжіння, та швидкості першого дотику до поверхні землі. На рисунку 3.2 відображається і крива падіння за висотою, і зміна швидкостей, що демонструє роботу ефекту відскоку, як маркера неприпустимості такої посадки, та початкову швидкість спрямовану догори, що зумовлена початковим обертанням гвинтів з мінімальною тягою.

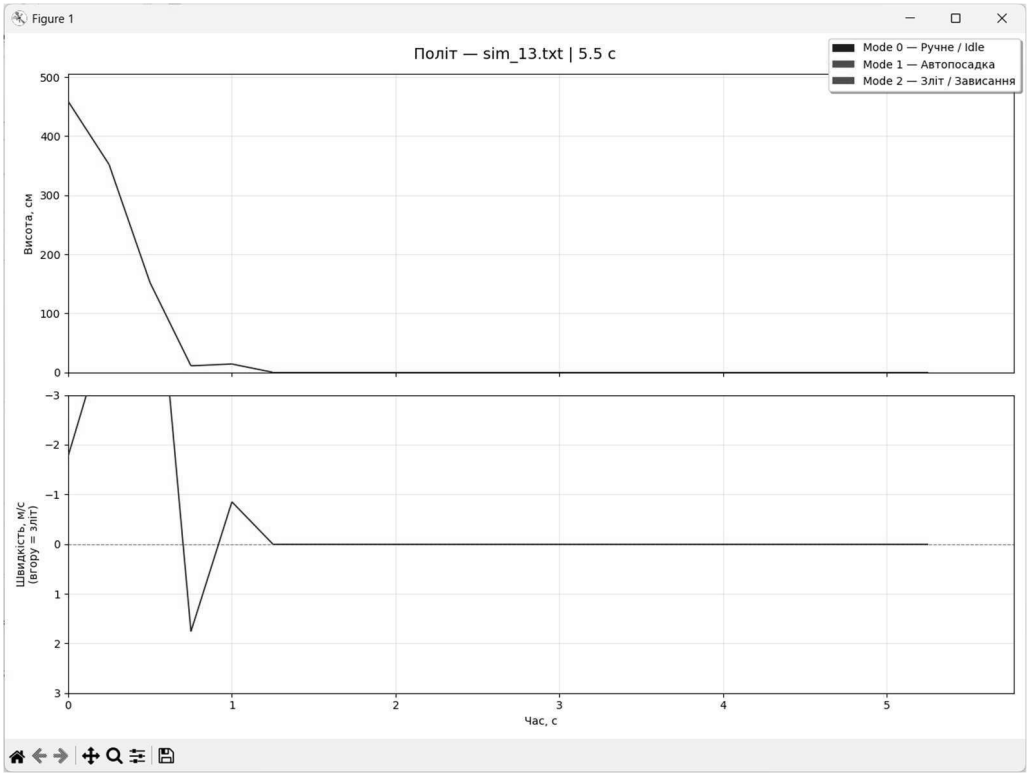


Рисунок 3.2 – Демонстрація роботи моделі сили тяжіння

3.3.Математична модель сумарної тяги двигунів

Стабілізація площини дрона горизонтально відносно поверхні, повністю покладається на польотний контролер, при тому, що реально контролер розподіляє навантаження нерівномірно, відповідно до зміщень центра мас, обурень та інших впливів, стабілізація повністю передана в зону відповідальності польотного контролера, що є занчущою перевагою модульності. Тому тяга двигунів, в поточній моделі розглядається як єдина величина що не потребує коригування всередині системи що моделюється.

Тяга двигунів залежить виключно від сигналу керуючого впливу:

$$u \in [1000, 2000] \ (\mu s):$$

$$u_{norm} = max(0, min(1, \frac{u - 1000}{1000})), \tag{3.5}$$

$$F_t(u) = F_{min} + (F_{max} - F_{min}) * (u_{norm})^{exp}, \tag{3.6}$$

$$F_{max} = m * g * F_{thrust} \tag{3.7}$$

$$F_{max} = 1{,}225 * 9{,}81 * 3{,}4 \approx 41{,}07\ Н,$$

$$exp = 1$$

де u – сигнал керуючого впливу;

$F_{min} = 1{,}0$ Н, – мінімальна тяга;

$F_{max} \approx 41{,}07$ Н – максимальна тяга;

F_{thrust} – ефективна тяга гвинтів (за відрахуванням сили тяжіння);

exp – експоненційна складова відхилення тяги.

Загальна затримка реакції системи на керуючий вплив, узагальнена як час відгуку двигунів, що моделюється як інерційна ланка першого порядку:

$$\tau = \tau_{max} * norm(\Delta F_t), \tag{3.8}$$

$$a = \frac{\Delta t}{\tau + \Delta t}, \tag{3.9}$$

$$F_{t,k+1} = F_{t,k} + a\ (F_t(u_{target}) - F_{t,k}). \tag{3.10}$$

Модель формування вимірів датчика висоти ToF:

$$\tilde{d} = \begin{cases} d + N(0,\sigma), & d \leq 4{,}0\ м \\ 0, & d > 4.0\ м \end{cases}$$

$$d = h - h_{min}, \tag{3.11}$$

де d – поточна дистанція до землі;

$\tilde{d}$ – дані датчика з урахуванням похибки;

$\sigma = 0.01\ м.$ – похибка.

В симуляторі автоматична посадка активується та вимикається натисканням клавіши “Пробіл” і базується на регуляторі, який керує швидкістю спуску $v_{target}(d)$, залежно від поточної відстані до землі.

Цільова швидкість спуску розраховується за функцією, що є лінійною інтерполяцією за законом мапи залежності швидкості від висоти:

$$v_{target}\,(d) = v_{min} + (\,d - 0{,}1)\frac{v_{max} - v_{min}}{4-0{,}1} \qquad (3.12)$$

де $v_{min} = 0{,}1$ *м/с;*

$v_{max} = 0{,}3$ *м/с.*

Розрахунок керуючого впливу відповідно ПІД-закону виконується на кожному кроці вимірювання. Відповідно до інтервалів проміжків вимірювання датчика, оновлення відбувається кожні 50 мс, та фіксується за умови, що датчик отримує валідні дані, тобто земля знаходиться в зоні вимірювання:

$$e_k = v_k - v_{target,k}, \qquad (3.13)$$

$$v_k = \frac{\tilde{d}_{k-1} - \tilde{d}_k}{\Delta t_{PID}} \qquad (3.14)$$

$$I_k = I_{k-1} + e_k\,\Delta t_{PID}, \qquad (3.15)$$

$$D_k = \frac{e_k - e_{k-1}}{\Delta t_{PID}} \qquad (3.16)$$

$$\Delta u = Pe_k + I_c I_k + D_c D_k, \qquad (3.17)$$

де $P = P_{coef} \in [5, 20]$, – пропорційний коефіцієнт;

$I_c = 0{,}1 + (P-5)\frac{0{,}25-0{,}1}{20-5}$ – інтегральний коефіцієнт;

$D_c = 5{,}0$ – коефіцієнт похідної;

e_k – поточна помилка;

v_k – поточна швидкість

v_{target} – цільова швидкість

Δu – розрахункова корекція керуючого впливу.

Оновлення керуючого впливу сигналу газу:

$$u_{target,k+1} = min\ (max\ (u_{target,k} + \Delta u, 1000), 1800) \qquad (3.18)$$

При поверненні значення $d \leq 0.01$ *м*, тобто поза зоною дії датчика, фіксується режим утримання поточного стану режиму керування та визначення дистанції поза межами вимірювання. При цьому керуючий вплив встановлюється без розрахунків, трохи менший за величину відповідну до зависання по вертикалі.

Задля визначення цієї умовної величини керуючого впливу, що відповідає нульовому прискоренню по вертикалі, була змодельована поведінка FPV-дрона при сталих обертах пропелерів, з фіксованою тягою. Наведений рисунок у вигляді графіків наочно демонструє як різниця лише в один пункт керуючого впливу змінює напрямок вертикального руху на межі числового значення що відповідає зависанню.

На практиці стабільне зависання з чітко фіксованими обертами неможливе, через ряд чинників та обурень що впливають на систему, котра характерізує положення апарату по вертикалі. Для теоретичного розуміння виведено значення керуючого впливу *и = 1340,5 мкс* (рис. 3.3) що на практиці є неможливим, але слугує відправною точкою при розрахунках.

При цьому вже мінімальне відхилення керуючого впливу навіть на 0,04% з постійним знаком, призводить до стійкого прискорення, що згодом може досягати помітних швидкостей, як демонструють графіки (рис. 3.4) та (рис. 3.5).

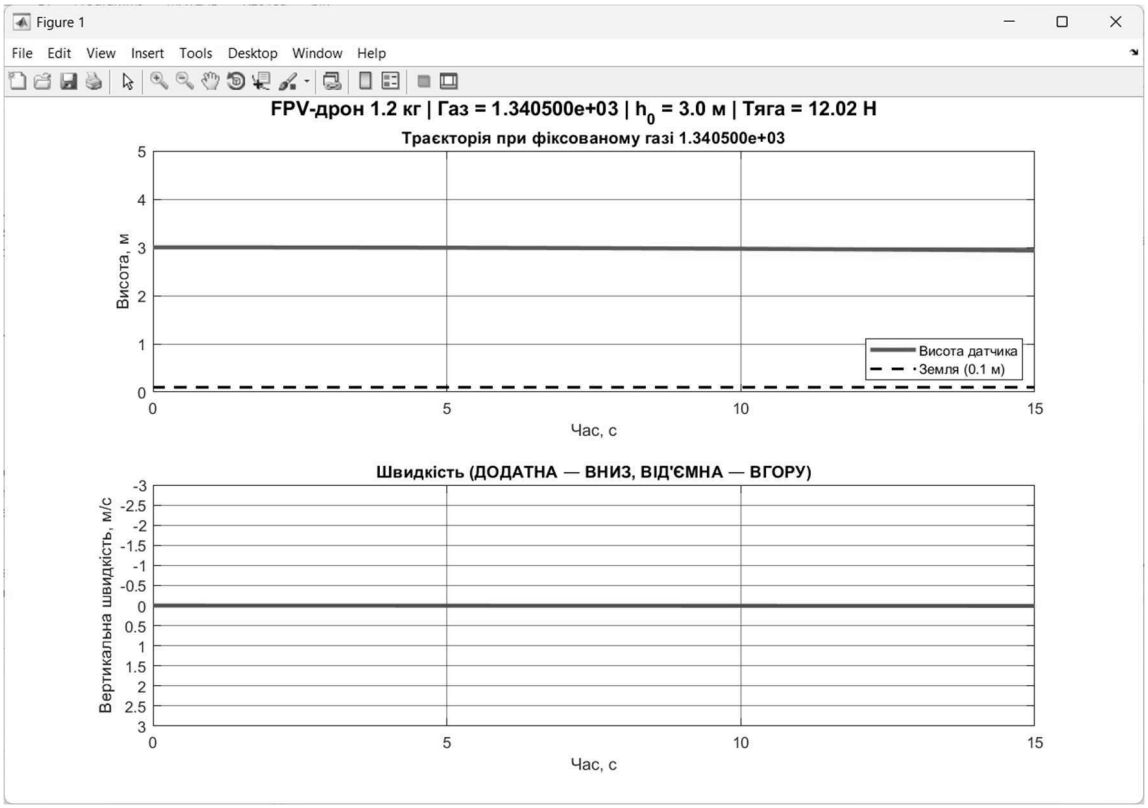


Рисунок 3.3 – Розрахункове значення керуючого впливу, відповідне зависанню (1340,5 мкс).

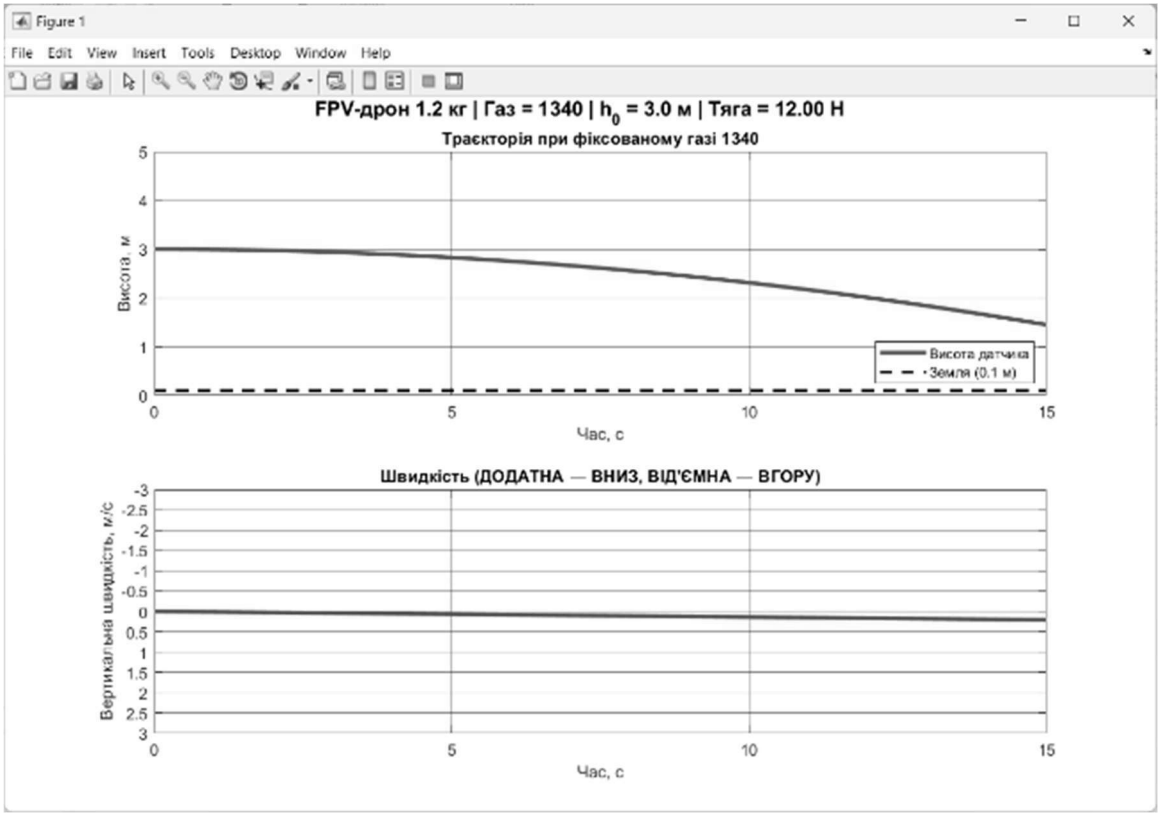


Рисунок 3.4 – Прогрес зниження при обертах зависання – 0,04%.

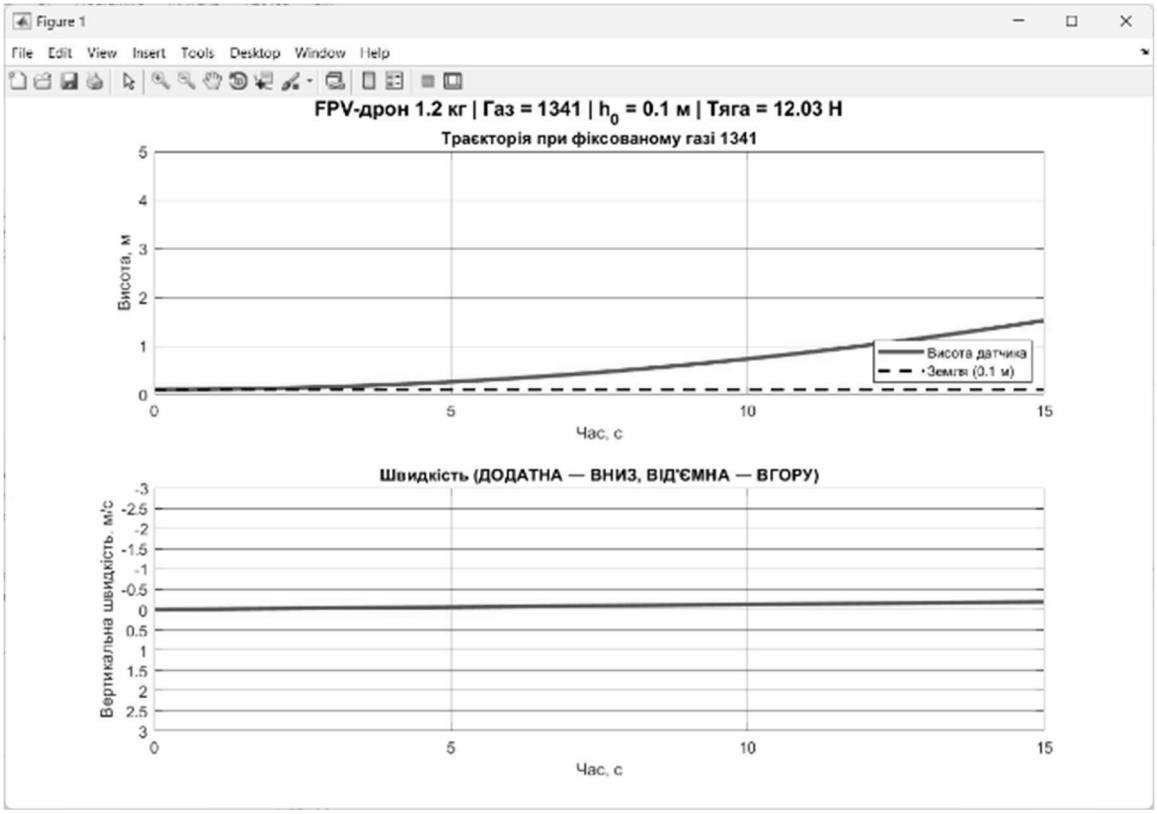


Рисунок 3.5 – Прогрес зльоту при обертах зависання + 0,04%.

3.4. Математична модель системи для стабілізації автоматичних режимів з ПІД-регулюванням

Вибір законів регулювання для режимів автоматичного зльоту-утримання висоти та автоматичного приземлення.

У модулі реалізовано два незалежні ПІД-контури, які працюють у одній функції, але розділені за станами «зависання» та «автопосадка». Обидва регулятори керують виключно вертикальною швидкістю, а не висотою безпосередньо, що забезпечує стійкість до коливань маси, вітру та розряду акумулятора.

ПІД-регулятор режиму зльоту та зависання.

Вхідні дані:

поточна висота (м), отримана з ToF-датчика: $d(t)$

вертикальна швидкість (м/с): $v(t) = -\frac{(d(t) - d(t-\Delta t))}{\Delta t}$;

цільова висота зльоту та зависання: $d_{target} = H_{hover} = 0{,}8$ м;

Помилка по висоті: $e_h(t) = d(t) - H_{hover.}$

Цільова швидкість формується нелінійним законом:

$$v_{target}(t) = \begin{cases} -0.1, & e_h(t) < -0.03 \\ 0, & |e_h(t)| \le 0.03 \\ +0.1, & e_h(t) > +0.03 \end{cases}$$

Помилка для ПІД:

$$e_v(t) = v(t) - v_{target}(t) \tag{3.19}$$

Керуючий вплив:

$$\Delta T(t) = P \cdot e_v(t) + I \int_0^t e_v(\tau)d\tau + D \cdot \frac{de_v(t)}{dt} \tag{3.20}$$

де P - пропорційний коефіцієнт;

I - інтегральний коефіцієнт;

D - коефіцієнт похідної складової;

e_v - помилка за швидкістю.

Оновлення швидкості обертання гвинтів зміни керуючого ШІМ сигналу:

$$T(t) = min\ (T_{max},\ max\ (T_{min},\ T(t-1) + \Delta T(t))) \tag{3.21}$$

ПІД-регулятор режиму посадки (autoland_active)

Вхідні дані:

поточна висота (м): d(t);

вертикальна швидкість (м/с): $v(t) = -\frac{(d(t) - d(t - \Delta t))}{\Delta t}$.

Цільова швидкість зниження формується лінійно залежно від висоти за формулою:

$$v_{target}(t) = v_{min} + \frac{d(t)-0.1}{H_{max}-0.1}(v_{max}-v_{min}), \quad (3.22)$$

Розрахункове значення визначається за поточною висотою. Чим ближче до точки цільової висоти, тим менше швидкість. Діапазон припустимих значень становить:

$$v_{min} = \pm 0.1 \text{ м/с},$$
$$v_{max} = \pm 0.3 \text{ м/с}.$$

Додатково встановлюється обмеження цільової швидкості за показниками межевих значень:

$$v_{target}(t) = min(v_{max}, max(v_{min}, v_{target}(t))) \quad (3.23)$$

Помилка розраховується за різницею поточної та цільової швидкостей:

$$e_v(t) = v(t) - v_{target}(t) \quad (3.24)$$

Керуючий вплив формується за сумою усіх складових ПІД-регулятора:

$$\Delta T(t) = P \cdot e_v(t) + I \int_0^t e_v(\tau)d\tau + D \cdot \frac{de_v(t)}{dt} \quad (3.25)$$

Оновлення значення керуючого впливу додатково обмежується верхнім та нижнім лімітами:

$$T(t) = min\ (T_{max},\ max\ (T_{min},\ T(t-1) + \Delta T(t))) \quad (2.26)$$

Розрахунок оптимальних коефіцієнтів ПІД-регулювання для режима посадки дрона в симуляторі.

Симулятор дозволяє варіювати Коефіцієнти законів регулювання, аналізуючи коливання висоти, час необхідний для переходу до цільової швидкості, та фіксувати швидкість дотику до землі. Початково змодельовані графіки з фіксованим керуючим впливом.

Зліт з максимальним керуючим впливом що було змодельовано, проявив на графіку поведінку, що створює надвисоке прискорення, та не підлягає реальному розрахунку в межах симулятора (рис. 3.6).

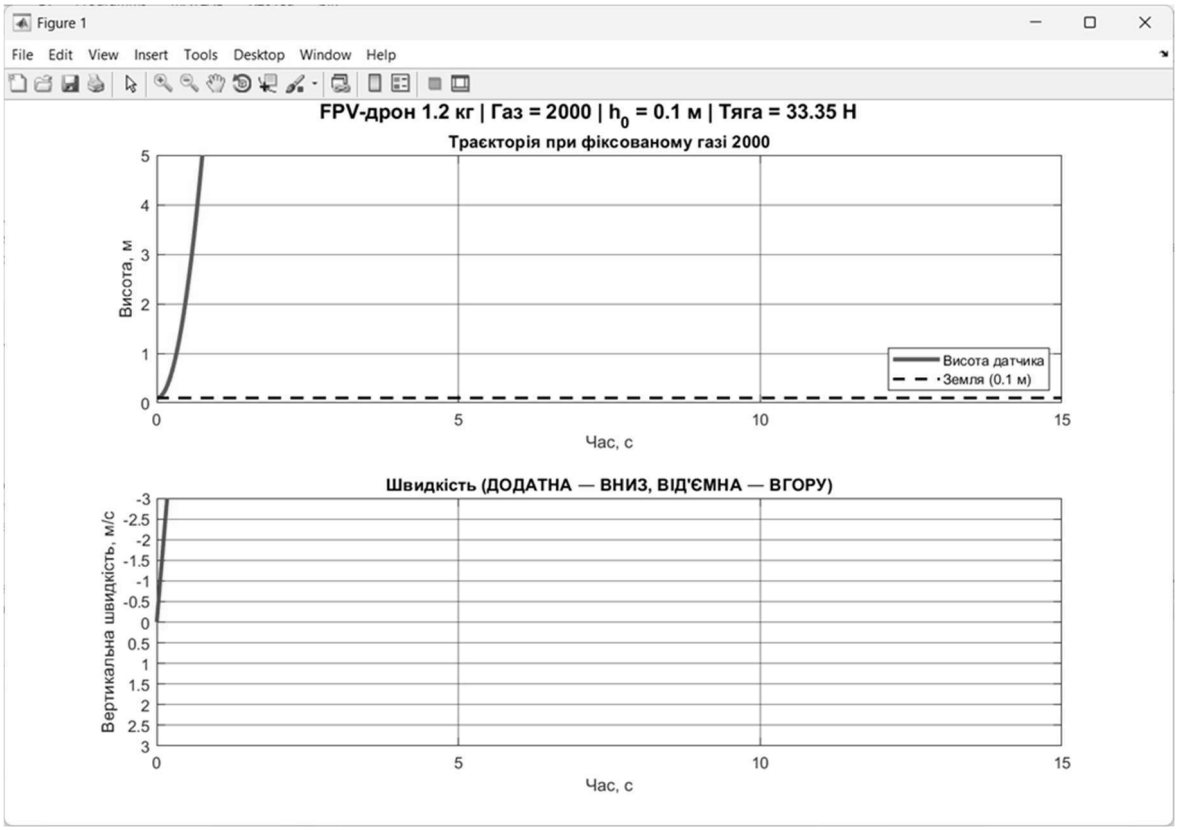


Рисунок 3.6 – Зліт з максимальним фіксованим керуючим впливом 2000 мкс

Через значне ускладнення динамічної моделі, що не є виправданим, при завданні з плавного регулювання. Отже, приблизний розрахунок виходу дрону

за межі вимірювання датчика складає близько 0,5 с та з прискоренням близько 17 м/с зліт продовжується. Така поведінка виправдовує застосування обмеження керуючого впливу поза зоною вимірювання датчика, задля погашення надлишкового прискорення.

Розгляд проміжних значень керуючого впливу в інтервалах між максимальним, впливом зависання, та мінімальним значенням, дозволяє отримати уявлення про вплив проміжних значень прискорення. Графік характеризує наскільки зростає швидкість при зниженні з висоти 3 метри зі сталим значенням керуючого впливу 1325 мкс (рис. 3.7), що менше за тягу зависання всього на 1,16%, таким чином отримуємо швидкість близько 1,5 м/с при торканні землі, що вже є неприпустимим.

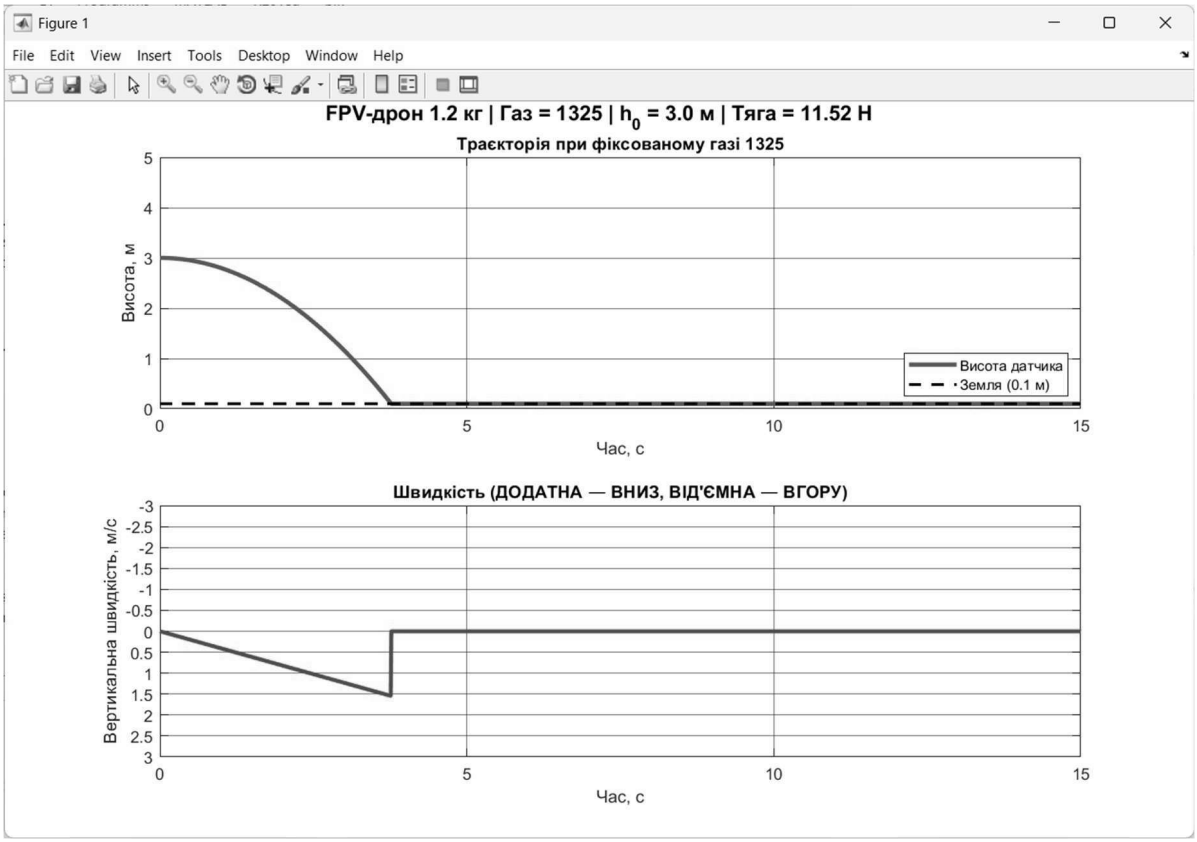


Рисунок 3.7 – Зниження з фіксованим впливом 1325 мкс

Розрахунки з помірним перевищенням тяги зависання на 4,44% та значенням керуючого впливу 1400 вже створює чимале прискорення (рис. 3.8), при якому дрон покидає зону вимірювання датчика відстані менше ніж за 2 секунди, що вже є прийнятним показником для впровадження регулювання.

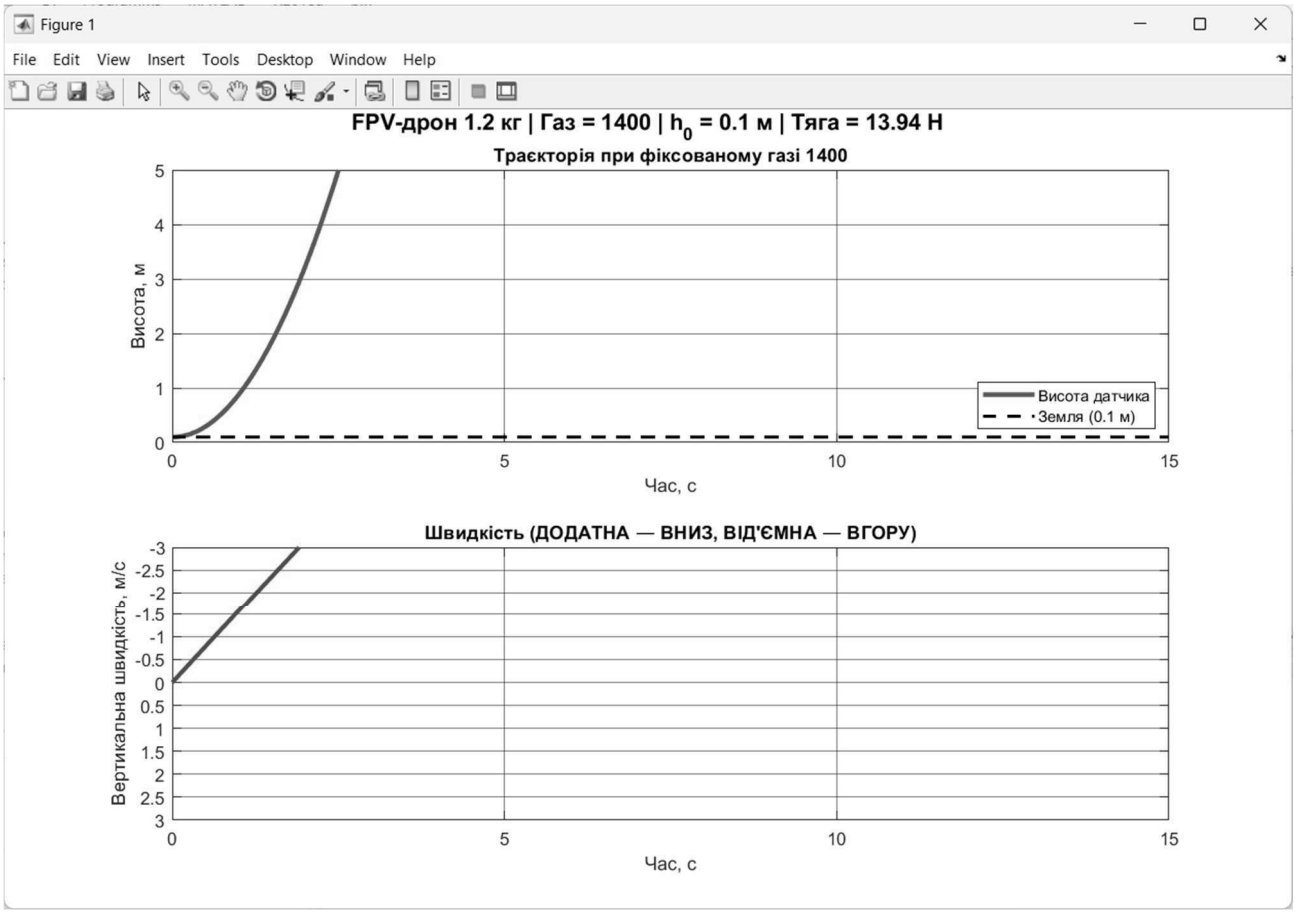


Рисунок 3.8 – Зліт з фіксованими впливом 1400 мкс.

3.5. Розробка дешифратора логів для аналізу польотних даних

Кожен тестовий політ фіксується на microSD-карті у вигляді текстового лог-файлу. Оскільки логування планується як на етапі симуляції, так і на етапі польових випробувань, дешифратор має працювати з файлами обох джерел походження. Для симулятора застосовується запис в файли «sim**.txt», тоді як логер формує файли «log**.txt». Обидва типи файлів мають дотримуватися єдиної структури для достовірної інтерпрітації даних. Маркером для

дешифратора, для визначення початку рядків, що містять корисну інформацію, слугує мітка « >ch ». Структура що несе корисне змістовне навантаження формує дані з частотою 4 Гц, відповідно до інтервалу запису. При цьому пакети логів на мікроконтролері оброблюються з частотою 4 Гц та, окрім ключових даних містять додаткову інформацію, що корисна при налагодженні. Така періодичність, разом з рішенням винесення функцій запису на окреме ядро процесора, має забезпечити захист від надмірного навантаження на ESP32. Ручний аналіз сотень або тисяч рядків у текстовому редакторі виявився неприйнятним за часом, тож створення спеціалізованої програми дешифратора, призначеної для швидкого завантаження, парсингу та візуалізації цих логів, значно заожаджує час та дає наочність при представленні даних.

При активації, програма автоматично відкриває діалогове вікно для вибору файлу, підраховує кількість рядків з міткою «> ch» і визначає загальну тривалість польоту, виходячи з інтервалу запису 0,25 с. Далі відбувається послідовне читання файлу з вилученням лише потрібних параметрів. Значення режиму польоту 0 – ручний режим, що характеризується транзитом керуючого впливу без втручання, 1 – автоматична посадка, при якій працює безпосередньо ПІД-регулятор за швидкістю, 2 – зліт з подальшим зависанням. Також дешифратор виокремлює висоту в сантиметрах та вертикальну швидкість у метрах за секунду. Покажчики керуючого впливу, як оригінального з пульта так і підмінних значень, хоча й присутні в логах, для даної візуалізації ігноруютсья.

Приклад запису даних в лог-файл симуляції:

```
# Симуляцію розпочато: 2025-12-09 22:14:37
> ch1000 0 0459 1000 +1.77
0.00; 1.77; 0.00; 0.00; 1000; 0.10; 20.00; 1; 273
> ch1000 0 0352 1000 +4.53
3.83; 4.53; 0.00; 0.00; 1000; 0.07; 20.00; 5; 306
```

> ch1000 0 0152 1000 +7.38

2.11; 7.38; 0.00; 0.00; 1000; 0.08; 20.00; 3; 317

> ch1000 0 0011 1000 -1.76

-0.01; -1.76; 0.00; 0.00; 1000; 0.06; 20.00; 3; 268

> ch1000 0 0014 1000 +0.85

0.21; 0.85; 0.00; 0.00; 1000; 0.08; 20.00; 3; 289

> ch1000 0 0000 1000 -0.00

-0.01; -0.00; 0.00; 0.00; 1000; 0.06; 20.00; 3; 266

> ch1000 0 0000 1000 -0.00

0.01; -0.00; 0.00; 0.00; 1000; 0.06; 20.00; 3; 270

> ch1000 0 0000 1000 -0.00

0.01; -0.00; 0.00; 0.00; 1000; 0.07; 20.00; 3; 291

Для коректного відображення переходів між режимами при зміні значення режиму поточна точка додається як до нового, так і до попереднього набору даних. Це усуває розриви ліній на графіках у моменти перемикання, що особливо важливо при аналізі швидких переходів з ручного режиму в автоматичні і навпаки.

Візуалізація побудована на двох синхронізованих графіках: верхній показує зміну висоти у сантиметрах, нижній – вертикальну швидкість у метрах за секунду додатне значення показує зниження, в той час як від'ємне характеризує зліт угору. Кожному режиму відповідає свій колір. Синій прийнятий для режима транзиту даних, червоний – для автоматичної посадки, зелений обрано для позначення зльоту та зависання. Легенда графіків формується у верхньому правому куті та дає миттєве розуміння, в якому режимі перебував дрон у той чи інший момент часу.

Анімація відтворюється з інтервалом 50 мс, що забезпечує плавне та зрозуміле сприйняття динаміки польоту в реальному часі навіть при тривалості запису понад 10 хвилин. Завдяки кольоровому кодуванню та суцільності ліній при аналізі завжди легко визначити моменти активації

режимів, характер зміни швидкості по вертикалі, наявність та інтенсивність коливань або різких стрибків положення по вертикалі, також діапазон дрейфу в режимі зависання.

Впроцесі симуляції випробувано дію зниженого впливу (1300), що прийнятий як фіксоване нерегульоване значення поза межами вимірювання датчика. Графік (рис. 3.9) демонструє, необхідність гасити надлишкове вертикальне прискорення вгору, що створює не менш небезпечне прискорення зниженя. Для ідеалізованої моделі симулятора, воно буде дорівнювати початковому прискоренню вгору, але з інверсією знаку і відповідно напрямку. В реальній ситуації, висота підйому як і прискорення зниження будуть дещо менші за змодельовані в симуляції, за рахунок дії супротиву повітря, що має квадратичну залежність відповідно до швидкості руху та протилежний по напрямку.

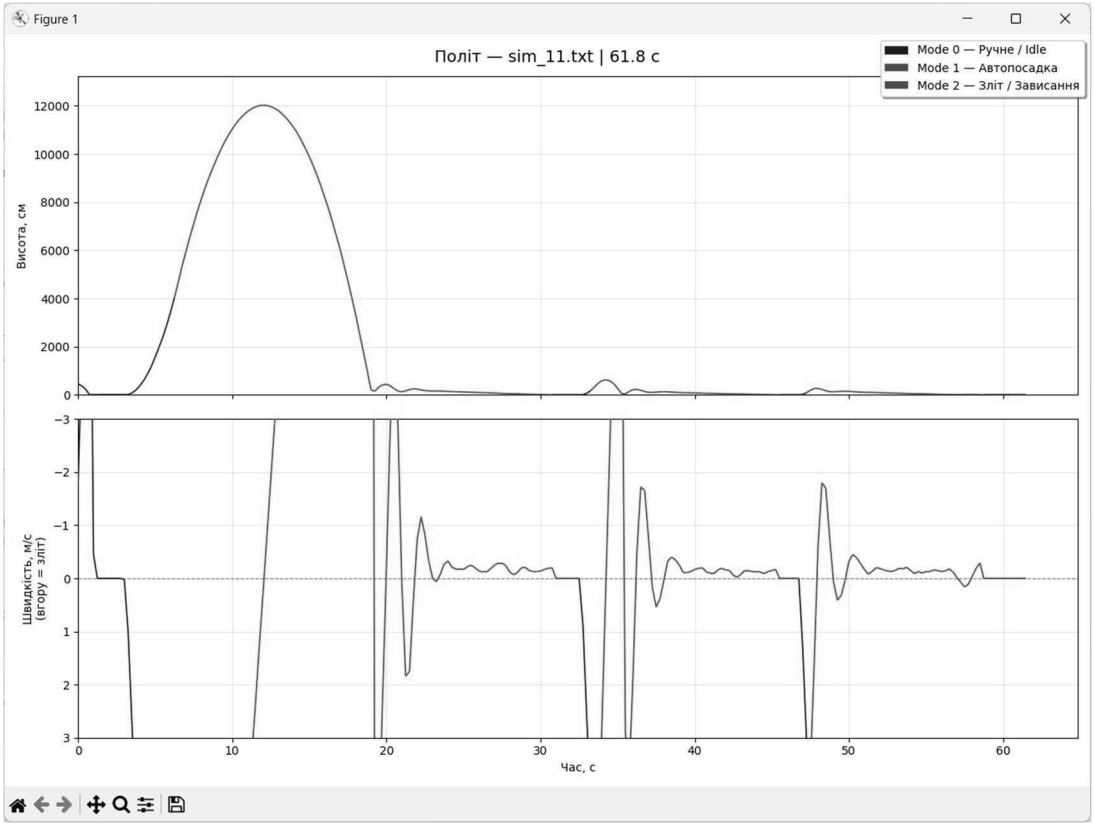
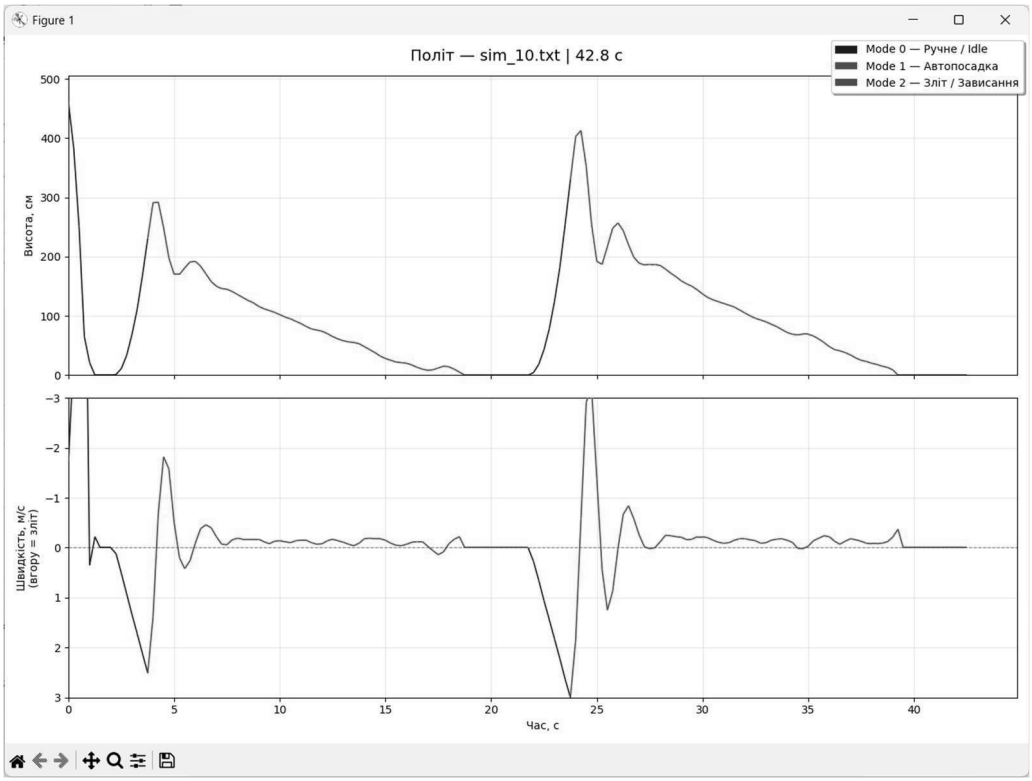


Рисунок 3.9 – Високе початкове прискорення вгору

При розробці та налагодженні дешифратор став незамінним інструментом щоденної роботи. Він дозволив, запобігти втратам апарату через неприпустимі значення регулювань, а також оперативно порівнювати ефективність різних наборів ПІД-коефіцієнтів (рис.3.10, рис. 3.11). Час аналізу кожного окремого польоту значно скоротився.



Рисунрк 3.10 – Випробовування регулятора в симуляції з виходом за межі вимірювання

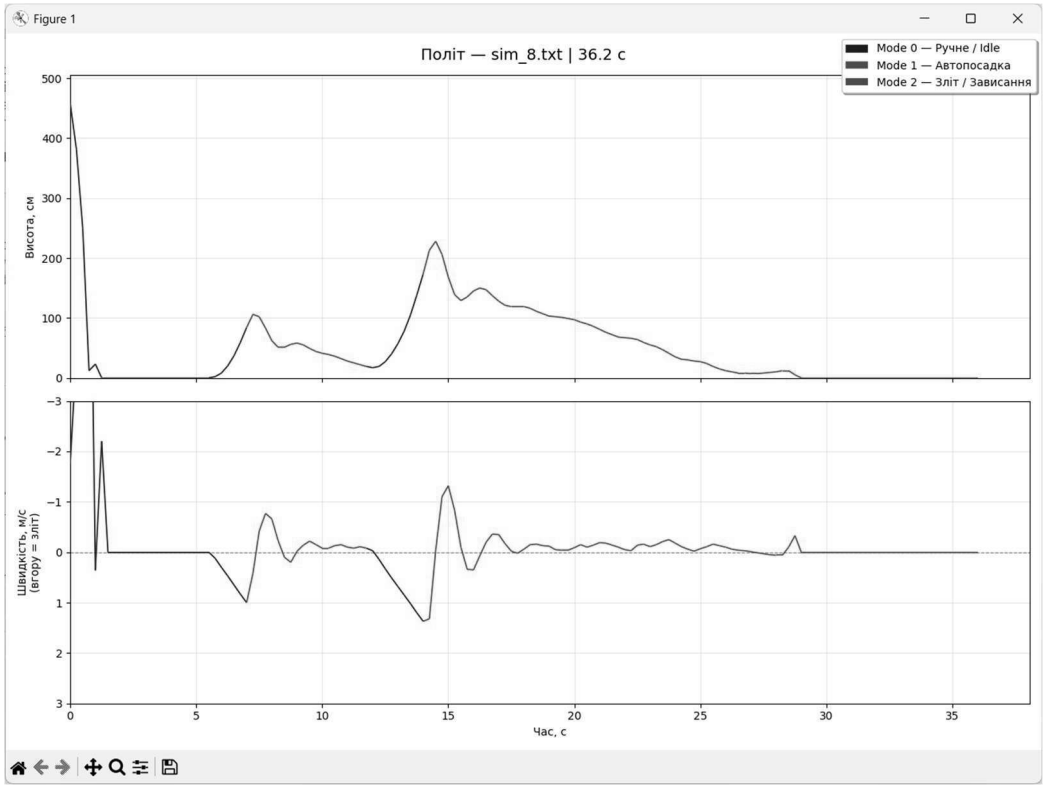


Рисунок 3.11 – Випробовування регулятора в симуляції без виходу за межі вимірювання

Таким чином, розроблений дешифратор логів став завершальним елементом математичного моделювання, що дозволив перейти до реальних випробувань з початково сформованим уявленням про процеси та значно прискорив доведення системи автоматизації до необхідного рівня надійності та стабільності.

Кожен тестовий виліт апарату, є джерелом корисної інформації як про стан датчику так і про поточний стан керуючого впливу та системи загалом. Безпосередньо в програмній частині системи автоматизації планується реалізація блоку запису поточних станів – блок логування. Корисна інформація фіксується на SDкарту, та використовується для подальшого аналізу та налагодження системи.

Висновок до розділу. Розроблене математичне моделювання формує надійне підґрунтя для створення алгоритмів автоматизації зльоту, зависання та посадки FPV-дрона без ризику пошкодження апаратури на етапі

відпрацювання та тестування. Симулятор польотних режимів забезпечив контрольоване віртуальне середовище для численних експериментів, де було підібрано орієнтовні коефіцієнти ПІД-регулятора, як і сам закон регулювання, та перевірено поведінку системи в критичних ситуаціях, різке початкове прискорення, надмалі та надвеликі коефіцієнти впливу, розглянуто умови за яких може виникати дрейф по висоті, та встановлені відправні обмеження в яких має працювати система.

Спрощена математична модель фізичних процесів, що враховує дію сили тяжіння та протидію їй з боку тяги двигунів, виявилася достатньою для узагальненого опису динаміки руху. Запланована подвійна структура ПІД-регулятора з окремими контурами для режима зльоту-зависання та режиму посадки дозволила врахувати специфіку кожного етапу окремо, забезпечивши стабільне утримання висоти та контрольоване зниження зі швидкістю 0,1 м/с – 0,3 м/с.

Дешифратор логів доповнив цей цикл, забезпечивши швидку візуалізацію даних як з симуляції, і так само придатний для обробки записів реальних польотів завдяки плануванню однакової структури файлів логування. Це дозволить оперативно порівнювати теоретичні траєкторії з практичними результатами, підтвердити стабільність роботи датчика ToF в діапазоні до 3 м та усунути небажані коливання по висоті на фінальному етапі посадки.

Отримані результати повністю підтверджують можливість реалізації надійної автоматизації вертикальних маневрів у діапазоні висот 0 м –3 м без значного втручання оператора.

4 ВИБІР АПАРАТНОЇ ТА СЕНСОРНОЇ БАЗИ СИСТЕМИ

Вибір апаратної та сенсорної бази системи було розпочато з аналізу ядра що задовольнить вимогам майбутньої платформи, оскільки саме обчислювальні можливості й апаратні інтерфейси визначають межі динаміки системи при оперативному реагуванні на зміни умов та режимів польоту. Під час вибору мікроконтролера були розглянуті численні варіанти, серед яких мікроконтроллери та одноплатні комп'ютери сімейств: ESP32, Arduino, Raspberry Pi та інші, що могли забезпечити прийнятний баланс між компактністю та продуктивністю. Основна увага приділялася здатності пристрою працювати в реальному часі, гарантуючи обробку потоку даних без суттєвих затримок та перевантаження ресурсів.

4.1.Вибір апаратної платформи

Спираючись на аналіз проведений в розділі 1, було розглянуто варіанти мікроконтролерів для ефективної взаємодії з апаратною та програмною складовими польотних контролерів. Для наочності, дані розглянутих мікроконтролерів зведено в таблицю (таб. 4.1).

Таблиця 4.1 – Прівняння мікроконтролерів

Мікроконтролер	Тактова частота / ядра/ ОЗП / Flash / EEPROM	Напруга живлення та логічних рівнів Споживання (типове)	Габарити (мм)	Ціна (орієнтовно, грн, 2025) Доступність в Україні (2025)
1	2	3	4	5
ESP32-Luna-DevKit	240 МГц/ 2 ядра/ 520 КБ SRAM/	5 В/ 3.3 В/ 80–250 мА	51× 30	~180 Висока

Продовження таблиці 4.1

1	2	3	4	5
	4–16 МБ Flash/ 16 КБ EEPROM			
ESP32 DevKit	240 МГц/ 2 ядра/ 520 КБ SRAM/ 4–16 МБ Flash/ 16 КБ EEPROM	5 В/ 3.3 В/ 80–250 мА	51 × 28	~295 Висока
ESP32-S3	240 МГц/ 2 ядра/ 512 КБ + до 8 МБ PSRAM/ 16 МБ Flash/ 16 КБ EEPROM	5 В/ 5 $В_{tolerant}$3.3 В/ 100–300 мА	52 × 28	~280 Висока
STM32F411 (Black Pill)	168 МГц/ 1 ядро/ 192 КБ SRAM/ 1 МБ Flash/ 4 КБ EEPROM	5 В/ 5 $В_{tolerant}$3.3 В/ 80–150 мА	53 × 23	~290 Середня

Продовження таблиці 4.1

1	2	3	4	5
RP2040 (Raspberry Pi Pico)	133 МГц/ 2 ядра/ 264 КБ SRAM/ 2 МБ Flash/ немає	5 В/ 5 $В_{tolerant}$3.3 В/ 80–150 мА	51 × 21	~200 Висока
Raspberry Pi Zero 2 W	1 ГГц/ 4 ядра/ 512 МБ LPDDR2 / microSD / немає	5 В/ 5 $В_{tolerant}$3.3 В/ 150–350 мА	65 × 30	>900 Середня

Після вивчення характеристик вибір був зроблений на користь ESP32 LUNA DEVKIT, який продемонстрував оптимальне співвідношення функціональності та апаратної гнучкості. Однією із ключових переваг стала наявність двох повноцінних UART-інтерфейсів, реалізованих на апаратному рівні, що дозволило виділити окремі канали зв’язку для приймача радіосигналу та польотного контролера з мінімальними затримками. У вибраній конфігурації кожен UART виконує свою окрему роль: один забезпечує прийом команд від радіоприймача, а інший – передачу на польотний контролер, забезпечуючи можливість модифікації або транзиту сигналів у режимі реального часу. Така конфігурація створила передумови для стабільного функціонування системи.

Однак застосування ESP32 потребувало вирішення питання узгодження логічних рівнів, адже мікроконтролер працює з рівнем логічних сигналів 3 В, тоді як більшість польотних контролерів оперують з сигналами на рівні 5 В. Для коректної взаємодії був застосований конвертор логічних

рівнів, що забезпечив надійну двосторонню комунікацію вузлів обчислення без ризику пошкодження апаратури. Завдяки цьому апаратна платформа отримала можливість безпечно працювати та реалізувати необхідний обмін даними.

Вибір сенсорної частини системи виявився значно тривалішим та вимагав серії практичних експериментів (таб. 4.2).

Таблиця 4.2 – Порівняння датчиків відстані

Датчик	Технологія	Максимальна заявлена дистанція (виробником)	Споживання струму (типове / максимальне)	Ціна орієнтовна (USD, 2025)
VL53L0X	ToF (laser)	2 м	20 мА / 40 мА	2.5–4
VL53L1X	ToF (laser)	4 м	20 мА / 40 мА	4–6
TF-Luna (8 м версія)	ToF (laser)	8 м	85 мА / 140 мА	18–22
TFmini-S	ToF (laser)	12 м	85 мА / 140 мА	38–45
TFmini Plus	ToF (laser)	12 м	85 мА / 140 мА	42–50
Garmin LIDAR-Lite v3	ToF (laser)	40 м	85 мА / 105 мА	120–150
HC-SR04	Ультразвук	4–5 м	15 мА / 20 мА	1–2
JSN-SR04T	Ультразвук	6 м	15 мА / 30 мА	4–6

Перші випробування проводилися з лазерними ToF датчиками VL53L0X і VL53L1X, які спочатку розглядалися як основні вимірювачі висоти. Попри заявлені виробниками характеристики, що передбачали стабільну роботу на кілька метрів, на практиці ці сенсори проявили себе провально. Найкращий із них – VL53L1X на відкритому просторі зміг продемонструвати максимум близько 20 см робочої дистанції, після чого дані просто втрачалися. Висока чутливість до зовнішнього освітлення, особливо

до прямого сонячного світла, практично унеможливила їхнє використання в умовах польоту.

На наступному етапі були протестовані ультразвукові датчики, які добре працювали в лабораторії, але на дроні виявили сильну чутливість до вібрацій та акустичних завад від пропелерів. Це призводило до значного рівня шуму у вимірюваннях та робило їх непридатними для використання в реальному польоті. Ультразвукові сенсори були відхилені як ненадійні.

Лише після численних тестувань та аналізу практичних обмежень, в роботу були включені сенсори типу TF Luna. На відміну від попередніх рішень, вони продемонстрували значно вищу стійкість до зовнішнього освітлення й здатність працювати на реальних польотних дистанціях. Практичні випробування показали, що стабільне та достовірне вимірювання вони забезпечують на висотах до 3,5–4,0 метра, що вже можна розглядати як мінімально необхідну та достатнью для реалізації системи автоматичного контролю висоти, в задачах зльоту та посадки. Саме ці сенсори були прийняті як вимірювальний елемент сенсорної частини системи.

4.2.Визначення та аналіз контрольованих параметрів для автоматизації польоту

У процесі побудови моделі симулятора системи, всі дійові параметри були поділені на контрольовані та такі, що не підлягають прямому керуванню. До контрольованих увійшли величини, на які система здатна впливати безпосередньо через роботу пропелерів, регулюючи їхню швидкість обертання та відповідну їй тягу. Саме ці дві величини утворюють первинний канал управління, через який система здійснює стабілізацію висоти або задає необхідну динаміку вертикального руху.

Висота польоту розглядається як основний вимірюваний параметр, що визначає відстань дрона відносно поверхні землі. Базуючись на цих вимірюваннях система формує регулюючий вплив, коригуючи тягу так, щоб

забезпечити утримання цільової швидкості чи то з метою утримання висоти, чи для контрольованого руху в заданому напрямку по вертикалі, чи то зниження, чи то підйом. Висотні дані доповнюються розрахунками зміни швидкості при вертикальних переміщеннях, що дозволяє не лише фіксувати зміну висоти, а й оцінювати тенденцію руху – чи дрон набирає висоту, чи втрачає її. Таким чином зміна швидкості відіграє роль параметра, що уточнює стан та забезпечує передбачуваність реакції системи.

Ці параметри формують основу автоматизації, оскільки безпосередньо визначають характер вертикальної динаміки дрона. Реакція системи здійснюється за рахунок модифікації сигналів, що передаються на польотний контролер, або шляхом транзиту початкових команд, отриманих від радіоприймача без змін. Такий підхід дозволяє модулю працювати як проміжна ланка, здатна виконувати роль адаптивного регулятора, що постійно відстежує фактичний стан дрона, формує керуючі впливи в режимі реального часу та готовий прийняти керування на себе.

Усі інші параметри польоту – такі як бічне зміщення, кут нахилу чи вплив зовнішніх факторів – у межах цієї частини системи розглядаються як неконтрольовані. Вони можуть впливати на поведінку дрона, однак не є безпосереднім об’єктом регуляції, оскільки основною задачею системи є саме управління вертикальною динамікою.

Висновок до розділу. На поточному етапі реалізації, стабілізація дрона в горизонтальній площині повністю покладена на штатний функціонал польотного контролера. Під час періодів регулювання висоти модуль примусово встановлює режим горизонту, перекладаючи на польотний контролер задачу з балансування та демпфування горизонтальних коливань корпусу. Завдяки цьому власна логіка регулятора зосереджується виключно на вертикальній динаміці. Таким чином система уникає конфлікту між регуляторами, зберігає узгодженість сигналів і забезпечує плавну поведінку дрона під час посадки чи утримання висоти, та забезпечує максимально адекватну реакцію у змінних умовах польоту та під дією помірних обурень (як

то примусовий нахил корпусу в обмеженому діапазоні, за необхідності оператора виконувати коригування позиції в горизонтальній площині). Водночас розмежування ролей між модулем і польотним контролером гарантує стабільність і безпеку роботи та незалежність складових системи в складних умовах.

5 ОПИС АЛГОРИТМУ АВТОМАТИЗАЦІЇ ЗЛЬОТУ, ЗАВИСАННЯ ТА ПОСАДКИ

Далі надано узагальнену схему, що дає уявлення про структуру реалізованого алгоритму на рівні системи вцілому (рис. 5.1). Алгоритм є центральною частиною програмної системи автоматизації. Він визначає поточний стан роботи, логіку переходів між режимами та порядок дій під час зльоту, зависання або посадки. Архітектура алгоритму побудована таким чином, щоб забезпечити стабільну роботу на наявному обладнанні з урахуванням можливих сценаріїв сбоїв помилок та невизначеностей, що можуть виникнути під час роботи програми керування та передбачити можливість розширення функціоналу в майбутньому.

Така відправна точка забезпечує можливість покрокового втілення системи з утриманням фокусу на загальній структурі, знижує ризик хибних рішень, та пришвидшує розробку вцілому.

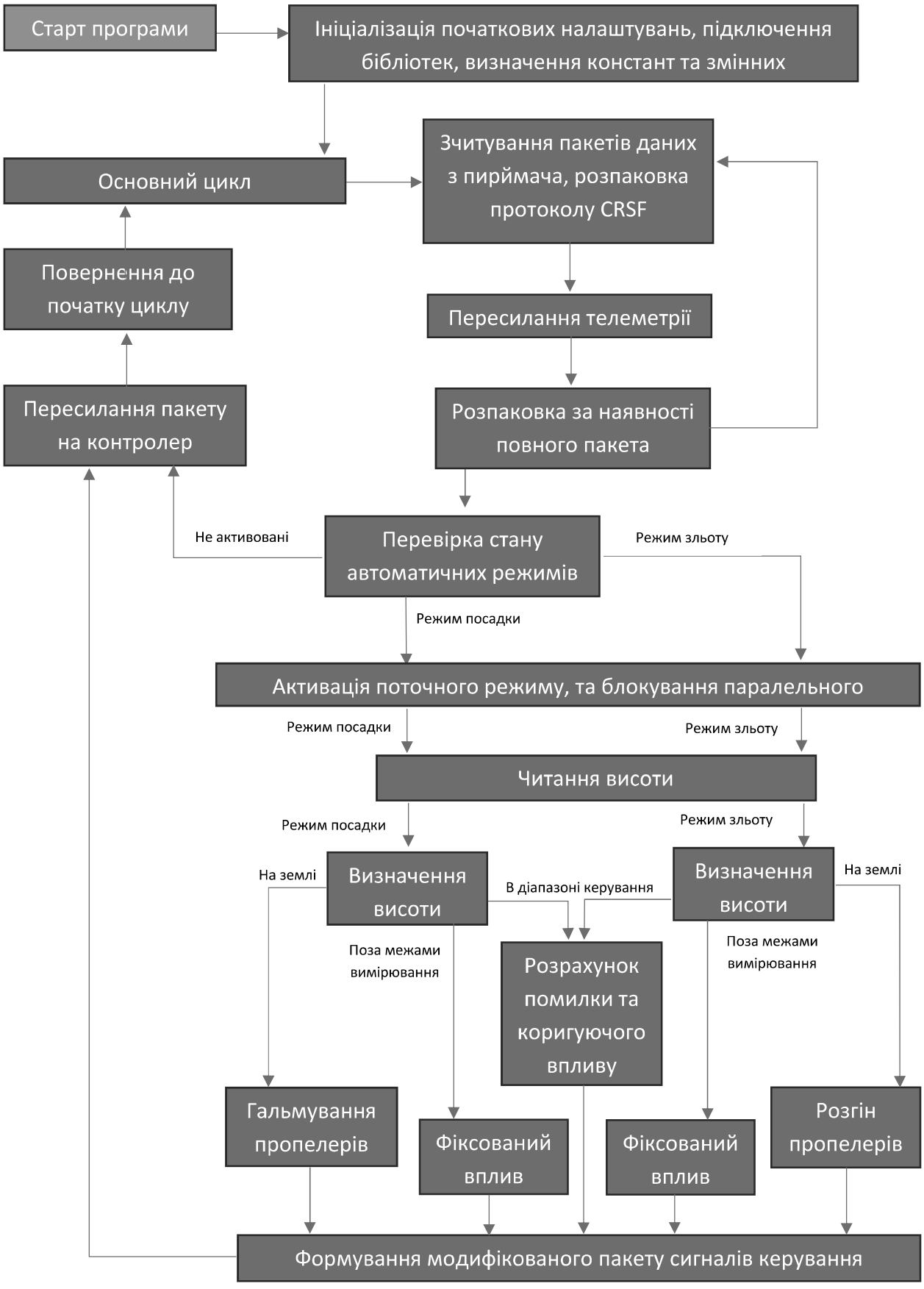


Рисунок 5.1 – Узагальнена схема алгоритму керування

5.1.Алгоритм роботи системи автоматизації

Система автоматизації працює у межах трьох режимів, що визначаються положенням трипозиційного тумблера на пульті оператора. Нульова позиція забезпечує режим транзиту: усі сигнали з пульта передаються без змін прямо у політний контролер. Перша позиція це режим автоматичного зльоту та утримання заданої початкової висоти. Друга позиція – режим автоматичної посадки в межах вимірювання датчику.

Перехід між режимами можливий лише послідовно, відповідно до механіки тумблера.

Активація автоматичних функцій відбувається в кожному з активних режимів та передбачає вихід зі стану транзиту та фіксацію цільової поведінки (зльоту або посадки).

Розгін двигунів при зльоті та їх гальмування при посадці виконуються за умови положення апарату на поверхні землі.

Встановлення примусових обмежень з метою стабілізації горизонтальної площини та отримання вимірювань по вертикалі з мінімальними відхиленнями. При цьому система переводить польотний контролер у режим стабілізації горизонту та обмежує канали нахилу по всіх осях створюючи умови для безпечної роботи алгоритму.

Вимірювання висоти забезпечує приймання даних від далекоміра, перевірка на валідність не проводится, щоб не втрачати дані на високих швидкостях.

ПІД-регулювання формує керуючий вплив відповідно до поточної похибки відносно заданого значення.

Оновлення керуючого впливу забезпечує транзит керуючих сигналів з пульта або часткову їх підміну відповідно до розрахунку керуючого впливу, в залежності від режиму роботи модуля.

Деактивація тумблером повертає апарат під повний контроль оператора, переводячи систему до режиму транзиту сигналів з пульта до польотного контролера.

Логування проводиться на всьому етапі роботи з моменту подачі живлення до його відключення. Є необхідною складовою на етапі розробки та налагодження модуля.

5.2.Режими роботи: транзит або підміна сигналів керування

Для модуля створено наступні режими роботи: транзит або підміна сигналів керування при зльоті чи посадці. Режими роботи визначаються поточним станом керуючого 3х позиційного тумблера.

Нульове положення відповідає режиму транзиту сигналів. Усі чотири канали керування обертання по осях, включно з дроселем, передаються напряму в політний контролер без будь-яких змін. Система автоматизації не втручається – оператор несе повну відповідальність за політ та меневри.

Положення керуючого тумблера в позиції один відповідає режиму зльоту до заданої висоти 0.8 м за замовчуванням, та з утриманням поточної висоти. При цьому, система примусово встановлює режим стабілізації горизонту в польотному контролері, накладає обмеження на канали нахилу, дозволяючи оператору лише помірні корекції по осях, для компенсації горизонтального дрейфу та повністю підміняє канал дроселю розрахованим керуючим впливом. Цільова висота встановлена за замовчуванням на рівні 0.8 м, але алгоритм коректно працює при активації на будь-якій висоті в межах вимірювання датчика.

Положення керуючого тумблера в позиції 2 відповідає режиму посадки. Система так само обмежує керування та бере на себе повний контроль над вертикальним каналом. Оберти двигунів підбираються так, щоб забезпечити кероване зниження. Шляхом інтерполяції в заданому діапазоні швидкостей відповідно до поточної висоти, алгоритм коригує швидкість відповідно до

мапи висот. В безпосередній близькості до землі вимикає регулювання й переходить у режим плавного зменшення обертів до мінімальних значень.

5.3.Алгоритм розгону двигунів, зльоту та досягнення заданої висоти

Режим зльоту може бути активований на будь-якій висоті в межах вимірювання датчика. Задача алгоритму привести положення дрона до заданої висоти незалежно від поточної позиції. При визначенні стабільного положення на землі перед переходом безпосередньо до регулювання, починається попередня плавна процедура розгону гвинтів до того моменту, поки датчик не побачить збільшення висоти. З цього моменту, керування передається до процедури формування керуючого впливу, що використовує ПІД-регулювання задля досягнення заданої висоти. Після того як висоту було досягнуто, функція продовжує оперувати керуючим впливом задля утримання висоти з припустимим відхиленням в межах ± 3 см. В цьому режимі модуль повністю підміняє в першу чергу режим польоту ПК, примусово встановлюючи його як утримання горизонту, задля забезпечення вертикалі при вимірюваннях. Повністю підміняє сигнал дроселю що надходить з пульта на розрахунковий керуючий вплив, що має забезпечити питоме положення за висотою. Також накладаються обмеження на канали нахилу за усіма осями, що забезпечує оператору можливість помірного втручання в горизонтальне переміщення, задля компенсації горизонтального дрейфу за вітром, або переміщення в необхідну позицію по горизонталі.

5.4.Алгоритм посадки та уповільнення двигунів

Алгоритм посадки передбачає контроль швидкості зниження відповідно до мапи висот в діапазоні вимірювання датчика висоти. Його функція – виконати плавну посадку з початкової висоти увімкнення режиму із цільовим значенням швидкості 0.1 м/с поблизу поверхні землі. За умови ввімкнення

поза межами вимірювання, система встановлює оберти, що дозволяють знижуватись з тягою трохи менше ніж тяга зависання. таким чином мінімізуючи прискорення в неконтрольованій зоні. З моменту отримання валідних даних, система починає працювати за законом регулювання. По досягненню висоти що відповідає максимальному наближенню до поверхні землі, закон регулювання вимикається та починається плавне зниження обертів двигунів до мінімального значення активного режиму.

5.5.Обробка даних датчика відстані та протоколу Crossfire для управління

Алгоритм обробки даних вживлено до безперервного циклу зчитування пакетів протоколу CRSF з виходу радіоприймача. На кожному кроці система очікує заголовок пакета з адресою модуля 0xC8, зчитує довжину та тип корисних даних «payload». За умови що пакет має відповідний тип каналів, виконується розрахунок і перевірка контрольної суми CRC8, щоб пересвідчитися, що дані отримані повністю. У разі успішної верифікації пакет розпаковується і значення 16 каналів витягуються з бітового поля за стандартним порядком протоколу, з подальшим приведенням до діапазону 989 мкс – 2000 мкс для зручності обробки даних в основному розрахунковому форматі.

Паралельно з обробкою пакетів, у тому ж циклі маэ виконуватися запит даних з датчика ToF. Отримане значення висоти перевіряється на валідність. якщо воно дорівнює 0, що відповідає висоті понад 4м, або менше 3 см, дані мають вважатися невалідними. При цьому фіксується стан «вихід за межі вимірювання». Відповідно до алгоритмічних залежностей, має бути встановлений фіксований керуючий вплив на рівні 1300 мкс, що забезпечує перехід до режиму безпечного спуску. Якщо висота валідна, розраховується вертикальна швидкість.

Далі алгоритм визначає активний режим за значенням обумовленого каналу пульта. У режимі транзиту пакет даних протоколу передається на польотний контролер без змін. В автоматичних режимах виконується підміна значення керуючого впливу за результатами ПІД-регулятора; каналів, що відповідають за нахил по осях дозволений діапазон керування обмежуються 1400 мкс – 1600 мкс для запобігання критичному викревленню траекторії вимірювання.

Після модифікації канали повторно упаковуються в корисне навантаження (payload), розраховується новий CRC8, і пакет відправляється на вхід польотного контролера. Телметрія з контролера передається назад до приймача без затримок.

Така послідовність забезпечить синхронну інтеграцію даних датчика в процес підміни сигналів протоколу, з мінімальною затримкою та ізоляцією помилок.

Висновок до розділу. Запропоновані алгоритми автоматизації зльоту, зависання та посадки, з урахуванням послідовної обробки даних датчика та протоколу CRSF, мають забеєпечити надійну та безпечну роботу модуля в реальних умовах. Вони зберігають модульність рішення, мінімізують затримки та гарантують ізоляцію помилок, дозволяючи оператору проводити коригування в горизонтальній площині. Практична реалізація цих алгоритмів на прототипі, включаючи апаратну інтеграцію та польові випробування, буде розглнута в наступному розділі.

6 ПРАКТИЧНА РЕАЛІЗАЦІЯ МОДУЛЯ ТА ВИПРОБУВАННЯ

Розділ висвічує методи реалізації розробленого модуля автоматизації функцій керування FPV-дроном, включаючи опис апаратної частини та конструктивних рішень, електричної схеми підключення та оптимізації розташування елементів, програмного забезпечення та результатів польових випробувань. На основі теоретичних обґрунтувань і математичного моделювання, викладених у попередніх розділах, було створено робочий прототип на базі мікроконтролера та датчика ToF. Реалізація забезпечує модульність, сумісність через протокол Crossfire та ізоляцію помилок, дозволяючи інтегрувати модуль у різні кастомні збірки без зміни прошивки польотного контролера. Випробування проводилися в діапазоні висот 0,1 м – 4.5 м, з виходом за межі вимірювання, та з акцентом на стабільність алгоритмів зльоту, зависання та посадки, а також на економічну доцільність рішення для зниження навантаження на оператора та підвищення живучості дрона в умовах логістики, моніторингу та відеозйомки.

6.1.Опис апаратної частини модуля автоматизації функцій

Апаратна частина модуля автоматизації збудована на мікроконтролері сімейства ESP32, який забезпечує достатню обчислювальну продуктивність для локальної обробки даних та реалізації алгоритмів автозльоту, зависання та автоматичної посадки.

Модуль включає: лазерний далекомір що працює за принципом ToF, підключений через інтерфейс I^2C, та виконує вимірювання висоти з частотою 50 Гц; конвертер логічних рівнів; мікроконтролер ESP32-Luna Dev Kit; модуль SD-карти.

Для забезпечення апаратної стабільності використовується живлення з окремого виходу польотного контролера, яке ізолює мікроконтролер і сенсорну частину від шумів силової шини двигунів.

Два ядра в архітектурі ESP32-Luna Dev Kit, дозволяють рознести обчислення: основний цикл керування виконується на першому ядрі, а логування перенесено на друге. У поточній реалізації більшість функцій виконується в межах одного ядра, що спрощує структуру, та залишає резерв для подальших оптимізацій.

Значну роль в апаратній реалізації відіграють елементи, виготовлені за допомогою 3D-друку, які забезпечують модульність, легкість та повну сумісність прототипу до рами FPV-дрона [12]. Зокрема, розроблено спеціальні кріплення для стійок (рис. 6.1), що виконують функцію посадкових опор. Ці кріплення являють собою компактні манжети з внутрішнім діаметром 16 мм, які фіксуються безпосередньо під променями рами FPV-дрона на тих самих болтах, що утримують двигуни. Виготовлені суцільним друком з матеріалу PETG, вони володіють високою міцністю та стійкістю до ударів. У манжети вставляються відрізки металопластикової труби однакової довжини в діапазоні 10 см –15 см, які слугують опорами з високим ступенем пружності.



Рисунок 6.1 – Кріплення опорних стійок

Таке рішення забезпечує необхідну висоту для датчика над поверхнею в стані спокою на землі (рис. 6.2), дозволяючи уникнути хибних вимірювань через випадкове або нестабільне розташування апарату на посадкових

поверхнях. Металопластикові труби додають конструкції пружності, ефективно амортизуючи удари під час посадки чи падінь, при цьому легко замінюються як доступний витратний матеріал у разі критичних ушкоджень. Кріплення ж залишаються неушкодженими навіть при значних дефрмаціях посадкових опор, що підтверджено практичними тестами.



Рисунок 6.2 – Зльотні стійки забезпечують сталу висоту датчика

Окремо розроблено корпус для розміщення всього модуля, який фіксується в передній частині рами FPV-дрона (рис. 6.3). Цей контейнер горизонтальної орієнтації слугує обрамленням для плати мікроконтролера, що встановлюється всередину. У корпусі передбачено спеціальне сідло для конвертера логічних рівнів, яке забезпечує мінімальну довжину з'єднувальних проводів між мікроконтролером та конвертером. Фіксація корпусу здійснюється за допомогою другої пластикової пластини, на якій монтується датчик (рис. 6.4). Обидві частини корпусу охоплюють нижню площину рами дрона стискаючи її, забезпечуючи жорстке та точне позиціонування датчика в площині основи рами. Кріплення датчика реалізовано на болтах, різьба яких нарізається безпосередньо в пластику корпусу. Завдяки високому коефіцієнту тертя та адгезії в матеріалі PETG. Таке з'єднання працює аналогічно

фіксуючим гайкам з нейлоновим кільцем, не вимагаючи додаткових елементів і забезпечуючи надійну фіксацію без люфтів.

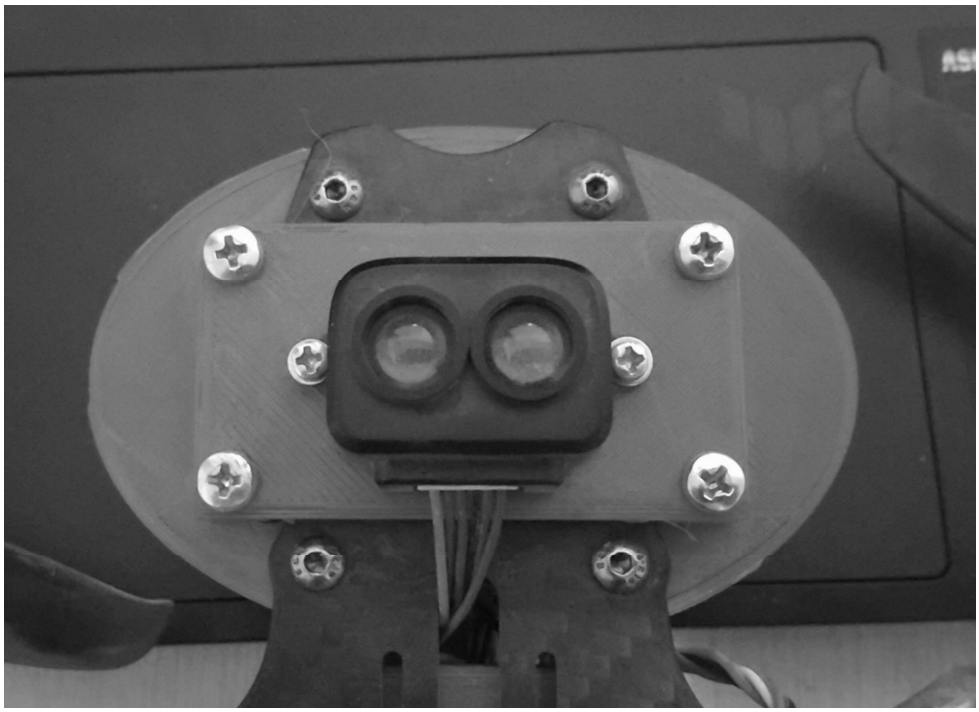


Рисунок 6.3 – Крпус мікроконтролера

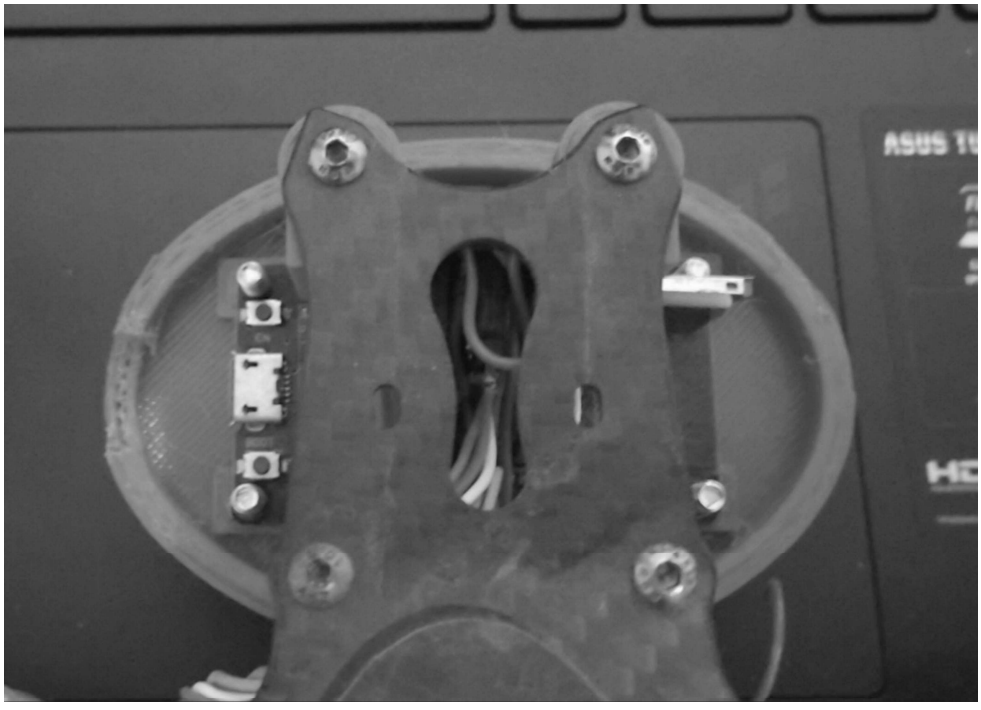


Рисунок 6.4 – Встановлення датчика відстані

Загалом, використання 3D-друку для цих елементів дозволяє швидко ітеративно вдосконалювати конструкцію, мінімізувати вагу модуля та забезпечити його сумісність з різноманітними збірками FPV-дронів у повній відповідності до рами.

6.2.Електрична схема підключення компонентів системи

Електрична схема модуля побудована за принципом мінімальної кількості зв’язків і максимальної зручності підключення. Пайка виконується при встановленні електронних компонентів в корпус, але при цьому існує можливість вільно їх виймати та встановлювати знов на свої місця. Схема зєднань показана на (рис. 6.5) Конструкція побудована так, щоб інтегруватися в ланцюг між радіоприймачем і польотним контролером без пайки на основних платах дрона. Завдяки цьому підключення відповідає концепції plug-and-play, що особливо важливо для кастомних FPV-збірок.

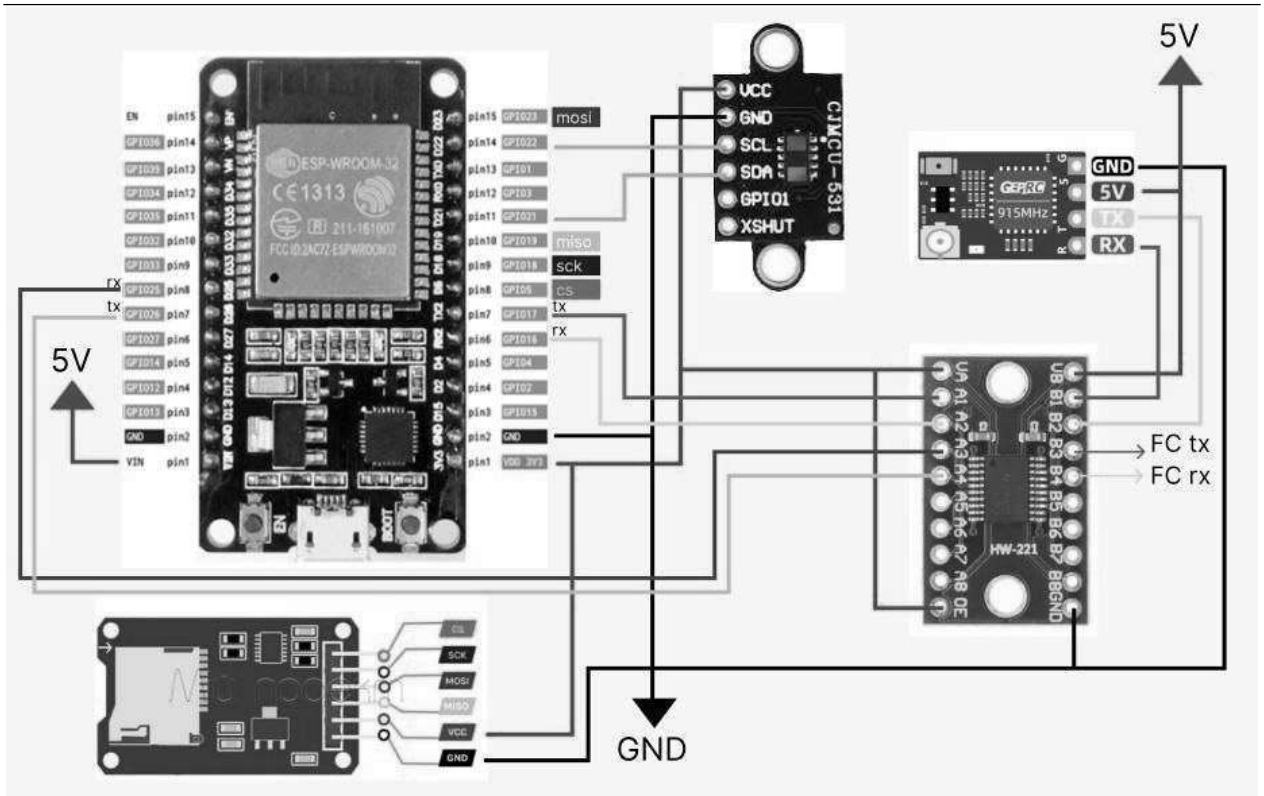


Рисунок 6.5 – Принципова електрична схема модуля

Центральним елементом схеми є проміжна плата з’єднань (рис. 6.6).

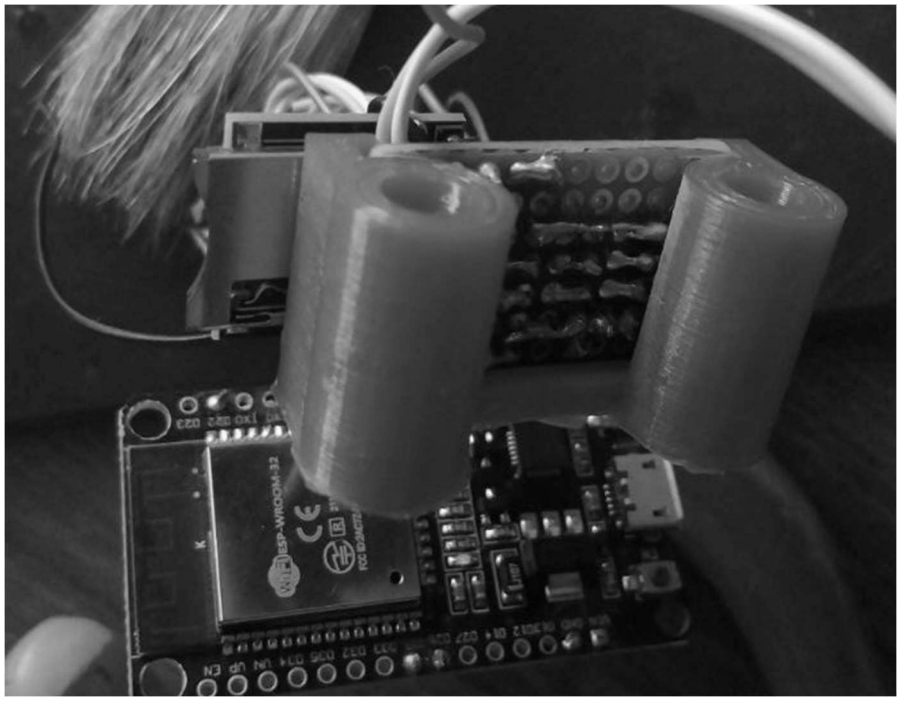


Рисунок 6.6 – Плата електричних з’єднань

Вона монтується в спеціальний тримач корпусу, виконаний методом 3D-друку, і виконує три основні функції: розподіл живлення, зведення всіх сигналів до роз’ємів та надійну механічну фіксацію дротів.

Плата формує три незалежні лінії, що забезпечують живлення: землю (GND), лінію +5 В та лінію +3,3 В. Живлення береться з польотного контролера через один стандартний роз’єм +5 В та GND, лінія поєднань +3.3 В формується після відповідного виводу плати мікроконтролера.

Усі плати ESP32 забезпечують необхідний для роботи мікроконтролера рівень живлення +3,3 В, завдяки вбудованому лінійному регулятору напруги Low-Dropout Regulator (LDO). Лінійний регулятор знижує напругу використовуючи прохідний транзистор, що діє як автоматично регульований резистор, що включений послідовно в лінію живлення. Регулятор коригує вихідну напругу, спираючись на ланцюжок зворотного зв’язку, та коригуючи опір транзистора за принципом звичайного резистивного дільника. При цьому надлишкові вольти поглинаються транзистором та перетворюються на тепло.

Створена гілка +3.3 В застосовується для живлення датчику, SD-карти та для створення опорної напруги конвертора логічних рівнів.

Така організація повністю ізолює сенсорну частину модуля від шумів силової шини двигунів. Жорстка фіксація всіх провідників на платі виключає рухомі контакти, що могли б замкнутися під час польоту. Для додаткового захисту плата покрита тонким шаром захисного діелектричного лаку.

Підключення датчика ToF приведено до окремого роз'єму на проміжній платі, що дозволяє легко від'єднати його для перевірки або заміни. Оскільки датчик формує вихідні сигнали на логічному рівні 3,3 В, конвертер рівнів не потрібен — це спрощує схему і зменшує зав'язків. Частота оновлення даних встановлена 50 Гц (період 200 мс), що забезпечує достатню точність для автоматичної посадки та зльоту без перевантаження шини I^2C.

Радіоприймач підключений до проміжної плати через стандартний штирьовий роз'єм. Сигнали TX і RX проходять через конвертер логічних рівнів і надходять на апаратний UART1 мікроконтролера. Польотний контролер підключений аналогічно до другого роз'єму на тій самій платі. Після обробки та узгодження рівнів, сигнали передаються на апаратний UART2. Використання двох незалежних апаратних UART гарантує одночасну роботу з приймачем і польотним контролером без затримок.

Модуль SD-карти змонтований безпосередньо на проміжній платі, що зменшує кількість дротових поєднань і підвищує жорсткість встановлення та забезпечує надійність логування. Всі роз'ємні з'єднання виконані через штирові конектори типу Dupont, які легко вставляються і витримують вібрацію (рис. 6.7).



Рисунок 6.7 – Роз’єми підключення польотного контролера та радіоприймача на проміжній платі

Така схема дозволяє підключити модуль до будь-якого FPV-дрона з протоколом CRSF буквально за кілька операцій: розріз лінії зав’язків між приймачем та польотним контролером, встановлення конекторів та доєднання їх до плати з’єднань модуля автоматизації функцій. При цьому модуль не впливає на основну електричну схему дрона і може бути швидко демонтований для тестування або перенесення на іншу збірку.

6.3.Програмна реалізація: код для ESP32

Програмна частина модуля розроблена в середовищі Arduino IDE, яке забезпечує зручний доступ до апаратних інтерфейсів ESP32 та швидку ітерацію коду під час тестування. Код побудований сегментами з використанням окремих функцій для кожної ключової операції, що спрощує налагодження, модифікацію та подальший аналіз. Загальний цикл програми організований навколо обробки пакетів протоколу CRSF, зчитування даних сенсора та умовного втручання в керуючі сигнали залежно від активного режиму.

Основою взаємодії з польотним контролером є функції роботи з протоколом CRSF. Програма виконує повний цикл обробки пакетів: зчитування байтового потоку з UART приймача, перевірку адреси та довжини пакета, контроль CRC. Розпаковку каналів реалізовано через функцію «**unpackChannels()**». Модифікацію окремих каналів при роботі з автоматичними режимами, наприклад, підміна значення керуючого впливу виконує основна функція «**updateThrottleFromDistance()**», обмеження обертання по осях, в своє чергу забезпечує функція «**LimitChannels()**». Зворотну упаковку даних у вихідний пакет «**crsfPrepareChannelsPacket ()**» (рис. 6.8) та відправку на польотний контролер здійснює «**sendCRSF_2_FC()**». Такий підхід гарантує наскрізну передачу всіх каналів у режимі транзиту без затримок, а в автоматичних режимах дозволяє точно втручатися тільки в потрібні параметри, не порушуючи решту сигналів з пульта. Це забезпечує повну сумісність з будь-якими конфігураціями польотних контролерів прошитих за допомогою Betaflight та iNav без зміни їх прошивки.

```
void crsfPrepareChannelsPacket(uint8_t* packet)
{
  uint16_t mapped_channels[16];
  for (uint8_t i = 0; i < 16; i++) {
    mapped_channels[i] = map(channels[i], 989, 2000,
CRSF_CHANNEL_MIN, CRSF_CHANNEL_MAX);
  }
  uint8_t payload[CRSF_TYPE_CHANNELS_PAYLOAD_LENGTH];
  payload[0] = (uint8_t)(mapped_channels[0] & 0x07FF);
  payload[1] = (uint8_t)((mapped_channels[0] & 0x07FF) >> 8 |
(mapped_channels[1] & 0x07FF) << 3);
  payload[2] = (uint8_t)((mapped_channels[1] & 0x07FF) >> 5 |
(mapped_channels[2] & 0x07FF) << 6);
  payload[3] = (uint8_t)((mapped_channels[2] & 0x07FF) >> 2);
  payload[4] = (uint8_t)((mapped_channels[2] & 0x07FF) >> 10 |
(mapped_channels[3] & 0x07FF) << 1);
```

```
  payload[5] = (uint8_t)((mapped_channels[3] & 0x07FF) >> 7 |
(mapped_channels[4] & 0x07FF) << 4);
  payload[6] = (uint8_t)((mapped_channels[4] & 0x07FF) >> 4 |
(mapped_channels[5] & 0x07FF) << 7);
  …
```

Рисунок. 6.8 – Фрагмент функції розпаковки та упаковки CRSF-пакетів

Функція зчитування датчика «**ReadTFluna()**» виконує періодичні запити через I²C з частотою 50 Гц або періодом 20 мс. При втраті сигналу або виході за межі вимірювання, повертається значення 0, що автоматично переводить систему в безпечний режим з фіксованим керуючим впливом. При висоті нижче 18 см –20 см активується плавне зниження обертів двигуна до мінімальних.

Функція визначення та перемикання режимів роботи «**checkAutoland()**» аналізує значення каналу відповідного перемикача пульта. При значеннях, що відповідають діапазону активації, вона забезпечує перехід у відповідний режим з ініціалізацією всіх змінних (скидання інтегральної помилки ПІД, фіксація початкової висоти, обмеження каналів обертання по осях, для запобігання критичному збільшенню дистанції вимірів за гіпотенузою. При деактивації режиму всі змінені канали миттєво повертаються до значень з пульта оператора викликом функції «**restoreValues()**».

Обчислення керуючого впливу виконується в функцією «**updateThrottleFromDistance()**» за законом ПІД-регулювання по швидкості вертикального переміщення. Цільова швидкість залежить від поточної висоти та режиму (0.1–0.3 м/с для посадки, фіксована для зависання). Помилка розраховується як різниця між виміряною та цільовою швидкістю, після чого формуються пропорційна, інтегральна та диференціальна складові. Отримане коригування накладається на поточне значення throttle з жорсткими обмеженнями 1000–1800 мкс. Окремі функції «**RampUp**» та «**RampDown**» забезпечують плавний розгін та гальмування відповідно пропелерів на землі,

уникаючи різких ривків, повторних зльотів або нездатності відірватися від поверхні злітного майданчика.

Логування реалізовано з періодом 250 мс і фіксує повний набір параметрів для подальшого аналізу: поточна висота, вертикальна швидкість, цільова швидкість, коригування керуючого впливу, значення P-I-D складових, стан поточного режиму, кількість оброблених пакетів за період. Дані записуються у текстовий файл на SD-карту у форматі, відповідно до сформованого в розділі 3.4 «Розробка дешифратора логів для аналізу польотних даних» . Це дозволяє точно відтворювати траєкторії польотів, аналізувати осциляції та оптимізувати коефіцієнти ПІД на основі реальних логів (рис. 6.9, рис. 6.10, рис. 6.11).

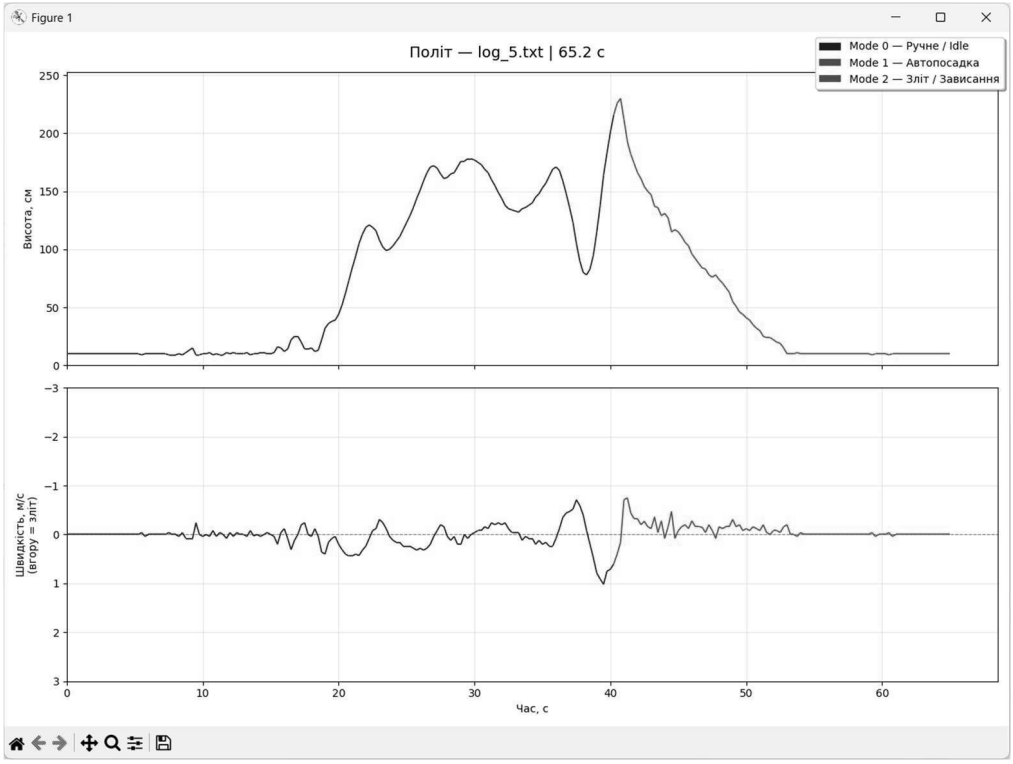


Рисунок 6.9 – Активація автоматичної посадки на висоті понад 2 метри

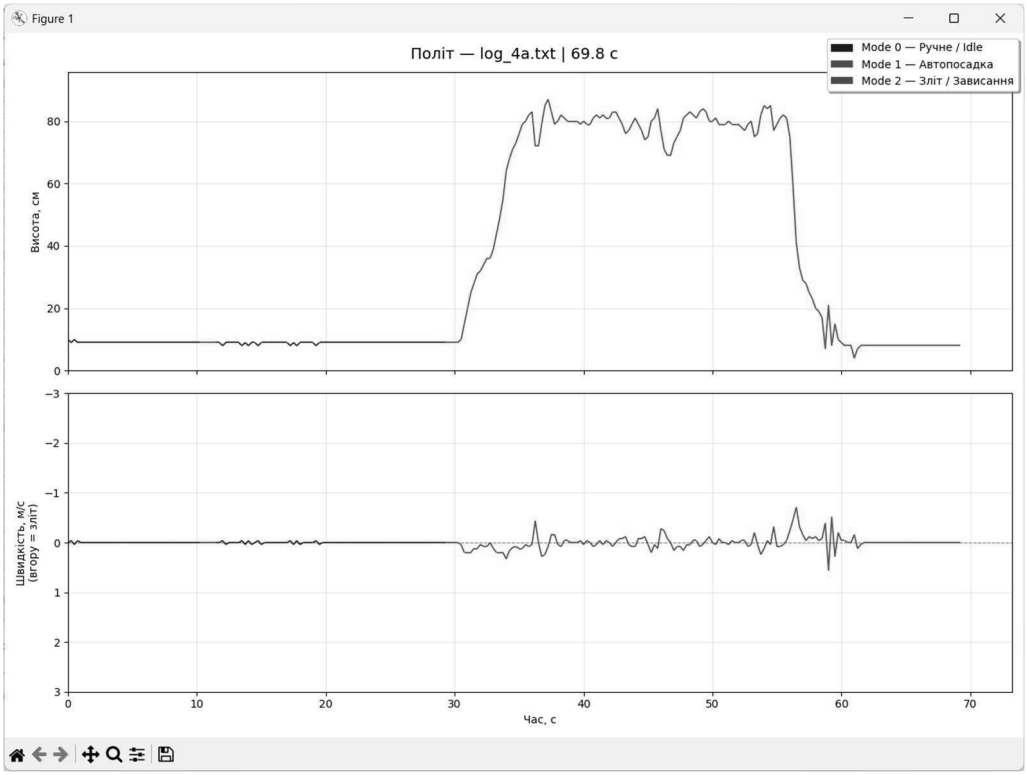


Рисунок. 6.10 – Активація автоматичного зльоту з подальшою автоматичною посадкою

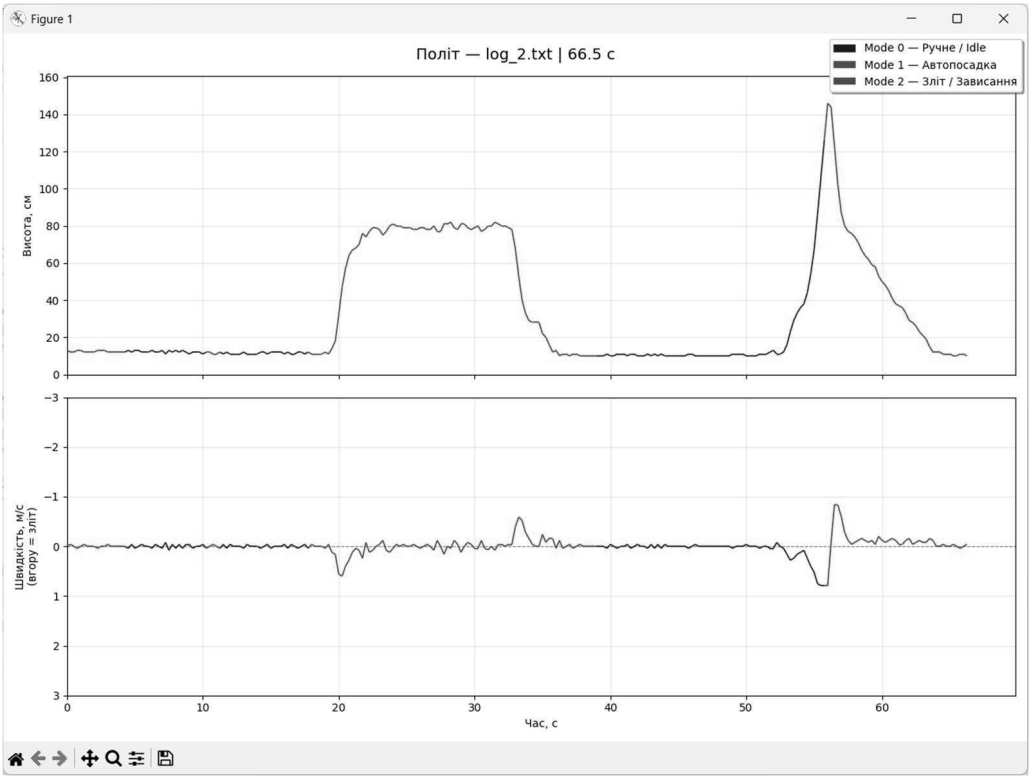


Рисунок 6.11 – Контрольний лог в умовах з мінімальним обуренням

Розроблена програмна структура повністю відповідає принципам модульності та ізоляції помилок, заявленим у розділах 1.4 та 5. Втручання в керування відбувається тільки за явної команди оператора, при будь-яких збоях система миттєво повертається в режим прозорого транзиту сигналів. Використання стандартного Arduino-середовища та бібліотек для CRSF та TF-Luna забезпечує легкість перенесення модуля на інші платформи ESP32 та швидку адаптацію під конкретні збірки дронів.

Для контролю версій коду та захисту від втрати даних використано локальну систему Git (рис. 6.11). Кожен етап розробки, від початкової реалізації транзиту CRSF-пакетів до інтеграції ПІД-регулювання та функцій ramp** фіксувався окремим комітом з описом змін. Це дозволило легко відстежувати еволюцію програми, повертатися до стабільних версій у разі конфліктів чи помилок, а також порівнювати ефективність різних налаштувань коефіцієнтів або алгоритмів обробки сенсора. Використання Git значно спростило ітеративну розробку, мінімізувало ризики втрати прогресу та забезпечило можливість швидкого відновлення після експериментальних змін.

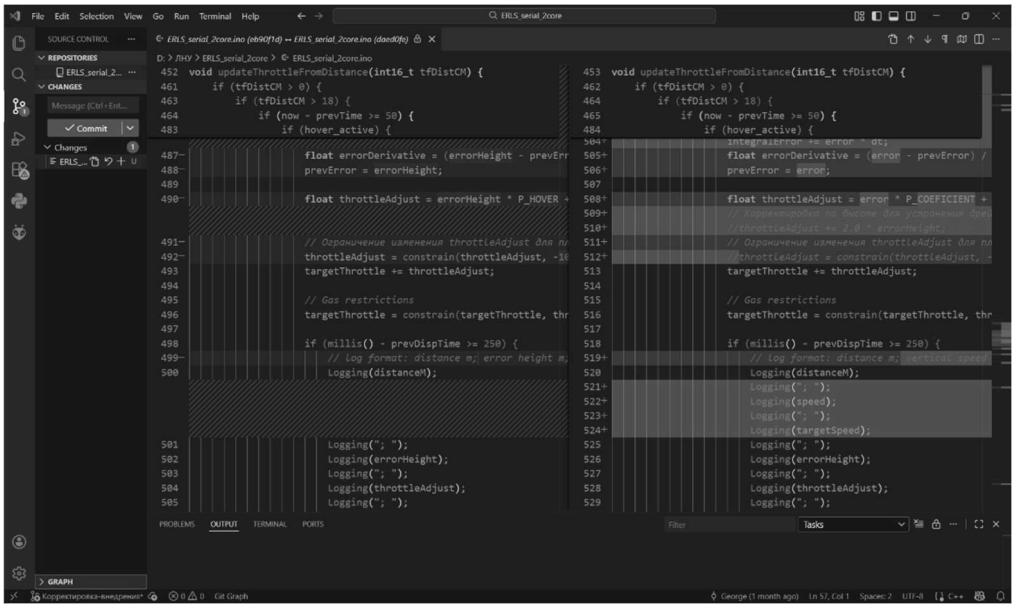


Рисунок. 6.11 – Інтерфейс локального репозиторію Git

6.4.Планування та результати польових випробувань

Метою польових випробувань було остаточне підтвердження працездатності модуля в реальних умовах експлуатації FPV-дрона, перевірка стабільності алгоритмів автоматичної посадки та зависання, а також отримання даних для фінального налаштування ПІД-регулятора, що були розглянуті в розділах 5 та 3 відповідно, цієї роботи. Вже перші спроби автоматичної посадки виявили значний дрейф через вітер, зміщення центру мас та активну роботу польотного контролера по утриманню горизонтального положення корпусу, що також додає деяку складову горизонтального дрейфу. Тому, особлива увага приділялася збереженню можливості обмеженого втручання оператора в коригування по горизонтальній площині по відповідних осях керування.

Робочий діапазон висот випробувань було обмежено 0,2 м – 3,0 м через характеристики датчика та міркування безпеки. Температура повітря під час тестів становила +10…+25 °C, швидкість вітру не перевищувала 5 м/с. Випробування проводилися на відкритому майданчику при різній освітленості що охоплювала діапазон від яскравого сонця до сутінків, та в приміщенні наприкінці циклу для аналізу поведінки з максимальною ізоляцією від зовнішніх обурень.

При випробуваннях застосовані наступні заходи безпеки. Перед будь-яким польотом FPV-дрон транспортувався без пропелерів, увімкнення завжди відбувалося в режимі «disarm» – повне відключення двигунів. Оператор обов’язково використовував захисні окуляри, розташовувався з підвітряного боку, мав аварійний вимикач живлення «disarm», та спрямовував основний напрямок руху FPV-дрона «від себе». Перші серії автоматичних посадок виконувалися на висоті не більше 1 м. Оскільки м’яка поверхня не характерна для природніх ландшафтів, покладалися на пружність 3D-друкованих стійок та демпфуючі властивості карбонової рами – саме ті чинники що мають убезпечувати посадку в реальних умовах.

Підготовка випробувань базувалася на фіксованій конфігурації 7-дюймового квадрокоптера масою 1225 г з модулем та штатною батареєю живлення. З метою створення додаткового обурення з боку нестабільної маси, було застосовано батареї різних формфакторів та різної ваги, що спричиняло варіативність розташування центру мас та самої загальної маси. Також, як додатковий обурюючий фактор випробування проводилися до повного розряду батареї, щоб перевірити вплив цього чинника на здатність системи до саморегулювання. Проводився обов'язковий запис вимірюваних та розраховуваних параметрів польоту з періодом 250 мс на SD-карту. Після кожної серії проводився аналіз та подальше калібрування ПІД-коефіцієнтів. Кожна серія режиму Autoland містила не менше 10 повторів, які могли включати різний початковий пропорційний коефіцієнт, відповідно до налаштувань встановлених оператором.

Відповідно до матеріалу розгляду п. 4.3 поточної роботи, попередні лабораторні тести відсіяли ультразвукові датчики типу HC-SR04 та ToF-далекоміри VL53L0X, VL53L1X. За підсумками залишився лише TF-Luna. Стабільна зона роботи становить 3–350 см. Нижче 3 см датчик видає хаотичні низькі значення, що неможливо інтерпретувати як надійні, тому вони розглядаються як втрата сигналу поза зоною вимірювання і система переводиться на фіксований керуючий вплив 1300 мкс. Обмеження мінімальних висот регулювання 18 см – 20 см встановлено для переходу до режиму RampDown, або початок з RampUp. Такий діапазон обрано через довжину стійок і початкову висоту датчика близько 12 см від поверхні майданчика, з урахуванням можливих нерівностей поверхні в зоні опори та вимірювань. Функції розгону та гальмування мають працювати лише в безпосередній близькості до поверхні землі.

Випробування режиму Autoland є пріоритетними, тому передували тестам зльоту-зависання. Саме автоматична посадка мала довести коректність роботи закону регулювання, щоб потім його можна було застосовувати і для зльоту. Починали з виведення апарату на невелику висоту в ручному режимі

керування, з подальшою активацією автоматичної посадки при майже нульовій вертикальній швидкості. Поступово, в наступних тестах, висота та початкова швидкість при активації режиму збільшувалися до 3 м та ±2 м/с. В ході чисельних коригувань вдалося досягти м'якої посадки, навіть с відносно агресивними початковими умовами вмикання автоматичного регулювання. Також проведені тести з перетином верхньої межі вимірювання датчика. Система переходила на фіксований керуючий вплив, після повторного входження в зону вимірювання відбувалося не більше двох осциляцій перед відновленням контрольованого плавного спуску.

Режим зависання тестувався лише після отримання задовільних результатів автоматичної посадки. Початкові спроби виявили нестабільність на цільовій висоті через особливості розрахунку мапи швидкості; після доопрацювання час виходу на цільову висоту 0,8 м скоротився до 0,5 с – 1,2 с, утримання відбувалося з похибкою ±3 см протягом 60 с навіть при наявності вітру та необхідності ручної корекції позиції апарату в горизонтальній площині.

Аналіз логів дозволив остаточно відпрацювати ПІД-коефіцієнти. Пропорційна складова залишилася в діапазоні 12 – 20 одиниць. Оптимальне значення не виводилося через необхідність охоплювати широкий діапазон початкових швидкостей. Інтегральна складова обмежена 0,15–0,22 для мінімізації дрейфу. Диференціальна складова 4,5 – 6,0 продемонструвала здатність ефективно гасити осциляції. На етапі планування, описаному в розділі 3 поточної роботи, визначено залежність цільової швидкості від висоти в діапазоні 0,3 м/с – 0,1 м/с, що саме і забезпечує плавне приземлення на фінальній фазі. Випробування проведені для режиму зльоту виявили, що ця залежність має бути інвертована відносно цільової висоти зависання, та підсвітили необхідність створення зони нечутливості ±3 см при досягненні цільової висоти.

Розроблений модуль повністю працездатний у реальних умовах, часто несприятливих умовах. Він забезпечує м'яку посадку та стабільне зависання

на висоті 0,8 м з мінімальною участю оператора, який залишається відповідальним лише за компенсацію горизонтальних обурень. Отримані логи з практичних польотів є надійною основою для подальшого вдосконалення системи (Додаток В).

6.5.Економічна доцільність та гнучкість для кастомних збірок

Розроблений модуль є зовнішнім пристроєм plug-and-play, який встановлюється доволі легко в розріз шлейфу польотного контролера та радіоприймача без модифікації прошивок цих комплектуючих. Сам модуль вимагає налаштування відповідно до визначення каналів керування, що обумовлено в польотному контролері. Синхронізація каналів виконується один раз і зберігається в енергонезалежній пам’яті ESP32. Це робить його ідеальним рішенням як для індивідуальних користувачів, так і для навчальних центрів та логістичних підрозділів. Економічну доцільність наведено в таблицях (таб. 6.1, таб. 6.2).

Собівартість одиничного модуля станом на грудень 2025 р. Може бути розрахована наступним чином:

$$К_{сум} = k_{мат} + k_{3D} + k_{пр}, \quad (6.1)$$

де $k_{мат}$ = ESP32 + TF-Luna + дрібна електроніка

k_{3D} = корпус, стійки, тримачі

$k_{пр}$ = складання, налаштування, тест

За розрахунком по даним таблиці отримуємо:

$$К_{сум} = 2080 \text{ грн}$$

Таблиця 6.1 – Розрахунок собівартості одиничного модуля

Складова	Ціна, грн	Пояснення
ESP32 Luna DevKit	280	Rozetka / AliExpress
TF-Luna (оригінал Benewake)	1050	AliExpress
Конвертер рівнів та плата з'єднань	80	локальний ринок
Корпус та тримач датчика	150	80–100 г матеріалу та час друку близько 3 годин
Стійки 4 шт. кріплення та металопластик	300	готові або 3D друк
Дрібна механіка та конектори	90	локальний ринок
Робота (складання + налаштування)	250	оплата найманої праці або власний час
Разом одиничний	2080	

Таблиця 6.2 – Розрахунок собівартості серійного модуля

Кількість, шт.	Коефіцієнт серійності	Ціна за одиницю, грн	Пояснення
1	1,00	2080	Роздріб
10	0,93	1930	Можливі знижки на комплектуючі
50-100	0,88	1830	Знижка на комплектуючі та економія на 3Д-друку

Для створення уявлення про економічну доцільність пропонується порівняти вартість модуля з потенційними втратами при ушкодженні дронів. Таким чином отримаємо економічний ефект за рахунок збереження та заощаджень на ремонтах та заміні як комплектуючих так і апаратів цілком.

Розглянемо типові пошкодження та втрати при ручній посадці FPV-дрона.

Пропелери. Найчастіша поломка як у новачків, так і у досвідчених пілотів. При посадці на нерівну поверхню або при втраті висоти один-два пропелери зачіпають землю, ламаються або отримують сколи. У новачків це трапляється в кожному третьому-четвертому польоті, бо вони ще не відпрацювали відчуття рівня землі через сприйняття перспективного викревлення зображення камери.

Камера FPV. Класична «проблема» новачка. Дрон перекидається на бік або на спину, часто падає саме на камеру. Об'єктив або корпус камери тріщиться об камінь чи то асфальт. Досвідчені пілоти ставлять захисні кільця або стійки, тому ймовірність падає до 4 %, а новачки втрачають камеру в кожному п'ятому серйозному інциденті як то жорстке приземлення або втрата орієнтації в просторі.

Відеопередавач та антена. Антена відривається при перевороті або чіпляє гілки, потрапляє під удари пропелерів. Удар, особливо відцентрової спрямованості від пропелера, часто вириває SMA-роз'єм із плати VTX, що призводить до короткого замикання або роботи на повній потужності без антени, і спалення передавача в підсумку. Для новачків це одна з найдорожчих поломок, і доволі поширена.

Рама карбонова. Може бути ушкоджена при сильному ударі «на плаский» або на ребро. У новачків трапляється кожні 12–15 жорстких посадок, у досвідчених операторів цей покажчик значно кращій, і знижується до одного разу на 100-300 польотів. Суттєві пошкодження рами можуть потягти за собою ушкодження інших компонентів.

Двигун. Періодично можливе заклинювання через потрапляння піску або волокон трави, є імовірність ушкодження обмоток, або трапляється перегрів при спробі «витягнути» дорн різким підвищенням керуючого впливу газу. Коли виходить з ладу один двигун, зазвичай радять міняти пару для балансу. Навіть при однаковому маркуванні, двигуни різних виробників не ідентичні за тягою.

Польотний контролер чи плата ESC. Найпоширеніша причина їх ушкоджень, це погано закріплений акумулятор або силовий шнур. При падінні, якщо батарея відривається, силовий роз'єм чи шнур потрапляє під пропелер. Відбувається коротке замикання, в наслідок якого вигорають ESC або сам польотний контролер. Така аварія може спричинити додаткові ушкодження інших компонентів. У новачків це трапляється приблизно у 8 % випадків, у досвідчених операторів вкрай рідко. Вони формують звичку надійно закріплювати та ретельно перевіряти як батарею так і її дроти, оскільки вже знають вартість таких аварій.

Акумулятор. Пошкодження корпусу, роздування або удар пропелерами при перевороті. У новачків трапляється приблизно раз на 10 польотів, бо часто не приділяють достатньо уваги фіксації. Ушкодження акумулятора нерідко тягне за собою й ушкодження електронних компонентів.

Повна втрата дрона. Виліт за межі видимості, чи втрата відеосигналу, посадка в густий ліс або на дерево, куди неможливо дістатися, падіння у водойму. Нерідко спричиняють саме повну втрату апарату. Для новачка ймовірність повної втрати зазвичай тримається в межах від 3 % до 5 %. Досвідчені оператори скорочують показник цього ризику до 0,3 %, та навіть менше.

Середня вартість 7" FPV-дрона без модуля становить близько 19 000 грн, таким чином питома вага вартості модуля складає близько 10 % від вартості всього літального апарату.

Для оцінки економічної доцільності розглянуто два типи сценаріїв використання модуля автоматизації функцій. В першому сценарії FPV-дрон використовує досвідчений оператор, що має понад 500 польотів. В другому сценарії апаратом керує новачок, з початковим досвідом менше 50 польотів, як наприклад при навчанні.

Типові аварії та середня вартість компонентів на грудень 2025 р. подано в таблиці (таб. 6.3).

Таблиця 6.3 – Вартість компонентів та коефіцієнти ризиків

Компонент	Середня ціна, грн	Коефіцієнт втрат «досвідчений»	Коефіцієнт втрат «новачок»
Пропелери (4 шт.)	400	0,12	0,35
Камера FPV	2000	0,04	0,20
Відеопередавач та антена	2500	0,02	0,12
Рама карбонова	1500	0,01	0,08
Двигун	1000	0,01	0,06
Польотний контролер або ESC	2500	0,003	0,08
Акумулятор	1800	0,005	0,10
Повна втрата дрона	19000	0,003	0,04

Для розрахунку доцільності та економії за один цикл «зліт-посадка», пропонується наступна формула:

$$K_{захист} = \Sigma\ (P_i \times Ціна_i) \qquad (6.2)$$

Таким чином точка окупності модуля може бути оцінена за даними таблиці (таб. 6.4), що демонструє розрахунок окупності для сценарію використання FPV-дрона спорядженого модулем автоматизації функцій досвідченим пілотом та початківцем в процесі навчання.

Таблиця 6.4 – Точка окупності модуля

Сценарій	Економія за цикл, грн	Окупність одиничний 2080 грн	Окупність партії 100 шт. х 1830 грн
«Досвідчений» пілот	276	7,5 циклів	6,6 циклів
«Новачок» у процесі навчання	2160	0,96 < 1 циклу	0,85 < 1 циклу

Висновок до розділу. Для досвідченого пілота модуль окупається за 7–8 польотів і далі працює «в плюс» значно підвищуючі збереженість апарату.

Для новачка або навчального процесу економічний ефект настільки великий, що модуль фактично стає необхідним елементом тренувального дрона: вже одна врятована камера або батарея повністю покриває його вартість, а одна врятована машина окупає приблизно 10 модулів. Таким чином, впровадження модуля знижує середню собівартість одного польоту на 40–60 % у навчальному процесі, та на 8–13 % у логістичному застосуванні.

ВИСНОВКИ

За результатами виконаної кваліфікаційної роботи магістерського рівня на тему: «Проміжний модуль автоматизації функцій керування безпілотним літальним апаратом» отримані наступні висновки:

1. У результаті аналізу технологій та засобів управління БПЛА встановлено, що системи ручного керування FPV-дронами мають значні недоліки, пов’язані з високим навантаженням на оператора, відсутністю автоматизації зльоту, зависання та посадки, а також залежністю від модифікації прошивки для інтеграції додаткових сенсорів. Огляд радіоприймачів, протоколів, комерційних польотних контролерів та їх конфігураторів підтвердив обмежену інтегрованість автоматизації в існуючих рішеннях.

2. Обґрунтовано актуальність модуля автоматизації польотних функцій. Модульний підхід здатний забезпечити зниження навантаження на оператора, підвищення безпеки та збереження цілісності FPV-дронів, модульність і сумісність через протокол Crossfire, зручність для навчання операторів, а також гнучку інтеграцію з кастомними рішеннями без втручання в базову прошивку.

3. Математичне моделювання автоматичних режимів, зокрема розробка симулятора польотних режимів та дешифратора логів у рамках сформованої математичної моделі фізичних процесів, дозволило визначити ефективний закон керування та підтвердити переваги ПІД-регулятора, який забезпечує найбільш плавну роботу з охопленням широкого діапазону початкових умов за висотою та швидкістю, а також кращу компенсацію похибок порівняно з альтернативними варіантами.

4. Обґрунтовано вибір апаратної та сенсорної бази за принципом необхідного та достатнього функціоналу, що забезпечує надійну реалізацію автоматизації при збереженні низької собівартості, доступності компонентів та можливостей подальшого вдосконалення системи.

5. Розроблено алгоритм автоматизації зльоту, зависання та посадки, що передбачає транзит для ручного керування або підміну сигналів в автоматичних режимах, обробку даних датчика відстані та протоколу Crossfire, алгоритми розгону та уповільнення двигунів з обмеженням каналів керування для збереження можливості коригування в горизонтальній площині. На базі алгоритму розроблено програму управляння модуля.

6. Практична реалізація модуля підтвердила його працездатність. Апаратна частина та програмна компонента модуля, у сукупності забезпечили надійну інтеграцію в ланку керування. Польові випробування в діапазоні висот 0,1 м – 4,5 м продемонстрували стабільність та ефективність ПІД-регулювання, що підтверджено логами. Оцінено та доведено економічну доцільність застосування модуля для кастомних збірок.

Отже, мету роботи досягнуто: розроблено прототип проміжного модуля автоматизації функцій керування безпілотним літальним апаратом на базі мікроконтролера та датчика ToF, що забезпечує ефективний зліт, зависання та посадку. Прототип характерізується високою модульністю, сумісністю через протокол Crossfire та зниженням навантаження на оператора в режимах безпосередньої близькості до поверхні землі. Система демонструє практичну цінність для застосування в логістичних місіях, моніторингу територій, аерофотозйомці та навчанні пілотів; подальший розвиток можливий через розширення алгоритмів та інтеграцію додаткових параметрів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Офіційний сайт виробника польотних контролерів Speedy Bee політний контролер f405-v3. URL:https://www.speedybee.com/speedybee-f405-v3-30x30-flight-controller/ (дата звернення: 25.09.2025).

2. Офіційний сайт виробника мікроконтролерів STM32. URL: https://www.st.com/en/microcontrollers-microprocessors/stm32f405-415.html (дата звернення: 25.09.2025)

3. Офіційний сайт спільноти розробників Betaflight. URL: https://www.betaflight.com/ (дата звернення: 28.09.2025)

4. Офіційний сайт розробника конфігуратора iNAV. URL: https://inavllc.com/ (дата звернення: 28.09.2025)

5. Дистрибутивна платформа конфігуратора iNAV. URL: https://github.com/iNavFlight/inav/releases (дата звернення: 03.10.2025)

6. Офіційний сайт виробника польотних контролерів JHEMCU. URL: https://www.jhemcu.com/e_productshow/?78-GHF405AIO-ICM-78.html (дата звернення: 03.10.2025)

7. Офіційний сайт виробника польотних контролерів Pixhawk. URL: https://www.pixhawk.store/docs/flight-controller/pixhawk-6x/pixhawk-6x-technical-specification-1810 (дата звернення: 05.10.2025)

8. Специфікація польотного контролера Tykho Lit S4. URL: https://drive.google.com/file/d/1C_Rkd-OyDaPQgh0tf4ESfEqy7gFkFONL/view (дата звернення: 10.10.2025)

9. Офіційний сайт виробника польотних контролерів Tykho. URL: https://www.tykho.com.ua/en (дата звернення: 20.10.2025)

10. Офіційний сайт спільноти розробників, сторінка завантаження конфігуратора Betaflight. URL: https://www.betaflight.com/download (дата звернення: 03.10.2025)

11. Офіційний сайт відкритої спільноти розробників Ardupilot. URL: https://ardupilot.org/ (дата звернення: 20.11.2025)

12. Вакула Ю., Міняйло Н. О. Виготовлення пристроїв для роботизованих систем із застосуванням 3D-технологій. Збірник наукових праць студентів, аспірантів, докторантів і молодих вчених «Молода наука-2025» Запоріжжя: Запорізький національний університет, 2025. Т. 5 С. 166-167.

13. Міняйло Н.О., Вакула Ю.О., Ендік Т.В. Модуль автоматичної посадки FPV-дрона. Інженерні інновації та розбудова національної економіки : матеріали II Міжнародної науково-практичної конференції Інженерного навчально-наукового інституту ім. Ю. М. Потебні (15–16 травня 2025 року, м. Запоріжжя) / наук. ред. Н. Г. Метеленко; Інженерний навчально-науковий інститут ім. Ю. М. Потебні Запорізького національного університету. – Одеса: Видавничий дім «Гельветика», 2025. С. 930-933

14. Grok xAI Large Language Model, інтерактивна консультація та технічна підтримка в режимі реального часу, 2025. URL: https://x.ai/grok (дата звернення 11.11.2025).

15. ChatGPT-4o (OpenAI), інтерактивний помічник. URL: https://chat.openai.com (дата звернення 15.11.2025).

16. Опис польотного контролера Matek на платформі відкритої спільноти. URL: https://ardupilot.org/copter/docs/common-matekf405.html (дата звернення: 20.11.2025)

17. Офіційний сайт виробника TBS. URL: https://www.team-blacksheep.com/shop/cat%3Acat_crossfire?srsltid=AfmBOopWGouJcnSqXv75dZ8yVm3AkHG6TKwHjZ8p5QdHTtljWjHwPpDu&utm_source=chatgpt.com(дата звернення: 25.11.2025)

ДОДАТКИ

Додаток А

ПОРІВНЯЛЬНА ТАБЛИЦЯ ПОЛЬОТНИХ КОНТРОЛЕРІВ

Виробник / Параметр	Speedy Bee F405 V3	JHEMCU GHF405	TykhoLit S4	Pixhawk 6X	Matek F405-STD
1	2	3	4	5	6
Мікроконтролер	STM32F405 (168 МГц, ARM Cortex-M4 з FPU) |	STM32F405 (168 МГц, ARM Cortex-M4 з FPU)	STM32F405 (168 МГц, ARM Cortex-M4 з FPU)	FMU: STM32H753 (480 МГц, ARM Cortex-M7 з FPU); IO: STM32F103 (72 МГц)	STM32F405 (168 МГц, ARM Cortex-M4 з FPU)
IMU (гіроскоп + акселерометр)	MPU6000 (SPI, <1 мс)	MPU6000 (SPI, <1 мс)	MPU6000 (SPI, <1 мс)	3× ICM-45686 (BalancedGyro™, SPI, redundancy)	ICM42688-P (SPI, <1 мс)
Барометр	BMP280 (I2C)	BMP280 (I2C, опціонально)	BMP280 (I2C)	2× (ICP20100 + BMP388, redundancy	BMP280 (I2C)

Продовження порівняльної таблиці польотних контролерів (додаток А)

1	2	3	4	5	6
Магнітометр	Опціонально (зовнішній)	Опціонально (зовнішній)	Опціонально (зовнішній,	BMM150 (зовнішній)	Опціонально (зовнішній, напр. RM3100)
UART	6	4	5	4 (з flow control) + 2 GPS + 2 CAN	5 (з flow control на UART6)
Моторні виходи	4–8 (залежно від стеку	4	4	16 PWM	8
Живлення (LiPo)	3-6S (до 50A/канал у стеку)	3-6S (до 30A/канал)	3-6S (до 40A/канал)	3-12S (до 50A/канал)	3-6S (до 35A/канал)
BEC	5V/2A (у стеку)	5V/2A	5V/2A + 9V/1A	| 5V/3A + 3.3V/400mA	5V/2A + 3.3V/500mA
Current sensor	Так (у стеку, до 100A)	Ні (зовнішній пін)	Так (до 80A)	Так (до 120A)	Так (до 100A)
Протоколи RX	SBUS/PPM/CRSF	SBUS/PPM/CRSF	SBUS/PPM/CRSF	Spektrum/DSM, PPM/S.Bus, CRSF (адаптер)	SBUS/PPM/CRSF

Продовження порівняльної таблиці польотних контролерів (додаток А)

1	2	3	4	5	6
ESC протоколи	BLHeli_32, DShot300/600	BLHeli_S/32, DShot300/600	BLHeli_32, DShot600	DShot, PWM, UAVCAN	BLHeli_32, DShot1200
OSD	Так (Betaflight OSD)	Ні (зовнішній)	Так (Betaflight OSD, AT7456E)	Так (MAVLink OSD)	Так (Betaflight OSD, AT7456E)
Blackbox	MicroSD (до 32 ГБ)	MicroSD (до 16 ГБ)	MicroSD (до 32 ГБ)	MicroSD (до 128 ГБ)	MicroSD (до 64 ГБ)
Інтерфейси	I2C, SPI, USB Type-C, Bluetooth/Wi-Fi	I2C, SPI, USB Micro-B	I2C, SPI, USB Type-C	I2C, SPI, CAN (2), Ethernet, USB-C	I2C, SPI, USB Type-C
Розміри (мм)	30.5×30.5 (стек)	30.5×30.5	30.5×30.5	38.8×31.8 (FC); 52.4×103.4 (baseboard)	30.5×30.5
Вага (г)	~10 (стек)	~6	~8	23 (FC); 51 (baseboard)	~7
Підтримка прошивок	Betaflight, iNav (оптимізовано)	Betaflight, iNav	Betaflight, iNav	PX4, ArduPilot	Betaflight, iNav

Продовження порівняльної таблиці польотних контролерів (додаток А)

1	2	3	4	5	6
Redundancy	Hi	Hi	Hi	Так (3×IMU, 2×baro, 3×power)	Hi
Захист	LC-фільтри, перегрів (ESC)	Базовий (зворотна полярність)	LC-фільтри, зворотна полярність	ESD, зворотна полярність, віброізоляція	TVS, зворотна полярність, LC-фільтри
Ціна (USD, орієнтовно, 2025)	35–45 (FC); 70 (стек)	15–20	25–35	200–300 (full kit)	25–35
MTBF (годин, оцінка)	>500	~400	~450	>1000	~500
Сумісність з CRSF	Так (повна)	Так	Так	Так (з адаптером)	Так (повна)
Особливості	Бездротовий тюнінг (Bluetooth/Wi-Fi), вбудований стек ESC	Мінімалізм, низька ціна	Локальна підтримка, українська документація	Redundancy, Ethernet, модульність	8 моторів, низький шум IMU

Додаток Б

КОД СИМУЛЯТОРА

```
# drone_sim_touch_in_block.py
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
from matplotlib.widgets import Slider
import numpy as np
import time
import keyboard
# ========================================
# КОНСТАНТИ
# ========================================
mass = 1.225
thrust_ = 3.4
DronHeight = 0.15
gravity = 9.81
P_min = 5
P_max = 20
P_step = 5
P_COEFICIENT = P_max
D_COEFICIENT = 5.0
throttle_min = 1000
throttle_max = 1800
minDescentSpeed = 0.1
maxDescentSpeed = 0.3
throttleForceMin = 1.0
expo = 1.0
```

```
motorResponseTimeMaxMs = 500.0

dataNoise = 0.01

UpdateIntervalms = 50

MaxSimHeightM = 500

MaxTFlunaheight = 4.0

MaxAltitudeCM = 400


def mapf(x, in_min, in_max, out_min, out_max):

    return (x - in_min) * (out_max - out_min) / (in_max
- in_min) + out_min


def getThrust(throttle, thrustMin, thrustMax, expo):

    throttleNorm = max(0, min(1, (throttle - 1000) /
1000.0))

    return thrustMin + (thrustMax - thrustMin) *
(throttleNorm ** expo)


def calcHoverThrottle(mass, thrustMin, thrustMax,
expo):

    f_hover = mass * gravity

    throttleNorm = max(0, min(1, (f_hover - thrustMin)
/ (thrustMax - thrustMin)))

    throttle = 1000 + 1000 * (throttleNorm ** (1.0 /
expo))

    return int(max(1000, min(2000, throttle)))

throttleForceMax = gravity * thrust_

throttleHover = calcHoverThrottle(mass,
throttleForceMin, throttleForceMax, expo)

# ======================================

# ВІЗУАЛІЗАЦІЯ
```

```
# ========================================

fig = plt.figure(figsize=(8, 7))

gs = fig.add_gridspec(2, 2, width_ratios=[1.4, 0.6],
height_ratios=[1.4, 0.1], hspace=0.4, wspace=0.35)

#2

ax_log = fig.add_subplot(gs[0, 0])

ax_log.axis('off')

ax_log.set_title("ЛОГ + THROTTLE", fontsize=11, pad=8)

log_text = ax_log.text(0.05, 0.95, "", fontsize=8.5,
fontfamily='monospace',

                       va='top', ha='left',
transform=ax_log.transAxes)

ax_slider = plt.axes([0.42, 0.35, 0.06, 0.55])

throttle_slider = Slider(ax_slider, '', throttle_min,
throttle_max,

                         valinit=throttle_min,
orientation='vertical', valstep=1,

                         color='lightblue', alpha=0.8)

throttle_slider.valtext.set_visible(False)

ax_sim = fig.add_subplot(gs[0, 1])

ax_sim.set_xlim(0, 1.0)

ax_sim.set_ylim(0, 5.0)

ax_sim.set_title("Дрон (5 м шкала)", fontsize=11)

ax_sim.set_ylabel("м", fontsize=9)

ax_sim.tick_params(axis='both', labelsize=8)

ax_sim.axhspan(0, DronHeight, color='green', alpha=0.3)

ax_sim.axhline(y=MaxTFlunaheight, color='blue',
linestyle='--', linewidth=1.5)

drone_width = 0.09
```

```
drone_height = 0.08
drone_x = (1.0 - drone_width) / 2
drone_rect = plt.Rectangle(
    (drone_x, DronHeight), drone_width, drone_height,
    fc='red', ec='black', lw=1.2
)
ax_sim.add_patch(drone_rect)

# --- ВІДОБРАЖЕННЯ ШВИДКОСТІ ДОТИКУ ---
ax_touch = fig.add_subplot(gs[1, 1])
ax_touch.axis('off')
touch_text = ax_touch.text(0.5, 0.5, "Швидкість дотику: —", fontsize=10, ha='center', va='center', fontweight='bold')
const_str = (
    f"mass = {mass} кг\n"
    f"thrust_ = {thrust_} кг\n"
    f"DronHeight = {DronHeight} м\n"
    f"gravity = {gravity} м/с²\n"
    f"P: {P_min}-{P_max} (крок {P_step})\n"
    f"P = {P_COEFICIENT}, D = {D_COEFICIENT}\n"
    f"throttle: {throttle_min}-{throttle_max}\n"
    f"Descent: {minDescentSpeed}-{maxDescentSpeed} м/с\n"
    f"throttleForceMin = {throttleForceMin} Н\n"
    f"expo = {expo}\n"
    f"motorResponse = {motorResponseTimeMaxMs} мс\n"
    f"noise = ±{dataNoise} м\n"
```

```
    f"MaxSim = {MaxSimHeightM} м\n"
    f"TF-Luna max = {MaxTFlunaheight} м\n"
    f"MaxCM = {MaxAltitudeCM} см\n"
    f"throttleForceMax = {throttleForceMax:.3f} H\n"
    f"throttleHover = {throttleHover}\n"
)
# ========================================
# ПОЧАТКОВИЙ СТАН
# ========================================
altitude = 5.0
velocity = 0.0
thrust = getThrust(1000, throttleForceMin,
throttleForceMax, expo)
prev_time = None
current_throttle = throttle_min
target_throttle = throttle_min
dragging = False
auto_landing = False
prev_error = 0.0
integral = 0.0
prev_tf_dist = 0.0
last_update_time = 0
first_touch_speed = None
touch_color = 'black'
# ========================================
# КЕРУВАННЯ
# ========================================
def on_slider_change(val):
```

```
    global current_throttle, target_throttle
    current_throttle = int(throttle_slider.val)
    if not auto_landing and not dragging:
        target_throttle = current_throttle
throttle_slider.on_changed(on_slider_change)


def toggle_auto_landing():
    global auto_landing, target_throttle, integral,
prev_error, prev_tf_dist, first_touch_speed,
last_update_time
    auto_landing = not auto_landing
    integral = 0.0
    prev_error = 0.0
    prev_tf_dist = 0.0
    first_touch_speed = None  # сброс при новому
пирземленні
    last_update_time = time.time()  # сброс часу при
вмиканні автопосадки
    if auto_landing:
        localSpeed = -velocity
        target_throttle = 1400 if localSpeed > 2.0 else
1200 if localSpeed < -2.0 else 1300
    else:
        target_throttle = current_throttle
keyboard.on_press_key("space", lambda _:
toggle_auto_landing())


def on_press(event):
    global dragging, altitude, velocity
```

```
    if event.button == 1 and event.inaxes == ax_sim and
not auto_landing:
        dragging = True
        altitude = np.clip(event.ydata, DronHeight,
6.0)
        velocity = 0.0
def on_motion(event):
    global altitude
    if dragging and event.inaxes == ax_sim:
        altitude = np.clip(event.ydata, DronHeight,
6.0)
def on_release(event):
    global dragging
    if event.button == 1:
        dragging = False
fig.canvas.mpl_connect('button_press_event', on_press)
fig.canvas.mpl_connect('motion_notify_event',
on_motion)
fig.canvas.mpl_connect('button_release_event',
on_release)
# ========================================
# PID
# ========================================
def compute_pid(nowSpeed, targetSpeed, dt):
    global prev_error, integral
    error = nowSpeed - targetSpeed
    integral += error * dt
    derivative = (error - prev_error) / dt if dt > 0
else 0
```

```
    prev_error = error
    P = P_COEFICIENT
    I = mapf(P_COEFICIENT, P_min, P_max, 0.1, 0.25)
    D = D_COEFICIENT
    return P * error + I * integral + D * derivative
# ========================================
# ОСНОВНИЙ ЦИКЛ РОЗРАХУНКІВ
# ========================================
def update(frame):
    global altitude, velocity, thrust, prev_time,
target_throttle, current_throttle
    global prev_tf_dist, last_update_time,
first_touch_speed, touch_color
    current_time = time.time()
    if prev_time is None:
        prev_time = current_time
        #drone_rect.set_y(altitude)
        return drone_rect, touch_text
    dt = min(current_time - prev_time, 0.1)
    prev_time = current_time
    acceleration = 0.0
    was_above = altitude > DronHeight + 0.005
    # === реальна висота (для фізики) ===
    distance = max(0.0, altitude - DronHeight)  # ЧИСТА
    # === ДАТЧИК (для PID та лога) ===
    if distance > MaxTFlunaheight:  # понад 4 м датчик
втрачає можливість вимірювати
        noisy_distance = 0.0
```

```
    else:
        noisy_distance = distance + np.random.uniform(-
dataNoise, dataNoise)
    tfDistCM = int(noisy_distance * 100)  # з похибкою
датчика — ДЛЯ PID
    if auto_landing:
        # === PID — отримує зашумлені дані ===
        if tfDistCM > 10 and current_time -
last_update_time >= UpdateIntervalms / 1000.0:
            distanceM = tfDistCM / 100.0  # ← зашумлена
висота
            dt_pid = current_time - last_update_time
            last_update_time = current_time

            if prev_tf_dist < 0.005:
                prev_tf_dist = distanceM
                integral = prev_error = 0
            nowSpeed = (prev_tf_dist - distanceM) /
dt_pid  # швидкість з похибкою датчика
            targetSpeed = mapf(distanceM, 0.1,
MaxAltitudeCM/100.0, minDescentSpeed, maxDescentSpeed)
            targetSpeed = np.clip(targetSpeed,
minDescentSpeed, maxDescentSpeed)
            prev_tf_dist = distanceM

            throttle_adjust = compute_pid(nowSpeed,
targetSpeed, dt_pid)
            target_throttle += throttle_adjust
```

```
            target_throttle =
int(np.clip(target_throttle, throttle_min,
throttle_max))
            mode = "АВТОПОСАДКА (PID с шумом)"
        else:
            mode = "АВТОПОСАДКА (OUT)"
    else:
        mode = "РУЧНЕ (drag/throttle)"
    if dragging:
        drone_rect.set_y(altitude)
        velocity = 0.0
        target_throttle = current_throttle
        mode = "УТРИМАННЯ (drag)"
    else:
        targetThrust = getThrust(target_throttle,
throttleForceMin, throttleForceMax, expo)
        thrustDelta = abs(targetThrust - thrust)
        maxDelta = throttleForceMax - throttleForceMin
        normDelta = np.clip(thrustDelta / maxDelta,
0.0, 1.0) if maxDelta > 0 else 0.0
        motorResponseTimeCurrentMs =
motorResponseTimeMaxMs * normDelta
        motorResponseTimeCurrentMs =
max(motorResponseTimeCurrentMs, 1.0)
        dtMs = dt * 1000.0
        alpha = dtMs / (motorResponseTimeCurrentMs +
dtMs)
        thrust += (targetThrust - thrust) * alpha
        # === фізика, розподілення удару та статичної
опопри ===
```

```
        N = 0.0
        on_ground = altitude <= DronHeight
        if on_ground:
            altitude = DronHeight

            # Фиксація швидкості дотику
            if first_touch_speed is None and velocity <
0:
                first_touch_speed = abs(velocity)
                touch_color = ('green' if
first_touch_speed <= 0.1 else
                               'orange' if
first_touch_speed <= 0.3 else 'red')

            # === відбиття: лише при сильному ударі
визначається за швидкістю ===
            if velocity < -0.5:  # порог удара
                velocity = -0.25 * velocity  # 25%
отскок
            else:
                # === статична опора: діє завжди коли
не відбиття при ударі ===
                N = max(0.0, mass * gravity - thrust)
                velocity = 0

        # === підсумкова сила ===
        net_force = thrust - mass * gravity + N
       acceleration = net_force / mass
        # === рух ===
```

```
        velocity += acceleration * dt
        altitude += velocity * dt
        if altitude < DronHeight:
            altitude = DronHeight
        drone_rect.set_y(altitude)

    # --- оновлення тексту дотику ---
    if first_touch_speed is not None:
        touch_text.set_text(f"Швидкість дотику:
{first_touch_speed:.3f} м/с")
        touch_text.set_color(touch_color)
    # --- ЛОГ ---
    landing_speed_str =
(f"[{touch_color}]{first_touch_speed:.3f}
м/с[/{touch_color}]"
                        if first_touch_speed is not
None else "—")
    status_lines = [
        f"Throttle Current: {current_throttle}",
        f"Throttle Target: {target_throttle}",
        f"Тяга: {thrust:.3f} ",
        f"Висота: {altitude:.3f} м",
        f"Датчик: {noisy_distance:.3f} м (±{dataNoise}
м)",
        f"Швидкість: {velocity:+.3f} м/с",
        f"Прискорення: {acceleration:.4f} м/с²",
        f"Дотик: {landing_speed_str}",
        f"Режим: {mode}",
```

```
        f"Space = автопосадка: {'УВІМК' if auto_landing
else 'ВИМК'}",
        f"dt: {dt*1000:.1f} мс"
    ]
    full_log = const_str + "\n" + "—" * 28 + "\n" +
"\n".join(status_lines)
    log_text.set_text(full_log)
    return drone_rect, touch_text
# ========================================
# ЗАПУСК
# ========================================
ani = FuncAnimation(fig, update, interval=0,
cache_frame_data=False, blit=False)
plt.subplots_adjust(left=0.05, right=0.98, top=0.94,
bottom=0.02)
plt.show()
```

Додаток В

ФРАГМЕНТ ЛОГ-ФАЙЛА

Приклад поведінки системи з мінімальним впливом обурень..

```
> ch1988 0 13 990 0.00
AL true at: 13 cm
1299.00
> ch1988 1 12 990 0.04
1244.00
. . .
> ch1988 1 12 990 0.00
1000.00
> ch1988 1 12 990 0.00
1000.00
> ch1988 1 12 990 0.00
Hover true at: 13 cm
Hover false
> ch1000 0 13 990 -0.04
> ch1000 0 12 990 0.04
. . .
> ch1000 0 12 990 -0.04
Hover true at: 11 cm
1000.00
> ch1988 2 12 990 0.00
AL true at: 12 cm
1287.00
> ch1988 1 11 990 0.04
1234.00
> ch1988 1 11 990 0.00
1186.00
> ch1988 1 12 990 -0.04
1133.00
Hover true at: 12 cm
Hover false
> ch1000 0 11 990 0.04
> ch1000 0 12 990 -0.04
. . .
> ch1000 0 11 990 0.04
```

```
> ch1000 0 12 990 -0.04
Hover true at: 12 cm
1000.00
> ch1497 2 11 990 0.04
1053.00
> ch1497 2 11 990 0.00
. . .
> ch1497 2 14 990 -0.12
1368.00
> ch1497 2 18 990 -0.16
0.24; -0.58; -0.10; -0.56; -23.25; 1346; 0.05; 10.00;
1558; 282
> ch1497 2 32 990 -0.56
0.44; -0.78; -0.10; -0.36; -8.33; 1294; 0.05; 10.00;
50; 260
> ch1497 2 47 990 -0.60
0.54; -0.38; -0.10; -0.26; -1.39; 1319; 0.05; 10.00;
55; 260
> ch1497 2 57 990 -0.40
0.62; -0.19; -0.10; -0.18; -0.94; 1329; 0.05; 10.00;
55; 262
> ch1497 2 64 990 -0.28
0.66; 0.00; -0.10; -0.14; 19.83; 1345; 0.05; 10.00; 52;
274
> ch1497 2 67 990 -0.12
0.68; 0.00; -0.10; -0.12; 19.47; 1346; 0.05; 10.00; 55;
260
> ch1497 2 68 990 -0.04
0.70; -0.19; -0.10; -0.10; -18.71; 1329; 0.05; 10.00;
55; 259
> ch1497 2 70 990 -0.08
0.75; -0.40; -0.10; -0.05; -24.16; 1303; 0.05; 10.00;
50; 262
> ch1497 2 76 990 -0.24
0.73; 0.00; -0.10; -0.07; -17.16; 1350; 0.05; 10.00;
55; 262
> ch1497 2 74 990 0.08
0.76; 0.00; -0.10; -0.04; 19.83; 1347; 0.05; 10.00; 55;
256
```

```
> ch1497 2 77 990 -0.12
0.78; 0.00; 0.00; -0.02; 19.21; 1338; 0.05; 10.00; 50;
263
> ch1497 2 79 990 -0.08
0.79; 0.00; 0.00; -0.01; -0.02; 1337; 0.05; 10.00; 55;
260
> ch1497 2 79 990 0.00
> ch1497 2 78 990 0.04
0.78; 0.00; 0.00; -0.02; -0.02; 1338; 0.05; 10.00; 56;
267
> ch1497 2 75 990 0.12
0.74; 0.17; -0.10; -0.06; 17.57; 1373; 0.06; 10.00; 53;
267
> ch1497 2 77 990 -0.08
0.77; -0.20; -0.10; -0.03; -21.01; 1335; 0.05; 10.00;
54; 260
> ch1497 2 80 990 -0.12
0.81; -0.20; 0.00; 0.01; -2.35; 1318; 0.05; 10.00; 55;
260
> ch1497 2 81 990 -0.04
0.82; -0.20; 0.00; 0.02; -22.02; 1314; 0.05; 10.00; 49;
256
> ch1497 2 80 990 0.04
0.79; 0.20; 0.00; -0.01; 21.17; 1360; 0.05; 10.00; 55;
261
> ch1497 2 80 990 0.00
0.78; 0.00; 0.00; -0.02; -17.82; 1342; 0.05; 10.00; 55;
261
> ch1497 2 79 990 0.04
0.80; 0.00; 0.00; 0.00; 16.53; 1337; 0.05; 10.00; 51;
264
> ch1497 2 79 1103 0.00
0.78; 0.00; 0.00; -0.02; -18.88; 1340; 0.05; 10.00; 54;
256
> ch1497 2 79 1284 0.00
0.78; 0.00; 0.00; -0.02; -0.02; 1340; 0.05; 10.00; 55;
265
> ch1497 2 78 1294 0.04
```

```
0.79; -0.20; 0.00; -0.01; -21.20; 1320; 0.05; 10.00;
56; 264
> ch1497 2 78 1415 0.00
0.78; 0.19; 0.00; -0.02; 56.67; 1357; 0.05; 10.00; 46;
260
> ch1497 2 79 1402 -0.04
0.79; 0.00; 0.00; -0.01; -0.02; 1338; 0.05; 10.00; 55;
259
> ch1497 2 79 1402 0.00
0.79; 0.00; 0.00; -0.01; 19.21; 1337; 0.05; 10.00; 55;
262
> ch1497 2 78 1427 0.04
0.78; 0.00; 0.00; -0.02; 18.84; 1341; 0.05; 10.00; 49;
260
> ch1497 2 78 1500 0.00
0.78; 0.00; 0.00; -0.02; -19.24; 1339; 0.05; 10.00; 55;
256
> ch1497 2 80 1494 -0.08
0.77; 0.00; -0.10; -0.03; -36.75; 1355; 0.05; 10.00;
55; 257
> ch1497 2 77 1492 0.12
0.79; -0.19; 0.00; -0.01; -1.56; 1324; 0.05; 10.00; 48;
264
> ch1497 2 77 1494 0.00
0.79; 0.00; 0.00; -0.01; 18.86; 1345; 0.05; 10.00; 55;
262
> ch1497 2 81 1251 -0.16
0.83; 0.00; 0.00; 0.03; 19.98; 1339; 0.05; 10.00; 55;
255
> ch1497 2 81 1127 0.00
0.81; 0.00; 0.00; 0.01; -0.01; 1344; 0.05; 10.00; 48;
259
> ch1497 2 82 1127 -0.04
0.80; 0.58; 0.00; 0.00; 24.25; 1399; 0.05; 10.00; 55;
259
. . .
```