

**МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ**

**ФАКУЛЬТЕТ ФІНАНСІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І
СИСТЕМ**

КВАЛІФІКАЦІЙНА РОБОТА

**на тему: Розробка системи керування безпілотними літальними апаратами
в локальній та інтернет мережі**

Здобувача вищої освіти першого (бакалаврського) рівня,
IV курсу, групи ЦДБ-22-1, спеціальності 122 «Комп'ютерні науки»,
освітня програма «Технології цифрового дизайну»


(підпис)

Здітовецького Данііла
Вячеславовича
(прізвище, ім'я, по батькові)

Науковий керівник


(підпис)

Ратушняк Тетяна Володимирівна
канд. фіз.-мат. наук, доцент
(прізвище, ім'я, по батькові, науковий ступінь,
вчене звання)

Керівник освітньо-
професійної програми


(підпис)

Ратушняк Тетяна Володимирівна
канд. фіз.-мат. наук, доцент
(прізвище, ім'я, по батькові, науковий ступінь,
вчене звання)

Допущено до захисту за рішенням кафедри комп'ютерних та інформаційних
технологій і систем протокол від «12» травня 2026 № 20

Завідувач кафедри


(підпис)

Філоненко Михайло
Миколайович, канд. фіз.-мат.
наук, доцент
(прізвище, ім'я, по батькові, науковий ступінь,
вчене звання)

Ірпінь – 2026

Реєстрація № 46

24.05.2026

(дата)



(підпис відповідальної особи)

Результати перевірки

до захисту
(до захисту, на доопрацювання)

12.05.26

(дата)



(підпис)

Результат захисту

5 (відмінно)
(оцінка)

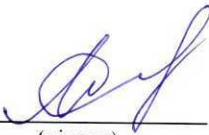
11.06.2026

(дата)



(підпис)

Голова комісії:



(підпис)

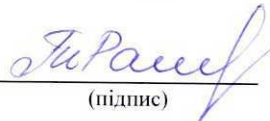
О.С. Дюбанов

Члени комісії:



(підпис)

М.М. Філоненко



(підпис)

Т.В. Ратушняк



(підпис)

В.О. Ніжегородцев

БЛАНК ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Об'єкти контролю	Діапазон оцінки в балах	Фактична кількість балів	Примітки	Дата, П.І.Б.: керівника, голови комісії членів комісії, секретаря, (підписи)
ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ КОМІСІЄЮ ПО ЗАХИСТУ				
1. Зміст випускної роботи (якість теоретичного матеріалу, його зв'язок з практичним, вміння представляти матеріал в логічній послідовності, наявність висновків та пропозицій)	від 0 до 30	28		11.06.2026 Ратушняк Т.В. <i>Т.В. Ратушняк</i> Дюбанов О.С. <i>О.С. Дюбанов</i>
2. Дотримання вимог щодо оформлення	від 0 до 10	9		Філоненко М.М. <i>М.М. Філоненко</i> Ратушняк Т.В. <i>Т.В. Ратушняк</i>
3. Вчасне виконання роботи на всіх етапах у встановлені кафедрою терміни	від 0 до 5	4		Ніжегородцев В.О. <i>В.О. Ніжегородцев</i> Суховерхівка Л.П. <i>Л.П. Суховерхівка</i>
4. Рецензент	від 0 до 5	5		
РАЗОМ БАЛІВ	від 0 до 50	46		

Оцінювання кваліфікаційної роботи науковим керівником – 46 балів.

ПЕРЕВЕДЕННЯ ОЦІНКИ ЗА 100-БАЛЬНОЮ ШКАЛОЮ ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою	Дата	П.І.Б.: голови, членів комісії, секретаря, (підписи)
92	A	5 (відмінно)	11.06.2026	Дюбанов О.С. <i>О.С. Дюбанов</i> Філоненко М.М. <i>М.М. Філоненко</i> Ратушняк Т.В. <i>Т.В. Ратушняк</i> Ніжегородцев В.О. <i>В.О. Ніжегородцев</i> Суховерхівка Л.П. <i>Л.П. Суховерхівка</i>

ПРАВИЛА ПЕРЕВЕДЕННЯ ОЦІНОК ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою
90-100	A	5 (відмінно)
80-89	B	4 (добре)
70-79	C	
60-69	D	
50-59	E	3 (задовільно)
35-49	FX	2 (незадовільно)

Факультет фінансів та цифрових технологій
Кафедра комп'ютерних та інформаційних технологій і систем
Галузь знань: 12 «Інформаційні технології»
Спеціальність: 122 «Комп'ютерні науки»
Освітня програма: «Технології цифрового дизайну»

Затверджую
Завідувач кафедри  Філоненко М.М.
«01» грудня 2025

Завдання на кваліфікаційну роботу
здобувачу вищої освіти
Здітовецькому Даніїлу Вячеславовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: Розробка системи керування безпілотними літальними апаратами в локальній та інтернет мережі

Затверджена наказом ректора від «28» листопада 2025 № 1526

2. Строк здачі закінченої роботи «11» травня 2026

3. Мета роботи: розробка та дослідна перевірка програмно-апаратної системи керування безпілотними літальними апаратами з комп'ютера в локальній та інтернет-мережі, яка забезпечує виявлення мережевого відеопристрою через ONVIF, прийом RTSP-відеопотоку, OSD-відображення службової інформації на основі MSP DisplayPort та передавання керуючих даних через Serial/HID-to-UDP bridge.

Об'єкт дослідження: процес мережевого керування безпілотними літальними апаратами з використанням локальних та віддалених IP-мереж.

Предмет дослідження: методи, програмні модулі та мережева архітектура передавання відеопотоку, OSD/службової інформації і керуючих даних між персональним комп'ютером оператора та бортовим мережевим вузлом БПЛА.

Орієнтовна база дослідження: тестовий програмно-апаратний стенд на базі ПК оператора, IP-камери OpenIPC/SSC30KQ, мережевого вузла MikroTik hEX S та локальної/VPN-мережі.

4. Вихідні дані до роботи: науково-технічні джерела щодо систем керування БПЛА, IP-мереж, відеопередавання та протоколів реального часу; документація та специфікації технологій ONVIF, RTSP, MSP DisplayPort, UDP, VPN, HID, CSRF; програмні засоби Python, Kivy, PyAV/FFmpeg, onvif-zeep; апаратні компоненти тестового стенда: ПК оператора, IP-камера OpenIPC/SSC30KQ, маршрутизатор MikroTik hEX S, HID/Serial-пристрої керування; результати експериментального тестування LAN, SFP/оптоволоконного та VPN+Wi-Fi/LTE-сценаріїв; показники RTT, packet loss, стабільності RTSP-потoku, коректності OSD-відображення та роботи UDP-моста.

5. Перелік графічного матеріалу: структурна схема програмно-апаратної системи керування БПЛА; схема взаємодії ПК оператора, IP-камери OpenIPC/SSC30KQ, маршрутизатора MikroTik hEX S та бортового мережевого вузла; схема потоків даних між RTSP-відео, OSD/MSP-пакетами та керуючим UDP-каналом; алгоритм роботи застосунку оператора; структура програмного проєкту з основними модулями Controller, ONVIF Discovery, RtspPlayer, MSPDisplayPortClient, OSDGrid, SerialToNetBridge та HID-to-CRSF converter; фрагменти програмної реалізації RTSP-прийому, MSP DisplayPort parser, OSD shadow buffer, CSRF-пакування каналів і Serial/HID-to-UDP bridge; таблиці функціональних і нефункціональних вимог; таблиці порівняння технологій відеопередавання, OSD та керування; таблиці сценаріїв експериментальної перевірки;

таблиці кількісних результатів тестування LAN і VPN+Wi-Fi/LTE; панель Link T Statistics із показниками UDP, MSP, відеопакетів, декодованих кадрів і помилок.

6. Зміст кваліфікаційної роботи (перелік питань за кожним розділом Вступ; Розділ 1. Теоретико-методологічні основи побудови систем мережевого керування БПЛА; 1.1. Сучасні підходи до керування БПЛА та вимоги до мережевого каналу; 1.2. Порівняльний аналіз мережевих, відеоінформаційних і сервісних технологій; 1.3. Обґрунтування вибору технологій і дослідницькі критерії; Висновки до розділу 1; Розділ 2. Системний аналіз та проектування системи керування БПЛА; 2.1. Постановка задачі та вимоги до системи; 2.2. Апаратна та програмна архітектура системи; 2.3. Алгоритми роботи, модель потоків даних і методика експерименту; Висновки до розділу 2; Розділ 3. Реалізація, тестування та оцінювання системи керування БПЛА; 3.1. Реалізація програмних модулів системи; 3.2. Організація тестування системи; 3.3. Аналіз результатів, обмеження та напрями вдосконалення; Висновки до розділу 3; Висновки до роботи; Список використаних джерел; Додатки.

7. Календарний план виконання роботи

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	Фактично
1	2	3	4
1	Вибір теми, підбір літературних джерел, формування структури кваліфікаційної роботи	До 30.10.2025	До 30.10.2025
2	I розділ (розкрити сутність і стан дослідження наукової проблематики роботи, після чого навести її загальну характеристику; огляд літератури за темою і вибір конкретних напрямів досліджень)	29.12.2025	29.12.2025
3	II розділ (аналіз експериментальної частини про методики досліджень)	23.02.2026	23.02.2026
4	III розділ (проведені теоретичні і (або) експериментальні дослідження та проектні рішення)	20.04.2026	20.04.2026
5	Висновки	04.05.2026	04.05.2026
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	11.05.2026	11.05.2026
7	Отримання відгуку керівника	01.05.2026	01.05.2026
8	Отримання зовнішньої рецензії	04.06.2026	04.06.2026

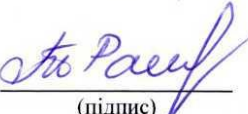
8. Дата видачі завдання «01» грудня 2025

9. Керівник кваліфікаційної роботи


(підпис)

Ратушняк Т.В.
(прізвище, ініціали)

10. Керівник освітньо-професійної програми


(підпис)

Ратушняк Т.В.
(прізвище, ініціали)

11. Завдання до виконання отримав здобувач вищої освіти


(підпис)

Здітовецький Д.В.
(прізвище, ініціали)

АНОТАЦІЯ

Розробка системи керування безпілотними літальними апаратами в локальній та інтернет-мережі

Мета роботи: розробка та дослідна перевірка програмно-апаратної системи керування безпілотними літальними апаратами з комп'ютера в локальній та інтернет-мережі, яка забезпечує виявлення мережевого відеопристрою через ONVIF, прийом RTSP-відеопотоку, OSD-відображення службової інформації на основі MSP DisplayPort та передавання керуючих даних через Serial/HID-to-UDP bridge.

Завданням роботи є: провести аналіз сучасних підходів до керування БПЛА через локальні та віддалені IP-мережі; дослідити особливості передавання відеопотоку, службової OSD-інформації та керуючих даних у мережевому середовищі; порівняти технології ONVIF, RTSP, MSP DisplayPort, UDP, VPN, HID та CRSF з погляду їх застосування у системі керування БПЛА; сформулювати функціональні та нефункціональні вимоги до програмно-апаратної системи; розробити апаратну та програмну архітектуру системи керування з використанням ПК оператора, IP-камери OpenIPC/SSC30KQ, маршрутизатора MikroTik hEX S та локальної/VPN-мережі; реалізувати модульний застосунок оператора з підтримкою ONVIF-пошуку, RTSP-прийому, OSD-відображення, UDP-моста та обробки HID/Serial-входу; провести експериментальне тестування системи у LAN, SFP/оптоволоконному та VPN+Wi-Fi/LTE-сценаріях; оцінити працездатність системи за показниками RTT, packet loss, стабільності RTSP-потoku, коректності OSD-відображення, кількості декодованих кадрів і роботи UDP-моста.

Об'єкт дослідження: процес мережевого керування безпілотними літальними апаратами з використанням локальних та віддалених IP-мереж.

Предмет дослідження: методи, програмні модулі та мережева архітектура передавання відеопотоку, OSD/службової інформації і керуючих даних між персональним комп'ютером оператора та бортовим мережевим вузлом БПЛА.

Методи дослідження: системний аналіз для визначення структури системи, її модулів і потоків даних; порівняльний аналіз мережових, відеоінформаційних і сервісних технологій; програмне проєктування для побудови модульного застосунку оператора; експериментальне тестування для перевірки працездатності ONVIF, RTSP, OSD та UDP-моста; кількісне оцінювання для фіксації RTT, втрат пакетів, стану OSD, кількості отриманих відеопакетів, декодованих кадрів і навантаження на ПК.

Базою дослідження слугує тестовий програмно-апаратний стенд, що включає персональний комп'ютер оператора, IP-камеру OpenIPC/SSC30KQ, маршрутизатор MikroTik hEX S, HID/Serial-пристрої керування, локальну мережу, SFP/оптоволоконне підключення та VPN-сценарій. Для реалізації програмної частини використано Python, Kivy, PyAV/FFmpeg, onvif-zeep, а також протоколи ONVIF, RTSP, MSP DisplayPort, UDP, HID і CRSF.

Одержані результати можуть бути використані в навчальній, дослідній та інженерній діяльності для побудови лабораторних прототипів наземних станцій керування БПЛА. Розроблена система дозволяє об'єднати в одному середовищі прийом відеопотоку, службове OSD-відображення, передавання керуючих даних і збір статистики роботи каналів. Результати тестування підтвердили працездатність основних модулів у локальному сценарії та показали можливість подальшого розвитку системи для віддалених конфігурацій з використанням VPN, Wi-Fi або LTE.

Кваліфікаційна робота містить 46 сторінок, 12 таблиць, список використаних джерел із 43 найменувань.

Ключові слова: БПЛА, система керування, наземна станція, локальна мережа, інтернет-мережа, ONVIF, RTSP, OSD, MSP DisplayPort, UDP, VPN, HID, CRSF, Python, Kivy, OpenIPC, MikroTik.

Рік виконання кваліфікаційної роботи: 2026.
Рік захисту роботи: 2026.

ABSTRACT

Development of a Control System for Unmanned Aerial Vehicles in Local and Internet Networks

The purpose of the work: is to develop and experimentally test a software and hardware system for controlling unmanned aerial vehicles from a computer in a local and Internet network. The system provides detection of a network video device through ONVIF, reception of an RTSP video stream, OSD display of service information based on MSP DisplayPort, and transmission of control data through a Serial/HID-to-UDP bridge.

Research objectives: To analyze modern approaches to UAV control through local and remote IP networks; To study the features of video stream transmission, service OSD information, and control data in a network environment; To compare ONVIF, RTSP, MSP DisplayPort, UDP, VPN, HID, and CRSF technologies in terms of their application in a UAV control system; To define the functional and non-functional requirements for the software and hardware system; To design the hardware and software architecture of the control system using an operator PC, an OpenIPC/SSC30KQ IP camera, a MikroTik hEX S router, and a local/VPN network; To implement a modular operator application with support for ONVIF discovery, RTSP reception, OSD display, a UDP bridge, and HID/Serial input processing; To conduct experimental testing of the system in LAN, SFP/fiber-optic, and VPN+Wi-Fi/LTE scenarios; To evaluate the system performance according to RTT, packet loss, RTSP stream stability, correctness of OSD display, number of decoded frames, and operation of the UDP bridge.

The object of the research: is the process of network control of unmanned aerial vehicles using local and remote IP networks.

The subject of the research: is the methods, software modules, and network architecture for transmitting a video stream, OSD/service information, and control data between the operator's personal computer and the onboard network node of the UAV.

Research methods: System analysis to determine the structure of the system, its modules, and data flows; Comparative analysis of network, video information, and service technologies; Software design for developing a modular operator application; Experimental testing to verify the functionality of ONVIF, RTSP, OSD, and the UDP bridge; Quantitative evaluation to record RTT, packet loss, OSD status, the number of received video packets, decoded frames, and PC load.

The research is based on a test software and hardware stand that includes an operator's personal computer, an OpenIPC/SSC30KQ IP camera, a MikroTik hEX S router, HID/Serial control devices, a local network, an SFP/fiber-optic connection, and a VPN scenario. Python, Kivy, PyAV/FFmpeg, onvif-zeep, as well as ONVIF, RTSP, MSP DisplayPort, UDP, HID, and CRSF protocols were used to implement the software part.

The obtained results can be used in educational, research, and engineering activities for developing laboratory prototypes of UAV ground control stations. The developed system makes it possible to combine video stream reception, service OSD display, control data transmission, and channel operation statistics collection in a single environment. The test results confirmed the functionality of the main modules in the local scenario and demonstrated the possibility of further system development for remote configurations using VPN, Wi-Fi, or LTE.

The qualification work contains 46 pages, 12 tables, and a list of references with 43 sources.

Keywords: UAV, control system, ground station, local network, Internet network, ONVIF, RTSP, OSD, MSP DisplayPort, UDP, VPN, HID, CRSF, Python, Kivy, OpenIPC, MikroTik.

Year of completion of the qualification work: 2026.
Year of defense of the qualification work: 2026.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ.....	3
ВСТУП	6
РОЗДІЛ 1. ТЕОРЕТИКО-МЕТОДОЛОГІЧНІ ОСНОВИ ПОБУДОВИ СИСТЕМ МЕРЕЖЕВОГО КЕРУВАННЯ БПЛА	12
1.1. Сучасні підходи до керування БПЛА та вимоги до мережевого каналу	12
1.2. Порівняльний аналіз мережевих, відеоінформаційних і сервісних технологій	14
1.3. Обґрунтування вибору технологій і дослідницькі критерії	16
РОЗДІЛ 2. СИСТЕМНИЙ АНАЛІЗ ТА ПРОЄКТУВАННЯ СИСТЕМИ КЕРУВАННЯ БПЛА	19
2.1. Постановка задачі та вимоги до системи	19
2.2. Апаратна та програмна архітектура системи.....	21
2.3. Алгоритм роботи, модель потоків даних і методика експерименту	25
РОЗДІЛ 3. РЕАЛІЗАЦІЯ, ТЕСТУВАННЯ ТА ОЦІНЮВАННЯ СИСТЕМИ КЕРУВАННЯ БПЛА	28
3.1. Реалізація програмних модулів системи	28
3.2. Організація тестування системи.....	34
3.3. Аналіз результатів, обмеження та напрями вдосконалення.....	36
ВИСНОВКИ.....	41
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	43

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

БПЛА - безпілотний літальний апарат, що використовується для виконання польотних, спостережних або дослідних завдань без перебування пілота на борту.

ПК - персональний комп'ютер оператора, на якому запускається програмний застосунок наземної станції.

Bitrate - обсяг даних, що передається за одиницю часу у відеопотоці або мережевому каналі.

Bridge - програмний міст між локальним джерелом даних і мережевим каналом передавання.

Checksum - контрольна сума, що використовується для перевірки цілісності отриманого пакета.

COM/tty - позначення послідовних портів у Windows- та Linux-системах.

CRSF - протокол передавання RC-каналів і телеметрії, який використовується у FPV-системах керування.

DHCP - протокол автоматичного надання IP-адрес та інших мережевих параметрів пристроям у локальній мережі.

FFmpeg - набір бібліотек і утиліт для обробки мультимедійних потоків.

FPS - кількість кадрів за секунду, що характеризує плавність відеопотоку.

GUI - графічний інтерфейс користувача, через який оператор взаємодіє із системою.

H.264 - стандарт стискання відео, який дозволяє передавати відеопотік із прийнятним співвідношенням якості та бітрейту.

HID - клас пристроїв введення, до якого належать джойстики, пульти та інші контролери оператора.

IP - Internet Protocol, мережевий протокол адресації та передавання даних між вузлами комп'ютерної мережі.

Jitter - нерівномірність затримки пакетів у мережі, яка може спричинити ривки відео або нестабільне оновлення службових даних.

Kivy - Python-фреймворк для створення графічного інтерфейсу оператора.

LAN - локальна комп'ютерна мережа, що використовується для базової лабораторної перевірки системи.

Link Test Statistics - панель статистики каналів відео, OSD і керування у застосунку оператора.

LTE - стандарт мобільного зв'язку четвертого покоління, який розглядається як один із можливих інтернет-каналів.

MikroTik hEX S - маршрутизатор, що використовується для локальної мережі, DHCP-адресації, маршрутизації та SFP-сценарію.

MJPEG - формат відеопередавання як послідовності JPEG-зображень, що може бути зручним для простих UDP-сценаріїв.

MSP - MultiWii Serial Protocol, протокол обміну службовими даними з польотним контролером або сумісними модулями.

MSP DisplayPort - режим передавання OSD-символів через MSP для подальшого відображення службової інформації.

MTU - максимальний розмір пакета, який може бути переданий через мережевий канал без фрагментації.

NAT - механізм трансляції мережевих адрес, який застосовується під час маршрутизації між приватними та зовнішніми мережами.

ONVIF - стандарт взаємодії мережевих відеопристроїв, який використовується для пошуку камери та отримання її профілів.

OpenIPC - відкрита Linux-подібна прошивка для IP-камер, використана як основа мережевого відеовузла.

OSD - екранне службове відображення, що накладається поверх відео та містить телеметричні або діагностичні дані.

Packet loss - втрата пакетів під час передавання мережею, що впливає на стабільність відео, OSD та керування.

Payload - корисне навантаження пакета даних без службових заголовків.

PyAV - Python-обгортка над FFmpeg, яка використовується для відкриття RTSP-потoku та декодування кадрів.

RC - радіокерування або логічний канал ручного керування літальним апаратом.

RTP - протокол передавання аудіо- та відеоданих у реальному часі.

RTSP - протокол керування потоковим відео, що використовується для отримання відеопотоку з мережевої камери.

RTT - час проходження пакета до вузла призначення та повернення відповіді до відправника.

Serial - послідовний канал передавання даних, який може використовуватися для MSP, CRSF або службового обміну.

SFP - модульний інтерфейс для підключення оптичних або мідних мережевих ліній до маршрутизатора чи комутатора.

Shadow buffer - буфер останнього коректного стану OSD, який зменшує вплив короткочасної втрати пакетів.

Socket - програмний інтерфейс для мережевого обміну даними між процесами або пристроями.

SSC30KQ - апаратна платформа IP-камери, використана у тестовому стенді роботи.

Stream 0 / Stream 1 - основний та допоміжний RTSP-профілі відеопотоку, що можуть відрізнятися роздільною здатністю, FPS і бітрейтом.

Tailscale - VPN-рішення на базі WireGuard, що може використовуватися як альтернатива для віддаленого доступу.

TCP - транспортний протокол із підтвердженням доставки, контролем цілісності та повторним передаванням втрачених даних.

UART - апаратний послідовний інтерфейс обміну даними між пристроями.

UDP - транспортний протокол без встановлення з'єднання, який використовується для актуальних даних реального часу.

VPN - віртуальна приватна мережа, що дозволяє об'єднувати віддалені пристрої в одну захищену логічну мережу.

WAN - зовнішня або глобальна мережа передавання даних, через яку може здійснюватися віддалена взаємодія.

Wi-Fi - технологія бездротового мережевого зв'язку, яка може використовуватися як частина каналу між оператором і бортовим вузлом.

WS-Discovery - механізм автоматичного виявлення пристроїв у мережі, застосований для пошуку ONVIF-сумісної камери.

ZeroTier - VPN-рішення для створення приватної віртуальної мережі між віддаленими пристроями.

ВСТУП

Актуальність теми. Розвиток безпілотних літальних апаратів підвищує вимоги до систем дистанційного керування, які мають поєднувати відеоспостереження, службове відображення, мережеву діагностику та передавання керуючих даних. У типових конфігураціях ці функції часто розділені між кількома пристроями або програмами, що ускладнює перевірку стану каналу, зміну конфігурації та аналіз помилок.

Практична проблема полягає не лише у передаванні відео з камери на ПК оператора. Для реального застосування важливо, щоб відео, OSD-дані та керуючі повідомлення оброблялися незалежно, мали власні критерії працездатності та могли окремо тестуватися. Відеопотік залежить від пропускної здатності та декодування, OSD - від своєчасного оновлення службових символів, а керування - від регулярного надходження актуальних пакетів.

У роботі розглянуто програмно-апаратну систему, у якій ПК оператора працює як наземна станція, а IP-камера OpenIPC/SSC30KQ та мережевий вузол MikroTik забезпечують локальну або віддалену взаємодію. Система орієнтована на лабораторні та регламентовані сценарії перевірки, де можна оцінювати затримку, втрати пакетів, стабільність RTSP-потoku, роботу OSD і стан керуючого моста.

Метою кваліфікаційної роботи є розроблення та дослідна перевірка системи керування БПЛА з комп'ютера в локальній та інтернет-мережі, яка забезпечує ONVIF-виявлення мережевого відеопристрою, прийом RTSP-відео, OSD-відображення на основі MSP DisplayPort та передавання керуючих даних через Serial/HID-to-UDP bridge.

Об'єктом дослідження є процес мережевого керування безпілотними літальними апаратами з використанням локальних та віддалених IP-мереж.

Предметом дослідження є методи, програмні модулі та мережева архітектура передавання відеопотоку, OSD/службової інформації і керуючих даних між ПК оператора та бортовим мережевим вузлом.

Для досягнення мети необхідно виконати такі завдання: проаналізувати підходи до керування БПЛА через IP-мережі; порівняти технології відеопередавання, виявлення пристроїв, службового відображення та керування; сформулювати функціональні й нефункціональні вимоги; спроектувати апаратну та програмну архітектуру; реалізувати застосунок оператора; перевірити ONVIF-пошук, RTSP-прийм, OSD-модуль і UDP-міст; виконати експериментальну оцінку затримки, втрат пакетів, стабільності та навантаження на ПК; визначити обмеження і напрями подальшого вдосконалення.

Методи дослідження. У роботі використано системний аналіз для виділення модулів і потоків даних, порівняльний аналіз для вибору технологій, програмне проектування для реалізації модульного застосунку, експериментальне тестування для перевірки працездатності та кількісне оцінювання для фіксації RTT, packet loss, стану OSD, кількості декодованих кадрів і навантаження на ПК.

Наукова новизна роботи полягає у поєднанні ONVIF-виявлення, RTSP-прийому, MSP DisplayPort OSD, UDP-статистики та HID-to-CRSF перетворення в єдиній модульній системі керування БПЛА з ПК. На відміну від описового підходу, у роботі додатково введено критерії експериментальної перевірки та програмні лічильники для аналізу стану каналів.

Практична цінність полягає у створенні робочого прототипу наземного застосунку, який може використовуватися для перевірки локальних, оптоволоконних, Wi-Fi та VPN-сценаріїв у контрольованих умовах. Результати можуть бути використані як основа для подальшого розширення системи, додавання автоматичного журналювання, перевірки залежностей, візуалізації метрик і підтримки нових конфігурацій підключення.

Практична цінність зумовлена кількома ключовими факторами. Серед них: «1. Актуальність в умовах сучасних загроз та бойових завдань.

Ефективність роботи екіпажів БПЛА напряму залежить від зручності та надійності інструментів, якими вони користуються. У складних операційних умовах немає часу та можливості налаштовувати складні, дорогі чи прив'язані

до конкретних моделей обладнання системи. Потрібен універсальний інструмент, який працюватиме з різними камерами, різними каналами зв'язку, різними маршрутизаторами і мережею будь-якого рівня стабільності.

Система орієнтована саме на такі умови: швидкий старт роботи, мінімум залежностей, можливість використання на ноутбучі чи навіть компактному польовому міні-ПК. Це підвищує виживаність екіпажу і дозволяє переносити пункт керування ближче або далі залежно від задачі.

2. Гнучкість і адаптивність для різних конфігурацій техніки.

У практиці застосування БПЛА майже не буває двох однакових конфігурацій. Тут інший роутер, там інша камера, десь RTSP, десь JPEG-over-UDP, десь нестандартна телеметрія. Більшість існуючих рішень або вимагають довгих налаштувань, або не дозволяють швидко комбінувати кілька потоків відео, або мають жорсткі обмеження щодо пристроїв.

Проект вирішує такі проблеми за рахунок:

- підтримки кількох видів відеотрансляцій,
- можливості одночасного використання основного та допоміжного потоків (наприклад, режим нічного бачення),
- універсальної роботи з UDP, RTSP та іншими типами трафіку,
- автоматичного визначення наявності потоку,
- кастомізованих сценаріїв роботи залежно від режиму зв'язку.

Така адаптивність дає змогу створювати конфігурації “під себе” без переписування програми.

3. Кросплатформеність і висока портативність.

Окремою перевагою є незалежність від конкретної операційної системи чи встановленого середовища. Програмний комплекс поставляється у вигляді портативних збірок, які:

- не потребують інсталяції Python,
- не потребують встановлення GStreamer,
- не вимагають адміністраторських прав,
- можуть запускатись прямо з флешки,

- не залишають слідів у системі.

Це робить інструмент зручним у польових умовах та дає можливість швидко розгортати його на нових робочих місцях.

4. Зручність для оператора та зниження навантаження на екіпаж.

Робота оператора БПЛА у стресових умовах вимагає максимальної концентрації на ситуації в повітрі, а не на технічних налаштуваннях обладнання. Будь-які зайві дії, що змушують переключатися між інтерфейсами, вручну вводити параметри або перезапускати потоки, збільшують ризик помилок та втрати управління апаратом. Саме тому одним із ключових напрямків розробки стало створення інтерфейсу, який дозволяє оператору працювати інтуїтивно, без постійного залучення технічних знань і без необхідності виконувати рутинні дії.

Система створює умови, за яких оператор взаємодіє лише з найнеобхіднішими елементами, а вся технічна логіка залишається «під капотом» і працює автоматично. Це зменшує когнітивне навантаження та підвищує ефективність виконання завдань. У системі реалізовано:

- збереження конфігурацій для кожного локального користувача,
- окремі профілі підключення (локальний та мережевий),
- автоматичні підказки та заповнення налаштувань,
- швидке перемикання потоків без втрати керування,
- відображення стану зв'язку, телеметрії, FPS та бітрейту,
- стабільний перезапуск відеопотоків із мінімальною затримкою.

Завдяки цьому оператор може повністю зосередитися на управлінні польотом та прийнятті рішень, не відволікаючись на технічні нюанси.

5. Масштабованість і можливості подальшого розвитку.

Технологічний прогрес у сфері БПЛА відбувається настільки швидко, що будь-який програмний інструмент має бути здатним еволюціонувати разом із потребами користувачів. Саме тому архітектура системи створена модульною та розширюваною. Кожен компонент може модернізуватися незалежно, без необхідності переписувати всю програму. Це особливо важливо для інтеграції

нових типів камер, оновлених протоколів телеметрії або додаткових каналів відео.

Побудова системи з розрахунком на тривалу експлуатацію забезпечує гнучкість і дозволяє впроваджувати новий функціонал без втручання в критичні частини застосунку. Завдяки цьому він може адаптуватися до появи нових сценаріїв використання та технологічних новацій. Серед потенційних напрямків розвитку:

- інтеграція з додатковими типами камер та сенсорів,
- покращена обробка нічного бачення,
- нові протоколи телеметрії та зв'язку,
- автоматична стабілізація та корекція відео,
- аналітичні модулі для розпізнавання об'єктів,
- підтримка зовнішніх наземних станцій,
- можливість підключення кількох БПЛА одночасно.

Таким чином, проєкт має міцний фундамент для довгострокового розвитку та масштабування у різних середовищах.

6. Орієнтація на реальні польові потреби, а не лабораторну теорію.

Багато існуючих програмних рішень створюються у комфортних умовах лабораторій, де немає перешкод у мережі, де ідеальний зв'язок, стабільне живлення та стандартизоване обладнання. Але у реальному використанні оператори зіштовхуються з постійними збоями, втратами пакетів, перезавантаженням камер, обривами каналів, перемиканням між різними мережами та VPN-тунелями. Саме ці умови стають справжнім випробуванням для ПЗ, яке не має стійкої архітектури. Тому при створенні системи ключовим принципом була "польова стійкість": здатність працювати стабільно в умовах, де більшість програм просто відмовляються функціонувати. Логіка програми побудована так, щоб адаптуватися до змін, а не ламатися від них, забезпечуючи оператору безперервність роботи навіть за несприятливих обставин. Під час розробки враховано:

- втрати пакетів та нерівномірність трафіку,

- джиттер UDP-каналів,
- вимкнення та перезапуск камери без попередження,
- нестабільність польових маршрутизаторів,
- різні схеми маршрутизації між пристроями,
- використання Tailscale та інших VPN-рішень,
- затримки та «просідання» тунелів.

7. Потреба в доступному комплексі для команд з обмеженим бюджетом.

Професійні ґрунтовні станції та ПЗ коштують дорого. У багатьох команд їх просто немає. Низька ціна входу та можливість використання будь-якого доступного заліза (навіть з ринку вживаних ноутбуків) роблять комплекс важливим саме в ресурсно обмежених середовищах. Це дозволяє зменшити бар'єр доступу до якісного керування БПЛА» [1].

Кваліфікаційна робота складається зі вступу, трьох розділів, висновків, списку використаних джерел і додатків. Перший розділ містить теоретичне та порівняльне обґрунтування технологій, другий - системний аналіз і проєктування, третій - реалізацію, тестування та оцінювання результатів.

РОЗДІЛ 1. ТЕОРЕТИКО-МЕТОДОЛОГІЧНІ ОСНОВИ ПОБУДОВИ СИСТЕМ МЕРЕЖЕВОГО КЕРУВАННЯ БПЛА

Побудова системи мережевого керування БПЛА потребує не лише переліку протоколів, а й пояснення їхньої ролі у загальній архітектурі. Для системи керування важливо визначити, які дані передаються, які затримки є прийнятними, які втрати можуть бути компенсовані програмно, а які призводять до деградації роботи.

У межах цієї роботи система розглядається як модульний програмно-апаратний комплекс. Його завданням є не заміна всіх існуючих засобів керування БПЛА, а створення дослідної наземної станції, у якій оператор може одночасно бачити відео, службове OSD-відображення, стан мережі та канал керування.

1.1. Сучасні підходи до керування БПЛА та вимоги до мережевого каналу

Класичне RC-керування базується на спеціалізованому радіоканалі між пультом оператора та бортовим приймачем. Його перевагою є поширеність готових компонентів і зрозуміла логіка підключення. Водночас класичний підхід не завжди зручний для дослідних задач, де потрібно об'єднати відео, OSD, мережеві логи, зміну профілів потоку та тестування альтернативних джерел керування.

Наземна станція на базі ПК дає змогу перенести частину логіки до програмного середовища. У такій моделі ПК оператора не лише відображає відео, а й виконує ONVIF-пошук, відкриває RTSP-потік, приймає OSD-пакети, опитує COM/tty або HID-джерело та запускає мережевий міст керування. Це збільшує складність програмної частини, але робить систему більш придатною до діагностики.

Мережеве керування відрізняється від прямого радіоканального підходу наявністю IP-рівня. Фізичний канал може бути різним: LAN, SFP/оптоволокно,

Wi-Fi або VPN через інтернет. Для застосунку ці сценарії зводяться до роботи з IP-адресою, портом і протоколом, але з погляду якості вони мають різні затримки, втрати та ризики.

LAN-сценарій є найбільш придатним для лабораторної перевірки. У ньому оператор контролює підмережу, DHCP-адресацію, доступність камери, маршрутизацію та порти сервісів. Саме тому локальний настільний тест використано як базовий для оцінювання працездатності модулів.

SFP/оптоволоконний сценарій розглядається як стабільний фізичний канал між землею та бортовою стороною у регламентованих умовах. Його перевагою є стійкість до радіозавад та стабільна якість передавання за умови справної лінії. Обмеженням є механічний стан кабелю: перегин, пошкодження або заплутування можуть впливати на зв'язок.

Wi-Fi та VPN-сценарії залежать від якості зовнішнього каналу. У тестах VPN через Wi-Fi/LTE дав більшу варіативність RTT і втрати пакетів, тому такий режим розглядається як дослідний, а не як основний спосіб оцінювання працездатності.

Таблиця 1.1

Порівняння підходів до організації каналу керування та відео

Підхід	Сильні сторони	Обмеження	Доцільність у роботі
Класичне RC-керування	Готова апаратна екосистема, мала затримка, звична логіка для БПЛА	Слабка інтеграція з мережевим відео, OSD і програмною діагностикою	Розглядається як базовий контекст, але не як основна архітектура
ПК як наземна станція	Єдине середовище для відео, OSD, керування та діагностики	Потребує розробки застосунку й контролю навантаження на ПК	Основний підхід у роботі

LAN/SFP локальна мережа	Контрольована адресація, низький RTT, стабільність у лабораторних умовах	Залежність від фізичного підключення та стану кабелю	Базовий тестовий сценарій
VPN/інтернет	Віддалений доступ без прямого відкриття портів	Змінна затримка, MTU, залежність від провайдера	Дослідний сценарій для подальшого розвитку

Джерело: сформовано автором.

1.2. Порівняльний аналіз мережевих, відеоінформаційних і сервісних технологій

Для відеопередавання у роботі обрано RTSP, оскільки OpenIPC-камера вже надає RTSP-потоки, а PyAV дозволяє інтегрувати їх у Python-застосунок без окремого медіасервера. Альтернативи, такі як WebRTC, MJPEG, raw UDP або GStreamer, мають власні переваги, але у межах цього прототипу поступаються за простотою інтеграції, відтворюваністю або кросплатформною підтримкою.

RTSP не є універсальним протоколом для всієї системи. Він відповідає лише за відео. Керуючі повідомлення та OSD передаються окремо, оскільки старі відеокадри, старі OSD-символи й старі команди мають різну цінність для оператора. Для реального часу важливо не накопичувати чергу застарілих даних.

ONVIF/WS-Discovery використано для зменшення ручного налаштування. У локальній мережі або VPN-підмережі оператор не повинен кожного разу шукати IP-адресу камери вручну. Модуль пошуку виявляє сумісний пристрій, перевіряє доступність сервісу та, якщо можливо, отримує RTSP URI профілю.

Для OSD застосовано MSP DisplayPort. У цій моделі службові символи не вбудовуються в RTSP-потік на камері, а приходять окремими UDP-пакетами та накладаються на стороні ПК. Це дає змогу змінювати логіку OSD без зміни відеомодуля та окремо оцінювати втрати OSD-пакетів.

UDP використано для OSD і керуючих даних через низькі накладні витрати та орієнтацію на актуальний стан. Для OSD втрата окремого пакета компенсується shadow buffer, а для керування важливо, щоб нові актуальні дані не блокувалися повторною доставкою старих повідомлень. Водночас це рішення потребує статистики і контролю помилок, тому в оновленій реалізації додано лічильники UDP-пакетів, помилок сокета та не оброблених повідомлень.

Таблиця 1.2

Порівняння альтернатив для відео, інтерфейсу та керування

Компонент	Обране рішення	Альтернативи	Причина вибору
Відео	RTSP + PyAV	WebRTC, MJPEG, raw UDP, GStreamer	RTSP уже підтримується OpenIPC, PyAV працює у Python і доступний для Linux/Windows/macOS
GUI	Kivy	PyQt, Tkinter, веб-інтерфейс	Швидка побудова прототипу, кросплатформність, пряме поєднання з Python-модулями; обмеження layout компенсовано окремою логікою вікна

OSD	MSP DisplayPort через UDP	Вбудоване OSD у відео, власний формат	Окреме тестування OSD, сумісність з FPV-логікою службових символів
Керування	HID/Serial-to- UDP, CRSF- подібні кадри	MAVLink, TCP bridge, WebSocket	Польотні контролери часто очікують CRSF- подібні RC-канали; UDP не накопичує старі команди
Віддалений доступ	VPN/ZeroTier як дослідний режим	Tailscale, WireGuard, проброс портів	VPN зменшує потребу відкривати службові порти; конкретний вибір залежить від підтримки маршрутизатора

Джерело: сформовано автором.

1.3. Обґрунтування вибору технологій і дослідницькі критерії

Вибір технологій у роботі визначався не лише доступністю бібліотек, а й тим, чи можна виміряти працездатність відповідного модуля. Для ONVIF критерієм є час виявлення пристрою та правильність отриманих параметрів. Для RTSP - стабільність потоку, FPS, кількість отриманих відеопакетів, декодованих кадрів і помилок декодування. Для OSD - кількість отриманих UDP-пакетів, валідних MSP-пакетів і не оброблених фрагментів. Для керування - наявність постійного потоку CRSF-даних і помилки сокета.

Python обрано як мову реалізації через доступність бібліотек для мережі, відео, GUI, послідовних портів і HID-пристроїв. Обмеження Python, зокрема багатопотоковість і навантаження при декодуванні відео, враховано через

винесення RTSP, OSD і bridge у окремі потоки та використання останнього доступного кадру замість накопичення черги.

PyAV використовується як обгортка над FFmpeg для відкриття RTSP-потoku, демультимплексування пакетів і декодування кадрів у RGB-масив. У реалізації задано параметри, орієнтовані на низьку затримку: UDP-транспорт, відключення буферизації, low_delay, max_delay=0 і тайм-аут відкриття потоку.

Kivy використано для побудови інтерфейсу оператора, вибору пристрою, запуску потоків, перемикання stream=0/stream=1 і відображення статистики. Хоча Kivy не є спеціалізованим фреймворком для наземних станцій, він достатній для прототипу та дозволяє працювати у Python без окремої веб-частини.

HID-to-CRFS є важливим компонентом, оскільки дозволяє використовувати HID-пристрої як джерело RC-каналів без додаткового апаратного перетворювача. Значення осей і кнопок нормалізуються у діапазон CRFS, пакуються у 11-бітні поля та формують кадр з контрольною сумою. Це переводить роботу з джойстиком із рівня “натиснута кнопка” на рівень структурованого потоку керування.

Таблиця 1.3

Метрики, які використовуються для оцінювання модулів

Модуль	Метрики	Очікувана інтерпретація
Мережевий канал	RTT min/avg/max, packet loss	Оцінка затримки та стабільності маршруту
RTSP Player	video_packets_received, decoded_frames, decode_errors, FPS	Оцінка стабільності відео та декодування
OSD/MSP	udp_packets_received, valid_msp_packets, not_processed_packets, delivery estimate	Оцінка коректності службового відображення

Serial/HID-to-UDP bridge	udp_packets_sent, udp_packets_received, socket_errors	Оцінка каналу керування та помилок мережі
ПК оператора	CPU/GPU load, стабільність інтерфейсу	Оцінка придатності платформи до реального часу

Джерело: сформовано автором.

Висновки до розділу 1

У першому розділі розглянуто теоретичні та порівняльні основи побудови системи мережевого керування БПЛА. Показано, що для дослідного прототипу доцільним є підхід наземної станції на ПК, де відео, OSD, керування та статистика об'єднані в одному програмному середовищі.

Обґрунтовано використання RTSP для відео, ONVIF/WS-Discovery для виявлення камери, MSP DisplayPort для OSD, UDP для актуальних службових і керуючих повідомлень, а також HID-to-CRSF для перетворення дій оператора у канали керування.

На відміну від суто описового підходу, у розділі визначено метрики, за якими надалі оцінюються модулі: RTT, packet loss, кількість отриманих і декодованих відеокадрів, валідність MSP-пакетів, помилки сокета та навантаження на ПК. Це формує основу для експериментальної частини роботи.

РОЗДІЛ 2. СИСТЕМНИЙ АНАЛІЗ ТА ПРОЄКТУВАННЯ СИСТЕМИ КЕРУВАННЯ БПЛА

У другому розділі система розглядається як інженерний прототип з вимірюваними характеристиками. Основна увага приділяється постановці задачі, апаратній конфігурації, програмній архітектурі, потокам даних, критеріям успішності та методиці тестування.

2.1. Постановка задачі та вимоги до системи

Задача проєктування полягає у створенні застосунку оператора, який працює з камерою OpenIPC/SSC30KQ і мережевим вузлом MikroTik hEX S у локальній або віддаленій IP-мережі. Система має забезпечити пошук пристрою, вибір RTSP-профілю, декодування відео, накладання OSD, прийом HID/Serial-входу та передавання керуючих даних на UDP-порт віддаленого вузла.

Важливою вимогою є відтворюваність. Тому в роботі фіксуються не лише загальні назви протоколів, а й параметри тестового стенда: підмережа MikroTik 192.168.88.0/24, ONVIF-порт 85, RTSP-порт 554, OSD UDP-порт 2002, керуючий UDP-порт 2001, serial-параметри 420000n81 для CRSF і 115200n81 для MSP/OSD.

Функціональні вимоги охоплюють ONVIF-пошук, ручне введення IP-адреси, перемикання потоків stream=0/stream=1, запуск RTSP-плеєра, запуск OSD-клієнта, вибір COM/tty або HID-джерела, запуск UDP-моста та відображення статистики. Нефункціональні вимоги охоплюють низьку затримку, стабільність при втраті окремих пакетів, безпечне поводження з конфігураціями, переносимість і можливість окремої діагностики кожного модуля.

Таблиця 2.1

Функціональні вимоги до системи

№	Вимога	Зміст
1	Пошук пристрою	ONVIF/WS-Discovery і fallback-перевірка доступності у заданій підмережі
2	Ручний режим	Введення IP-адреси, якщо автоматичне виявлення не спрацювало
3	RTSP-прийом	Відкриття stream=0 або stream=1, декодування кадрів через PyAV
4	OSD	Прийом MSP DisplayPort через UDP, оновлення OSDGrid, shadow buffer
5	Керування	Прийом COM/tty або HID, формування/передавання CRSF-подібного потоку
6	Статистика	Відображення лічильників UDP, MSP, відеопакетів, декодованих кадрів і помилок
7	Модульність	Можливість тестувати відео, OSD і керування окремо

Джерело: сформовано автором.

Таблиця 2.2

Нефункціональні вимоги та способи перевірки

Вимога	Критерій перевірки	Засіб фіксації
Низька затримка	RTT, візуальна оцінка затримки відео, відсутність накопичення кадрів	ping, PyAV-статистика, візуальний тест
Стабільність OSD	Частка валідних MSP-пакетів, коректність сітки	OSDStats, shadow buffer

Стабільність керування	Постійне надходження CRSF-подібних пакетів, відсутність socket_errors	BridgeStats
Переносимість	Можливість запуску на Linux і потенційно на інших ОС	Python/Kivy/PyAV, відсутність жорсткої прив'язки до одного каналу
Безпека конфігурацій	Відсутність реальних секретів у публічному тексті	Маскування паролів і ключів
Діагностованість	Окремий статус для відео, OSD і bridge	Панель Link Test Statistics

Джерело: сформовано автором.

2.2. Апаратна та програмна архітектура системи

Основним середовищем роботи ПК оператора є Linux. Тестування проводилося на ноутбукі Asus ROG Zephyrus G14 з процесором Ryzen 7 4800HS, графічним адаптером RTX 2060 Max-Q та 16 ГБ DDR4. За результатами спостереження, навантаження на CPU/GPU під час роботи відео та OSD становило до приблизно 20 %, що є прийнятним для лабораторного прототипу.

Мережевий вузол MikroTik hEX S використовується для DHCP-адресації, маршрутизації, LAN-підключення та можливого SFP/оптоволоконного сценарію. У базовому тесті камера отримує адресу у підмережі 192.168.88.0/24, після чого застосунок оператора виконує пошук або ручне підключення.

Камера OpenIPC/SSC30KQ розглядається як Linux-подібний IP-вузол. У тестовій конфігурації ONVIF працює на порту 85, RTSP-профілі мають шляхи /stream=0 і /stream=1. Перший профіль може використовуватися для якіснішого відео, другий - для меншої затримки або роботи через нестабільний канал. Для

інтернет-сценарію доцільно використовувати 720p, 30 FPS і бітрейт близько 1500 кбіт/с, оскільки LTE/VPN може давати втрати та фризи.

На стороні камери або бортового вузла ser2net може проксувати послідовні інтерфейси у UDP. У тестовій конфігурації керуючий канал використовує UDP-порт 2001 і serial-параметри 420000n81, а OSD/MSP - UDP-порт 2002 і 115200n81. У публічному тексті паролі наведено як тестові або масковані значення, оскільки виробничі секрети не повинні потрапляти до документації.

Таблиця 2.3

Відтворювана конфігурація тестового стенда

Компонент	Параметри	Призначення
ПК оператора	Linux, Ryzen 7 4800HS, RTX 2060 Max-Q, 16 ГБ RAM	Запуск Python/Kivy-застосунку, декодування RTSP, OSD, bridge
Маршрутизатор	MikroTik hEX S, підмережа 192.168.88.0/24, DHCP	Локальна мережа, маршрутизація, LAN/SFP
Камера	OpenIPC/SSC30KQ, ONVIF port 85, RTSP port 554	Джерело відео і мережевий вузол
RTSP профілі	/stream=0, /stream=1; H.264; приклад: 1080p/60 FPS і 720p/30 FPS	Перемикання між якістю та меншою вимогливістю до каналу
OSD	UDP 2002, MSP DisplayPort, 115200n81	Службове відображення поверх відео
Керування	UDP 2001, CRSF-подібний потік, 420000n81	Передавання RC-каналів до віддаленого вузла
Віддалений доступ	VPN/ZeroTier як дослідний сценарій	Робота через приватну мережу без прямого відкриття портів

Джерело: сформовано автором.

Програмна архітектура складається з окремих модулів. Controller зберігає стан застосунку, вибрану IP-адресу, список пристроїв, RTSP-параметри, OSD-порт і посилання на RtspPlayer, MSPDisplayPortClient та SerialToNetBridge. UI не працює напряду з сокетми або відеоконтейнерами, а викликає методи Controller, що зменшує зв'язність.

Таблиця 2.4

Основні програмні модулі системи

Модуль	Роль у системі	Оцінювані параметри
main.py / UI	Запуск Kivy, меню, відеоекран, кнопки stream і statistics	Коректність переходів, доступність статусів
Controller	Координація ONVIF, RTSP, OSD, bridge	Стан selected_ip, статуси модулів
onvif_discovery.py	WS-Discovery, перевірка порту, отримання профілів	Час пошуку, кількість знайдених пристроїв
rtsp_player.py	PyAV RTSP reader, останній RGB-кадр, VideoStats	FPS, пакети, кадри, помилки
mvp_osd.py	UDP MSP DisplayPort, OSDGrid, shadow buffer, OSDStats	Валідні MSP-пакети, втрати, помилки
hid_crsf.py	HID-порти, нормалізація осей, пакування CRSF	Частота кадрів, коректність каналів
serial_bridge.py	Двосторонній Serial/HID-to-UDP bridge, BridgeStats	UDP sent/received, socket_errors

Джерело: сформовано автором.

Для уточнення практичної реалізації програмної архітектури нижче наведено структуру проєкту, яка відображає поділ системи на модулі запуску, координації, мережевого виявлення, прийому відео, OSD-відображення та керуючого моста. Такий поділ дозволяє тестувати окремі частини системи незалежно та зменшує зв'язність між графічним інтерфейсом, відеомодулем і мережевими компонентами.

Структура програмного проєкту

```
launcher/  
  main.py  
  controller.py  
  onvif_discovery.py  
  config.py  
  control/  
    serial_bridge.py  
    hid_crsf.py  
  video/  
    rtsp_player.py  
  osd/  
    msp_osd.py  
    osd_grid.py  
  desg/  
    menu_screen.py  
    video_screen.py  
    widgets.py  
  assets/  
    fonts/  
    icons/  
    osd_glyphs/
```

2.3. Алгоритм роботи, модель потоків даних і методика експерименту

Алгоритм роботи починається із запуску застосунку та вибору мережевого пристрою. Оператор може виконати ONVIF-сканування або ввести IP вручну. Після вибору пристрою запускається RTSP-потік, OSD-клієнт і, за потреби, міст керування. Дані не зводяться в один універсальний канал: відео, OSD і керування мають окремі модулі та окремі лічильники стану.

Експериментальна методика передбачає п'ять запусків для кожного основного сценарію. Для ONVIF критерієм успіху є виявлення або ручне підключення до камери після її ініціалізації, максимальний зафіксований час якої становив до 15 секунд. Для RTSP критерієм є робота потоку без обриву протягом тестового інтервалу. Для OSD - коректне оновлення символів і висока частка валідних MSP-пакетів. Для керування - регулярне надходження CRSF-подібних пакетів без помилок сокета.

Оцінювання мережі проводиться через ping і статистику застосунку. Для LAN-тесту тривалістю близько 1 хвилини зафіксовано 68 переданих і 68 отриманих пакетів, packet loss 0 %, RTT min/avg/max = 1,422/1,750/3,523 мс. Для VPN+Wi-Fi/LTE-тесту зафіксовано 90 переданих, 89 отриманих пакетів, packet loss 1 %, RTT min/avg/max = 17/28/104 мс. Це демонструє, що локальний сценарій є значно стабільнішим, а VPN-сценарій потребує окремого налаштування MTU та якості каналу.

Таблиця 2.5

Сценарії та критерії експериментальної перевірки

Сценарій	Що перевіряється	Критерій успішності
LAN на столі	DHCP, ONVIF, RTSP, OSD, UDP bridge	0 % втрат ping, стабільне відео, коректний OSD

SFP/оптика	Фізична стабільність каналу, відсутність радіозавад	Стабільна картинка за справної оптичної лінії
Wi-Fi/VPN	Робота через віддалений маршрут	Допустимі втрати, відсутність критичного зависання, контроль MTU
RTSP stream=0/1	Якісний і легший профіль відео	Можливість перемикання профілів у застосунку
OSD/MSP	Прийом і розбір MSP DisplayPort	Валідні MSP-пакети, shadow buffer при втратах
HID/Serial-to-UDP	Передавання CRSF-подібного потоку	Пакети йдуть регулярно, socket_errors = 0

Джерело: сформовано автором.

Окрема увага приділяється ситуації, коли відео може залишатися активним, а канал керування - зупинитися або працювати з помилкою. Для цього у VideoScreen додано панель Link Test Statistics, яка показує статистику Control UDP bridge, OSD UDP stream і Video stream, а також ознаку “Video active while control is down”. Це перетворює застосунок з простого вікна відео на інструмент діагностики.

Висновки до розділу 2

У другому розділі сформульовано задачу створення модульної IP-системи керування БПЛА з ПК та визначено функціональні й нефункціональні вимоги до неї. На відміну від загального опису, вимоги прив’язано до конкретних критеріїв перевірки: RTT, втрати пакетів, OSD-статистика, відеостатистика та стан bridge.

Описано апаратну конфігурацію з ПК оператора, MikroTik hEX S, OpenIPC/SSC30KQ, LAN/SFP-з'єднанням і VPN-сценарієм. Зафіксовано ключові параметри відтворюваності: підмережа 192.168.88.0/24, порти 85, 554, 2001 і 2002, профілі /stream=0 і /stream=1, а також serial-параметри для CRSF і MSP.

Розроблено програмну архітектуру із Controller, ONVIF Discovery, RtspPlayer, MSPDisplayPortClient/OSDGrid, SerialToNetBridge і HID-to-CRSF converter. У проєкт закладено не лише запуск модулів, а й збір статистики, що підвищує дослідницьку цінність роботи.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ, ТЕСТУВАННЯ ТА ОЦІНЮВАННЯ СИСТЕМИ КЕРУВАННЯ БПЛА

Третій розділ присвячено реалізації програмних модулів, експериментальній перевірці та аналізу результатів. Основний акцент зроблено не на описі кнопок інтерфейсу, а на тих частинах коду, які забезпечують роботу з протоколами, буферами, статистикою та реальними потоками даних.

3.1. Реалізація програмних модулів системи

Застосунок реалізовано мовою Python із використанням Kivy для графічного інтерфейсу. Головний модуль створює Controller, меню та відеоекран. Controller ініціалізує SerialToNetBridge, RtspPlayer, OSDGrid і MSPDisplayPortClient, зберігає вибрану IP-адресу, OSD-порт і параметри RTSP. Така організація дозволяє не змішувати логіку інтерфейсу з низькорівневою роботою мережі.

ONVIF Discovery використовує WS-Discovery, TCP-перевірку порту та onvif-zeep для отримання профілів. Якщо профілі доступні, система може отримати RTSP URI; якщо ні, використовується fallback-формування RTSP URL за IP-адресою, логіном, паролем і шляхом /stream=0 або /stream=1.

RtspPlayer відкриває RTSP-потік через PyAV. У коді використовуються параметри, орієнтовані на низьку затримку: `rtsp_transport=udp`, `fflags=nobuffer`, `flags=low_delay`, `max_delay=0`. Плеєр не накопичує чергу кадрів для відображення, а зберігає останній доступний RGB-кадр. Додатково реалізовано VideoStats: кількість отриманих відеопакетів, декодованих кадрів, не оброблених пакетів і помилок декодування.

Лістинг 3.1 - Параметри RTSP-прийому з орієнтацією на низьку затримку

```

options = {
    "rtsp_transport": "udp",
    "fflags": "nobuffer",
    "flags": "low_delay",
    "max_delay": "0",
    "stimeout": "5000000",
    "threads": "0",
}
container = av.open(rtsp_url, options=options)

```

Окремим елементом реалізації є підрахунок статистики відеопотоку без втручання в основний цикл відображення. Під час прийому пакетів система збільшує лічильник отриманих відеопакетів, фіксує помилки декодування та окремо рахує кількість кадрів, які були успішно перетворені у RGB-формат для подальшого відображення в інтерфейсі.

Лістинг 3.1.1 - Статистика відеопотоку у RtspPlayer

```

self._count("video_packets_received")
try:
    decoded_frames = packet.decode()
except Exception:
    self._count("decode_errors")
    self._count("not_processed_packets")
    continue
for frame in decoded_frames:
    img = frame.to_ndarray(format="rgb24")
    self._last_frame = img
    self._count("decoded_frames")

```

OSD-модуль приймає UDP-пакети, накопичує їх у буфері та розбирає як MSP DisplayPort. Парсер перевіряє сигнатуру пакета, напрям, довжину, команду, payload і XOR-checksum. Валідні MSP-пакети передаються до обробника команд, а пошкоджені або неочікувані фрагменти враховуються у `not_processed_packets`.

Лістинг 3.2 - Розбір MSP DisplayPort-пакета з UDP-буфера

```
def _consume(self, buf: bytearray):
    i = 0
    while i + 6 <= len(buf):
        if buf[i] != 36 or buf[i + 1] != 77: # '$' 'M'
            self._count("not_processed_packets")
            i += 1
            continue
        direction = buf[i + 2]
        if direction not in (60, 62): # '<' or '>'
            self._count("not_processed_packets")
            i += 1
            continue
        length = buf[i + 3]
        cmd = buf[i + 4]
        frame_len = 6 + length
        if i + frame_len > len(buf):
            break
        payload = bytes(buf[i + 5:i + 5 + length])
        checksum = buf[i + 5 + length]
        calc = length ^ cmd
        for b in payload:
            calc ^= b
        if calc != checksum:
            self._count("not_processed_packets")
```

```

        i += 1
        continue
    self._count("valid_msp_packets")
    self._handle_msp(cmd, payload)
    i += frame_len
if i > 0:
    del buf[:i]

```

Перевірка checksum є необхідною для відсіювання пошкоджених або зміщених фрагментів UDP-буфера. Якщо контрольна сума не збігається, пакет не передається до обробника OSD-команд, а враховується у `not_processed_packets`. Це дозволяє не тільки уникнути некоректного відображення службових символів, а й оцінювати якість прийому OSD-даних кількісно.

Лістинг 3.2.1 - Перевірка checksum у MSP DisplayPort parser

```

calc = length ^ cmd
for b in payload:
    calc ^= b
if calc != checksum:
    self._count("not_processed_packets")
    i += 1
    continue
self._count("valid_msp_packets")

```

Для стійкості OSD до втрати пакетів застосовано shadow buffer. Якщо новий кадр OSD порожній або пакет втрачено, система тимчасово показує останній коректний стан. Якщо він старіший за 2 секунди, OSD очищується, щоб оператор не бачив застарілу службову інформацію.

Лістинг 3.3 - Shadow buffer для зменшення впливу втрати OSD-пакетів

```
def present(self):
    with self._lock:
        has_data = np.any(self.back)
        now = time.time()
        if has_data:
            self.front[:, :] = self.back
            self.shadow_buffer[:, :] = self.back
            self.last_valid_ts = now
        else:
            if now - self.last_valid_ts < 2.0:
                self.front[:, :] = self.shadow_buffer
            else:
                self.front[:, :] = 0
```

HID-to-CRSF converter дозволяє використовувати сумісні HID-пристрої як джерело каналів керування. У тестах використовувалися EdgeTX-пульт, пульт від Mavic і контролер PlayStation 5. Осі нормалізуються у діапазон CRSF_MIN...CRSF_MAX, а 16 каналів пакуються у 11-бітні поля. Частота генерації кадрів становить близько 150 Гц, що відповідає потребі регулярного потоку керуючих даних.

Лістинг 3.4 - Пакування 16 CRSF-каналів у 11-бітні поля

```
def pack_crsf_channels(channels: list[int]) -> bytes:
    dest = bytearray(22)
    bits_merged = 0
    bits_merged_count = 0
    byte_idx = 0
    for channel in channels[:CRSF_FRAME_CHANNELS]:
        channel = _clamp(channel, CRSF_MIN, CRSF_MAX)
```

```

        bits_merged |= (channel & 0x7FF) <<
bits_merged_count
        bits_merged_count += 11
        while bits_merged_count >= 8:
            dest[byte_idx] = bits_merged & 0xFF
            bits_merged >>= 8
            bits_merged_count -= 8
            byte_idx += 1
    return bytes(dest)

```

SerialToNetBridge реалізує двосторонню передачу між локальним джерелом і мережею. У режимі UDP socket підключається до remote_host:remote_port, після чого окремі потоки читають дані з serial/HID і передають їх у мережу, а також приймають відповідь з мережі та записують у serial. В оновленій версії додано BridgeStats: sent/received UDP packets, not_processed_packets і socket_errors.

Лістинг 3.5 - Фрагмент Serial/HID-to-UDP bridge зі статистикою

```

def pump_serial_to_net():
    while not self._stop.is_set():
        try:
            data = ser.read(4096)
            if data:
                try:
                    if protocol == "tcp":
                        sock.sendall(data)
                    else:
                        sock.send(data)
                        self._count("udp_packets_sent")
                except OSError:

```

```

        self._count("socket_errors")
        self._count("not_processed_packets")
        raise

    else:
        time.sleep(0.001)
except Exception as e:
    _fail(e)
    break

```

У VideoScreen додано панель Link Test Statistics. Вона виводить статистику Control UDP bridge, OSD UDP stream і Video stream: кількість UDP-пакетів, валідних MSP-пакетів, не оброблених фрагментів, socket errors, отриманих відеопакетів, декодованих кадрів і помилок декодування. Це усуває попередню проблему, коли результат тесту описувався лише суб'єктивно.

3.2. Організація тестування системи

Тестування проводилося у кількох сценаріях: LAN на столі, SFP/оптика, VPN+Wi-Fi/LTE. Кожний сценарій запускався по п'ять разів, а окремі вимірювання тривали близько однієї хвилини. У базовій схемі ноутбук підключався до MikroTik через LAN, камера отримувала адресу DHCP у підмережі 192.168.88.0/24, після чого застосунок запускав ONVIF, RTSP, OSD і, за потреби, bridge.

Критерієм успішного ONVIF-підключення було знаходження камери або ручне підключення після її ініціалізації. Максимальний час ініціалізації камери в тестовому середовищі становив до 15 секунд. RTSP-потік вважався працездатним, якщо відео відображалося без обриву протягом тестового інтервалу. OSD вважався працездатним, якщо символи коректно накладалися на відео, а MSP-пакети проходили перевірку. Керування вважалося працездатним, якщо CRSF-подібні пакети надходили регулярно і не фіксувалися socket_errors.

Для оцінювання відеозатримки застосовано візуальний метод: рух руки або таймер перед камерою порівнювався з відображенням у застосунку. Такий метод не є точним лабораторним вимірюванням, але дозволяє оцінити порядок затримки. За спостереженням, затримка становила приблизно 100-200 мс. У подальшому доцільно реалізувати окремий модуль точного timestamp-вимірювання.

Для оцінювання навантаження використовувався ноутбук Asus ROG Zephyrus G14. Обробка RTSP виконувалася переважно процесором через PyAV/FFmpeg, а накладання OSD і відображення задіювали графічну частину інтерфейсу. Загальне навантаження під час тестів оцінювалося приблизно до 20 %, що залишає запас для додаткової діагностики.

Таблиця 3.1

Умови тестування системи

Параметр	Значення
Кількість запусків на сценарій	5
Тривалість окремого мережевого тесту	близько 1 хвилини
Базова підмережа	192.168.88.0/24
ONVIF-порт	85
RTSP-профілі	/stream=0, /stream=1
OSD UDP-порт	2002
Control UDP-порт	2001
CRSF serial	420000n81
MSP/OSD serial	115200n81
Тестові HID-пристрої	EdgeTX-пульт, Mavic-пульт, PlayStation 5 controller

Джерело: сформовано автором.

Для конкретизації методики тестування окремо виділено сценарії перевірки, тривалість запусків, ключові метрики та очікуваний результат. Це

дозволяє відокремити загальні умови тестування від практичних перевірок окремих каналів: відео, OSD та керування.

Таблиця 3.2

Деталізація сценаріїв експериментальної перевірки

Сценарій	Тривалість/запуски	Метрики	Очікуваний результат
LAN на столі	≈1 хв, 5 запусків	RTT, packet loss, RTSP, OSD	0 % втрат, стабільний відеопотік
VPN+Wi-Fi/LTE	≈1 хв, 5 запусків	RTT, packet loss, фриз	Оцінка впливу нестабільного інтернету
OSD/MSP	під час відеотесту	valid_msp_packets, not_processed_packets	Коректне накладання службових символів
HID/Serial-to-UDP	під час керування	udp_packets_sent, socket_errors	Постійний потік CRSF-подібних пакетів

Джерело: сформовано автором.

3.3. Аналіз результатів, обмеження та напрями вдосконалення

Результати тестування показали, що локальний LAN-сценарій є найстабільнішим. Ping до камери 192.168.88.252 протягом тесту показав 68 переданих і 68 отриманих пакетів, 0 % втрат і $RTT_{min/avg/max/mdev} = 1,422/1,750/3,523/0,456$ мс. Це підтверджує, що локальна мережа придатна для базової перевірки програмної архітектури.

VPN+Wi-Fi/LTE-сценарій показав нижчу стабільність: 90 пакетів передано, 89 отримано, 1 % втрат, RTT min/max/avg = 17/104/28 мс. У цьому режимі спостерігалися пропуски кадрів і підлагування відео. Причиною є змінна якість інтернет-каналу, а також можливі проблеми з MTU у VPN-сценарії. Тому VPN-режим у роботі визначено як дослідний і такий, що потребує окремого налаштування.

Панель Link Test Statistics дала змогу перейти від суб'єктивної оцінки до кількісної. В одному з тестів OSD UDP stream зафіксував 4230 отриманих UDP-пакетів, 19905 валідних MSP-пакетів, 46 не оброблених фрагментів, 0 socket errors і delivery estimate 99,8 % (19905 з 19951). Video stream зафіксував 980 отриманих відеопакетів, 958 декодованих кадрів і 22 не оброблені пакети. Ці значення показують, що система не лише відображає відео, а й може фіксувати якість роботи модулів.

Таблиця 3.3

Кількісні результати тестування

Показник	LAN	VPN+Wi-Fi/LTE	Коментар
Ping packets	68/68	89/90	LAN без втрат, VPN має 1 % loss
RTT min/avg/max	1,422/1,750/3,523 мс	17/28/104 мс	VPN має значно більшу варіативність
Тривалість тесту	≈1 хв	≈1 хв	Однаковий порядок вимірювання
OSD valid MSP	19905 з 19951	не фіксувалося окремо	Delivery estimate 99,8 % у тесті статистики
OSD socket errors	0	не фіксувалося окремо	Помилки сокета не зафіксовано

Video packets / decoded frames	980 / 958	нестабільно через LTE	Для LAN відео декодувалося стабільно
Оцінка відеозатримки	≈100-200 мс	залежить від каналу	Візуальна оцінка, потребує точного модуля
Навантаження ПК	до ≈20 % CPU/GPU	залежить від профілю	Тестовий ноутбук мав достатній запас

Джерело: сформовано автором.

Для відео передбачено два профілі. Stream 0 може використовувати якісніший режим, наприклад 1080p/60 FPS з вищим бітрейтом, а stream 1 - легший режим для нестабільного каналу, наприклад 720p/30 FPS і близько 1500 кбіт/с. Такий поділ дозволяє оператору адаптувати відео до умов мережі без зміни архітектури застосунку.

Тестування OSD показало, що службові символи відображаються коректно, хоча іноді можливе візуальне миготіння через особливості прозорості самих glyph-ресурсів. Це не є помилкою транспортного рівня, але потребує доопрацювання графічної частини OSD.

Канал керування працює як постійний потік, що важливо для CRSF-логіки: якщо пакети перестають надходити, польотний контролер може перейти у стан втрати зв'язку. Саме тому bridge має не тільки передавати дані, а й фіксувати socket_errors і not_processed_packets. У тестах керуючі дані не губилися у базовому локальному сценарії.

Обмеження системи та напрями вдосконалення

Обмеження	Вплив	Напрямок вдосконалення
Повноцінне польове випробування не проводилося	Результати підтверджують лабораторну, а не повну експлуатаційну готовність	Провести регламентовані польові тести з журналюванням
Відеозатримка вимірювалася візуально	Точність оцінки 100-200 мс обмежена	Додати timestamp/LED/камерний метод вимірювання latency
VPN-сценарій залежить від MTU та якості інтернету	Можливі фризи й втрати через LTE/VPN	Оптимізувати MTU, бітрейт, профіль stream=1
Система перевірялася на обмеженій кількості ПК	Немає повної статистики переносимості	Додати тести на Linux/Windows/одноплатних ПК
Інтерфейс залишається прототипним	Потрібне покращення UX і фіксації layout	Доопрацювати UI, перевірити адаптивність
Немає повної перевірки залежностей при запуску	Можливі приховані помилки збірки	Додати startup self-check для pygame, av, Kivy, serial
Статистика є екранною, але не повністю журнальованою	Складно порівнювати довгі тести	Додати CSV/JSON-лог метрик

Джерело: сформовано автором.

Загалом реалізований прототип демонструє працездатність архітектури: ONVIF знаходить камеру, RTSP-потік відкривається стабільно у локальному сценарії, OSD накладається окремо від відео, а HID/Serial-to-UDP bridge передає керуючі дані. Додавання статистики суттєво підвищує дослідницьку цінність системи, оскільки дає змогу аналізувати не лише факт запуску, а й якість роботи окремих модулів.

Висновки до розділу 3

У третьому розділі описано практичну реалізацію системи керування БПЛА з ПК. Показано, що основними інженерними частинами є не лише інтерфейс, а й ONVIF-пошук, PyAV RTSP-прийом, MSP DisplayPort parser, OSD shadow buffer, HID-to-CRSF converter і Serial/HID-to-UDP bridge зі статистикою.

Експериментальна перевірка показала стабільну роботу локального LAN-сценарію з RTT avg 1,750 мс і 0 % втрат. VPN+Wi-Fi/LTE-сценарій показав більшу затримку і 1 % втрат, тому його визначено як перспективний режим, що потребує додаткової оптимізації.

Кількісні показники OSD і відео підтверджують працездатність системи: у тесті статистики зафіксовано 99,8 % валідних MSP-пакетів, 980 отриманих відеопакетів і 958 декодованих кадрів. Водночас робота має обмеження: неповне польове тестування, візуальний характер оцінки відеозатримки, залежність VPN від каналу та потреба у автоматичному журналюванні метрик.

ВИСНОВКИ

У кваліфікаційній роботі розроблено та дослідно перевірено програмно-апаратну систему керування БПЛА з комп'ютера в локальній та інтернет-мережі. Система об'єднує ONVIF-виявлення мережевої камери, RTSP-прийом відео, OSD-відображення на основі MSP DisplayPort, Serial/HID-to-UDP bridge і HID-to-CRSF перетворення.

У теоретичній частині проаналізовано підходи до керування БПЛА та порівняно технології відео, OSD, керування, GUI і віддаленого доступу. Обґрунтовано вибір RTSP + PyAV для відео, UDP для актуальних службових і керуючих повідомлень, ONVIF/WS-Discovery для пошуку пристроїв, Kivu для прототипу інтерфейсу та CRSF-подібного потоку для керування.

У проєктній частині сформовано апаратну та програмну архітектуру системи. Апаратна конфігурація включає ПК оператора, MikroTik hEX S, камеру OpenIPC/SSC30KQ, LAN/SFP або VPN-сценарій. Програмна архітектура включає Controller, ONVIF Discovery, RtspPlayer, MSPDisplayPortClient/OSDGrid, VideoScreen, SerialToNetBridge та HID-to-CRSF converter.

У практичній частині реалізовано робочі модулі й додано статистику для аналізу каналів. RTSP-модуль фіксує кількість отриманих відеопакетів, декодованих кадрів і помилок; OSD-модуль фіксує UDP-пакети, валідні MSP-повідомлення й не оброблені фрагменти; bridge фіксує UDP-пакети та помилки сокета. Це дозволило перейти від суто описової оцінки до кількісної.

Експериментальні результати підтвердили працездатність системи у локальному сценарії: для LAN отримано $RTT_{min/avg/max} = 1,422/1,750/3,523$ мс і 0 % втрат. У VPN+Wi-Fi/LTE-сценарії зафіксовано 1 % втрат і $RTT_{min/avg/max} = 17/28/104$ мс, що демонструє потребу у подальшій оптимізації віддаленого режиму.

OSD-статистика показала 99,8 % валідних MSP-пакетів у тесті, а відеомодуль зафіксував 980 отриманих відеопакетів і 958 декодованих кадрів.

Візуально оцінена затримка відео становила приблизно 100-200 мс, однак для більш точної наукової оцінки в майбутньому необхідно додати спеціальний вимірювальний модуль.

Практична цінність роботи полягає у створенні лабораторного прототипу наземної станції, який не обмежується виведенням відео, а дає змогу одночасно керувати підключенням, OSD, HID/Serial-входом і статистикою каналів. Основними напрямками розвитку є точне вимірювання latency, автоматичне логування метрик, перевірка залежностей під час запуску, оптимізація VPN/MTU, покращення інтерфейсу та тестування на ширшому наборі апаратних платформ.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Здітовецький Д. В., Ратушняк Т. В. Розробка системи керування безпілотними літальними апаратами в локальній та інтернет мережі // Електронні інформаційні ресурси: створення, використання, доступ та управління: матеріали Міжнар. наук.-практ. конф., Суми/Вінниця: НІКО/КЗВО «ВАБО», 2025. С. 111–113. URL: <https://docs.academia.vn.ua/handle/123456789/2545>
2. Bekmezci I., Sahingoz O. K., Temel Ş. Flying Ad-Hoc Networks (FANETs): A survey. Ad Hoc Networks. 2013. Vol. 11, No. 3. P. 1254–1270.
3. Bemporad A., Heemels M., Johansson M. Networked Control Systems. Springer, 2010. 735 p.
4. Betaflight. DisplayPort MSP Extensions. URL: <https://betaflight.com/docs/development/API/DisplayPort>
5. Betaflight. MSP Extensions. URL: <https://betaflight.com/docs/development/API/MSP-Extensions>
6. Chiumento Alessandro & Pollin S. Ultra-Reliable IEEE 802.11 for UAV Video Streaming: From Network to Application, 2017. DOI:10.1007/978-981-10-1627-1_50.
7. Donenfeld J. A. WireGuard: Next Generation Kernel Network Tunnel. NDSS Symposium. 2017. URL: <https://www.wireguard.com/papers/wireguard.pdf>
8. ExpressLRS. Serial Protocols Documentation. URL: <https://www.expresslrs.org/software/serial-protocols/>
9. FFmpeg Developers. FFmpeg Documentation and Protocols. URL: <https://ffmpeg.org/ffmpeg-protocols.html>
10. Gupta L., Jain R., Vaszkun G. Survey of Important Issues in UAV Communication Networks. IEEE Communications Surveys & Tutorials. 2015. Vol. 18, No. 2. P. 1123–1152.
11. Harris C. R. et al. Array programming with NumPy. Nature. 2020. Vol. 585. P. 357–362.
12. Hassanalain M., Abdelkefi A. Classifications, applications, and design

challenges of drones: A review // Progress in Aerospace Sciences, Volume 91, 2017, Pages 99-131. URL: <https://doi.org/10.1016/j.paerosci.2017.04.003> (<https://www.sciencedirect.com/science/article/pii/S0376042116301348>)

13. Hayat S., Yanmaz E., Muzaffar R. Survey on Unmanned Aerial Vehicle Networks for Civil Applications: A Communications Viewpoint. IEEE Communications Surveys & Tutorials. 2016. Vol. 18, No. 4. P. 2624–2661.

14. IEEE Standards Association. IEEE 802.3-2022: IEEE Standard for Ethernet. 2022. URL: <https://standards.ieee.org/ieee/802.3/10422/>

15. Information on RFC 3550-RFC Editor. URL: <https://www.rfc-editor.org/info/rfc3550>

16. ISO/IEC/IEEE. 12207:2017 Systems and software engineering — Software life cycle processes. 2017. URL: <https://www.iso.org/standard/90219.html>

17. Kivy Organization. Kivy Documentation: Cross-platform Python Framework. URL: <https://kivy.org/doc/stable/>

18. Koubâa A. et al. Micro Air Vehicle Link (MAVlink) in a Nutshell: A Survey. IEEE Access. 2019. Vol. 7. P. 87658–87680.

19. Kurose J. F., Ross K. W. Computer Networking: A Top-Down Approach. 8th ed. Pearson, 2021. 864 p.

20. Liechti C. pySerial Documentation. URL: <https://pyserial.readthedocs.io/en/latest/>

21. MikroTik. RouterOS Documentation. URL: <https://help.mikrotik.com/docs/spaces/ROS/pages/328059/RouterOS>

22. MikroTik. ZeroTier - RouterOS Documentation. URL: <https://help.mikrotik.com/docs/spaces/ROS/pages/83755083/ZeroTier>

23. Namuduri K., Chaumette S., Kim J. H., Sterbenz J. P. G. UAV Networks and Communications. Cambridge University Press, 2017.

24. OASIS. Web Services Dynamic Discovery (WS-Discovery) Version 1.1. 2009. URL: <https://docs.oasis-open.org/ws-dd/discovery/1.1/os/wsdd-discovery-1.1-spec-os.html>

25. ONVIF. Network Interface Specifications. Open Network Video Interface

Forum. URL: <https://www.onvif.org/profiles/specifications/>

26. ONVIF. ONVIF Core Specification. Open Network Video Interface Forum. URL: <https://www.onvif.org/specs/core/ONVIF-Core-Specification.pdf>

27. OpenIPC. Majestic streamer documentation. URL: <https://github.com/OpenIPC/wiki/blob/master/en/majestic-streamer.md>

28. OpenIPC. OpenIPC Firmware Documentation. URL: <https://docs.openipc.org/>

29. Postel J. User Datagram Protocol. RFC 768. IETF, 1980. URL: <https://datatracker.ietf.org/doc/html/rfc768>

30. PyAV Developers. PyAV Documentation. URL: <https://pyav.org/docs/stable/>

31. Pygame Developers. Pygame Documentation. URL: <https://www.pygame.org/docs/>

32. Python Software Foundation. Python 3 Documentation. URL: <https://docs.python.org/3/>

33. Python Zeep. Zeep: A fast and modern Python SOAP client. URL: <https://docs.python-zeep.org/en/master/>

34. Schulzrinne H., Casner S., Frederick R., Jacobson V. RTP: A Transport Protocol for Real-Time Applications. RFC 3550. IETF, 2003. URL: <https://datatracker.ietf.org/doc/html/rfc3550>

35. Schulzrinne H., et al. Real-Time Streaming Protocol Version 2.0. RFC 7826. IETF, 2016. URL: <https://datatracker.ietf.org/doc/html/rfc7826>

36. Schulzrinne H., Rao A., Lanphier R. Real Time Streaming Protocol (RTSP). RFC 2326. IETF, 1998. URL: <https://datatracker.ietf.org/doc/html/rfc2326>

37. Tailscale. About WireGuard. Tailscale Documentation. URL: <https://tailscale.com/docs/concepts/wireguard>

38. Tanenbaum A. S., Feamster N. Computer Networks. 6th ed. Pearson, 2020. 944 p.

39. TBS FPV. CRSF Protocol Specification. URL: <https://github.com/tbs-fpv/tbs-crsf-spec/blob/main/crsf.md>

40. Wang J. et al. Low-latency video transmission for UAVs: A review. *Journal of Network and Computer Applications*. 2019. Vol. 144. P. 39–53.
41. War Books: Drone Warfare - Modern War Institute. URL: <https://mwi.westpoint.edu/war-books-drone-warfare/>
42. Yaacoub E., Abu-Dayya A. Multihop Routing for UAV Swarms. *IEEE Communications Magazine*. 2020. Vol. 58, No. 2. P. 66–72.
43. ZeroTier. ZeroTier Documentation and Architecture. URL: <https://docs.zerotier.com/>