МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В. Н. Каразіна

Факультет: **ННІ Каразінський банківський інститут**
Кафедра: **Інформаційних технологій та математичного моделювання**
Спеціальність: **122 Комп'ютерні науки**
Освітня програма: **Комп'ютерні науки та інформаційні технології в бізнесі**

Група: АК-21М **денна форма навчання**

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему:

«ДОСЛІДЖЕННЯ ТЕХНОЛОГІЙ ЗБОРУ, ЛОГУВАННЯ ТА ПЕРЕДАЧІ ДАНИХ БЕЗДРОТОВИМИ ПРИСТРОЯМИ ДЛЯ ДОКАЗОВОГО СПОСТЕРЕЖЕННЯ У СФЕРІ БЕЗПЕКИ» ЗА НАКАЗОМ № 4601-5/3045 ВІД 25 ВЕРЕСНЯ 2024 РОКУ

здобувача вищої освіти **Волік Вячеслав Вікторович**

Робота допущена до захисту в ЕК
протокол кафедри ІТММ №4 від 30.11.2024 р.

Завідувач кафедри
к. п. н., доцент
_____ **Н. І. Стяглик**

Науковий керівник
к. п. н., доцент
_____ **Н. І. Стяглик**

м. Харків 2024 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Харківський національний університет імені В. Н. Каразіна

Факультет навчально-науковий інститут "Каразінський банківський інститут"
Кафедра інформаційних технологій та математичного моделювання
Рівень вищої освіти другий (магістерський)
Спеціальність 122 Комп'ютерні науки
Освітня програма Комп'ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ **Н. І. Стяглик**
Підпис ініціали, прізвище

“25” вересня 2024 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ (ПРОЄКТ)

Воліка Вячеслава Вікторовича
(прізвище, ім’я, по батькові студента)

1. Тема роботи «ДОСЛІДЖЕННЯ ТЕХНОЛОГІЙ ЗБОРУ, ЛОГУВАННЯ ТА ПЕРЕДАЧІ ДАНИХ БЕЗДРОТОВИМИ ПРИСТРОЯМИ ДЛЯ ДОКАЗОВОГО СПОСТЕРЕЖЕННЯ У СФЕРІ БЕЗПЕКИ»

керівник роботи к.п.н., доцент Стяглик Н.І.
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від “25” вересня 2024 року № 4601-5/3045

2. Строк подання студентом роботи 20 листопада 2024 року

3. Перелік питань, які потрібно розробити:
У розділі 1: розглянути теоретичні основи сучасного стану, розвиток та класифікацію датчиків та методи дослідження існуючих рішень доказового спостереження у сфері безпеки.
У розділі 2: розглянути розвиток технологій збору, логування та передачі даних за допомогою бездротових пристроїв доказового спостереження у сфері безпеки.
У розділі 3: описати алгоритми аналізу відеоданих, стиснення, логування та збереження та алгоритми стиснення та передачі параметричної інформації.

4. План роботи

№ з/п	Назви етапів роботи
1	Вибір здобувачем теми кваліфікаційної магістерської роботи
2	Затвердження плану і завдання кваліфікаційної магістерської роботи
3	Здача кваліфікаційної магістерської роботи керівнику
4	Підпис кваліфікаційної магістерської роботи керівника
5	Підпис кваліфікаційної магістерської роботи у нормоконтролера
6	Допуск завідувачем кафедри до захисту кваліфікаційної магістерської роботи
7	Захист кваліфікаційної магістерської роботи

5. Дата видачі завдання 25 вересня 2024 року

Студент _____ В.В.Волік
підпис ініціали, прізвище

Керівник роботи _____ Н.І.Стяглик
підпис ініціали, прізвище

РЕФЕРАТ
НА КВАЛІФІКАЦІЙНУ МАГІСТЕРСЬКУ РОБОТУ
«ДОСЛІДЖЕННЯ ТЕХНОЛОГІЙ ЗБОРУ, ЛОГУВАННЯ ТА ПЕРЕДАЧІ ДАНИХ БЕЗДРОТОВИМИ ПРИСТРОЯМИ ДЛЯ ДОКАЗОВОГО СПОСТЕРЕЖЕННЯ У СФЕРІ БЕЗПЕКИ»
Воліка Вячеслава Вікторовича

Кваліфікаційна магістерська робота містить: 96 сторінок, 15 таблиць, 39 рисунків, 18 формул, список літератури з 60 найменувань.

Об'єкт дослідження: працездатність бездротових пристроїв для доказового моніторингу.

Предмет дослідження: методи збору, логування, безвтратного стиснення, передачі даних бездротовими пристроями.

Мета кваліфікаційної магістерської роботи: підвищити ефективність збору, логування, безвтратного стиснення та передачі даних бездротовими пристроями для доказового моніторингу шляхом їх дослідження та стиснення.

Завдання кваліфікаційної магістерської роботи:

- у першому розділі розглянути теоретичні основи сучасного стану, розвиток та класифікацію датчиків та методи дослідження існуючих рішень доказового спостереження у сфері безпеки;

- у другому розділі розглянути розвиток технологій збору, логування та передачі даних за допомогою бездротових пристроїв доказового спостереження у сфері безпеки;

- у третьому розділі описати алгоритми аналізу відеоданих, стиснення, логування та збереження та алгоритми стиснення та передачі параметричної інформації.

Актуальність дослідження полягає у підвищенні рівня безпеки, ефективності та надійності систем спостереження та забезпечення відповідності сучасним технологічним та правовим вимогам.

За результатами дослідження сформульовані алгоритми аналізу відеоданих, стиснення, логування та їх збереження та алгоритми аналізу стиснення та передачі параметричної інформації.

Практична новизна: полягає у впровадженні інноваційних підходів та рішень, які значно підвищують ефективність, надійність та безпеку систем доказового спостереження, що забезпечує їх адаптивність до сучасних викликів і потреб.

Одержані результати можуть бути використані в різних сферах життя для підвищення рівня безпеки та забезпечення ефективного моніторингу у різних установах, організаціях і підприємствах для безпеки життєдіяльності населення.

КЛЮЧОВІ СЛОВА: PYTHON, АЛГОРИТМИ СТИСНЕННЯ, БЕЗПЕКОВІ ДАНІ, БЕЗДРОТОВІ ПРИСТРОЇ, ОБРОБКА ЗОБРАЖЕНЬ,

СПОСТЕРЕЖЕННЯ.

ABSTRACT
AT QUALIFICATION MAGISTER WORK
«INVESTIGATION OF WIRELESS DATA COLLECTION, LOGGING AND TRANSMISSION TECHNOLOGIES FOR SECURITY EVIDENCE SURVEILLANCE»
Viacheslav Volik

The master's thesis contains 96 pages, 15 table, 39 drawings, 18 formulas, a list of references of 60 titles.

The object of the study is performance of wireless devices for evidentiary monitoring.

The subject of the study is methods of collection, logging, lossless compression, data transmission by wireless devices.

The purpose of the qualifying master's thesis: to improve the efficiency of collection, logging, lossless compression and transmission of data by wireless devices for evidentiary monitoring through their research and compression.

The tasks of the master's qualification work are:

- in the first section to consider the theoretical foundations of the current state, the development and classification of sensors, and research methods for existing evidence-based surveillance solutions in the field of security;

- in the second section to consider the development of technologies for data collection, logging and transmission using wireless evidence surveillance devices in the field of security;

- in the third section describe the algorithms of video data analysis, compression, logging and saving, and algorithms of compression and transmission of parametric information.

The relevance of the study in increasing the level of security, efficiency and reliability of surveillance systems and ensuring compliance with modern technological and legal requirements.

According to the results of the research, algorithms for video data analysis, compression, logging and their preservation and algorithms for compression analysis and transmission of parametric information were formulated.

Main theoretical provisions on the topic of the practical relevance of the consists in the implementation of innovative approaches and solutions that significantly increase the efficiency, reliability and security of evidentiary surveillance systems, which ensures their adaptability to modern challenges and needs.

The results obtained can be used in various spheres of life to increase the level of safety and ensure effective monitoring in various institutions, organizations and enterprises for the safety of the population.

KEYWORDS: COMPRESSION ALGORITHMS, IMAGE PROCESSING, OBSERVATION, PYTHON, SECURE DATA, WIRELESS DEVICES.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧОК, СИМВОЛІВ І ТЕРМІНІВ

AES – Advanced Encryption Standard

API – Application Programming Interface

AWS – Amazon Web Services

CoAP – Constrained Application Protocol

GDPR – General Data Protection Regulation

HTTP/HTTPS – HyperText Transfer Protocol/ HyperText Transfer Protocol Secure

I2C – Inter-Integrated Circuit

IoT – Internet of Things

ITS – інтелектуальні транспортні системи

LoRaWAN – Long Range Wide Area Network

LZMA – Lempel-Ziv-Markov chain-Algorithm

MQTT – Message Queuing Telemetry Transport

MQTT – Message Queuing Telemetry Transport

NB-IoT – Narrow Band Internet of Things

NPM – SolarWinds Network Performance Monitor

SPI – Serial Peripheral Interface

TSIEM – Tivoli Security Information and Event Manager

WBAN – Wireless Body Area Network

ВСТУП

В умовах швидкого розвитку технологій та загальної цифровізації, питання безпеки та захисту даних постає дуже гостро. За останні роки можна побачити як зростає кількість бездротових пристроїв. Вони не тільки спрощують наше життя, але й створюють виклики щодо безпеки даних. Бездротові пристрої потребують усе нових і нових розроблень алгоритмів для забезпечення захисту даних, їх збереження і моніторингу. Вони надають унікальні можливості для збору, логування та передачі даних, що необхідні для доказового спостереження. Упровадження ефективних методів збору та аналізу даних дозволяє значно підвищити точність та оперативність у виявленні та застереженні різного роду інцидентів [49, 59].

Наприклад, одна з сфер життя, яка швидко розвивається – сфера Інтернет речей (IoT). Безпека в даній сфері є дуже важливим аспектом, бо з кожним роком кількість підключених к мережі пристроїв збільшуються в геометричній прогресії. Бездротові пристрої IoT надають унікальні можливості для збору, логування та передачі даних, необхідних для доказового спостереження. Ці дані можуть використовуватися для забезпечення безпеки на різних рівнях - від особистої та корпоративної до національної. Технології IoT дозволяють збирати дані в реальному часі, аналізувати їх та оперативно реагувати на погрози, що особливо важливо в умовах середовища, яке душе швидко змінюється [31, 40, 42-43].

Актуальністю дослідження технологій збору, логування та передачі даних бездротовими пристроями для доказового спостереження в сфері безпеки обумовлена наступними факторами:

- кількість бездротових засобів є важливими джерелами даних для різних систем безпеки;

- сучасні загрози безпеки стають все більш складними та різноманітними, що потребує нових підходів та інструментарію для їх

ефективного виявлення та попередження;

- розвиток технологій відкриває нові можливості для підвищення швидкості та надійності передачі даних, що особливо важливо в контексті доказового спостереження.

Мета дослідження – підвищити ефективність збору, логування, безвтратного стиснення та передачі даних бездротовими пристроями для доказового моніторингу шляхом їх дослідження.

Завданнями дослідження є наступні пункти:

- розглянути сучасний стан доказового спостереження у сфері безпеки;
- розглянути розвиток і класифікацію датчиків доказового спостереження у сфері безпеки;
- розглянути методи дослідження існуючих рішень доказового спостереження у сфері безпеки;
- дослідити збір відео- та аудіоінформації з використанням бездротових пристроїв під час спостереження;
- дослідити проєктування модулів для збору параметричних даних з використанням бездротових пристроїв;
- дослідити проєктування структури модуля синхронізації даних та порівняльного аналізу;
- виконати розробку алгоритмів аналізу відеоданих, стиснення, логування та їх збереження;
- виконати розробку алгоритмів аналізу стиснення та передачі параметричної інформації;
- виконати порівняльний аналіз розроблених алгоритмів.

Об’єкт дослідження – працездатність бездротових пристроїв для доказового моніторингу.

Предмет дослідження – методи збору, логування, безвтратного

стиснення, передачі даних бездротовими пристроями.

Робота складається з вступу, трьох розділів та висновків.

У вступі наведені актуальність роботи, мета, поставлені завдання на роботу, визначені об’єкт і предмет дослідження та представлена загальна структура роботи.

У першому розділі «Теоретичні основи доказового спостереження за допомогою бездротових пристроїв» проаналізовані сучасний стан, розвиток та методи дослідження існуючих рішень доказового спостереження у сфері безпеки.

У другому розділі «Дослідження розвитку технологій збору, логування та передачі даних за допомогою бездротових пристроїв у сфері безпеки» проаналізовані збір відео- та аудіоінформації, проєктування модулів для збору параметричних даних та структури модуля синхронізації даних доказового спостереження у сфері безпеки.

У третьому розділі «Методи розвитку алгоритмів збору, логування, стиснення та передачі даних у сфері безпеки» виконана розробка алгоритмів аналізу відеоданих, стиснення, логування та їх збереження і алгоритмів аналізу стиснення та передачі параметричної інформації доказового спостереження у сфері безпеки. Також було здійснено порівняння ефективності даних алгоритмів.

Висновки підводять та акцентують увагу на результатах роботи і показують важливість проведеного дослідження.

РОЗДІЛ 1.
ТЕОРЕТИЧНІ ОСНОВИ ДОКАЗОВОГО СПОСТЕРЕЖЕННЯ ЗА ДОПОМОГОЮ БЕЗДРОТОВИХ ПРИСТРОЇВ

1.1. Сучасний стан доказового спостереження у сфері безпеки

Системи доказового спостереження, які включають технологій збору, логування та передачі даних, використовуються в різних сферах, як в країнах Європи, так і в Україні. Дані системи застосовуються, як в урядових, так і в приватних секторах, які охоплюють правопорядок, охорону здоров’я, транспорт, промисловість тощо [6, 24, 37, 48].

1.1.1. Поточний стан систем доказового спостереження

В Україні системи доказового спостереження активно використовуються в правоохоронних органах, особливо для спостереження за громадським порядком та розслідування злочинів. Відеоспостереження, використання дронів та системи для перехоплення даних ж прикладами технологій, які застосовуються для цих цілей. У транспортній сфері системи спостереження використовуються для контролю за рухом транспорту, застереження дорожньо-транспортних пригод та слідування правилам дорожнього руху. У сфері охорони здоров’я системи спостереження допомагають відстежувати стан пацієнтів, особливо в умовах пандемій. Використовуються носимі пристрої, які збирають дані про здоров’я в реальному часі та передають їх медичній установі [48].

В Європі системи доказового спостереження широко використовуються для боротьби зі злочинністю та тероризмом. Технології включають камери спостереження, системи розпізнавання обличь та цифрові сліди. Також активно впроваджуються інтелектуальні транспортні системи (ITS), які використовують доказове спостереження для покращення дорожньої безпеки

та керування трафіком. Прикладами таких систем можуть бути камери контролю швидкості, системи моніторингу транспортних потоків і системи відстежування вантажів. Зріст використання IoT пристроїв для спостереження відкриває нові можливості для збору та аналізу даних, але також надає виклики в області кібербезпеки. Забезпечення захисту даних є важливою задачею, тому в Європі використовують норми Загальним регламентом про захист даних (GDPR). Використання штучного інтелекту та машинного навчання для аналізу даних доказового спостереження стає все більш розповсюдженим, що допомагає виявити аномалії та прогнозувати події [24].

Наприклад, Лондон відомий своєю великою мережею відеоспостереження, яка активно використовується для забезпечення загального порядку та безпеки. У Берліні впроваджені системи інтелектуального керування трафіком, які допомагають оптимізувати рух на дорогах та знижують кількість аварій [6, 37].

Таким чином, системи доказового спостереження розвиваються та стають все більш інтегрованими, забезпечуючи широкий спектр застосування: від забезпечення безпеки до покращення якості життя громадян.

1.1.2. Основні технології та методи, що використовуються в сучасних системах

Сучасні системи доказового спостереження включають різноманітні технології та методи, направлені на збір, зберігання, аналіз та передачу даних із високим ступенем достовірності та безпеки [50-51].

Для зручності розгляду та порівняння систем доказового спостереження була створена таблиця, в якій наведені найменування, опис та застосування основним технологій і методів сучасних систем доказового спостереження. Результат наведений у таблиці 1.1.

Таблиця 1.1

Порівняння технологій і методів сучасних систем доказового

спостереження

Технологія / метод	Опис	Застосування
1	2	3
Бездротові мережі	Використовуються для передачі даних від датчиків та пристроїв Wi-Fi, Bluetooth, Long Range Wide Area Network (LoRaWAN), Narrow Band Internet of Things (NB-IoT)	Моніторинг оточуючого середовища, розумні міста, промислова автоматизація
Датчики та IoT пристрої	Збирають дані про параметри оточуючого середовища (температури, вологості, руху тощо)	Медичний моніторинг, керування ресурсами, безпека та обладнання
Хмарні технології	Надають зберігання даних та обчислювальні ресурси через хмарні платформи (Amazon Web Services (AWS), Google Cloud, Azure)	Аналізування великих даних, машинне навчання, гнучкість та масштабованість
Криптографія та безпека	Забезпечення захисту даних при передачі та зберігання з використанням шифрування та методів автентифікації	Захист конфіденційності, запобігання несанкціонованого доступу
Аналіз даних	Включає машинне навчання, штучний інтелект, статистичні методи для обробки та інтерпретації даних	Прогностична аналітика, виявлення аномалій, оптимізація процесів
Системи логування та	Реєструють та зберігають дані про події та дії для	Забезпечення цілісності даних, керування

аудиту	забезпечення трасування та аналізу	ризиками, відповідність нормативним вимогам
Протоколи передачі даних	Визначають способи передачі даних між пристроями та серверами (Message Queuing Telemetry Transport (MQTT), Constrained Application Protocol (CoAP), HyperText Transfer Protocol / HyperText Transfer Protocol Secure (HTTP/HTTPS)	Ефективна комунікація, мінімізація витрат на передачу даних
Системи резервування	Включає регулярне створення резервних копій даних та використання розподілених систем зберігання для забезпечення доступності та відмовостійкості	Відновлення після збоїв, неперервність роботи та захист від втрати даних

Джерело: розробка автора

За таблицею 1.1 можна побачити різноманітність технологій і методів, їх застосування в сучасних системах доказового спостереження для забезпечення високого ступеня надійності, безпеки та ефективності в зборі, аналізі та передачі даних. Таким чином, сукупність наведених факторів підкреслюють значимість інтеграції передових технологій, суворого дотримання нормативних вимог та етнічних стандартів для досягнення стабільного та ефективного доказового спостереження.

1.1.3. Огляд найновіших досягнень і викликів у сфері безпеки

За останнє десятиріччя сфера безпеки доказового спостереження зазнала

значних змін та досягла чимало наступного [14, 46]:

1) розвиток бездротових технологій дозволив створювати компактні та потужні пристрої для збору та передачі даних, що покращило можливості для доказового спостереження;

2) впровадження точних геопозиційних систем, систем відеоспостереження високої якості та сенсорів із множиною функцій, що значно покращило здатність фіксування та аналізування даних у реальному часі;

3) упровадження хмарних технологій дозволяють зберігати більші об’єми даних безпечно та доступно, забезпечуючи можливість аналізу в будь-який час;

4) застосування алгоритмів машинного навчання та аналітики даних дозволяє виявляти аномалії та передбачати події, що критично важливо для оперативного реагування;

5) розширення використання розумних пристроїв та IoT технологій надає більше даних для моніторингу та збільшує вразливості в мережевій безпеці;

6) постійна розробка та покращення стандартів безпеки та нормативів для забезпечення ефективності та юридичного захисту при використанні технологій доказового спостереження тощо.

Не дивлячись на досягнення, існують і виклики, з якими стикаються розробники:

- із зростанням об’єму та чутливості даних зростають ризики їх витоку або отримання несанкціонованого доступу;

- необхідність в інтеграції різноманітних технологій та стандартів для забезпечення сумісності та ефективності систем спостереження;

- збір та використання даних можуть стикнутися з етичними питаннями такими, як права на конфіденційність і приватність;

- із збільшенням числа кібератак та зростом погроз для цифрових

систем безпеки, необхідність у захисті від кіберпогроз стає все більш актуальною.

Указані досягнення та виклики надають розуміння динамічності та значимості сфери безпеки доказового спостереження в сучасному світі.

1.2. Розвиток і класифікація датчиків доказового спостереження у сфері безпеки

Одним із найважливіших елементів у розвитку та класифікації грає роль датчики доказового спостереження, які використовуються для збору, аналізу та передачі даних. Вони широко застосовуються в системах безпеки, забезпечуючи точний і надійний контроль за різноманітними об’єктами та процесами [14, 50].

1.2.1. Особливості принципів роботи та сучасні конструкції датчиків

Принципи роботи датчиків доказового спостереження засновані на перетворення фізичних величин (температура, тиск, рух, звук тощо) в електричні сигнали, які можуть бути оброблені та передані для подальшого аналізу. Розуміння особливостей принципів їх роботи є одним із ключових етапів задля застосування в системах безпеки. Це дозволяє обрати найбільш відповідні типи датчиків у залежності від конкретних задач та умов експлуатації, а також інтегрувати їх у комплексні системи спостереження та керування. Також датчики можна класифікувати за різними критеріями та типами [46, 51].

Основні особливості роботи датчиків можна виділити за наступними аспектами [46, 51]:

- вимір температури об’єкту або середовища;
- перетворення тиску на електричний сигнал;

- реагування на теплові випромінювання;
- реагування на відображення звукових хвиль;
- вимір прискорення;
- перетворення звукових хвиль на електричні сигнали;
- надання неперервного сигналу, що напряму пропорційний параметру, що вимірюється;
- генерування цифрових сигналів;
- передання даних, використовуючи проводи або радіоканали тощо.

Сучасні датчики доказового спостереження розроблюються з використанням передових технологій та матеріалів, що дозволяє досягати високої точності, надійності та ефективності при їх застосуванні [46].

У таблиці 1.2 наведені приклади датчиків доказового спостереження.

Таблиця 1.2

Приклади датчиків доказового спостереження

Тип датчику	Опис	Застосування
1	2	3
Біосенсори	Використовують біологічні елементи (ферменти, антитіла тощо) для виявлення та виміру хімічних речовин	Медична діагностика, контроль середовища, харчова промисловість
Наноматеріали	Засновані на використання наноматеріалів (вуглецеві нанотрубки, графен тощо), що забезпечують високу чутливість та селективність	Газоаналізатори, біосенсори, датчики тиску та температури
Оптоелектронні	Використовують світло	Оптичні волокняні

	(оптичні волокна, лазери, світлодіоди тощо) для вимірювання різних параметрів, таких як відстань, температура, тиск	сенсори для інфраструктурного моніторингу, лазерні дальноміри
Бездротові	Передають дані за допомогою бездротових технологій (Wi-Fi, Bluetooth, LoRa, NB-IoT)	Розумний будинок, промислові IoT системи, віддалене спостереження
Мемс	Мікроелектромеханічні системи, інтегровані на мікрочіпах, що використогвуються для виміру прискорення, тиску, магнітних полів тощо	Смартфони, автомобілі, медичні пристрої, промислова автоматизація
Інтегральні	Зменшення розміру датчиків та покращення їх продуктивності	Споживна електроніка, медичні пристрої, автоматизація процесів
Гнучкі та носимі	Виготовлені з гнучків матеріалів, які можна інтегрувати в одяг або шкіру людини	Носимі пристрої для здоров’я та фітнесу, інтелектуальний одяг
Фотоакустичні	Засновані на вимірі акустичного відгуку на оптичне випромінювання	Медична діагностика, контроль забруднення повітря, промислові процеси

Продовження таблиці 1.2

1	2	3

Квантові	Використання квантових явищ (заплутаність, суперпозиція тощо) для досягнення високої точності та чутливості	Наукові дослідження, точні виміри магнітних та гравітаційних полів, квантова метрологія

Джерело: розробка автора

Додатковими прикладами сучасних конструкцій датчиків доказового спостереження є наступні [14, 46]:

1) температурні датчики на основі наноматеріалів, які дозволяють створити датчики з високою чутливістю та швидким відгуком;

2) мемс акселерометри дозволяють вимірювати прискорення, вібрацію та орієнтацію;

3) бездротові датчики розумного будинку, які спостерігають за температурою, вологістю, освітленням тощо;

4) гнучкі носимі датчики дозволяють спостерігати біометричні дані (серцеві скорочення, температура тіла, рівень кисню) тощо.

Конструкції датчиків доказового спостереження відрізняються високою точністю, надійністю та ефективністю. Вони знаходять широке застосування в різних сферах: від медичної до промислової та домашньої. Це забезпечує безпеку та зручність у повсякденному житті.

1.2.2. Тенденції розвитку сенсорних технологій

Тенденції розвитку сенсорних технологій у доказовому спостереженні в сфері безпеки швидко зросли та зростають і сьогодні. Якщо переглянути тенденції за останнє десятиріччя, то можна декілька наступних направлень, за якими відбувається зріст [25]:

- зріст IoT пристроїв, які надають можливість збору даних у режимі реального часу;

- мініатюризація сенсорів, які можна використовувати в більш широкому спектрі, з різними пристроями;

- покращення якості даних за допомогою використання новітніх сенсорів;

- надійні алгоритми калібрування, які корегують показання сенсорів, що забезпечує стабільність та достовірність даних;

- використання складних аналітичних систем для проведення глибокого аналізу даних, що були отримані із сенсорів;

- шифрування даних для їх захисту;

- розробка сенсорів з низьким рівнем енергоспоживання, що дозволяє збільшити їх строк служби та зменшити необхідність частої заміни;

- інтеграція із системами відеоспостереження, які інтегровані з іншими системами, що надає комплексний підхід до безпеки.

Описані вище тенденції розвитку надають розуміння про значний прогрес в області сенсорних технологій та їх використанні в доказовому спостереженні, що значно підвищує рівень безпеки та надійності систем моніторингу.

1.2.3. Класифікація датчиків за критеріями

Датчики доказового спостереження використовуються для забезпечення безпеки та моніторингу різних об’єктів і процесів. Їх класифікація можу бути проведена за різними критеріями, такими як вимірювані параметри, принципи роботи, методи передачі даних тощо [21].

За вже наведеними даними про датчики доказового спостереження, можна зробити узагальнену таблицю за їх класифікацією та розписати більш детально який тип датчику відноситься до якого критерію. Класифікація датчиків доказового спостереження наведена в таблиці 1.3.

Таблиця 1.3

Класифікація датчиків доказового спостереження

Класифікаційний критерій	Тип датчику	Опис
1	2	3
Вимірювані параметри	Температурний	Виміряють температуру об’єктів і середовища
	Тиску	Перетворюють тиск в електричний сигнал
	Руху та положення	Виміряють переміщення, прискорення та орієнтацію об’єктів
	Звуку	Перетворюють звукові сигнали в електричні сигнали
	Оптичні	Виміряють інтенсивність світла або оптичні характеристики об’єкту
	Хімічні	Визначають концентрацію хімічних речовин у середовищі
	Газові	Виміряють концентрацію газів у повітрі або іншому середовищі
	Вологості	Виміряють відносну або абсолютну вологість середовища
Принцип роботи	Резистивні	Вимірюють власний супротив у залежності від вимірюваного параметру
	П’єзоелектричні	Генерують електричний заряд під дією механічної напруги
Принцип роботи	Індуктивні	Вимірюють зміни індуктивності в

		залежності від положення або руху об’єкту
	Ємнісні	Вимірюють зміни ємкості при зміні положення або руху об’єкту
	Напівпровідникові	Використовують напівпровідникові матеріали для перетворення вимірюваних параметрів в електричний сигнал
Метод передачі даних	Дротові	Передають дані по дротам, забезпечуючи надійний і стабільний зв’язок
	Бездротові	Передають дані по радіоканалам, які є зручними для мобільних та тимчасових установок
Тип використовуваної енергії	Пасивні	Не потребують зовнішнього джерела живлення, використовують енергію вимірюваного сигналу
	Активні	Потребують зовнішнього джерела живлення (батарейні датчики, підключені до мережі датчики)
Область застосування	Промислові	Використовують в промислових процесах для контролю та автоматизації
Область застосування	Медичні	Застосовуються в медицині для моніторингу стану здоров’я пацієнтів
	Автомобільні	Використовуються в автомобілях

		для контролю різноманітних параметрів (тиск шин, рівень палива тощо)
	Розумний будинок	Застосовуються в системах автоматизації домашнього господарства (контроль температури, освітлення, безпека тощо)
Функціональні особливості	Однофункціональні	Виміряють тільки один параметр
	Багатофункціональні	Можуть вимірювати декілька параметрів одночасно
Середовище експлуатації	Внутрішні	Використовуються всередині приміщень
	Зовнішні	Призначені для роботи на відкритому повітрі та в суворих умовах

Продовження таблиці 1.3

1	2	3
Технологія зв’язку	Дротовий зв’язок	Використовують дроти для передачі даних і живлення
	Бездротовий зв’язок	Використовують радіосигнал для передачі даних, що дозволяє їх встановлювати в важкодоступних місцях та при мобільних застосувань

Джерело: розробка автора

Наведена класифікація (табл. 1.3) допомагає систематизувати інформацію про датчики доказового спостереження, що спрощує вибір

необхідних рішень для різних задач та умов експлуатації в сфері безпеки.

1.3. Методи дослідження існуючих рішень доказового спостереження у сфері безпеки

Доказове спостереження включає в себе різні технології та підходи, як уже було зазначено вище. Але є ряд існуючих програмних рішень, які необхідно було б розглянути.

1.3.1. Огляд і аналіз поточних рішень, що використовуються в галузі доказового спостереження

Програмні рішення для доказового спостереження включають у себе різноманітні інструменти та платформи, які забезпечують збір, аналіз та візуалізацію даних для прийняття інформованих рішень [7].

Розглянемо такі програмні рішення більш детально.

Системи керування подіями та інцидентами збирають, аналізують та інтерпретують дані мережевої безпеки для виявлення погроз та інцидентів. Наприклад, існує система Tivoli Security Information and Event Manager (TSIEM), яка є комплексним рішенням, що призначений для керування інформацією та подіями безпеки [20].

Даний програмний продукт має наступні можливості [20]:

- збір даних з різних джерел, включаючи мережеві пристрої, сервери, застосунки та бази даних;
- аналіз подій безпеки для виявлення аномалій та потенційних погроз;
- кореляція подій із різних джерел для побудови повної картини інцидентів;
- постійне спостереження безпеки в реальному часі;

- створення налаштовування звітності для візуалізації стану безпеки та інцидентів;

- інтеграція з іншими системами керування безпекою;

- збирання, зберігання та аналіз журналів подій для забезпечення відповідності нормативних вимог та аудиту.

Один з екранів TSIEM наведений на рисунку 1.1.

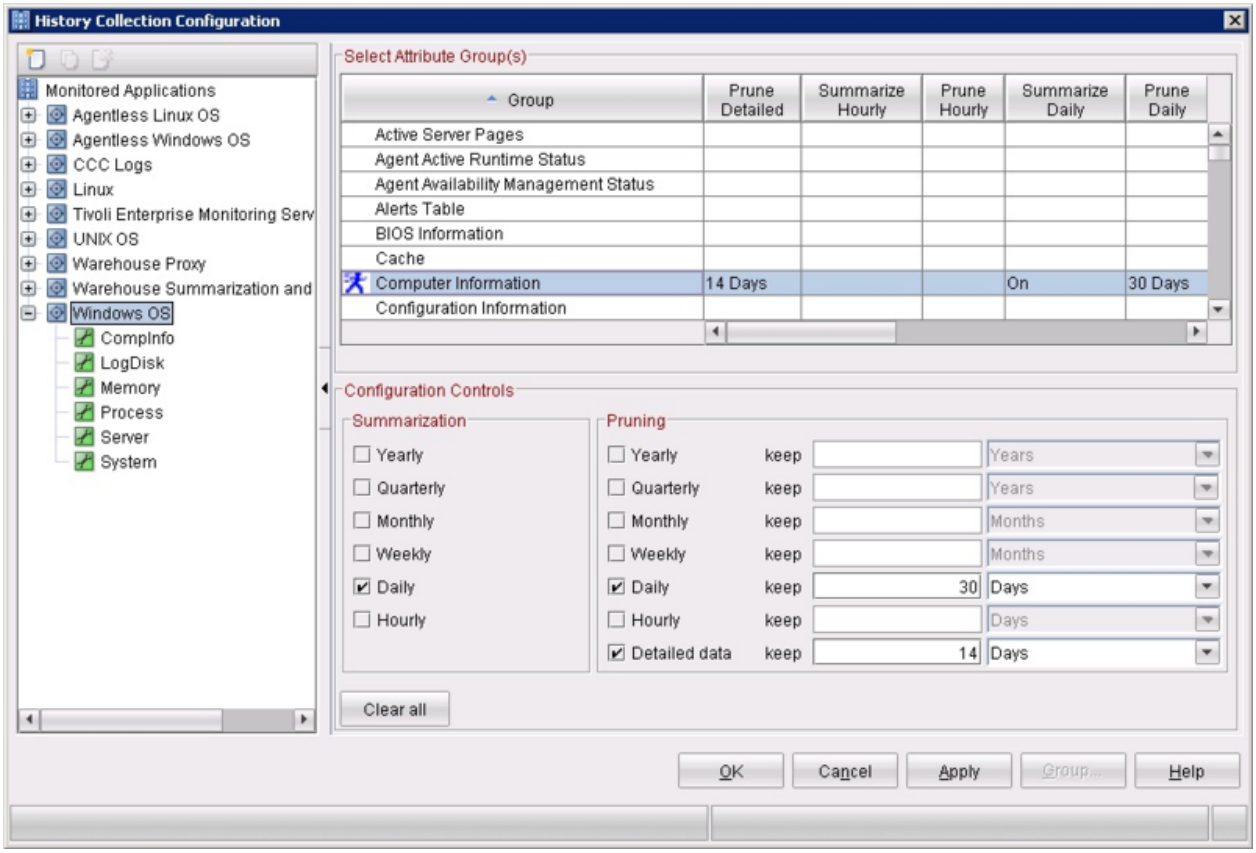

Рис. 1.1. Екран програмної системи TSIEM [20]

TSIEM надає потужні інструменти для проактивного керування та забезпечення безпеки, допомагаючи організаціям захищатися від кіберзагроз та своєчасного реагування на інциденти.

Платформи моніторингу продуктивності для аналізу продуктивності мережі, застосунків та сервісів. Наприклад, SolarWinds Network Performance Monitor (NPM). Програма допомагає відстежувати, діагностувати та вирішувати проблеми з продуктивністю мережі, надаючи деталізовану

інформацію та аналітику про стан мережевих пристроїв та сервісів.

Основними можливостями системи є [33]:

- постійний моніторинг стану та продуктивності мережевих пристроїв, включаючи маршрутизатори, комутатори, сервери та бездротові точки доступу;

- підтримка різноманітних виробників обладнання, що дозволяє спостерігати мережі з багатовендерною інфраструктурою;

- швидке виявлення проблем з продуктивністю та доступністю мережі;

- можливість створювати налаштовані звіти для візуалізації ключових метрик продуктивності тощо.

Один з екранів NPM наведений на рисунку 1.2.

NPM є надійним рішенням для спостереження мережевої продуктивності, яке допомагає організаціям підтримувати високу працездатність і доступність своєї мережевої інфраструктури, а також оперативно реагувати на проблеми.

Платформи моніторингу хмарних ресурсів, які необхідні для спостереження та керування хмарними ресурсами, включаючи керування продуктивністю, безпекою та витратами. Наприклад, ManageEngine Applications Manager. Система допомагає адміністраторам забезпечити високу доступність, продуктивність застосунків, сервісів, ресурсів в організації [36].

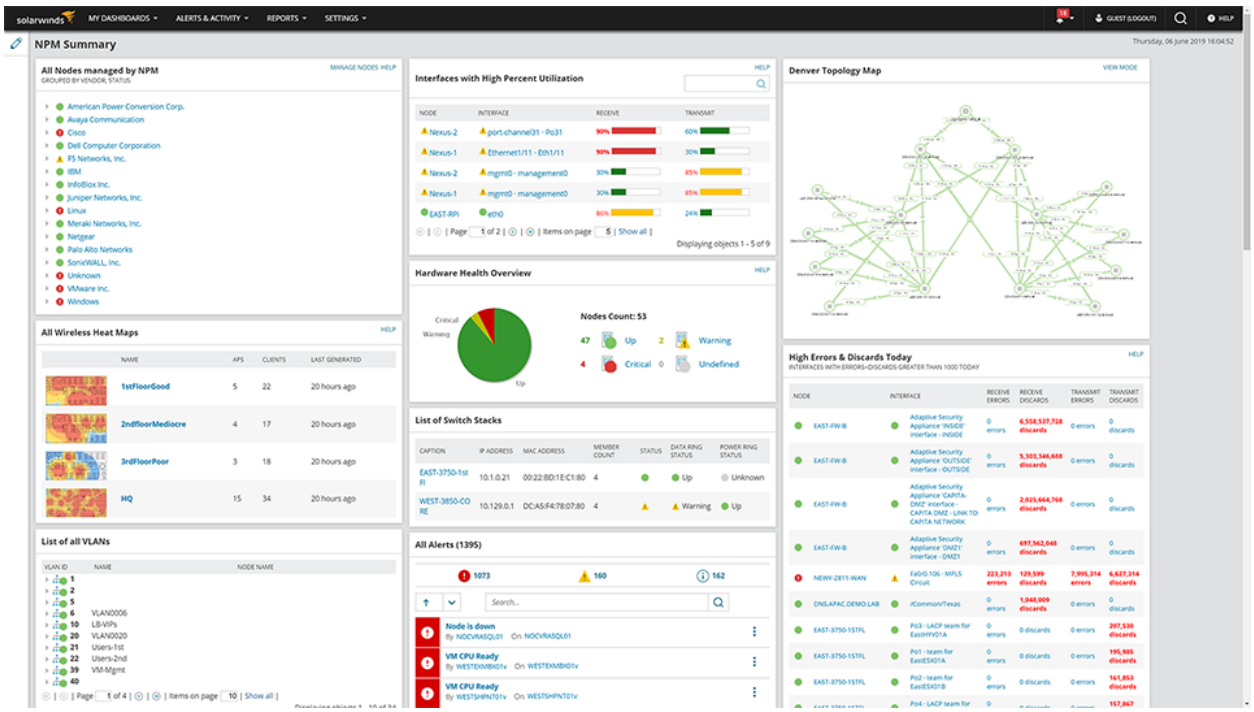

Рис. 1.2. Екран програмної системи NPM [33]

Основними можливостями програмного продукту є [36]:

- підтримка спостереження широкого спектру застосунків, включаючи вебсистеми, сервери, бази даних, поштові тощо;

- підтримка популярних хмарних сервісів та інструменти для відстежування стану та продуктивності хмарних сервісів і застосунків;

- підтримка різного роду баз даних та спостереження їх запитів, працездатності та стану для забезпечення безперебійної роботи;

- підтримка скриптів та автоматизації для виконання повторюваних задач тощо.

Один з екранів ManageEngine Applications Manager наведений на рисунку 1.3.

Дана система має надійне та багатофункціональне рішення для спостереження та керування хмарними даними, їх аналізу та зберігання.

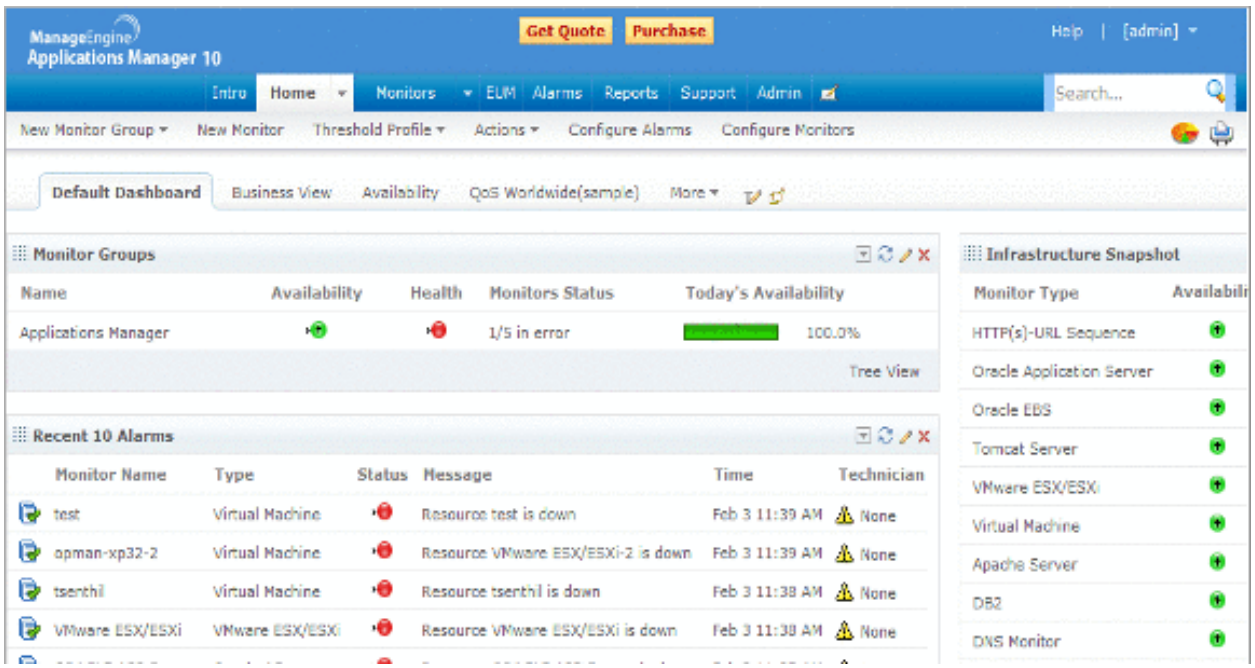

Рис. 1.3. Екран програмної системи ManageEngine Applications Manager [36]

Застосування аналізу даних та штучний інтелект надає виявити аномалії, прогнозування ризиків та автоматизації процесів прийняття рішень у реальному часі. Наприклад, Apache Spark, яка дуже швидко оброблює дані в пам’яті, забезпечуючи високу продуктивність та підтримку різноманітних завдань, включаючи машинне навчання та потокову обробку даних. Система підтримує різні типи аналітичних задач, обробку даних у реальному часі, пакетну обробку, машинне навчання та графові обчислення [10].

Основний функціонал програмної системи наступний [10]:

- має потужне ядро, яке відповідає за керування розподіленими задачами та їх виконання;

- забезпечує можливість для роботи з даними в пам’яті, що значно прискорює обробку в порівнянні з традиційними системами, шо засновані на введенні-виведенні;

- має модуль для роботи з даними та підтримку з різними джерелами даних;

- має модуль для обробки потоків даних у реальному часі;

- дозволяє виконувати операції над потоковими даними з мінімальною затримкою;

- має бібліотеку машинного навчання, що містить різноманітні алгоритми для класифікації, регресії, кластеризації та рекомендацій;

- підтримує роботу з великим об’ємом пам’яті , забезпечуюси високу продуктивність та масштабованість;

- має модуль для виконання графових обчислень;

- забезпечує можливість для створення, зміни та обробки графових структур.

Один з екранів Apache Spark наведений на рисунку 1.4.

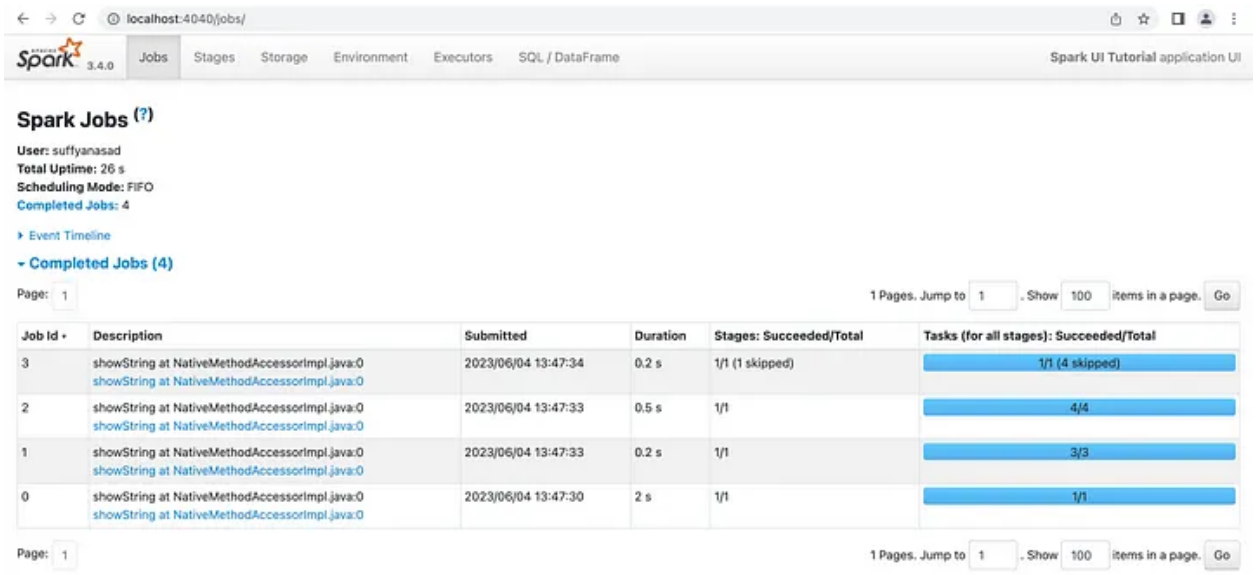

Рис. 1.4. Екран програмної системи Apache Spark [10]

За наведеними виде властивостям програмної системи, можна сказати, що Apache Spark потужний та гнучкий сервіс, який надає широкий спектр інструментів для аналізу даних та оброці великих об’ємів інформації, що робить її незамінною для багатьох сучасних застосунків та систем.

Системи керування журналами необхідні для збору, зберігання та аналізу журналів подій для виявлення аномалій та погроз безпеки. Наприклад, система NetWrix Event Log Manager надає можливості для керування та спостереження за подіями, які дозволяють централізовано збирати, архівувати та аналізувати журнали подій з різних джерел [38].

Основними можливостями системи є наступний функціонал [38]:

- збір подій з різних підключених джерел до системи;
- об’єднання даних в єдину централізовану базу для спрощеного аналізу та керування;
- безпечне зберігання журналів подій для довгострокового зберігання та відповідне нормативним вимогам;
- можливість налаштування періоду зберігання даних у залежності від потреб організації;
- інструменти для аналізу даних журналів подій, які дозволяють виявити аномалії та корелювати події для виявлення потенційних погроз;
- підтримка створення налаштовуваних правил та фільтрів для точного аналізу даних;
- створення деталізованих звітів за подіями для цілей аудиту та аналізу безпеки;
- підтримка автоматичного створення та відправлення звітів за розкладом для регулярного спостереження;
- інтеграція з іншими системами, такими як TSIEM, для розширення можливостей.

Один з екранів NetWrix Event Log Manager наведений на рисунку 1.5.

Програмний продукт NetWrix Event Log Manager має дуже зручний функціонал для керування журналами подій, яке забезпечує організаціям покращення безпеки, спрощує процеси аудиту та забезпечує відповідність

нормативним вимогам.

Системи моніторингу мережевої безпеки допомагають контролювати за безпекою трафіку та захищають від потенційних погроз. Наприклад, система Splunk збирає, індексує та аналізує машинні дані в режимі реального часу. Вона дозволяє отримувати цінну інформацію з даних, що генерують інфраструктура, застосунки, системи безпеки та IoT пристрої [52].

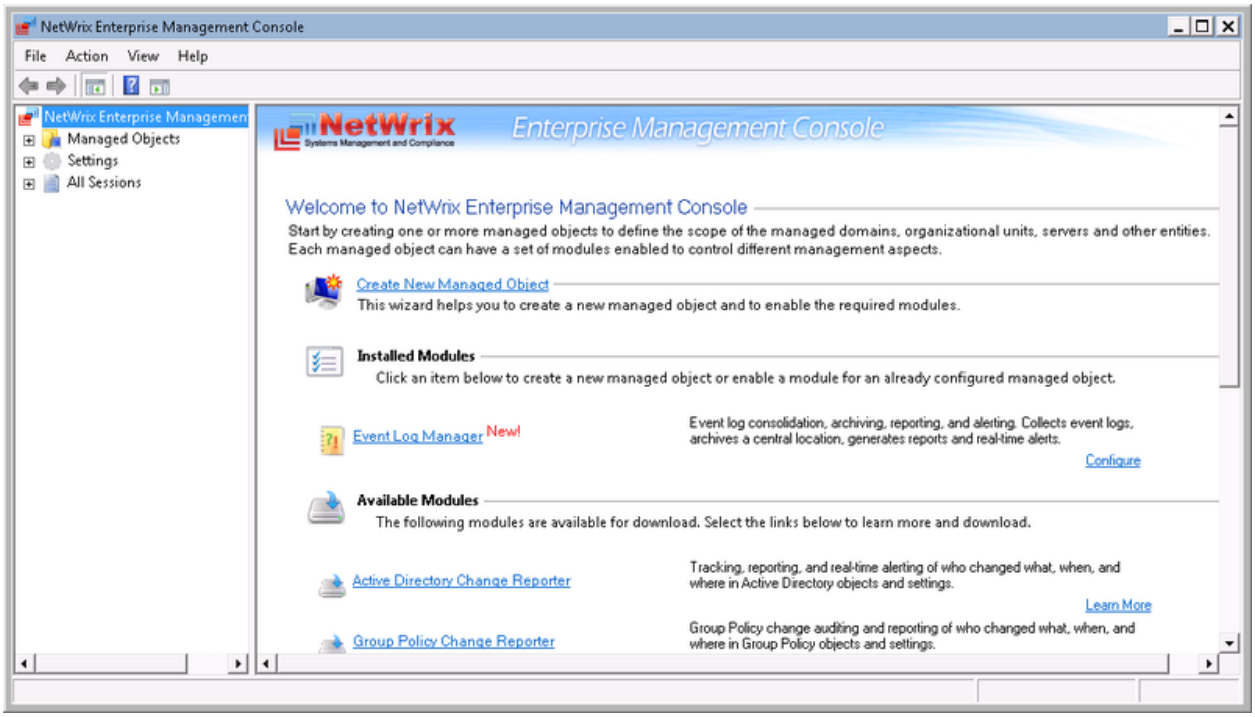

Рис. 1.5. Екран програмної системи NetWrix Event Log Manager [38]

Основними можливостями програми Splunk є наступні [52]:

- збір даних з різних джерел, включаючи журнали подій, мережі пристрої, бази даних і застосунки;

- індексування даних для швидкого пошуку та аналізу;

- має потужну пошукову мову для виконання складних запитів та аналізу даних;

- можливість проведення аналітики, включаючи створення кореляцій та виявлення аномалій;

- налаштування спостереження в реальному часі для відстежування ключових показників і подій;

- створення звітів для візуального представлення отриманих результатів;

- можливість створення налаштовуваних візуалізацій для різних типів даних;

- підтримка інтеграції з різними системами та застосунками через програмний інтерфейс і конектори;

- масштабованість системи, яка дозволяє оброблювати великі об’єми даних та підтримувати зростаючі вимоги організацій;

- підтримка сценаріїв використання сторонніх програм, таких як TSIEM;

- має інструменти для забезпечення безпеки та відповідності, включаючи аудит, спостереження та розслідування інцидентів;

- налаштування спостереження в реальному часі для моніторингу ключових показників і подій.

Один з екранів Splunk наведений на рисунку 1.6.

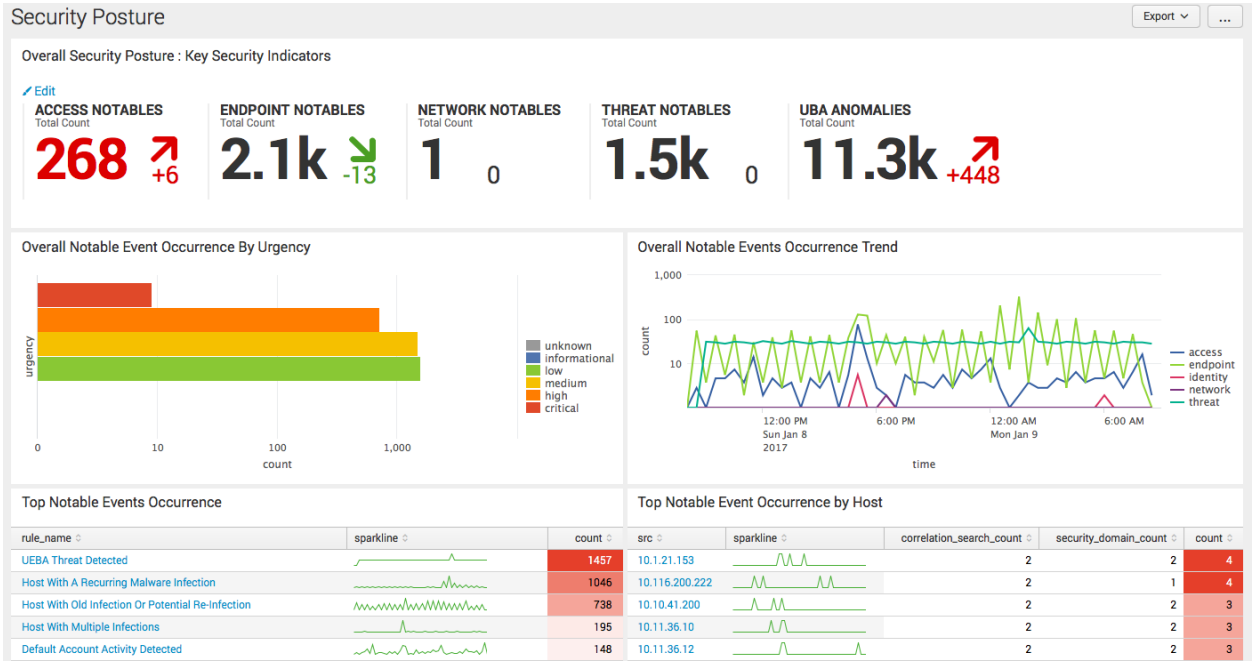

Рис. 1.6. Екран програмної системи Splunk [52]

Програмна система Splunk має багато універсальних рішень для аналізу даних, як допомагають витягувати цінну інформацію з наданих даних та забезпечити високу якість, продуктивність і безпеку.

1.3.2. Порівняння ефективності та надійності різних систем і технологій

Наразі важливість спостереження та аналіз даних неможливо переоцінити. Вищеописані програмні системи були порівняні за певними показниками:

1) призначення;
2) ефективність;
3) надійність;
4) основні функції;
5) підтримка джерел даних;
6) масштабованість;
7) інтеграція.

Їх порівняння наведені в таблиці 1.4.

Таблиця 1.4

Порівняльна характеристика програмних систем доказового спостереження

Параметр	Програмна система доказового спостереження					
	Tivoli Security Information and Event Manager	SolarWinds Network Performance Monitor	ManageEngine Applications Manager	Apache Spark	NetWrix Event Log Manager	Splunk
1	2	3	4	5	6	7
Призначення	Керування безпекою та подіями	Моніторинг мережевої продуктивності	Спостереження продуктивності застосунків	Обробка великих даних та аналітика	Керування та аналіз журналів подій	Аналіз машинних даних та спостереження
Ефективність	Керування безпекою та кореляціями подій	Мережевий моніторинг та діагностика	Моніторинг застосунків і серверів	Обробка та аналіз великих даних	Централізоване керування журналами	Аналіз даних у реальному часі
Надійність	Корпоративне рішення з добою підтримкою	Популярне рішення з гарною підтримкою	Популярне рішення з підтримкою та оновленнями	Відкритий код з широкою спільнотою та корпоративною підтримкою	Забезпечення довготривалого зберігання та відповідність нормативним вимогам	Корпоративне рішення з підтримкою та документацією
Основні функції	Кореляція подій,	Спостереження	Спостереження	Обробка великих	Збір, аналіз та	Аналіз даних,

	аналіз погроз, спостереження безпеки	продуктивності мережі, візуалізація сповіщення	застосунків, серверів, баз даних, звітність	даних, машинне навчання, потокова обробка	зберігання журналів подій	моніторинг у реальному часі, візуалізація
Підтримка джерел даних	Журнали подій, мережеві пристрої, застосунки	Мережеві пристрої, сервери	Застосунки, бази даних, сервери, хмарні сервіси	Різні формати даних	Журнали подій Windows, Syslog	Журнали подій, мережеві пристрої, застосунки, бази даних

Продовження таблиці 1.4

1	2	3	4	5	6	7
Масштабованість	Для великих організацій	Для організацій будь-якого розміру	Для організацій будь-якого розміру	Для великих організацій	Для організацій будь-якого розміру	Для великих організацій
Інтеграція	З іншими продуктами розробника та сторонніми системами	З іншими продуктами розробника	З іншими продуктами розробника	З Hadoop та іншими через програмний інтерфейс застосунку	З SIEM та іншими через програмний інтерфейс застосунку	Із застосунками через програмний інтерфейс застосунку

Джерело: розробка автора

Як можна побачити з порівняльної характеристики розглянутих програмних продуктів (табл. 1.4), вони мають високу ефективність і надійність. Кожна із систем надає унікальні можливості та преваги, забезпечуючи високу продуктивність та надійність в області застосування. Вибір конкретного інструмента залежить від специфічних потреб користувача та задач, які необхідно вирішити.

1.4. Висновки до розділу 1

За першим розділом було виконано низку поставлених завдань. Розглянуто поточний стан систем доказового спостереження, який надав розуміння, що сфера розвивається дуже активно й на сьогоднішній день. Розглянуті основні методи та технологій, що використовуються в сучасних системах доказового моніторингу у сфері безпеки. Проаналізований розвиток датчиків у різного розу пристроїв доказового спостереження. Оглянута класифікація датчиків за критеріями та тенденції їх розвитку. Досліджені поточні програмні рішення доказового спостереження у сфері безпеки. Виконаний опис програмних продуктів та досліджена порівняльна характеристика за відповідними параметрами.

РОЗДІЛ 2.
ДОСЛІДЖЕННЯ РОЗВИТКУ ТЕХНОЛОГІЙ ЗБОРУ, ЛОГУВАННЯ ТА ПЕРЕДАЧІ ДАНИХ ЗА ДОПОМОГОЮ БЕЗДРОТОВИХ ПРИСТРОЇВ У СФЕРІ БЕЗПЕКИ

2.1. Збір відео- та аудіоінформації з використанням бездротових пристроїв під час спостереження

Збір відео- та аудіоінформації широко використовується в системах безпеки для різного роду спостережень. Безпека та конфіденційність – два важливі фактори, що забезпечують якість і надійність систем відеоспостереження у відповідних програмних системах. Тільки автентифіковані користувачі можуть активувати камери спостереження, а перегляд даних доступний тільки уповноваженим особам. Для запису процесу відеоспостереження слід використовувати модулі з камерами відео- та аудіозапису. Таким чином можна отримати запис на сервер та безпосередньо пряму трансляцію, яка дозволяє робити висновки про об’єкт, за яким ведеться моніторинг та порівнювати їх з показниками системи безпеки [2, 54].

Такі модулі повинні мати можливість до наступного [3, 56]:

- спостерігати за об’єктом у прямому ефірі;
- шифрувати відеопотік при прямих трансляціях;
- записувати відео за запитом;
- стискати відео, використовуючи відповідні алгоритми та передавати їх на сервер;
- виконувати резервні копії локально.

У даному дослідженні пропонується система охорони, яка містить модулі відеокамер для спостерігання за об’єктами, бездротову мережу Wireless Body Area Network (WBAN) з датчиками та модуль синхронізації,

стиснення та протоколювання даних до зовнішньої мережі. Після того, як певний датчик у WBAN повідомляє про подію, користувач відповідає за активацію камери. Конфіденційність визначає, що сторонні користувачі не зможуть подавати запити. Після успішної автентифікації відеокамера фіксує відео та відправляє його в зашифрованому вигляді до хмарного сховища. Сховище, у свою чергу, забезпечує актуалізацію даних, зберігає зашифровані дані, керує автентифікуванням користувачів та надсилає повідомлення про отримані зашифровані відео уповноваженим особам (охоронникам, адміністраторам тощо). Після чого особи з відповідними вповноваженнями можуть розшифрувати відео та прийняти відповідні мірі безпеки [28, 58].

Схема системи наведена на рисунку 2.1.

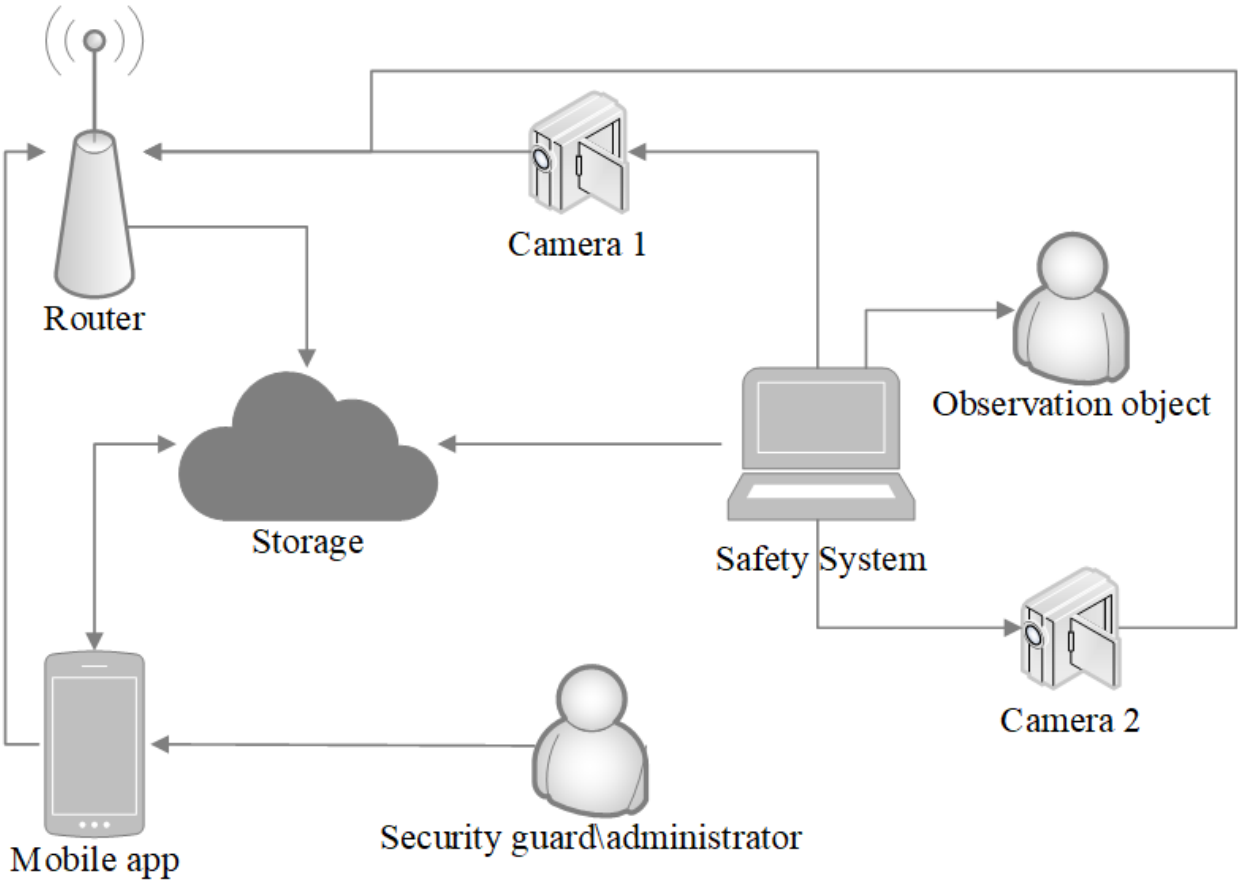

Рис. 2.1. Схема системи безпеки доказового спостереження

За наданою схемою (рис. 2.1) важливо зауважити, що існує декілька наступних проблем з використанням бездротових Wi-Fi відеопристроїв для

спостереження:

1) затримка та якість зображення пропорційно залежні один від одного; чим вище якість відеозображень, тим більше місця займає файл у сховищі;

2) існує ймовірність перехватити та змінити відеосигнал сторонніми особами.

Для вирішення описах проблем можна вдатися до таких рішень, як стиснення відео за допомогою спеціально розроблених алгоритмів і використання алгоритмів шифрування.

Наприклад, існує програмне рішення від компанії Verkada, які широко розповсюджені в системах безпеки для спостереженням за об’єктами. Серед переваг системи можна виділити високу якість зображення, можливість у реальному часі переглядати трансляцію та записи з відеокамери як віддалено, так і локально. Для проєктування власних рішень можна використовувати мікроконтролери. Найбільш популярними є ESP та Raspberry Pi, вони наведені на рисунках 2.2-2.3. Обидва мають вбудовані модулі Wi-Fi [9, 11, 29].

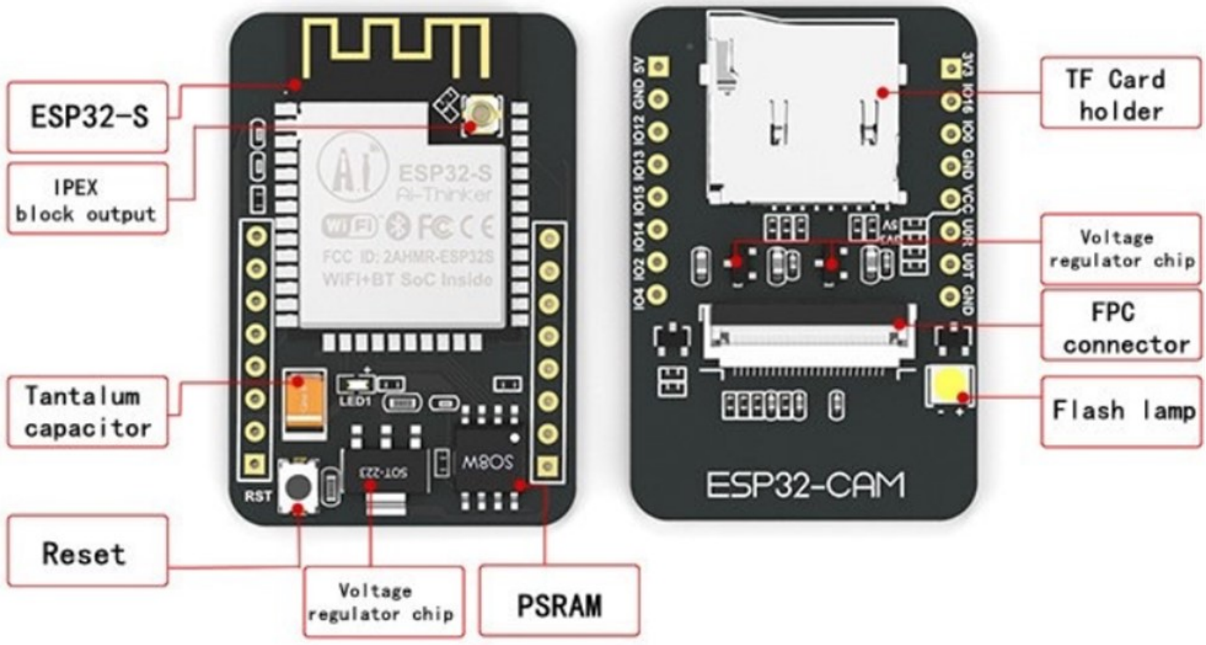

Рис. 2.2. Мікроконтролер ESP [11]

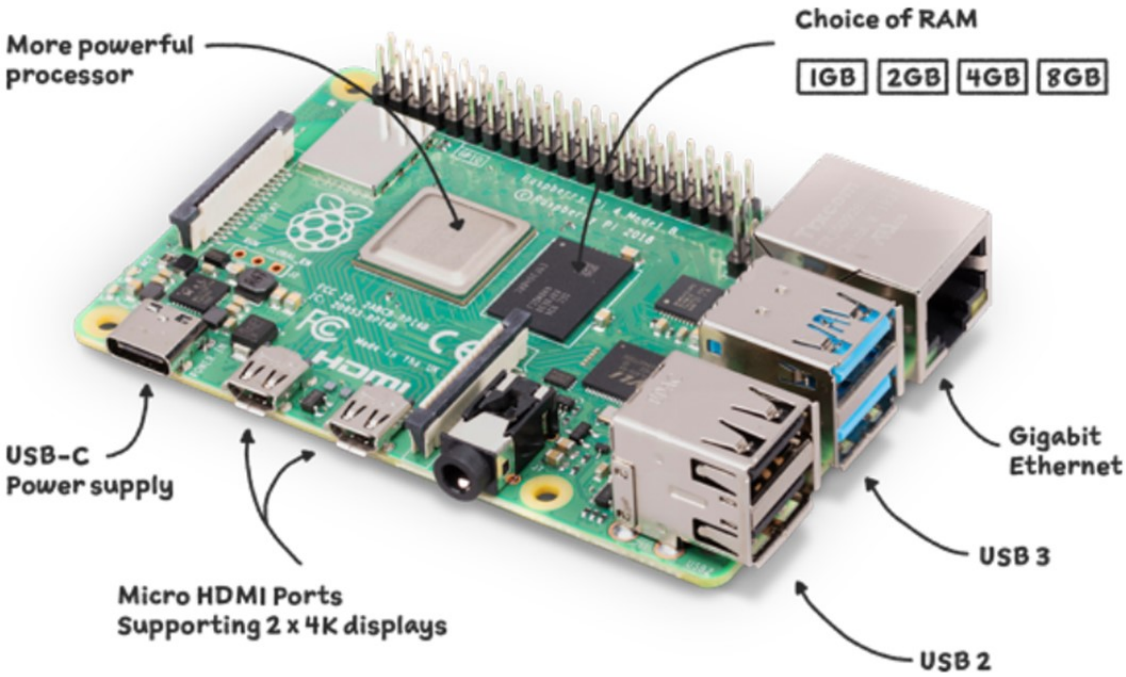

Рис. 2.3. Мікроконтролер Raspberry Pi [9]

Для порівняння даних мікроконтролерів була створена порівняльна характеристика, яка наведена в таблиці 2.1.

Таблиця 2.1

Порівняльна характеристика мікроконтролерів ESP та Raspberry Pi

Характеристика	ESP	Raspberry Pi
1	2	3
Процесор	32-біт, 2 ядра, до 160 МГц	64-біт, Broadcom BCM2711 ARM Cortex-A72, Quad Core 1.5GHz
Вбудована пам’ять	520 КБ, зовнішня 4MB статична пам’ять з довільним доступом	Пам’ять з довільним доступом 2 / 4/ 8 Гб
Бездротові інтерфейси	Wi-Fi 802.11b/g/n, Bluetooth	Wi-Fi 802.11b/g/n/ac, Bluetooth 5.0
Наявність камер	OV2640	Підтримка різних камер через порт

Порти введення / виведення	UART, SPI, I2C, PWM, ADC, DAC	GPIO, UART, SPI, I2C, PWM, Ethernet, USB
Підтримка карт пам’яті	Так	Так
Розміри модуля, мм	27 x 39 мм	85.6 x 56.5 мм
Енергоспоживання	Низьке	Середнє
Криптографія	AES, SHA-2, RSA, ECC, RNG	Підтримка через програмне забезпечення
Відео	Немає	VideoCore VI GPU
Підтримка програмування	Arduino IDE, PlatformIO	Python, Scratch тощо
Інтерфейси для камери	Вбудований інтерфейс для камери	Підтримка USB-камер

Джерело: розробка автора

Перевагами ESP є наступне [11]:

- компактність і можливість використання для прямої трансляції;

- підключення до загальної мережі та трансляція через веббравзер із шифруванням відео в режимі реального часу з використанням алгоритму Advanced Encryption Standard (AES).

Для відображення відео, модуль приєднується до мережі. Після отримання відеозапису, воно шифрується з використанням алгоритму та дані зберігаються на картці пам’яті та направляється на локальний сервер. Відповідна схема роботи контролера наведена на рисунку 2.4.

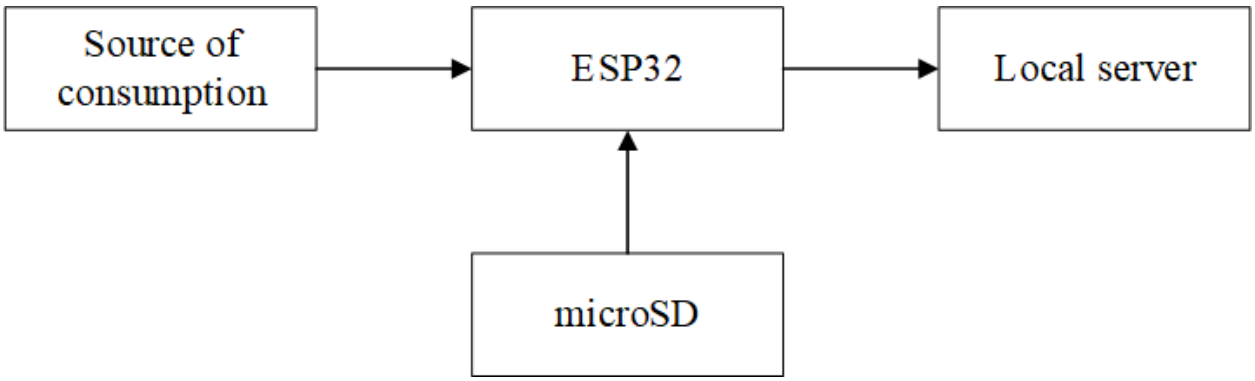

Рис. 2.4. Схема модуля трансляції ESP

Алгоритм прямої трансляції ESP наведений на рисунку 2.5.

Як можна побачити (рис. 2.5) алгоритм у контролера ESP невеликий, але він є дієвим і надійним за рахунок алгоритмів шифрування та дешифрування.

Перевагами Raspberry Pi є [9]:

- більш потужний варіант контролера для роботи із записом відео та його стисненням завдяки більш потужному процесору та відеочіпу;
- можливість запуску програм на різних мовах програмування для реалізації алгоритмів стиснення відео;
- підключення до більш потужної вебкамері завдяки наявності портів.

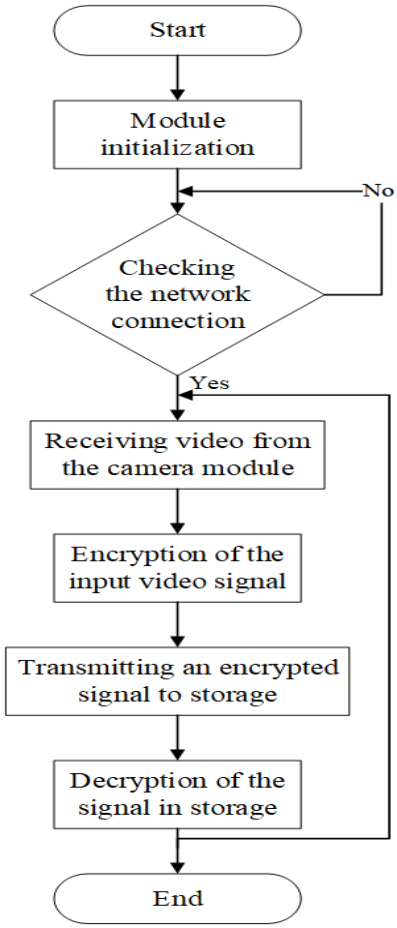

Рис. 2.5. Алгоритм прямої трансляції модуля ESP

Запис відео починається, коли користувач натискає на відповідну кнопку. Після його завершення, запис оброблюється в режимі реального часу та зберігається на карті пам’яті до передачі на сховище. Схема модулю трансляції контролеру наведена на рисунку 2.6.

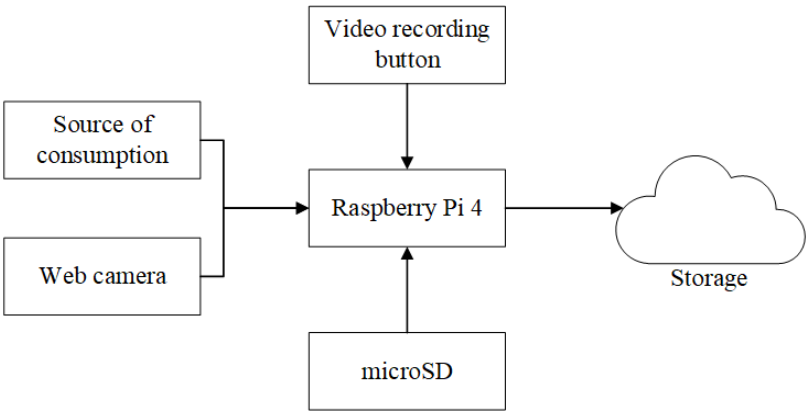

Рис. 2.6. Схема модуля трансляції Raspberry Pi

Алгоритм прямої трансляції Raspberry Pi наведений на рисунку 2.7.

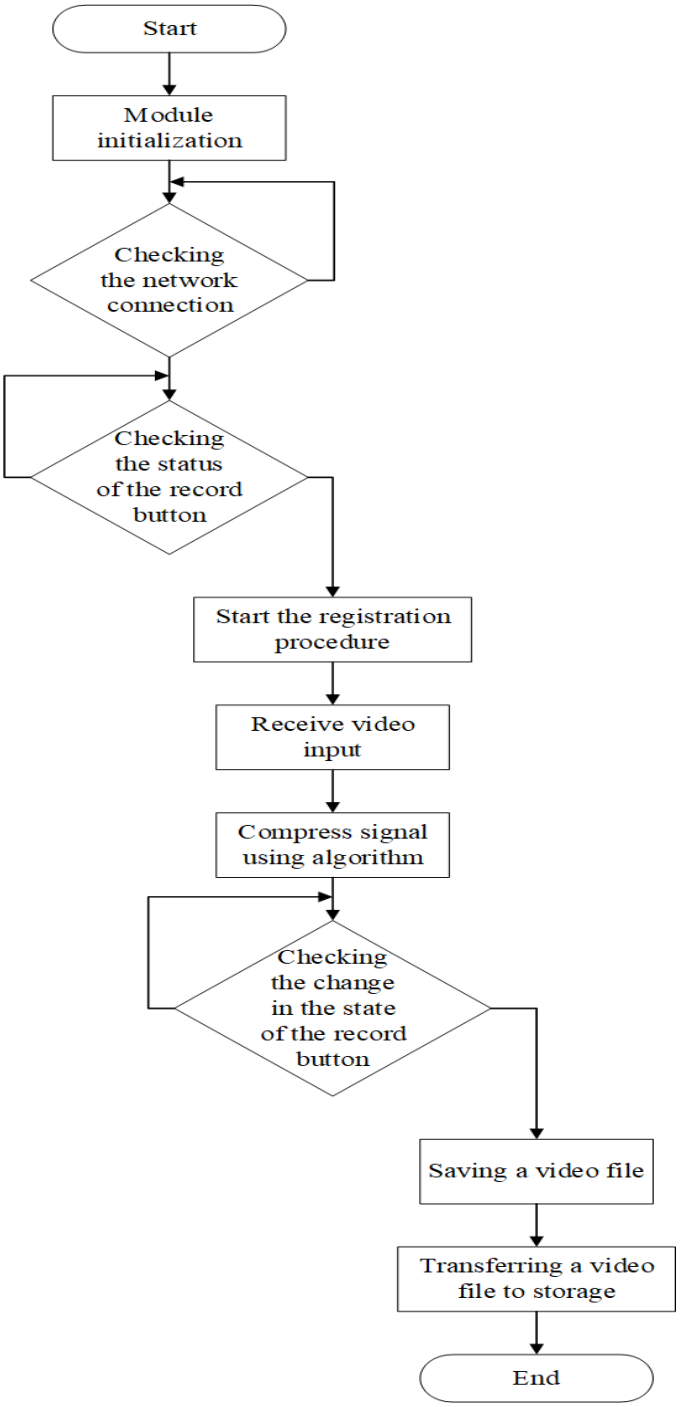

Рис. 2.7. Алгоритм прямої трансляції модуля Raspberry Pi

У алгоритмах (рис. 2.5, 2.7) показана здатність забезпечення передачі даних через обидва контролери. Модулі мають майже однаковий функціонал, але ширші можливості має Raspberry Pi. Він забезпечує запис, стиснення та безпосередній зв’язок камери із сервером.

2.2. Проєктування структури модуля синхронізації даних та розробка

порівняльного аналізу збору параметрів про об’єкт спостереження

Для збору спостережуваної інформації необхідні невеликі розміри модулю, автономне живлення та наявність бездротового інтерфейсу для передачі даних. Основним елементом для створення таких модулів є контролери (ESP або Raspberry Pi), які в значному ступені впливають на функціонал (табл. 2.1). Якщо ж обрати їх моделі, то найкращим для аналізу підійдуть дієві мікроконтролери ESP32 WROOM та Raspberry Pi Zero 2 W [4, 15].

Їх зображення наведені на рисунках 2.8-2.9.

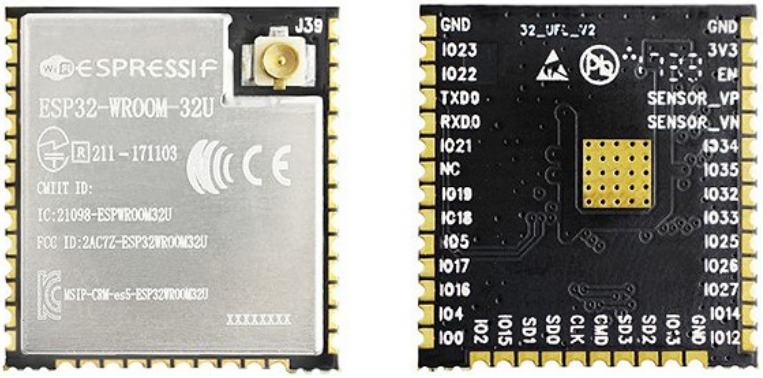

Рис. 2.8. Мікроконтролер ESP32 з модулем WROOM [15]

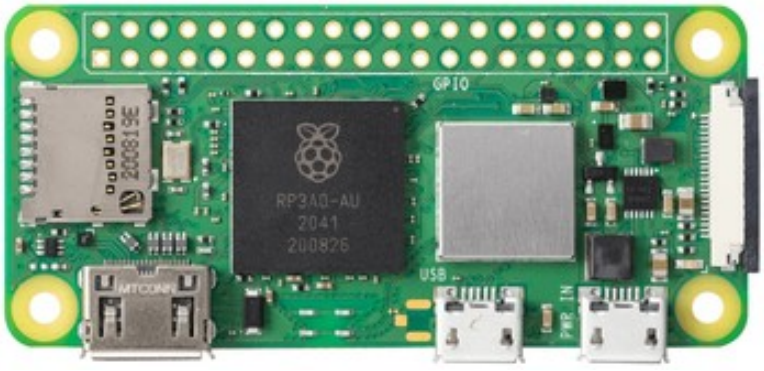

Рис. 2.9. Мікроконтролер Raspberry Pi з модулем Zero 2 W [4]

Порівняльна характеристика наведена в таблиці 2.2.

Таблиця 2.2

Порівняльна характеристика мікроконтролерів ESP32 WROOM та

Raspberry Pi Zero 2 W

Характеристика	ESP32 WROOM	Raspberry Pi Zero 2 W
Процесор	ESP32, 2 ядра, 32-біт	Broadcom BCM2710A1, 4 ядра, 64-біт
Бездротовий інтерфейс	Bluetooth + WiFi 2.4GHz	Bluetooth + WiFi 2.4GHz
Живлення, В	2,7-3,6 В	5 В
Кількість портів	48	40
Підтримка інтерфейсу	UART, SPI, I2C, PWM, ADC, DAC	UART, SPI, I2C, PWM, ADC, DAC
Розміри, мм	25,5 х 18	65 х 30
Ціна, $	10	50

Джерело: розробка автора

За виконаним аналізом (табл. 2.2) можна стверджувати, що мікроконтролер ESP32 WROOM є вигідним рішенням через ідентичні характеристики до опонента, меншу ціну та розміри.

Для більш точного спостереження необхідно додатково датчик температури повітря, який буде допомагати більш точно моніторити за об’єктом. Таким датчиком є AM2302. Він широко використовується в сфері безпеки для вимірювання температури та вологості повітря. Також він підходить для застосунків, що потребують високоточної реєстрації даних. Датчик має низьке енергоспоживання, вимірювання температури від мінус сорока до плюс вісімдесяти градусів за Цельсію, виміряє вологість повітря в діапазоні від 0 до 100%, має цифровий інтерфейс і протоколи передачі даних. AM2302 споживає до 3,6 В, що гарно стикається із обраним контролером та має вбудовані можливості для керування енергоспоживанням. Це дозволяє йому бути інтегрованим у системи з автономним живленням [19].

Датчик AM2302 наведений на рисунку 2.10.

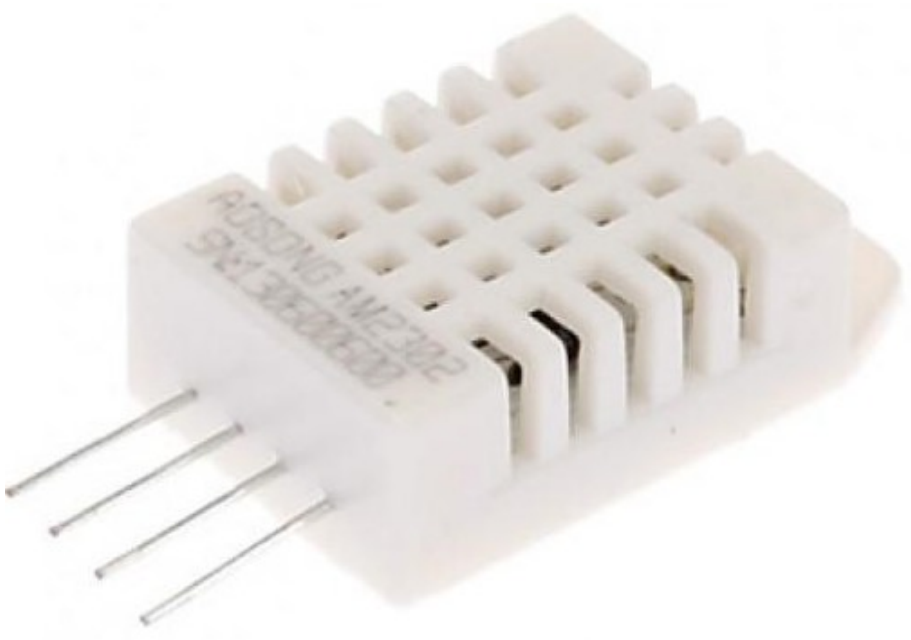

Рис. 2.10. Датчик AM2302 [19]

Датчик використовується часто у системах безпеки для наступних завдань [19]:

- підтримка оптимальних умов температури та вологості в приміщеннях для комфортного перебування в них;

- у комбінації з іншими датчиками, наприклад, повітря, можна своєчасно відреагувати та застережити пожежу;

- захист обладнання, які чутливі до вологості або температури;

- спостерігання умов зберігання матеріалів тощо.

На основі датчику побудований модуль збору даних про темпераутрні дані. Модуль містить в собі мікроконтролер, який оброблює дані, що надходять з датчика та передає їх по бездротовому інтерфейсу до центральної системи спостереження. AM2302 дозволяє відстежувати будь-які зміни в положенні та руху об’єктів. Такі можливості роблять його ідеальним рішенням для систем безпеки, що потребують високої точності та надійності.

Електрична схема датчику наведена на рисунку 2.11.

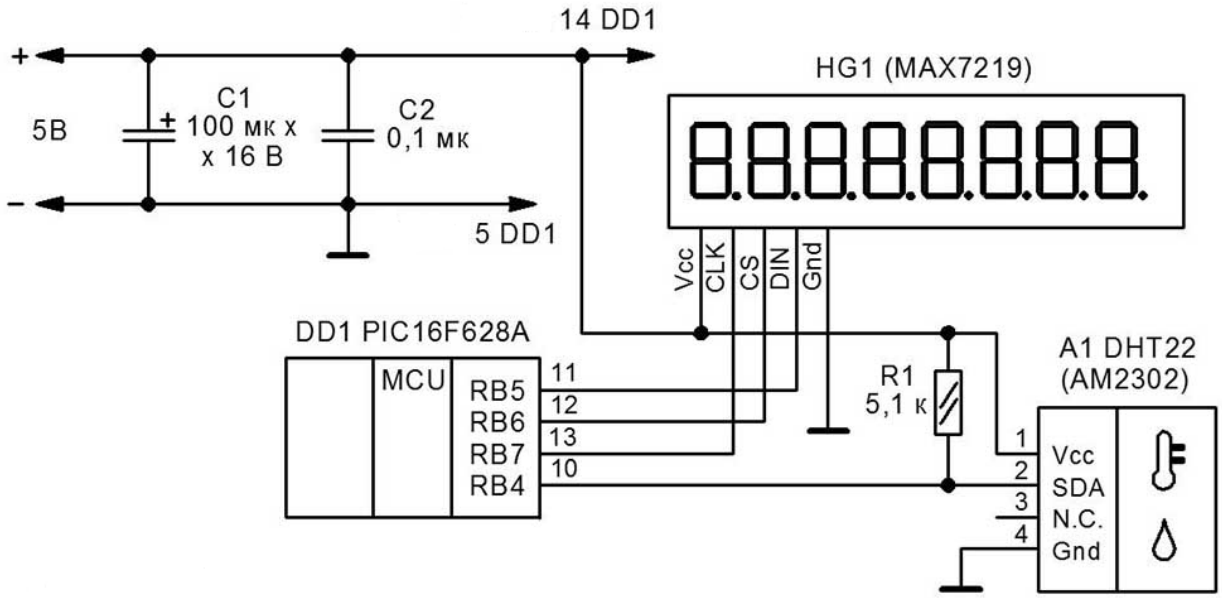

Рис. 2.11. Електросхема датчика AM2302 [19]

Схема модуля збору даних температури наведена на рисунку 2.12.

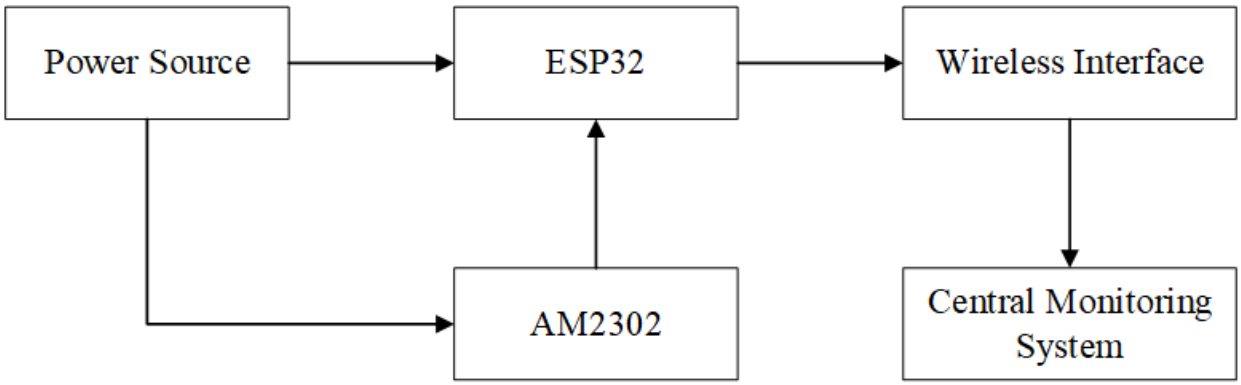

Рис. 2.12. Схема модуля збору даних температури та вологості датчика AM2302

За схемою (рис. 2.12) можна побачити такі елементи в роботі:

- джерело живлення, яке забезпечує енергією мікроконтролер та датчик;

- мікроконтролер, який оброблює дані з датчика та передає їх до бездротового інтерфейсу;

- датчик, який реєструє дані руху та передає їх до мікроконтролера;

- бездротовий інтерфейс, який відправляє оброблені дані в центральну систему спостереження;

- центральна система спостереження, яка приймає та аналізує дані, що надходять з бездротового інтерфейсу.

Така схема дозволяє ефективно збирати та передавати дані про температуру та вологість для подальшого аналізу та спостереження.

Для додаткового спостереження за об’єктом можна також взяти датчик виміру рівня газу MQ-135. Даний датчик може виявляти різні види шкідливих газів. Він широко використовується в системах спостереження якості повітря для забезпечення безпеки [16].

Датчик MQ-135 наведений на рисунку 2.13.

Рис. 2.13. Датчик газу MQ-135 [16]

На основі датчику побудований модуль збору даних про якість повітря. Він включає в себе мікроконтролер, який оброблює дані, що надходять із датчика та передає їх до бездротовому пристрою інтерфейсу в центральну систему спостереження [16].

Електрична схема датчику газу MQ-135 наведена на рисунку 2.14.

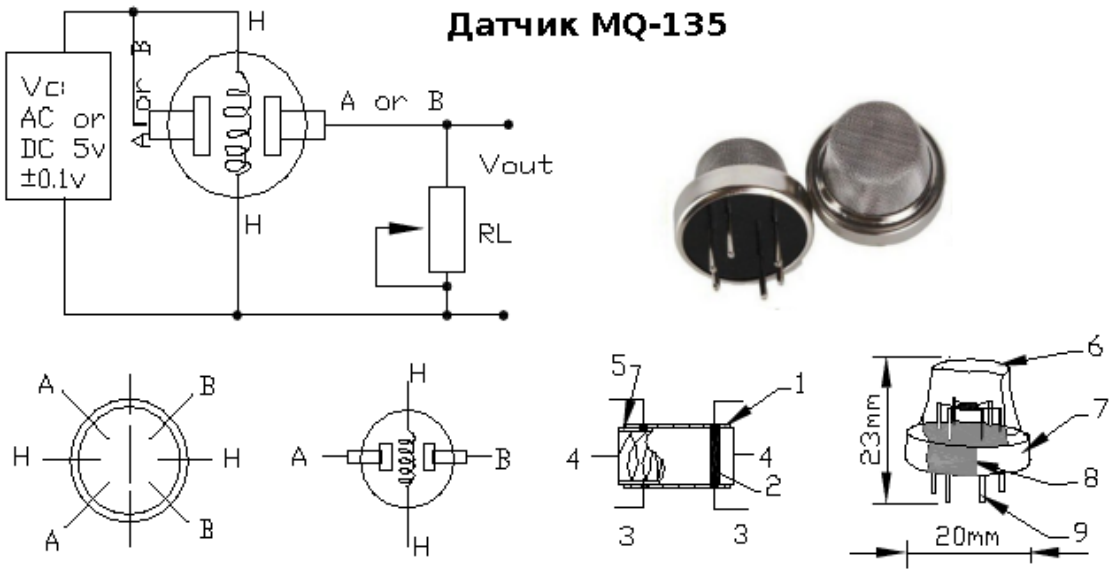

Рис. 2.14. Електрична схема датчику газу MQ-135 [16]

Датчик дозволяє відстежувати концентрацію різних шкідливих газів у повітрі, що робить його ідеальним розширенням для схем безпеки, що потребують високої точності та надійності. Він має інтегрований високочутливий інтерфейс, простий у використанні для підключення до будь-якого мікроконтролера та має низьке енергоспоживання [16].

Схема модуля збору даних про якість повітря з використанням датчику газу MQ-135 наведена на рисунку 2.15.

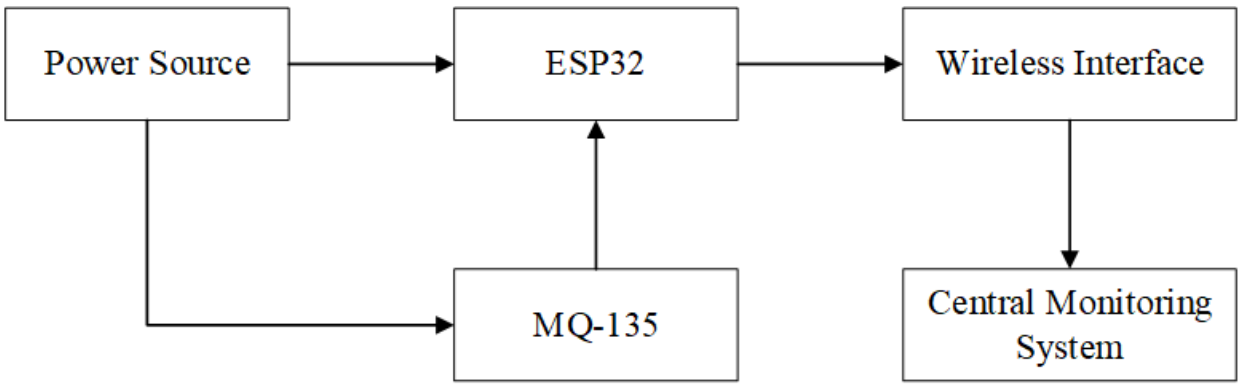

Рис. 2.15. Схема модуля збору даних про якість повітря датчика MQ-135

Алгоритм роботи подібний попередньому датчику.

2.3. Проєктування структури модуля синхронізації даних та розробка порівняльного аналізу

У випадку поточного дослідження передбачено два датчики. При виконанні проєктування IoT рішень передбачено, що дані датчиків надходять до сховища / серверу при умові, що вони є синхронізованими, проаналізовані, стисненні та відправлені до сховища. Дані повинні передаватися, використовуючи програмний інтерфейс служби безпеки Application Programming Interface (API).

Схема передачі таких даних наведена на рисунку 2.16.

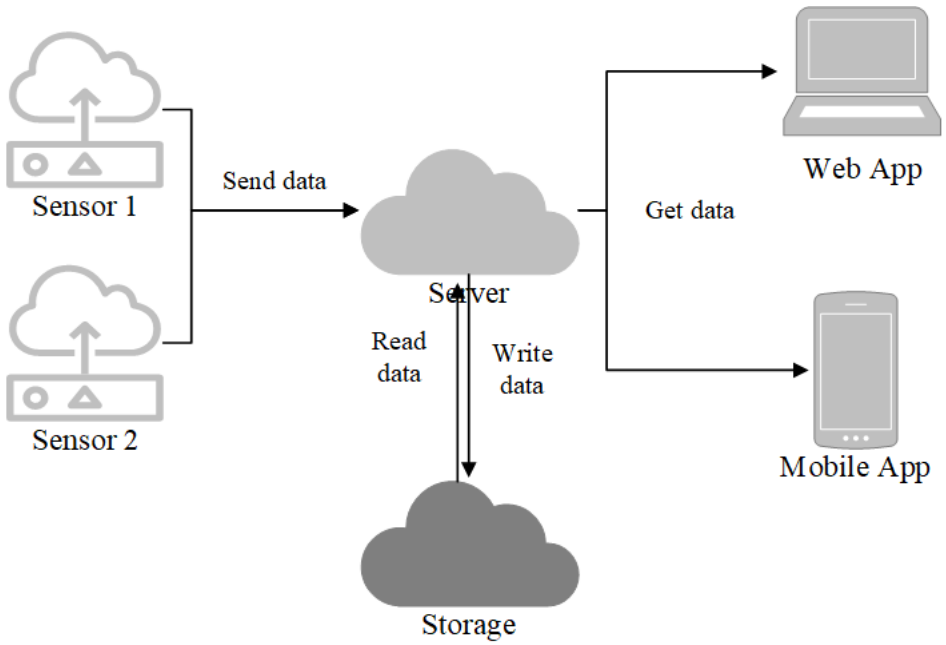

Рис. 2.16. Схема передачі даних із синхронізованих датчиків

Для отримання даних із обраних датчиків необхідний бездротовий інтерфейс передачі даних. Наприклад, можна використати Wi-Fi HaLow. Він забезпечує велику дальність передачі, низьке енергоспоживання та здатність роботи в низькочастотних діапазонах. Усі ці фактори забезпечують краще проникнення сигналу через стіни та інші перешкоди [57].

Перевагами використання Wi-Fi HaLow є такі аспекти [57]:

- дальність передачі до одного кілометра, що дозволяє використати датчики у великих та багатозонних об’єктах спостереження;

- низьке енергоспоживання, що важливо для роботи пристрою на батареї тривалий час;

- працездатність на низьких частотах, що забезпечує краще проникнення сигналу через перешкоди;

- підтримка сучасних проколів безпеки та стабільність передачі даних.

Наприклад, є сценарій спостереження якості повітря та температури у будівлі для забезпечення безпеки та своєчасного виявлення небезпечних умов. У системі за обраним інструментарієм відбувається наступний алгоритм дій:

1) датчик газу MQ-135 вимірює концентрацію газів та надсилає аналогові дані до контролера ESP32;

2) датчик AM2302 вимірює прискорення за трьома осями та надсилає цифрові дані до контролера ESP32, використовуючи інтерфейс проколу зв’язку Inter-Integrated Circuit (I2C) або Serial Peripheral Interface (SPI);

3) контролер ESP32 WROOM виконує обробку даних з датчиків та виконує попередній аналіз і передає їх по Wi-Fi HaLow на центральній сервер або сховище;

4) центральний сервер або сховище отримує дані, аналізує їх та генерує попередження у випадку порогових значень концентрації повітря або температури.

Схема описаного алгоритму дій наведена на рисунку 2.17.

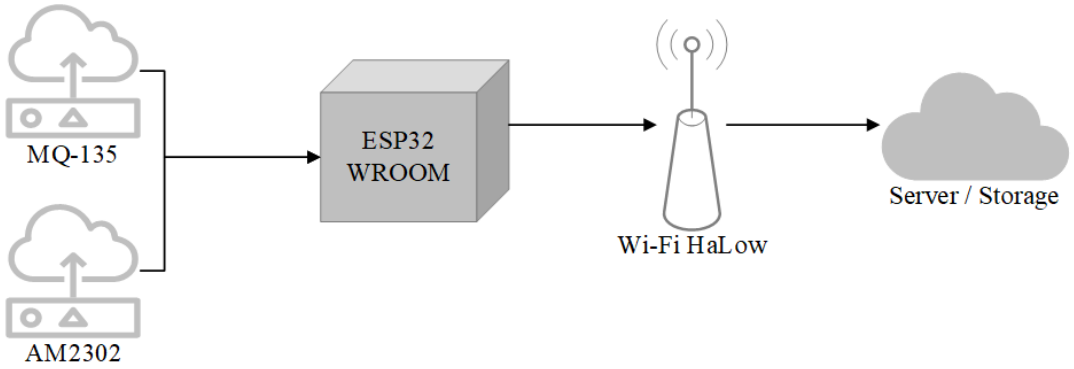

Рис. 2.17. Схема системи спостереження безпеки з використанням ESP32 WROOM, MQ-135, AM2302

Для роботи алгоритму доданий таймер, який координує завершення отриманих даних із датчиків. Після цього дані проходять аналіз та надсилаються до серверу/сховища. Якщо таймер не отримав ніякої інформації, він надсилає повідомлення про помилку та йде на перезапуск. Алгоритм синхронізації датчиків, аналіз їх даних, виконання стиснення даних та передача даних наведений на рисунку 2.18.

Таким чином, у дослідженні приведений остаточний алгоритм роботи системи безпеки доказового спостереження за об’єктом. Для обчислення інтегральної оцінки необхідно знати критичні значення із датчиків повітря та температури. За експериментальними даними були сформовані вибірки з дев’яти величин, які необхідно вимірювати, на підставі яких виноситься результат про їх відповідність. На основі аналізу даних необхідно розробити модель, яка забезпечує та алгоритм, що вимірює подальші спостереження роботи системи безпеки. Для цього необхідно використати статичний аналіз та критерій Ст’юдента [47].

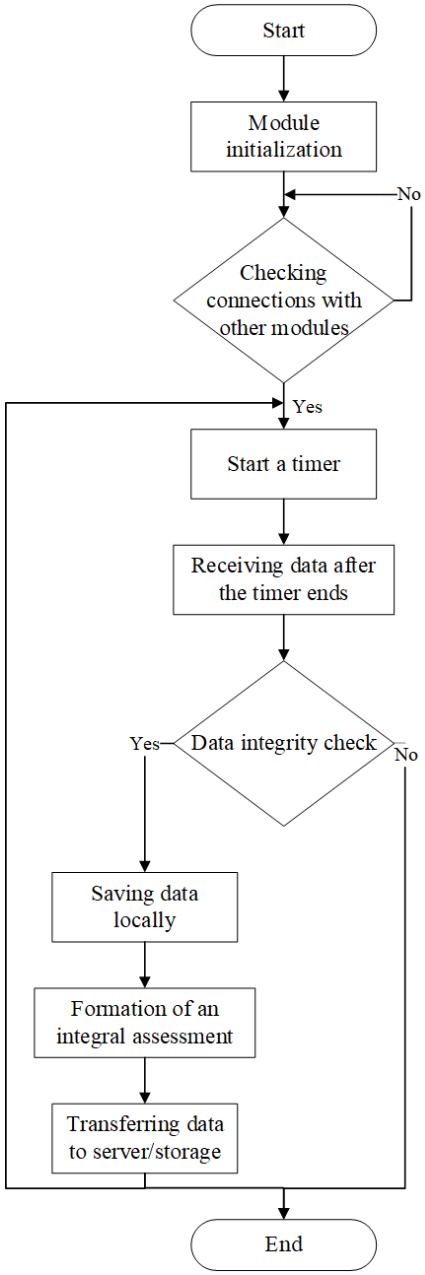

Рис. 2.18. Алгоритм синхронізації датчиків, аналізу даних, стиснення наданих даних та передачі даних до серверу або сховища

Т-критерій Ст’юдента підходить для написання необхідного скрипту, частина якого наведена на рисунку 2.19, а повний уміст в додатку А, файл А.1.

```python
import numpy as np

# Дані з датчика повітря: концентрація аміаку
sensor1_1 = [30, 32, 29, 34, 31, 33, 28, 35, 30]
sensor1_2 = [22, 25, 27, 23, 24, 26, 28, 21, 25]

# Обчислення середнього арифметичного і стандартного відхилення
s1 = np.sum(sensor1_1)
av1 = np.round(s1 / len(sensor1_1), decimals: 1)
print(f"X1 = {av1}")

arrs1 = []
for i in sensor1_1:
    f = np.square(i - av1)
    arrs1.append(f)

b1 = np.sum(arrs1)
c1 = np.sqrt(b1 / (len(sensor1_1) - 1))
c1_rounded = np.round(c1, decimals: 2)
print(f"Sigma1 = {c1_rounded}")

d1 = c1 / np.sqrt(len(sensor1_1))
d1_rounded = np.round(d1, decimals: 2)
print(f"Delta1 = {d1_rounded}")
```

Рис. 2.19. Скрипт розрахунку критерію Ст’юдента

Наведений скрипт (рис. 2.19) обчислює середні значення, дисперсії, стандартні відхилення і помилки, t-критерії для обох датчиків.

У першу чергу необхідно знайти арифметичне середнє ($\underline{X_k}$), далі стандартне відхилення (σ_k) та значення стандартної помилки (Δ_k) для обох датчиків.

Формула арифметичного середнього наведена нижче [39]:

$$\underline{X_k} = \frac{a_1+a_2+\cdots+a_n}{n}, \tag{2.1}$$

де: a_n – дані, які надійшли з датчику;

n – загальна кількість даних, які надійшли з датчику.

Для визначення середньоквадратичного відхилення, необхідно

використати наступну формулу [8]:

$$\sigma_k = \sqrt{\frac{(a_1-\underline{X_k})^2+(a_2-\underline{X_k})^2+\cdots+(a_n-\underline{X_k})^2}{n-1}}, \quad (2.2)$$

де: $(a_n - \underline{X_k})^2$ – квадратична сума даних з датчику та арифметичного середнього.

Значення стандартної помилки визначається за формулою [53]:

$$\Delta_k = \frac{\sigma_k}{\sqrt{n}} \quad (2.3)$$

Роздивимося приклад, у якому необхідно виміряти дані повітря у декількох класифікаціях та температури й вологості у приміщенні з ідеальними умовами. У таблиці 2.3 наведені експериментальні дані замірів з датчику повітря у вигляді концентрації аміаку в повітрі.

Таблиця 2.3

Експериментальні дані з датчику повітря: концентрація аміаку

№	Концентрація аміаку, ppm								
	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
1	30	32	29	34	31	33	28	35	30
2	22	25	27	23	24	26	28	21	25

Джерело: розробка автора

Для першої групи даних обчислення наведено нижче. Обчислення арифметичного середнього для другої групи даних з датчику повітря у вимірі аміаку за формулою 2.1:

$$\underline{X_1} = \frac{30 + 32 + 29 + 34 + 31 + 33 + 28 + 35 + 30}{9} = 31{,}3$$

Обчислення середнього квадратичного відхилення для першої групи даних з датчику повітря у вимірі аміаку за формулою 2.2:

$$\sigma_1 = \sqrt{\frac{(30 - 31{,}3)^2 + (32 - 31{,}3)^2 + (29 - 31{,}3)^2 + \cdots + (30 - 31{,}3)^2}{9 - 1}}$$

$$= 2{,}35$$

Тоді стандартна помилка для першої групи даних з датчику повітря у вимірі аміаку за формулою 2.3:

$$\Delta_1 = \frac{2{,}4}{\sqrt{9}} = 0{,}78$$

Для другої групи обчислення наведено нижче. Обчислення арифметичного середнього для першої групи даних з датчику повітря у вимірі аміаку за формулою 2.1:

$$\underline{X_2} = \frac{22 + 25 + 27 + 23 + 24 + 26 + 28 + 21 + 25}{9} = 24{,}6$$

Далі необхідно обчислити середнє квадратичне відхилення для другої групи даних з датчику повітря у вимірі аміаку за формулою 2.2:

$$\sigma_2 = \sqrt{\frac{(22 - 24{,}6)^2 + (25 - 24{,}6)^2 + (26 - 24{,}6)^2 + \cdots + (25 - 24{,}6)^2}{9 - 1}} = 2{,}3$$

Тоді стандартна помилка для другої групи даних з датчику повітря у вимірі аміаку за формулою 2.3:

$$\Delta_2 = \frac{2,3}{\sqrt{9}} = 0,77$$

Отож, можна порахувати критерій достовірності за наступною формулою [5]:

$$t = \frac{X_n - X_{n+1}}{\sqrt{\Delta_n^2 + \Delta_{n+1}^2}} \quad (2.4)$$

За формулою 2.4 необхідно порахувати критерій достовірності датчику повітря за даними про аміак:

$$t_{1-2} = \frac{31,3 - 24,6}{\sqrt{2,35^2 + 2,3^2}} = 2,03$$

Після знаходження критерію, необхідно порівняти його з табличним значенням, яке наведено в додатку Б, рис. Б.1. За таблицею Ст'юдента для заданого числа ступеню свободи ($f = 16$) та рівня значимості ($\alpha = 0,05$) табличне значення приблизно дорівнює $t_t = 2,12$. Тому можна стверджувати, що дані на 99% є достовірними, оскільки $t_{1-2} < t_t$.

У таблиці 2.4 наведені експериментальні дані замірів з датчику повітря у вигляді концентрації диму в повітрі. Так як умови ідеальні, то й концентрація диму повинна бути мінімальною.

Таблиця 2.4

Експериментальні дані з датчику повітря: концентрація диму

№	Концентрація диму, ppm								
	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
1	1	2	1	3	1	2	2	1	0

2	0	1	1	2	0	1	1	0	2

Джерело: розробка автора

Для першої групи даних обчислення наведено нижче. Обчислення арифметичного середнього для першої групи даних з датчику повітря у вимірі диму за формулою 2.1:

$$\underline{X_3} = \frac{1+2+1+3+1+2+2+1+0}{9} = 1{,}4$$

Обчислення середнього квадратичного відхилення для першої групи даних з датчику повітря у вимірі диму за формулою 2.2:

$$\sigma_3 = \sqrt{\frac{(1-1{,}4)^2 + (2-1{,}4)^2 + (1-1{,}4)^2 + \cdots + (0-1{,}4)^2}{9-1}} = 0{,}88$$

Тоді стандартна помилка для першої групи даних з датчику повітря у вимірі диму за формулою 2.3:

$$\Delta_3 = \frac{0{,}88}{\sqrt{9}} = 0{,}29$$

Для другої групи даних обчислення наведено нижче. Обчислення арифметичного середнього для першої групи даних з датчику повітря у вимірі диму за формулою 2.1:

$$\underline{X_4} = \frac{0+1+1+2+0+1+1+0+2}{9} = 0{,}9$$

Обчислення середнього квадратичного відхилення для другої групи даних з датчику повітря у вимірі диму за формулою 2.2:

$$\sigma_4 = \sqrt{\frac{(0-0{,}9)^2 + (1-0{,}9)^2 + (1-0{,}9)^2 + \cdots + (2-0{,}9)^2}{9-1}} = 0{,}78$$

Тоді стандартна помилка для другої групи даних з датчику повітря у вимірі диму за формулою 2.3:

$$\Delta_4 = \frac{0{,}78}{\sqrt{9}} = 0{,}26$$

Далі за формулою 2.4 необхідно порахувати критерій достовірності датчику повітря за даними про дим:

$$t_{3-4} = \frac{1{,}4 - 0{,}9}{\sqrt{2{,}4^2 + 2{,}3^2}} = 1{,}27$$

За таблицею Ст'юдента (додаток Б, рис. Б.1) для заданого числа ступеню свободи ($f = 16$) та рівня значимості ($\alpha = 0{,}05$) табличне значення приблизно дорівнює $t_t = 2{,}12$. Тому можна стверджувати, що дані на 99% є достовірними, оскільки $t_{3-4} < t_t$.

У таблиці 2.5 наведені експериментальні дані замірів з датчику температури повітря.

Таблиця 2.5

Експериментальні дані з датчику температури

№	Температура, °C								
	1	2	3	4	5	6	7	8	9
1	22	23	22	23	21	23	24	24	23
2	21	23	22	24	22	22	22	22	21

Джерело: розробка автора

Для першої групи даних обчислення наведено нижче. За формулою 2.1 потрібно обчислити арифметичне середнє для першої групи даних датчику температури:

$$\underline{X_5} = \frac{22 + 23 + 22 + 23 + 21 + 23 + 24 + 24 + 23}{9} = 22{,}8$$

За формулою 2.2 потрібно обчислити значення середнього квадратичного відхилення для першої групи даних датчику температури:

$$\sigma_5 = \sqrt{\frac{(22 - 22{,}8)^2 + (23 - 22{,}8)^2 + (22 - 22{,}8)^2 + \cdots + (23 - 22{,}8)^2}{9 - 1}}$$

$$= 0{,}97$$

Тоді стандартна помилка для першої групи даних датчику температури за формулою 2.3:

$$\Delta_5 = \frac{0{,}97}{\sqrt{9}} = 0{,}32$$

Для другої групи даних обчислення наведено нижче. За формулою 2.1 потрібно обчислити арифметичне середнє для другої групи даних датчику температури:

$$\underline{X_6} = \frac{21 + 23 + 22 + 24 + 22 + 22 + 22 + 22 + 21}{9} = 22{,}1$$

Обчислення середнього квадратичного відхилення для другої групи даних з датчику повітря у вимірі диму за формулою 2.2:

$$\sigma_6 = \sqrt{\frac{(21-22{,}1)^2 + (23-22{,}1)^2 + (22-22{,}1)^2 + \cdots + (21-22{,}1)^2}{9-1}}$$

$$= 0{,}93$$

Тоді стандартна помилка для другої групи даних з датчику повітря у вимірі диму за формулою 2.3:

$$\Delta_6 = \frac{0{,}93}{\sqrt{9}} = 0{,}31$$

Далі за формулою 2.4 необхідно порахувати критерій достовірності датчику повітря за даними про дим:

$$t_{5-6} = \frac{22{,}8 - 22{,}1}{\sqrt{0{,}32^2 + 0{,}31^2}} = 1{,}56$$

За таблицею Ст'юдента (додаток Б, рис. Б.1) для заданого числа ступеню свободи ($f = 16$) та рівня значимості ($\alpha = 0{,}05$) табличне значення приблизно дорівнює $t_t = 2{,}12$. Тому можна стверджувати, що дані на 99% є достовірними, оскільки $t_{5-6} < t_t$.

У таблиці 2.6 наведені експериментальні дані замірів з датчику вологості повітря.

Таблиця 2.6

Експериментальні дані з датчику вологості

№	Вологість, %								
	1	2	3	4	5	6	7	8	9
1	35	42	47	50	55	49	45	53	40
2	37	44	49	52	48	43	45	50	44

Джерело: розробка автора

За формулою 2.1 потрібно обчислити арифметичне середнє для першої групи даних датчику вологості:

$$\underline{X_7} = \frac{35 + 42 + 47 + 50 + 55 + 49 + 45 + 53 + 40}{9} = 45{,}9$$

За формулою 2.2 потрібно обчислити значення середнього квадратичного відхилення для першої групи даних датчику вологості:

$$\sigma_7 = \sqrt{\frac{(35 - 45{,}9)^2 + (42 - 45{,}9)^2 + (47 - 45{,}9)^2 + \cdots + (40 - 45{,}9)^2}{9 - 1}}$$

$$= 7{,}32$$

Тоді стандартна помилка для першої групи даних датчику вологості за формулою 2.3:

$$\Delta_7 = \frac{7{,}32}{\sqrt{9}} = 2{,}44$$

Для другої групи даних обчислення наведено нижче. За формулою 2.1 потрібно обчислити арифметичне середнє для другої групи даних датчику температури:

$$\underline{X_8} = \frac{37 + 44 + 49 + 52 + 48 + 43 + 45 + 50 + 44}{9} = 45$$

Обчислення середнього квадратичного відхилення для другої групи даних з датчику повітря у вимірі диму за формулою 2.2:

$$\sigma_8 = \sqrt{\frac{(37-45)^2 + (44-45)^2 + (49-45)^2 + \cdots + (44-45)^2}{9-1}} = 6{,}86$$

Тоді стандартна помилка для другої групи даних з датчику повітря у вимірі диму за формулою 2.3:

$$\Delta_8 = \frac{6{,}86}{\sqrt{9}} = 2{,}29$$

Далі за формулою 2.4 необхідно порахувати критерій достовірності датчику повітря за даними про дим:

$$t_{7-8} = \frac{45{,}9 - 45}{\sqrt{7{,}32^2 + 6{,}86^2}} = 0{,}27$$

За таблицею Ст'юдента (додаток Б, рис. Б.1) для заданого числа ступеню свободи ($f = 16$) та рівня значимості ($\alpha = 0{,}05$) табличне значення приблизно дорівнює $t_t = 2{,}12$. Тому можна стверджувати, що дані на 99% є достовірними, оскільки $t_{7-8} < t_t$.

Отож, за проведеним дослідженням, у приміщенні з ідеальними умовами, вибірка даних є прийнятою. Усі дані відповідають установленим нормам та є дійсними [22].

2.4. Висновки до розділу 2

За розділом були виконані всі поставлені завдання. Проаналізовано збір відео- та аудіоінформації з використанням бездротових пристроїв. Спроєктована схема системи безпеки доказового спостереження. Досліджені існуючий інструментарій для доказового спостереження та обраний необхідний і прийнятий. Спроєктовані алгоритми різних модулів для їх порівняння. Досліджені датчики для приміщення, що будуть контролювати

безпеку та обрані необхідні: датчики повітря та температури і вологості. Спроєктований алгоритм синхронізації обраних датчиків, аналізу, стиснення та передачі даних до серверу або сховища. Проведено статистичний аналіз для визначення допустимих значень із датчиків повітря (концентрацій аміаку та диму) та температури (температури повітря та вологості) та сформована оцінка стану приміщення з ідеальними умовами.

РОЗДІЛ 3.
МЕТОДИ РОЗВИТКУ АЛГОРИТМІВ ЗБОРУ, ЛОГУВАННЯ, СТИСНЕННЯ ТА ПЕРЕДАЧІ ДАНИХ У СФЕРІ БЕЗПЕКИ

3.1. Алгоритми аналізу відеоданих, стиснення, логування та їх збереження

Алгоритми стиснення даних грають важливу роль в ефективному зберіганні та передачі інформації. Вони дозволяють зменшити об’єм даних без значних витрат їх вмісту, що особливо важливо при умовах обмежених ресурсів зберігання та пропускної здатності мережі. Існує множина різних алгоритмів стиснення даних, які можна розподілити на дві категорії стиснення з та без втрат. Перше допускає деяку втрату інформації для досягнення більш високого ступеня стиснення, друге – без витрат зберігає вихідні дані [1, 30, 41, 45].

Види обох категорій стиснення наведені в таблиці 3.1.

Таблиця 3.1.

Огляд категорій стиснення з та без втрат

Вид стиснення	Опис
1	2
Стиснення з втратами	
Дискретне перетворення Уолша	Застосовується для стиснення зображень та аудіо, розкладає дані на набір ортогональних функцій
Дискретне вейвлет-перетворення	Використовується для багатомасштабного аналізу сигналів в зображень, забезпечуючи гарне стиснення при зберіганні важливих характеристик
Стиснення з втратами	

Дискретне перетворення Чебишева	Застосовується для апроксимації функцій та стиснення даних шляхом розкладання на полігони Чебишева
Алгоритми K-means	Застосовується для стиснення зображень шляхом кластеризації пікселей та зменшенні кількості кольорів
Векторне квантування	Використовується для стиснення аудіо та зображень, розбиваючи дані на невеликі блоки та замінюючи їх найближчими векторами з попереднього створеного набору
Стиснення без втрат	
Кодування Гаффмана	Використовує частоту символів для створення змінно довгих кодів, щоб символи, що часто зустрічаються в коді мали більш короткі коди
Алгоритм Лемпеля-Зіва-Маркова (LZMA)	Стискає дані шляхом пошуку повторюваних послідовностей символів і заміни їх кодами
Кодування довжин серій	Замінює послідовність однакових символів на один символ та кількість повторень
Арифметичне кодування	Представляє дані у вигляді єдиного числа в діапазоні [0, 1), використовуючи вірогідність символів
Алгоритм Deflate	Комбінує алгоритми LZMA та кодування Гаффмана
Перетворення Барроуза-Уілера	Перетворює дані таким чином, щоб вони стали більш придатними для стиснення за рахунок групування однакових символів

Джерело: розробка автора

Дані алгоритми є основними компонентами сучасних систем аналізу обробки відеоданих. Їх вибір та реалізація залежать від конкретних задач, а саме алгоритм стиснення даних з або без втрати якості повинні бути

сумісними з обраними датчиками та мікроконтролером, які передають дані через бездротовий пристрій до серверу або сховища у системі безпеки.

Для стиснення даних, які передають обрані датчики, мікроконтролер і бездротовий пристрій до сховища або сервера у сфері безпеки, можна розглянути наступні [13, 26, 35, 55, 60]:

- кодування Гаффмана, яке підходить для стиснення даних, який розподілення значень даних нерівномірне та визначені значення зустрічаються частіше інших;

- алгоритм LZMA, який забезпечує високий ступінь, особливо для великих об’ємів даних, але він може бути ресурсоємним, що слідує враховувати при використанні на пристроях з обмеженими ресурсами, такими як мікроконтролер ESP32;

- дискретне косинусне перетворення використовується для стиснення зображень і може бути застосовано, якщо дані з датчиків можуть бути представлені у вигляді зображень;

- дискретне перетворення Уолша застосовується для стиснення сигналів і зображень, що можна використовувати для масштабного аналізу даних із датчиків;

- дискретне перетворення Чебишева використовується для апроксимації та стиснення даних шляхом розкладання на поліноми, що може використовуватися для часових рядів даних з датчиками;

- алгоритми K-means використовують кластеризацію даних для вимірювання та передачі тільки центроїдних кластерів, що значно знижує об’єм даних.

3.1.1. Алгоритми аналізу відеоданих та їх оптимізоване стиснення

Кодування Гаффмана використовується для кодування даних на основі

частоти символів. Він будує префіксне дерево, де символи з більш високою частотою мають більш короткі коди. Алгоритм кодування наступний [17]:

1) підрахунок частоти появи кожного символу $f(C)$ в наборі даних;

2) для кожного символу необхідно створити та помістити його в чергу з пріоритетом, де пріоритет визначається частотою символу;

3) поки в черзі більше одного вузла необхідно:

1) вилучити два вузли з найменшою частотою $f(A)$ і $f(B)$;

2) створити новий вузол з цими двома як дочірніми та частотою $f(C)$:

$$f(C) = f(A) + f(B); \tag{3.1}$$

3) помістити новий вузол до черги;

4) повторювати попередній крок, доки не залишиться тільки один вузол – корінь дерева Гаффмана.

Для оцінювання середньої довжини кодового слова, яка допоможе побачити ефективність стиснення даних необхідно використовувати формулу середньої довжини кодового слова [17]:

$$L_{avg} = \sum_{C \in D} \quad p(C) \cdot l(C), \tag{3.2}$$

де: $p(C)$ – вірогідність (частота) символу C;

$l(C)$ – довжина кодового слова для символу C.

Приклад дерева Гаффмана наведений на рисунку 3.1.

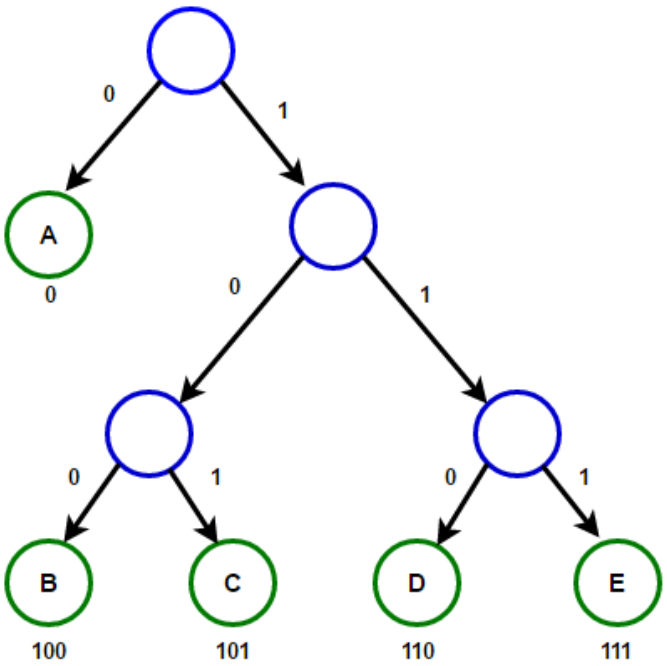

Рис. 3.1. Приклад збудованого дерева Гаффмана [17]

Кодування Гаффмана використовується для ефективного стиснення даних на основі частоти символів. Будуючи дерево Гаффмана, алгоритм створює оптимальні коди для символів, де символи з більш високою частотою мають більш короткі коди. Даний алгоритм широко застосовується в різного роду форматах стиснення даних, включаючи текстові та графічні файли.

Алгоритм Лемпеля-Зіва-Маркова заснований на пошуку повторюваної послідовності в даних та їх заміною на короткі коди. Алгоритм виконується за такими кроками [27]:

- вхідні дані розбиваються на блоки, для кожного з яких будується дерево пошуку (наприклад, дерево Лемпеля-Зіва, що наведений на рисунку 3.2);

- алгоритм шукає збіги з попередніми послідовностями, які вже зустрічалися в даних;

- збіги замінюються вказівками на їх попередні коди, зменшуючи об’єм даних.

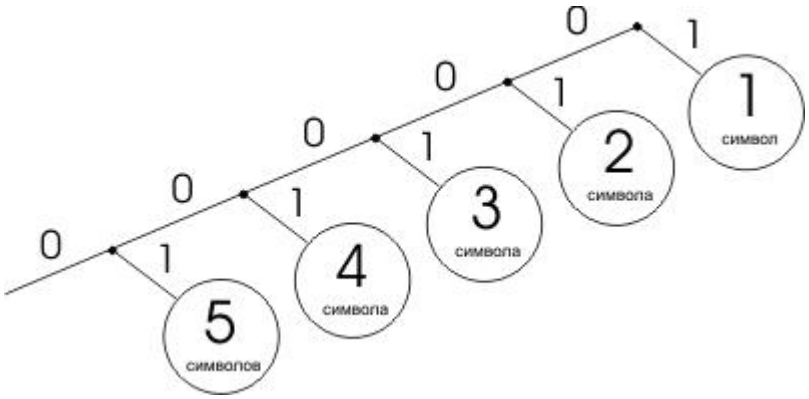

Рис. 3.2. Приклад дерева Лемпеля-Зіва [27]

Для представлення повторюваних послідовностей, які зменшують загальний об’єм даних необхідно використовувати наступну формулу [27]:

$$S' = (d_1, l_1), (d_2, l_2), \dots, (d_n, l_n), \tag{3.3}$$

де: S – вихідний рядок;

S' - стиснутий рядок;

d_n – відстань до попереднього входження;

l_n – довжина послідовності, що співпадає.

Алгоритм LZMA стискає дані, знаходячи повторювані послідовності та замінює їх кодами. Стиснутий рядок представляється як набір пар. Даний метод широко використовується в різних форматах архівів, забезпечуючи високе стиснення без втрат даних.

Дискретне косинусне перетворення використовує в алгоритмах стиснення зображень та відео. Воно перетворює дані в частотний простір, акцентуючи увагу на низькочастотні компоненти, які найбільш значимі для людського ока [32].

Алгоритм роботи наступний [32]:

- дані розбиваються на блоки (зазвичай 8x8 пікселів для зображень);
- для кожного блоку обчислюється дискретне косинусне представлення

на наступною формулою:

$$C(u,v) = \frac{1}{4}\alpha(u)\alpha(v)\sum_{x=0}^{N-1} \quad \sum_{y=0}^{N-1} \quad f(x,y)\ cos\ cos\ \left[\frac{(2x+1)u\pi}{2N}\right]$$

$$cos\ cos\ \left[\frac{(2y+1)v\pi}{2N}\right], \qquad (3.4)$$

де: $f(x,y)$ – значення пікселя в координатах (x,y);

$C(u,v)$ – коефіцієнти дискретного косинусного представлення;

$\alpha(u)$ або $\alpha(v)$ – нормовані коефіцієнти, які використовуються для забезпечення правильного перетворення коефіцієнтів за формулою [32]:

$$\alpha(u) \text{ або } \alpha(v) = \{\frac{1}{\sqrt{2}} \text{ якщо } u = 0, 1 \text{ інакше} \qquad (3.5)$$

Формула (3.5) дозволяє перетворити просторові дані в частотний простір, де можна ефективно застосувати стиснення.

Дискретне косинусне перетворення перетворює просторові дані в частотний простір, акцентуючи увагу на низькочастотних компонентах. Воно використовується в алгоритмах стиснення зображень та відео, дозволяючи більш ефективно видаляти високочастотні компоненти, які менш значимі для людського ока, тим самим зменшуючи об’єм даних.

Дискретне перетворення Уолша є аналогічним до алгоритму дискретного синусного перетворення, але використовує функції, які приймають значення ±1 (базисні функції Уолша Алгоритм дій наступний [34]:

- дані діляться на блоки;
- застосовується перетворення Уолша за наступною формулою:

$$W(u,v) = \frac{1}{N}\sum_{x=0}^{N-1} \quad \sum_{y=0}^{N-1} \quad f(x,y)R_{u,v}(x,y), \qquad (3.6)$$

де: $f(x,y)$ – значення пікселя в координатах (x,y);

$R_{u,v}(x,y)$ – функція, яка приймає значення ±1 та задається через суму бітів.

Приклад перетворення Уолша наведений на рисунку 3.3.

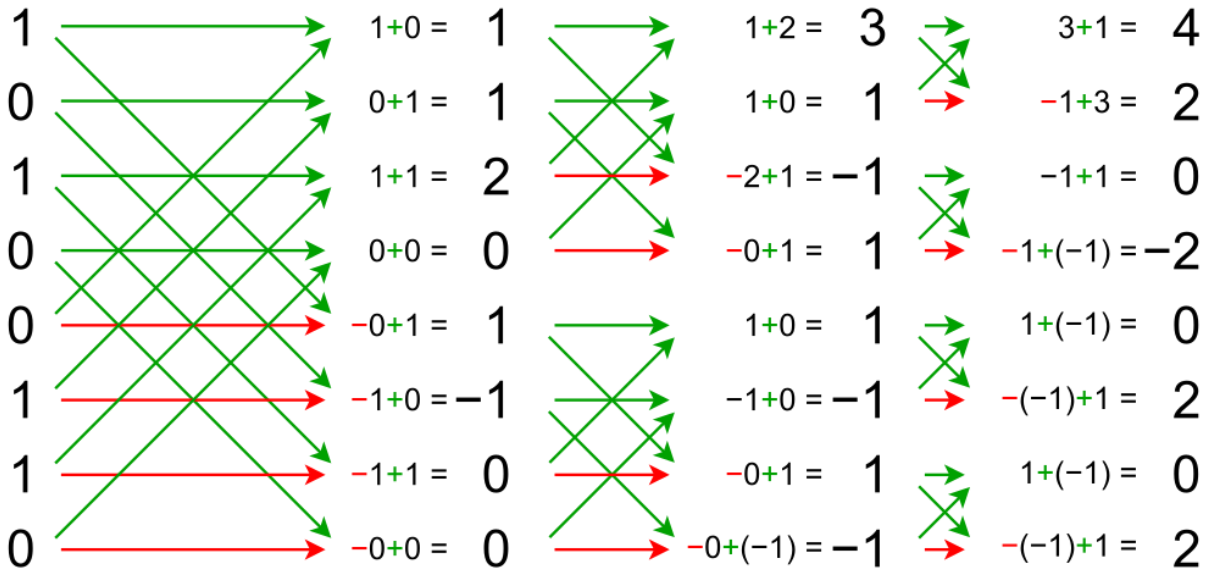

Рис. 3.3. Приклад роботи алгоритму перетворення Уолша [34]

Дана формула (3.6) використовується для перетворення даних з використанням функції, які прості в обчисленні та можуть бути корисні в застосунках, потребуючі швидкої обробки. Воно широко застосовується в системах цифрового зв'язку та в різних задачах обробки сигналів.

Дискретне перетворення Чебишева заснований на поліномах Чебишева, використовується для зближення функцій та стиснення даних. Алгоритм дій наступний [23]:

- дані поділяються на блоки;
- застосовується перетворення Чебишева за наступною формулою:

$$C_n = \sum_{k=0}^{N-1} f(k)T_n(x_k), \quad (3.7)$$

де: $f(k)$ – значення функцій у точці k;

$T_n(x_k)$ – поліном Чебишева першого ряду, який визначається рекурсією за наступною формулою [23]:

$$T_n(x) = \{1, \text{якщо } n = 0\ x, \text{якщо } n = 1\ 2xT_{n-1}(x) - T_{n-2}(x), \text{якщо } n \geq 2 \quad (3.8)$$

За формулою (3.7) можна обчислити перетворення даних з використанням поліномів Чебишева, що дозволяє ефективно апроксимувати функції та стискати дані. Даний вид алгоритму стиснення використовується в задачах численного аналізу, обробки сигналів та стиснення даних, де важлива точність апроксимації.

Алгоритми K-means використовуються для кластеризації даних, у тому числі в задачах стиснення даних, де дані групуються в кластери, що представлені центроїдами. Алгоритм діє наступним чином [18]:

1) визначається кількість кластерів K;

2) ініціалізується K центроїдів випадковим чином;

3) для кожного елементу даних необхідно:

1) визначити найближчий центроїд за формулою Евклідової відстані:

$$d(x, \mu_i) = \sqrt{\sum_{j=1}^{m} (x_j - \mu_{ij})^2} \quad (3.9)$$

де: x – елемент даних;

μ_i – центроїд і-го кластеру;

2) назначити елемент у відповідний кластер;

4) оновити центроїди, перерахувавши їх як середнє всіх елементів кластеру за наступною формулою:

$$\mu_i = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j, \quad (3.10)$$

де: C_i – набір елементів кластера і;

5) виконання повтору кроків три та чотири поки центроїди не стабілізуються, тобто їх зміни будуть мінімальними.

Даний алгоритм K-means використовується для кластеризації даних, групуючи їх у кластери та представляючи центроїдам. Метод повторює процес призначення елементів кластерам та оновлення центроїдам поки не досягне стабільності. Алгоритм широко використовується в аналізі даних, машинному навчанні та стисненні даних, забезпечуючи ефективне групове представлення більших об’ємів інформації.

Зазначені алгоритми описують стиснення як з та і без втрат інформації. Вони використовуються в різних застосунках від кодування даних до обробки зображень і відео.

3.1.2. Методи передачі та зберігання даних та оцінка можливих втрат

Для обраних датчиків, мікроконтролера та серверу або сховища даних необхідно застосовувати різні методи передачі та зберігання. Але їх можна об’єднати в одну схему. Методами передачі даних можуть виступати наступні [44]:

- Message Queuing Telemetry Transport (MQTT) є протоколом публікації-підписки, який ідеально підійде для пристроїв з обмеженими ресурсами; наприклад, мікроконтролер ESP32 WROOM зчитує дані з датчиків MQ-135 та AM2302 та надсилає їх до MQTT за Wi-Fi HaLow, а серсер або сховище підписує та отримує дані для подальшої обробки та зберігання;

- HTTP/HTTPS є стандартними протоколами для передачі даних; наприклад, мікроконтролер зчитує дані з датчиків та надсилає їх до сервера або сховища за допомогою запитів;

- WebSocket дозволяє встановлювати постійне з’єднання між клієнтами

та сервером, забезпечуючи низьку затримку в передачі даних; наприклад, ESP32 WROOM установлює WebSocket з'єднанню із сервером або сховищем та передає дані датчиків у реальному часі.

Для даного випадку буде підходити MQTT. Фрагмент коду наведений на рисунку 3.4 (додаток А.2).

```
# Налаштування MQTT
MQTT_BROKER = "mqtt.example.com"
MQTT_PORT = 1883
MQTT_TOPIC = "sensor/data"

def on_connect(client, userdata, flags, rc):
    print("Connected with result code " + str(rc))

client = mqtt.Client()
client.on_connect = on_connect
client.connect(MQTT_BROKER, MQTT_PORT, 60)

# Дані для відправлення
data_to_send = {
    "sensor_data": sensor_data
}

# Відправлення даних
client.publish(MQTT_TOPIC, json.dumps(data_to_send))
client.loop_forever()
```

Рис. 3.4. Фрагмент програмного коду передачі даних MQTT

Методами зберігання даних, переданих з датчиків повинен бути сервер або сховище. Необхідно обрати з наступних [44]:

- реляційні бази даних, такі як MySQL або PostgreSQL, у яких дані записуються в таблиці, що дозволяє легко виконувати запити та аналіз даних;

- нереляційні бази даних, такі як MongoDB або InfluxDB, які зберігають дані в колекціях або у вигляді часових рядків, що дозволяє масштабувати

систему;

- сховища на базі файлів, в які записуються дані з датчиків, їх можна легко прочитати та проаналізувати іншими системами.

Найкращим варіантом буде використовувати MySQL. Фрагмент коду зберігання даних наведено на рисунку 3.5 (додаток А.3).

```
# Функція для зберігання даних у MySQL
def save_data_to_mysql(sensor_type, group_name, values):
    try:
        connection = mysql.connector.connect(
            host=MYSQL_HOST,
            user=MYSQL_USER,
            password=MYSQL_PASSWORD,
            database=MYSQL_DATABASE
        )
        cursor = connection.cursor()
        create_table_if_not_exists(cursor)

        insert_query = """
        INSERT INTO sensor_data (sensor_type, group_name, values)
        VALUES (%s, %s, %s)
        """
        cursor.execute(insert_query, (sensor_type, group_name, json.dumps(values)))
        connection.commit()
        print(f"Data saved: {sensor_type}, {group_name}, {values}")

    except mysql.connector.Error as err:
        print(f"Error: {err}")
```

Рис. 3.5. Фрагмент коду зберігання даних у MySQL

Таким чином, дані будуть надійно передаватися та зберігатися. Їх завжди можна буде переглянути, проаналізувати та оцінити.

При передачі та зберігання даних може виникнути така ситуація, коли вони можуть зникнути. Для цього необхідно оцінити можливість втрати даних. Потрібно виконати симуляцію втрати даних при передачі даних від MQTT до MySQL та оцінити такий можливий варіант. Втрата даних при передачі через MQTT може відбуватися через нестабільне з’єднання або мережеві проблеми. Помилки запису в MySQL можуть відбуватися через помилки підключення до

бази даних або через проблеми із записом даних.

Для виконання розрахунку оцінки втрати даних необхідно використовувати наступні кроки [12]:

1) розрахувати вірогідність втрати даних при отриманні через MQTT за такою формулою:

$$P_{loss} = \frac{\text{Кількість втрачених повідомлень}}{\text{Загальну кількість відправлених повідомлень}}; \quad (3.11)$$

2) розрахувати вірогідність помилок запису в базу даних за наступною формулою:

$$P_{error} = \frac{\text{Кількість помилок запису}}{\text{Загальна кількість спроб запису}}. \quad (3.12)$$

Тобто, наприклад, якщо існую одна тисяча надісланих повідомлень, а втрачено 10, то вірогідність втрати буде 0,1. Якщо з однієї тисячі спроб записати дані до бази даних відбулося п’ять помилок, то вірогідність помилок становитиме 0,005. Подібна симуляція втрати даних була програмно реалізована та наведена в додатку А.4.

3.2. Алгоритми аналізу стиснення та передачі параметричної інформації

Для створення алгоритму моніторингу інтегральної оцінки стиснення та передачі даних параметричної інформації можна використовувати наступний підхід:

- збір даних з датчиків MQ-135 та AM2302;
- стиснення даних обраними алгоритмами, нехай це будуть алгоритми K-means та Лемпеля-Зіва-Маркова;
- передача даних через мережу Wi-Fi HaLow з використанням

протоколу MQTT;

- приймання та розпакування даних на зберігання в базі даних MySQL;
- розрахунок коефіцієнту стиснення, втрати даних, швидкості передачі даних та інших метрик.

Схема алгоритму наведена на рисунку 3.6.

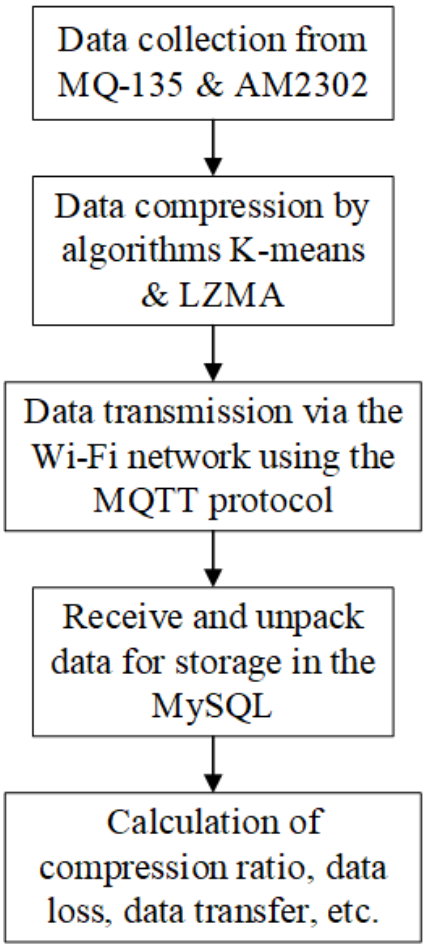

Рис. 3.6. Схема алгоритму моніторингу інтегральної оцінки стиснення та передачі даних

Програмна реалізація даного алгоритму наведена в додатку А.5.

3.2.1. Розробка алгоритмів для стиснення та передачі параметричних даних з максимальною ефективністю

Перед збереженням даних, переданих з датчиків, необхідно виконати їх стиснення. Є два види даних, які передають датчики: цифрові дані (наприклад, температура повітря в приміщенні) та зображення приміщення. Обидва види даних необхідно стиснути для економії місця в сховищі. Для порівняння стиснення даних можна використати алгоритми кластеризації та Лемпеля-Зіва-Маркова [18, 27].

Для алгоритму K-means необхідно визначитися із розміром зображення та кількістю кластерів, за якими файл буде стискатися. Схема алгоритму дій кластеризації наведений на рисунку 3.7.

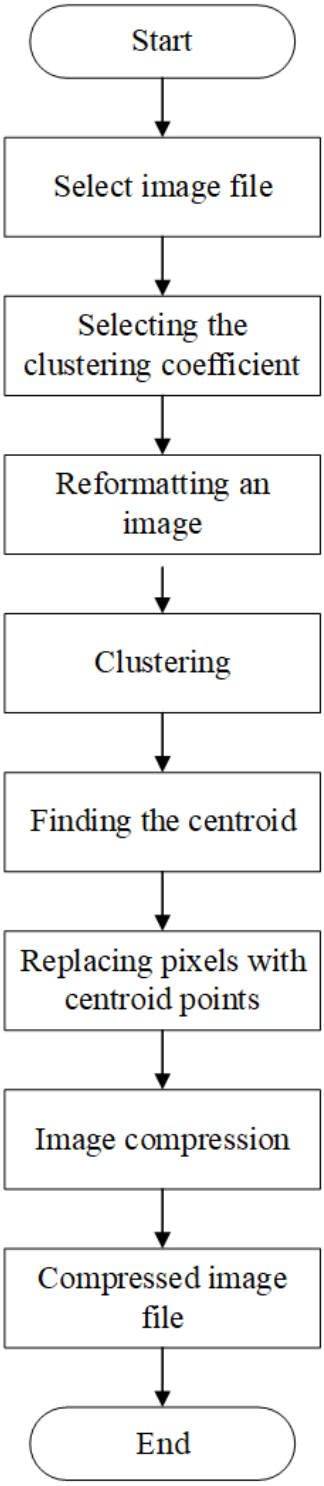

Рис. 3.7. Схема алгоритму K-means

За оригінальне зображення було обрано кафе, у якому можуть бути встановлені обрані датчики повітря та температури. Оригінальне зображення без стиснення наведено на рисунку 3.8.

Рис. 3.8. Оригінальне зображення приміщення

На зображенні (рис. 3.8) можна помітити різні кольори, за якими проведення кластеризації буде провести легше. Дане фото містить розмір 80x80 пікселів, а отже максимальний коефіцієнт стиснення також буде дорівнювати 80. Якщо поставити коефіцієнт більше, програмний алгоритм виведе помилку, що кількість обраних кластерів більша за кількість пікселів. Кількість кластерів визначається одразу та прописується в програмному коді. У даному випадку роботу алгоритму можна перевірити на дев’яти кластерах та перевірити якість і розмірність стиснення. Програма надає на виході два зображення: стисненні дані з датчику (наприклад, температурна карта) та стиснене зображення. Блок схема алгоритму роботи програмного коду наведений на рисунку 3.9, а програмний код у додатку А.6.

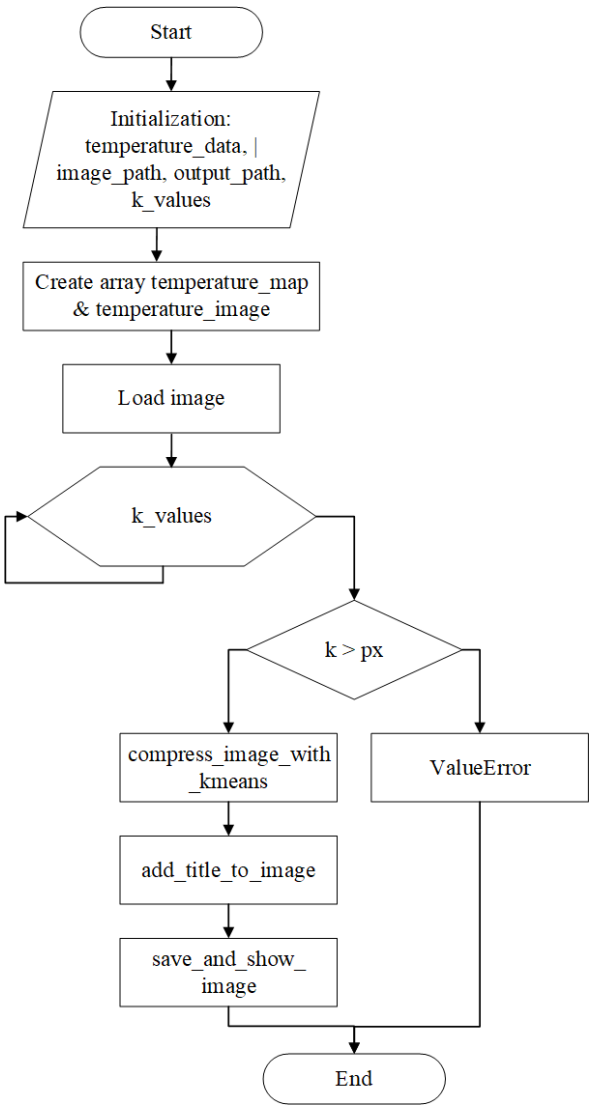

Рис. 3.9. Блок схема роботи алгоритму K-means

На виході маємо в сумі 18 стиснутих зображень по два на пару, стиснутими за дев'ятьма різними кластерами. За таблицею 3.2 можна побачити порівняння розмірності файлів до стиснення та після.

Таблиця 3.2

Результати стиснення даних і зображення за алгоритмом K-means

Коефіцієнт стиснення	Розмір файлу даних, КБ	Розмір файлу зображення, КБ
1	2	3
Оригінал	36,5	2,03

k=2	5,46	1,22
k=4	5,77	1,61
k=8	5,82	1,67
k=16	5,86	1,72
k=32	5,86	1,75

Продовження таблиці 3.2

1	2	3
k=48	5,82	1,77
k=64	5,82	1,78
k=72	5,83	1,79
k=80	5,83	1,79

Джерело: розробка автора

Результат стиснення наведений на рисунку 3.10.

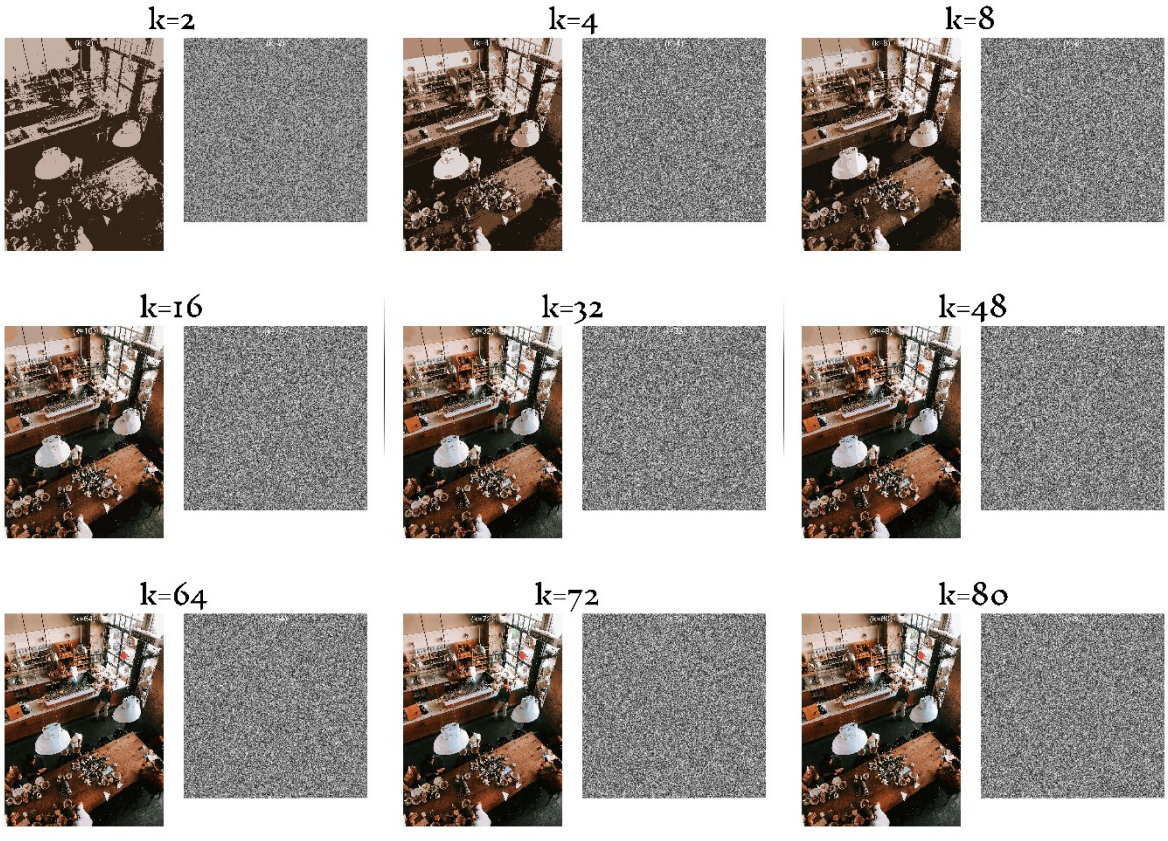

Рис. 3.10. Результат стиснення даних алгоритмом K-means

Як можна побачити (табл. 3.2, рис. 3.10), алгоритм стиснення є дієвим. За його допомогою можна регулювати якість стиснення, використовуючи коефіцієнти.

Другим алгоритмом стиснення є LZMA. Його можна використати декількома способами:

1) з використанням допоміжних алгоритмів, наприклад K-means;

2) без використання інших методів стиснення.

У даному випадку, буде розроблятися звичайний алгоритм LZMA. Він працює за схемою, наведеною на рисунку 3.11.

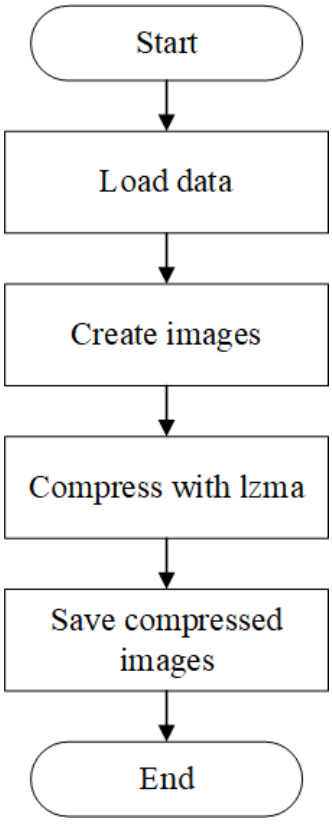

Рис. 3.11. Схема алгоритму LZMA

Зображення для стиснення було обрано таке, як і для попереднього алгоритму (рис. 3.8). Воно допоможе якісніше порівняти результати алгоритмів та оцінити їх ефективність. Даний алгоритм видає на результаті лише пару зображень: стиснені дані з датчику (наприклад, температурна

карта) та стиснене зображення. Програмна блок схема алгоритму наведена на рисунку 3.12, а програмна реалізація в додатку А.7.

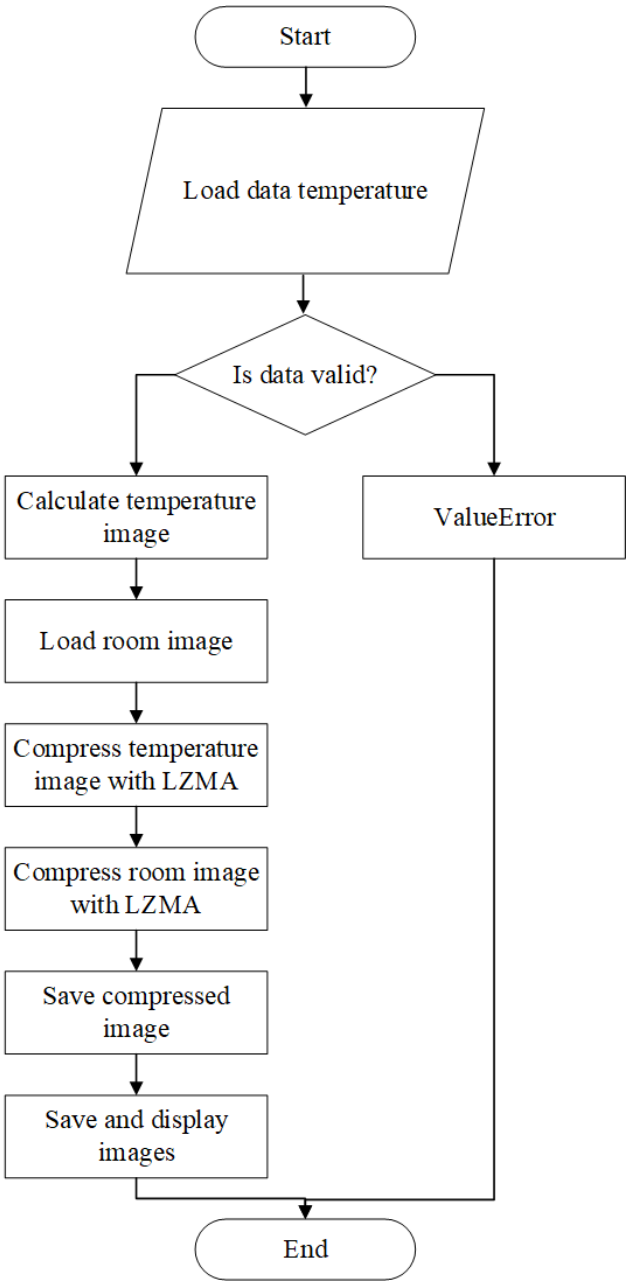

Рис. 3.12. Блок схема роботи алгоритму LZMA

У результаті спрацьовування алгоритму маємо два зображення: температурну карту та стиснуте зображення. Так як даний алгоритм відповідає за стиснення без втрат якості, проскановане зображення кімнати буде виглядити в гарній якості, але його розмір буде значно зменшений. У таблиці 3.3 наведені результати стиснення, а рисунок 3.13 відображає результат.

Таблиця 3.3

Результати стиснення даних і зображення за алгоритмом LZMA

Коефіцієнт стиснення	Розмір файлу даних, КБ	Розмір файлу зображення, КБ
Оригінал	36,5	2,03
Алгоритм LZMA	5,27	1,71

Джерело: розробка автора

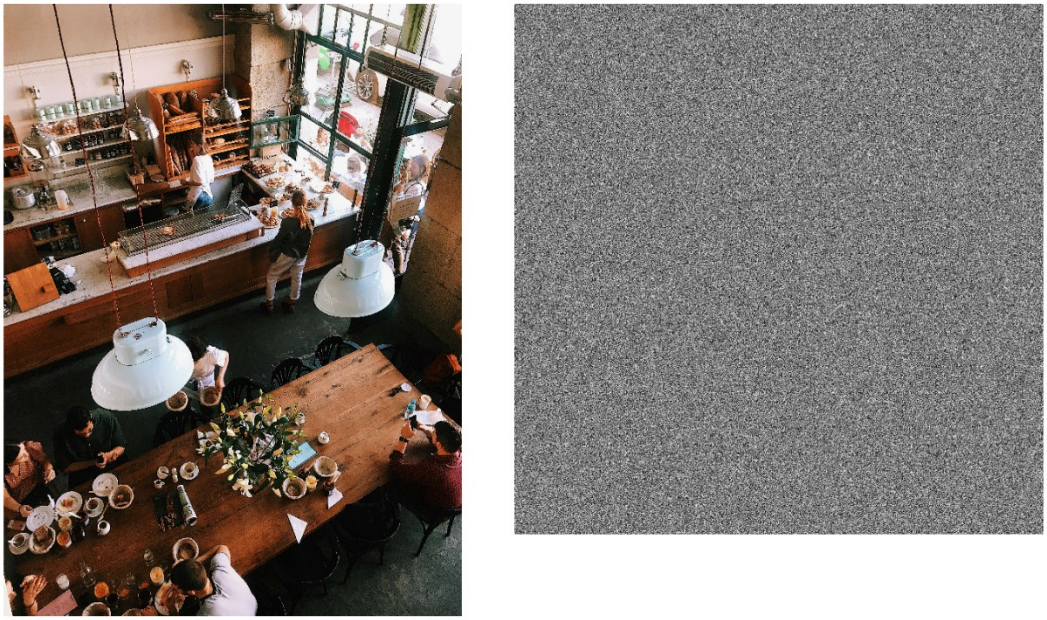

Рис. 3.13. Результат стиснення даних алгоритмом LZMA

Як можна побачити, результат дієвий у разі виконання обох алгоритмів. Обидва стискають дані так, як потрібно на велику кількість пам’яті, що дозволить зекономити місце у сховищі.

3.2.2. Оцінка коефіцієнтів стиснення та потенційних втрат даних

Коефіцієнт стиснення показує ефективність стиснутих даних у

порівнянні з його оригінальним розміром. Розраховується за наступною формулою:

$$Compression\ Ratio = \frac{Original\ size}{Compressed\ size}, \quad (3.13)$$

де: $Original\ size$ – розмір оригінального зображення у байтах;

$Compressed\ size$ – розмір стиснутого зображення у байтах.

Середньоквадратичну похибку, яка вимірює різницю між пікселями оригінальних і стиснутих даних, необхідно обчислювати на наступною формулою:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (I_i - \widehat{I_i})^2}, \quad (3.14)$$

де: N – загальна кількість пікселів у зображенні;

I_i – значення пікселя на позиції i в оригінальному зображенні;

$\widehat{I_i}$ – значення пікселя на позиції i у стисненому зображенні.

Результати обчислення обох значень для обох алгоритмів наведені в таблиці 3.4.

Таблиця 3.4

Результати оцінки коефіцієнтів стиснення та потенційних втрат даних

Коефіцієнт стиснення	Результат обчислення для температурної карти		Результат обчислення для зображення	
	Compression Ratio	RMSE	Compression Ratio	RMSE
1	2	3	4	5
Алгоритм K-means				

k=2	1,9	40,33	1,98	37,29
k=4	1,78	22,16	1,8	21,93
k=8	1,62	10,03	1,67	16,03
k=16	1,6	5,01	1,61	12,48
k=32	1,54	5,14	1,56	10,40
k=48	1,44	5,25	1,43	9,68
k=64	1,32	5,23	1,33	9,21
k=72	1,29	5	1,27	8,79
k=80	1,26	5,30	1,21	9,03
Алгоритм Лемпеля-Зіва-Маркова				
lzma	6,27	0,00	1,86	0,00

Джерело: розробка автора

Отож, за результатами (табл. 3.4) можна стверджувати, що алгоритми працюють вірно та стиснення відбувається. Особливо це видно з алгоритму LZMA, в якого за результатами розрахунків значення RMSE дорівнює 0, що означає вірність роботи алгоритму без втрати якості файлів, які були стиснені.

3.3. Проведення порівняльного аналізу розроблених алгоритмів

За проведеними розрахунками (табл. 3.4) можна сказати, що обидва алгоритми є дієвими в стисненні даних, але по-різному. Алгоритм стиснення K-means може забезпечувати гарне стиснення зображень із високим ступенем кореляції пікселів і має налаштованість кластерів. Але, даний алгоритм надає велику кількість втрати якості при дуже малому коефіцієнті (рис. 3.10) та може бути дуже чутливим при виборі початкових центроїдів. Алгоритм LZMA має високий коефіцієнт стиснення, особливо для даних із високим ступенем надмірності та забезпечує стиснення без втрат (рис. 3.13). Але, при більш низькому коефіцієнті стиснення для стиснення випадкових або шумових даних може втрачати якість.

Для порівняння часу виконання алгоритмів, були розраховано час

загальний та час компресії. Результати наведені в таблиці 3.5.

Таблиця 3.5

Час виконання алгоритмів K-means та LZMA

Алгоритм	Загальний час, с	Час компресії, с	Час декомпресії, с
K-means	133,51	71,02	0,13
LZMA	37,67	33,39	1,05

Джерело: розробка автора

Для кращого порівняння, у алгоритмі кластеризації був обраний максимальний кластер, з мінімальними втратами якості (табл. 3.2). Як можна побачити з результатів (табл. 3.5) час виконання алгоритму кластеризації повільніший за Лемпеля-Зіва-Маркова більш, ніж у три рази. Але час декомпресії у K-means менший за LZMA.

3.4. Висновки до розділу 3

За даним розділом був виконаний аналіз існуючих алгоритмів стиснення даних та обрані два алгоритми для стиснення існуючих даних. Були розроблені алгоритми K-means та LZMA для взаємодії з датчиками, мікроконтролером і бездротовим пристроєм. Був розроблений алгоритм передачі та зберігання даних через MQTT та MySQL. Були порівняні виконані алгоритми стиснення за оцінкою стиснення, їх втратами та часом виконання.

ВИСНОВКИ

Стиснення даних, переданих із пристроїв доказового спостереження є важливими аспектами через необхідність зберігання великої кількості переданих даних. Були розглянуті теоретичні основи доказового спостереження з використанням бездротових пристроїв. Було проаналізовано поточний стан, основні технології та досягнення в сфері безпеки. Досліджені розвиток і класифікація датчиків, а також існуючі рішення у сфері безпеки.

Були досліджені збір різного роду інформації з використанням бездротових пристроїв під час спостереження. Була спроєктована структура модуля синхронізації даних та розроблена схема порівняльного аналізу збору параметрів про об’єкт спостереження.

Були досліджені алгоритми аналізу даних, стиснення, логування та їх збереження. Були реалізовані алгоритми стиснення даних, переданих пристроями та х зберігання. Було проведено оцінювання алгоритмів, їх можливі втрати та час виконання. Визначено, що вибір алгоритму залежить від поставлених задач і бажаного рівня стиснення. Для критичних даних, для яких необхідне точне відновлення бажано використовувати алгоритм LZMA. Для менш критичних даних, для яких припустимі втрати можливе використання алгоритму K-means. Але, за часом виконання та компресії, а також якості, у даному дослідженні найкращим є LZMA.

Науковою новизною даного дослідження є розробка якісного алгоритму стиснення переданих даних з датчиків доказового спостереження через бездротові пристрої, який має невеликий відсоток втрат якості та швидке виконання за часом.

Практичною новизною є використання алгоритму стиснення з датчиками спостереження та передачі даних за бездротовим пристроєм, логування даних і їх зберігання.

Одержані результати можуть бути використаними в будь-якій сфері, де необхідне спостереження та передача даних через бездротові пристрої, їх

стиснення та зберігання.

Подальшим розвитком дослідження є програмна реалізація програмного забезпечення, у якому була б можливість налаштовувати необхідні датчики спостереження, керування передачею даних, їх стиснення за різними алгоритмами та зберігання даних.

ПЕРЕЛІК ПОСИЛАНЬ

1. A Generic Real Time Autoencoder-Based Lossy Image Compression / A. Tawfik et al. Communications, Signal Processing, and their Applications (ICCSPA) : 5th International Conference, Cairo. 2022. P. 1–6.

2. An Intelligent and Portable Air Pollution Monitoring System Based on Chemical Sensor Array / Y. Chen et al. Frontiers of Sensors Technologies (ICFST) : 4th International Conference, Shanghai. 2020. P. 21–25.

3. Application of improved on-line monitoring method using PMU data in recent European blackout / B. Li et al. Condition Monitoring and Diagnosis (CMD) : International Conference, Xi'an. 2016. P. 815–818.

4. Application of Raspberry Pi microcontroller for management and monitoring of IoT Systems / R. Minailenko et al. Central Ukrainian Scientific Bulletin Technical Sciences. 2023. Vol. 2, no. 7 (38). P. 12–18.

5. Application of student's t-test, analysis of variance, and covariance / P. Mishra et al. Annals of Cardiac Anaesthesia. 2019. Vol. 22, no. 4. P. 407.

6. A real time IoT based System Prediction and Monitoring of Landslides / R. Pol et al. International Journal of Food and Nutritional Sciences. 2024. Vol. 11, no. 7. P. 5204–5223.

7. A Survey of Motion Data Processing and Classification Techniques Based on Wearable Sensors / X. Xiaoqiong et al. IgMin Res. 2023. Vol. 1, no. 1. P. 105–115.

8. Attach importance of the bootstrap t-test against Student's t-test in clinical epidemiology: A demonstrative comparison using COVID-19 as an example / S. Zhao et al. Epidemiology and Infection. 2021. Vol. 149.

9. Automated Level Crossing System: A Computer Vision Based Approach with Raspberry Pi Microcontroller / R. U. Murshed et al. Electrical and Computer Engineering (ICECE) : 12th International Conference, Dhaka. 2022. P. 180–183.

10. Aydoğan M., Karci A. Spam Mail Detection using Naive Bayes method with Apache Spark. Artificial Intelligence and Data Processing (IDAP) :

International Conference, Malatya. 2018. P. 1–6.

11. Babalola T., Babalola A., Olokun M. Development of an ESP-32 Microcontroller Based Weather Reporting Device. Journal of Engineering Research and Reports. 2022. Vol. 22, no. 11. P. 27–38.

12. Bae T., Miljkovic T. Loss modeling with the size-biased lognormal mixture and the entropy regularized EM algorithm. Insurance: Mathematics and Economics. 2024. Vol. 117. P. 182–195.

13. Bedruz R. A., Quiros A. R. F. Comparison of Huffman Algorithm and Lempel-Ziv Algorithm for audio, image and text compression. Humanoid, Nanotechnology, Information Technology,Communication and Control, Environment and Management (HNICEM) : International Conference, Cebu. 2015. P. 1–6.

14. Bicamumakuba E., Habineza E., Chung S.-O. Sensor rechnologies for remote monitoring of automated orchard irrigation. Precision Agriculture Science and Technology. 2024. Vol. 6, no. 2. P. 81–95.

15. Cameron N. ESP32 Microcontroller. ESP32 Formats and Communication : Book. Berkeley, 2023. P. 1–54.

16. Capable of Gas Sensor MQ-135 to Monitor the Air Quality with Arduino uno / F. Neamah et al. International Journal of Engineering Research and Technology. 2020. Vol. 13, no. 10. P. 2955–2959.

17. Chen C.-C., Chang C.-C., Chen K. High-capacity reversible data hiding in encrypted image based on Huffman coding and differences of high nibbles of pixels. Journal of Visual Communication and Image Representation. 2021. Vol. 76.

18. Cluster Analysis / B. Balusamy et al. Big Data: Concepts, Technology, and Architecture. Wiley, 2021. P. 259–292.

19. Design and Implementation of IoT-Based Monitoring Battery and Solar Panel Temperature in Hydroponic System / R. Rahmatullah et al. Jurnal Ilmiah Teknik Elektro Komputer dan Informatika. 2023. Vol. 9, no. 3. P. 810–820.

20. Ding Y.-P., Stager G. Security tools: TSIEM and AppScan source for security. The Center for Advanced Studies on Collaborative Research : Proceedings

of the 2011 Conference, 7 November 2011.

21. Effective monitoring and classification of hydrogen and ammonia gases with a bilayer Pt/SnO2 thin film sensor / N. X. Thai et al. International Journal of Hydrogen Energy. 2019. Vol. 45, no. 3. P. 2418–2428.

22. Evangeline J., Dewi O. C., Sari J. S. Comfortable Room Condition for Working and Resting. Journal of Architectural Design and Urbanism. 2021. Vol. 3, no. 2. P. 59–71.

23. Germider O. V., Popov V. N. Mathematical Modeling of Elastically Deformed States of Thin Isotropic Plates Using Chebyshev Polynomials. Zhurnal Srednevolzhskogo Matematicheskogo Obshchestva. 2024. Vol. 26. P. 20–31.

24. Glascock A. P., Kutzik D. M. An Evidentiary Study of the Uses of Automated Behavioral Monitoring. Advanced Information Networking and Applications Workshops (AINAW'07) : 21st International Conference, Niagara Falls. 2007. P. 858–862.

25. Hamid M. A., Abdullah-Al-Wadud M., Alam M. M. A reliable structural health monitoring protocol using wireless sensor networks. Computer and Information Technology (ICCIT 2011) : 14th International Conference, Dhaka. 2011. P. 601–606.

26. Han Y., Hu Y., Huang X. Research and Application of K-means Clustering and DCT Double Compression Algorithm in Vibration Sensor. Information, Cybernetics, and Computational Social Systems (ICCSS) : 7th International Conference, Guangzhou. 2020. P. 180–183.

27. Hema M., Shyry P. fficient Compression of Multimedia Data using Lempel–Ziv–Markov Chain Adaptive Block Compressive Sensing (LZMC-ABCS). Wireless Personal Communications. 2024.

28. Ilias I. S. C., Ibrahim M. S. Performance analysis of audio video codecs over Wi-Fi/WiMAX network. Ubiquitous Information Management and Communication : Proceedings of the 8th International Conference. 2014. P. 1–5.

29. Imdad S. B. Security Institutes Threat to Human Security in Cyberspace? Choice in Threat Labelling (Mols Third Question of Ontological Politics). 2023.

30. Korycki R. Detection of tampering in lossy compressed digital audio recordings. New Trends In Audio & Video And Signal Processing: Algorithms, Architectures, Arrangements And Applications (NTAV/SPA) : Joint Conference, Lodz. 2012. P. 97–101.

31. Kumar J., Ramesh P. R. Low Cost Energy Efficient Smart Security System with Information Stamping for IoT Networks. Internet of Things: Smart Innovation and Usages (IoT-SIU) : 3rd International Conference, Bhimtal. 2018. P. 1–5.

32. Lama R. K., Kwon G.-R. New interpolation method based on combination of Discrete cosine transform and wavelet transform. Information Networking (ICOIN) : International Conference. 2015. P. 363–366.

33. Lapsley I. Hard and Soft NPM in City Management. The Resilience of New Public Management : Book. Oxford, 2024. P. 178–194.

34. Li B., Zhang S., Chen J. Generalized Walsh Transform Sequency Domain-Based Match Filtering for Electromagnetic Flowmeter Signal Measurement. IEEE Sensors Journal. 2024. Vol. 24, no. 7. P. 10203–10220.

35. Ma K., Li C., Li H. A new hybrid data compression algorithm for weather radar. Antennas, Propagation and EM Theory (ISAPE) : 11th International Symposium, Guilin. 2016. P. 886–889.

36. ManageEngine Application Manager. ManageEngine. Zoho Corp.

37. Monitoring Urban-Freight Transport Based on GPS Trajectories of Heavy-Goods Vehicles / S. Hadavi et al. IEEE Transactions on Intelligent Transportation Systems. 2019. Vol. 20, no. 10. P. 3747–3758.

38. Netwrix Corporation. Netwrix Event Log Manager. Netwrix Corporation.

39. Novak S. Y. Student's T -test: what's wrong?. ISNPS 2022 : Conference, London, 16 June 2022. 2022. P. 1–7.

40. Prawiro S. Y., Murti M. A. Device Discovery and Wireless Power Transfer-Solving Algorithm with Resonant Coupling. Wireless Personal Multimedia Communications (WPMC) : 21st International Symposium, Chiang Rai, 25–28 November 2018. P. 236–238.

41. P S. Optimization of Lossless Compression Algorithms using Multithreading. Journal of Information Technology and Sciences. 2023. Vol. 9, no. 1. P. 36–42.

42. Review on Blockchain for IoT Security and Data Integrity / M. Shaima et al. Security, Privacy and Trust Management : 12th International Conference. 2024. P. 115–126.

43. Santana T. V. On the Use of Device-to-Device in Wireless Networks. PhD : Thesis, 9 November 2018.

44. Selvaraj P., Doraikannan S. Privacy and Security Issues on Wireless Body Area and IoT for Remote Healthcare Monitoring. Intelligent Pervasive Computing Systems for Smarter Healthcare. Wiley, 2019. P. 227–253.

45. Semunigus W., Pattanaik B. Analysis for Lossless Data Compression Algorithms for Low Bandwidth Networks. Journal of Physics Conference Series. 2021. Vol. 1964, no. 4. P. 42–46.

46. Sensor-based intelligent tool online monitoring technology: Applications and progress / J. Huang et al. Measurement Science and Technology. 2024.

47. Sheth S. B., Sheth B. R. A variant of the student's t-test for data of varying reliability. Houston. 12 p. (Preprint. University of Houston ; TX 77204-4005).

48. Shevchuk B., Ivakhiv O., Geraimchuk M. Implementation of Effective Evidence-based Monitoring of the Object State by Means of Wireless Network Object Systems. Technology and Applications (IDAACS) : 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems, Cracow. 2021. P. 7–11.

49. Simultaneous Wireless Power and Data Transfer in Different Applications / P. B. Bobba et al. E3S : Web of Conferences, Hamza, 23 July 2024.

50. Singh S. Confined Space Safety: A Modern Approach for Remote Monitoring. 2024.

51. Sleep patterns and risk of chronic disease as measured by long-term monitoring with commercial wearable devices in the All of Us Research Program / N. S. Zheng et al. Nature Medicine. 2024. Vol. 2024.

52. Splunk Inc. Splunk Enterprise. Splunk Inc.

53. Stoltzfus J. Student's t-test for independent samples. International Journal of Academic Medicine. 2015. P. 27.

54. Su Z. Evidentiary value and evidentiary status of blockchain evidence. International Journal of Evidence and Proof. 2024.

55. Tan C., Zhang L., Wu H.-t. A Novel Blaschke Unwinding Adaptive-Fourier-Decomposition-Based Signal Compression Algorithm With Application on ECG Signals. IEEE Journal of Biomedical and Health Informatics. 2019. Vol. 23, no. 2. P. 672–682.

56. The Design and Implementation of GPS Controlled Environment Monitoring Robotic System based on IoT and ARM / H. Salman et al. Control and Robotics Engineering (ICCRE) : 4th International Conference, Nanjing. 2019. P. 93–98.

57. What Is the Fastest Way to Connect Stations to a Wi-Fi HaLow Network? / D. Bankov et al. Sensors. 2018. Vol. 18, no. 9.

58. Wi-Fi Network Analysis of University Campus / A. H. Faiz et al. International Journal of Computer Science and Network Security. 2020. Vol. 20, no. 10. P. 40–45.

59. Wireless and battery-free technologies for neuroengineering / S. M. Won et al. Nat. Biomed. Eng. 2023. Vol. 7. P. 405–423.

60. Yaguchi K., Kobayashi N., Shinohara A. Efficient Algorithm and Coding for Higher-Order Compression. Data Compression : Conference, Snowbird. 2014. P. 434–434.

ДОДАТКИ

Додаток А

Програмні алгоритми

A.1. Students_t-test.py

```
import numpy as np
# Дані з датчика повітря: концентрація аміаку
sensor1_1 = [30, 32, 29, 34, 31, 33, 28, 35, 30]
sensor1_2 = [22, 25, 27, 23, 24, 26, 28, 21, 25]
# Обчислення середнього арифметичного і стандартного відхилення концентрації аміаку для першої групи
s1 = np.sum(sensor1_1)
av1 = np.round(s1 / len(sensor1_1), 1)
print(f"X1 = {av1}")
arrs1 = []
for i in sensor1_1:
    f = np.square(i - av1)
    arrs1.append(f)
b1 = np.sum(arrs1)
c1 = np.sqrt(b1 / (len(sensor1_1) - 1))
c1_rounded = np.round(c1, 2)
print(f"Sigma1 = {c1_rounded}")
d1 = c1 / np.sqrt(len(sensor1_1))
d1_rounded = np.round(d1, 2)
print(f"Delta1 = {d1_rounded}")
# Обчислення середнього арифметичного і стандартного відхилення  концентрації аміаку для другої групи
s2 = np.sum(sensor1_2)
av2 = np.round(s2 / len(sensor1_2), 1)
print(f"X2 = {av2}")
arrs2 = []
for i in sensor1_2:
    f = np.square(i - av2)
    arrs2.append(f)
b2 = np.sum(arrs2)
c2 = np.sqrt(b2 / (len(sensor1_2) - 1))
c2_rounded = np.round(c2, 2)
print(f"Sigma2 = {c2_rounded}")
d2 = c2 / np.sqrt(len(sensor1_2))
d2_rounded = np.round(d2, 2)
print(f"Delta2 = {d2_rounded}")
# Дані з датчика повітря: концентрація диму
sensor2_1 = [1, 2, 1, 3, 1, 2, 2, 1, 0]
sensor2_2 = [0, 1, 1, 2, 0, 1, 1, 0, 2]
# Обчислення середнього арифметичного і стандартного відхилення концентрації диму для першої групи
s3 = np.sum(sensor2_1)
av3 = np.round(s3 / len(sensor2_1), 1)
print(f"X3 = {av3}")
arrs3 = []
for i in sensor2_1:
    f = np.square(i - av3)
    arrs3.append(f)
b3 = np.sum(arrs3)
c3 = np.sqrt(b3 / (len(sensor2_1) - 1))
c3_rounded = np.round(c3, 2)
print(f"Sigma3 = {c3_rounded}")
d3 = c3 / np.sqrt(len(sensor2_1))
d3_rounded = np.round(d3, 2)
```

```
print(f"Delta3 = {d3_rounded}")
# Обчислення середнього арифметичного і стандартного відхилення концентрації диму для другої групи
s4 = np.sum(sensor2_2)
av4 = np.round(s4 / len(sensor2_2), 1)
print(f"X4 = {av4}")
arrs4 = []
for i in sensor2_2:
    f = np.square(i - av4)
    arrs4.append(f)
b4 = np.sum(arrs4)
c4 = np.sqrt(b4 / (len(sensor2_2) - 1))
c4_rounded = np.round(c4, 2)
print(f"Sigma4 = {c4_rounded}")
d4 = c4 / np.sqrt(len(sensor2_2))
d4_rounded = np.round(d4, 2)
print(f"Delta4 = {d4_rounded}")
# Дані з датчика температури
sensor3_1 = [22, 23, 22, 23, 21, 23, 24, 24, 23]
sensor3_2 = [21, 23, 22, 24, 22, 22, 22, 22, 21]
# Обчислення середнього арифметичного і стандартного відхилення датчика температури для першої групи
s5 = np.sum(sensor3_1)
av5 = np.round(s5 / len(sensor3_1), 1)
print(f"X5 = {av5}")
arrs5 = []
for i in sensor3_1:
    f = np.square(i - av5)
    arrs5.append(f)
b5 = np.sum(arrs5)
c5 = np.sqrt(b5 / (len(sensor3_1) - 1))
c5_rounded = np.round(c5, 2)
print(f"Sigma5 = {c5_rounded}")
d5 = c5 / np.sqrt(len(sensor3_1))
d5_rounded = np.round(d5, 2)
print(f"Delta5 = {d5_rounded}")
# Обчислення середнього арифметичного і стандартного відхилення датчика темпераутри для другої групи
s6 = np.sum(sensor3_2)
av6 = np.round(s6 / len(sensor3_2), 1)
print(f"X6 = {av6}")
arrs6 = []
for i in sensor3_2:
    f = np.square(i - av6)
    arrs6.append(f)
b6 = np.sum(arrs6)
c6 = np.sqrt(b6 / (len(sensor3_2) - 1))
c6_rounded = np.round(c6, 2)
print(f"Sigma6 = {c6_rounded}")
d6 = c6 / np.sqrt(len(sensor3_2))
d6_rounded = np.round(d6, 2)
print(f"Delta6 = {d6_rounded}")
# Дані з датчика вологості
sensor4_1 = [37, 44, 49, 52, 57, 36, 48, 51, 39]
sensor4_2 = [35, 42, 47, 50, 55, 38, 45, 53, 40]
# Обчислення середнього арифметичного і стандартного відхилення датчика вологості для першої групи
s7 = np.sum(sensor4_1)
av7 = np.round(s7 / len(sensor4_1), 1)
print(f"X7 = {av7}")
arrs7 = []
for i in sensor4_1:
    f = np.square(i - av7)
```

```
    arrs7.append(f)
b7 = np.sum(arrs7)
c7 = np.sqrt(b7 / (len(sensor4_1) - 1))
c7_rounded = np.round(c7, 2)
print(f"Sigma7 = {c7_rounded}")
d7 = c7 / np.sqrt(len(sensor4_1))
d7_rounded = np.round(d7, 2)
print(f"Delta7 = {d7_rounded}")
# Обчислення середнього арифметичного і стандартного відхилення датчика вологості для другої групи
s8 = np.sum(sensor4_2)
av8 = np.round(s8 / len(sensor4_2), 1)
print(f"X8 = {av8}")
arrs8 = []
for i in sensor4_2:
    f = np.square(i - av8)
    arrs8.append(f)
b8 = np.sum(arrs8)
c8 = np.sqrt(b8 / (len(sensor4_2) - 1))
c8_rounded = np.round(c8, 2)
print(f"Sigma8 = {c8_rounded}")
d8 = c8 / np.sqrt(len(sensor4_2))
d8_rounded = np.round(d8, 2)
print(f"Delta8 = {d8_rounded}")
# Обчислення t-статистики для кожної пари масивів
t1 = (av1 - av2) / np.sqrt(np.square(d1) + np.square(d2))
print(f"t1-2: {np.round(t1, 2)}")
t2 = (av3 - av4) / np.sqrt(np.square(d3) + np.square(d4))
print(f"t3-4: {np.round(t2, 2)}")
t3 = (av5 - av6) / np.sqrt(np.square(d5) + np.square(d6))
print(f"t5-6: {np.round(t3, 2)}")
t4 = (av7 - av8) / np.sqrt(np.square(d7) + np.square(d8))
print(f"t7-8: {np.round(t4, 2)}")
```

A.2. mqtt.py

```
import numpy as np
import paho.mqtt.client as mqtt
import json
# Дані з датчиків
sensor_data = {
    "NH3": {
        "group1": [30, 32, 29, 34, 31, 33, 28, 35, 30],
        "group2": [22, 25, 27, 23, 24, 26, 28, 21, 25]
    },
    "smoke": {
        "group1": [1, 2, 1, 3, 1, 2, 2, 1, 0],
        "group2": [0, 1, 1, 2, 0, 1, 1, 0, 2]
    },
    "temperature": {
        "group1": [22, 23, 22, 23, 21, 23, 24, 24, 23],
        "group2": [21, 23, 22, 24, 22, 22, 22, 22, 21]
    },
    "humidity": {
        "group1": [37, 44, 49, 52, 57, 36, 48, 51, 39],
        "group2": [35, 42, 47, 50, 55, 38, 45, 53, 40]
    }
}
# Налаштування MQTT
MQTT_BROKER = "mqtt.example.com"
MQTT_PORT = 1883
MQTT_TOPIC = "sensor/data"
```

```
def on_connect(client, userdata, flags, rc):
    print("Connected with result code " + str(rc))
client = mqtt.Client()
client.on_connect = on_connect
client.connect(MQTT_BROKER, MQTT_PORT, 60)
# Дані для відправлення
data_to_send = {
    "sensor_data": sensor_data
}
# Відправлення даних
client.publish(MQTT_TOPIC, json.dumps(data_to_send))
client.loop_forever()
```

A.3. mysql.py

```
# Налаштування MySQL
MYSQL_HOST = "localhost"
MYSQL_USER = "root"
MYSQL_PASSWORD = "yourpassword"
MYSQL_DATABASE = "sensor_db"
# Функція підключення до MySQL та створення таблиці, якщо вона існує
def create_table_if_not_exists(cursor):
    cursor.execute("""
    CREATE TABLE IF NOT EXISTS sensor_data (
        id INT AUTO_INCREMENT PRIMARY KEY,
        sensor_type VARCHAR(50),
        group_name VARCHAR(50),
        values JSON,
        timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    )
    """)
# Функція для зберігання даних у MySQL
def save_data_to_mysql(sensor_type, group_name, values):
    try:
        connection = mysql.connector.connect(
            host=MYSQL_HOST,
            user=MYSQL_USER,
            password=MYSQL_PASSWORD,
            database=MYSQL_DATABASE
        )
        cursor = connection.cursor()
        create_table_if_not_exists(cursor)

        insert_query = """
        INSERT INTO sensor_data (sensor_type, group_name, values)
        VALUES (%s, %s, %s)
        """
        cursor.execute(insert_query, (sensor_type, group_name, json.dumps(values)))
        connection.commit()
        print(f"Data saved: {sensor_type}, {group_name}, {values}")
    except mysql.connector.Error as err:
        print(f"Error: {err}")
    finally:
        cursor.close()
        connection.close()
# Callback функції для MQTT
def on_connect(client, userdata, flags, rc):
    print("Connected with result code " + str(rc))
    client.subscribe(MQTT_TOPIC)
def on_message(client, userdata, msg):
    try:
        data = json.loads(msg.payload)
```

```
        sensor_data = data.get('sensor_data', {})

        for sensor_type, groups in sensor_data.items():
            for group_name, values in groups.items():
                save_data_to_mysql(sensor_type, group_name, values)

    except json.JSONDecodeError as e:
        print(f"JSON Decode Error: {e}")
```

A.4. data_loss.py

```
NUM_MESSAGES = 1000
NUM_ERRORS = 5
error_probability = NUM_ERRORS / NUM_MESSAGES
# Симуляція втрати даних
lost_messages = simulate_mqtt_transfer()
print(f"Simulated lost messages: {lost_messages} out of {NUM_MESSAGES}")
# Оцінка втрати даних при записі в базу даних
num_errors = 0
client.loop_start()
time.sleep(10)  # Дайте клиенту время для получения и обработки сообщений
client.loop_stop()
total_attempts = NUM_MESSAGES
p_loss = lost_messages / NUM_MESSAGES
p_error = num_errors / total_attempts
print(f"Probability of message loss (P_loss): {p_loss:.2f}")
print(f"Probability of database error (P_error): {p_error:.2f}")
```

A.5. compress&parametric_transmission.py

```
# Моніторинг
def monitoring_loop():
    while True:
        start_time = time.time()
        # Крок 1 - збір даних
        sensor_data = read_sensors()
        # Крок 2 - кластеризація даних
        clustered_data = cluster_data(sensor_data)
        # Крок 3 - стиснення даних
        compressed_data = compress_data(clustered_data)
        original_size = len(json.dumps(clustered_data.tolist()).encode('utf-8'))
        compressed_size = len(compressed_data)
        # Крок 4 - передача даних
        publish_data(mqtt_client, mqtt_topic, compressed_data)
        # Крок 5 - оцінка ефективності
        end_time = time.time()
        duration = end_time - start_time
        compression_ratio = original_size / compressed_size
        print(f"Original size: {original_size} bytes")
        print(f"Compressed size: {compressed_size} bytes")
        print(f"Compression ratio: {compression_ratio:.2f}")
        print(f"Transmission time: {duration:.2f} seconds")
        time.sleep(10)  # Затримка перед наступним циклом
monitoring_loop()
```

A.6. K-means.py

```
import numpy as np
from sklearn.cluster import KMeans
```

```
from PIL import Image, ImageDraw, ImageFont
import matplotlib.pyplot as plt
import paho.mqtt.client as mqtt
import mysql.connector
import random
import time
def create_temperature_image(temperature_map, min_temp, max_temp):
    """
    Створення зображення на основі карти температур.
    """
    height, width = temperature_map.shape
    normed_temp = (temperature_map - min_temp) / (max_temp - min_temp) * 255
    rgb_array = np.zeros((height, width, 3), dtype=np.uint8)
    rgb_array[:, :, 0] = normed_temp
    rgb_array[:, :, 1] = normed_temp
    rgb_array[:, :, 2] = normed_temp
    return rgb_array
def load_image(image_path):
    """
    Завантаження зображення з вказаного шляху.
    """
    img = Image.open(image_path)
    return np.array(img)
def compress_image_with_kmeans(image_array, k):
    """
    Стиснення зображення з використанням алгоритму K-means.
    """
    height, width, channels = image_array.shape
    pixels = image_array.reshape(-1, channels)
    if k > len(pixels):
        raise ValueError(f"n_clusters={k} більше, ніж кількість пікселів={len(pixels)}.")
    kmeans = KMeans(n_clusters=k, random_state=0).fit(pixels)
    compressed_pixels = kmeans.cluster_centers_[kmeans.labels_]
    compressed_image_array = compressed_pixels.reshape(height, width, channels)
    return compressed_image_array.astype(np.uint8)
def save_and_show_image(image_array, output_path):
    """
    Збереження та відображення зображення.
    """
    img = Image.fromarray(image_array)
    img.save(output_path)
    plt.imshow(img)
    plt.axis('off')
    plt.title("Стиснуте зображення")
    plt.show()
def add_title_to_image(image_array, title):
    """
    Додавання заголовка до зображення.
    """
    img = Image.fromarray(image_array)
    draw = ImageDraw.Draw(img)
    try:
        font = ImageFont.truetype("arial.ttf", size=150)
    except IOError:
        font = ImageFont.load_default()
    text_bbox = draw.textbbox((0, 0), title, font=font)
    text_width = text_bbox[2] - text_bbox[0]
    draw.text(((img.width - text_width) // 2, 10), title, fill="red", font=font)
    return np.array(img)
def simulate_am2302_data():
    """
    Дані температури та вологості, отриманих від сенсора AM2302.
    """
```

```
            temperature_data = [random.uniform(21.0, 25.0) for _ in range(80)]
            humidity_data = [random.uniform(35.0, 60.0) for _ in range(80)]
            return temperature_data, humidity_data
        def on_message(client, userdata, message):
            """
            Обробка отриманих повідомлень від MQTT брокера.
            """
            data = message.payload.decode('utf-8')
            temperature_data, humidity_data, ammonia_data, smoke_data = map(eval, data.split('|'))
            store_data_in_mysql(temperature_data, humidity_data, ammonia_data, smoke_data)
        def store_data_in_mysql(temperature_data, humidity_data, ammonia_data, smoke_data):
            """
            Збереження даних у MySQL базу даних.
            """
            conn = mysql.connector.connect(
                host="localhost",
                user="your_username",
                password="your_password",
                database="your_database"
            )
            cursor = conn.cursor()
            cursor.execute("CREATE TABLE IF NOT EXISTS sensor_data (timestamp TIMESTAMP DEFAULT
CURRENT_TIMESTAMP, temperature FLOAT, humidity FLOAT, ammonia FLOAT, smoke FLOAT)")
            for temp, hum, ammonia, smoke in zip(temperature_data, humidity_data, ammonia_data, smoke_data):
                cursor.execute("INSERT INTO sensor_data (temperature, humidity, ammonia, smoke) VALUES (%s,
%s, %s, %s)", (temp, hum, ammonia, smoke))
            conn.commit()
            conn.close()
        def publish_data_to_mqtt():
            """
            Публікація даних з сенсорів на MQTT брокер.
            """
            broker = "mqtt.eclipse.org"
            port = 1883
            topic = "sensor/data"
            client = mqtt.Client()
            client.connect(broker, port)
            temperature_data, humidity_data = simulate_am2302_data()
            ammonia_data = [30, 32, 29, 34, 31, 33, 28, 35, 30, 22, 25, 27, 23, 24, 26, 28, 21, 25]
            smoke_data = [1, 2, 1, 3, 1, 2, 2, 1, 0, 0, 1, 1, 2, 0, 1, 1, 0, 2]
            data_str = '|'.join([
                str(temperature_data),
                str(humidity_data),
                str(ammonia_data),
                str(smoke_data)
            ])
            client.publish(topic, data_str)
            client.disconnect()
        def start_mqtt_client():
            """
            Запуск MQTT клієнта для підписки та обробки повідомлень.
            """
            broker = "mqtt.eclipse.org"
            port = 1883
            client = mqtt.Client()
            client.on_message = on_message
            client.connect(broker, port)
            client.subscribe("sensor/data")
            client.loop_start()
        def main(temperature_data, humidity_data, image_path, output_path='compressed_image.jpg', k_values=[2,
4, 8,16, 32, 48, 64, 72, 80]):
            """
            Основна функція для обробки температурних даних та зображень.
```

```
    """
    width, height = 20, 4
    temperature_map = np.array(temperature_data).reshape((height, width))
    temperature_image_array = create_temperature_image(temperature_map, min_temp=21, max_temp=25)
    room_image_array = load_image(image_path)
    for k in k_values:
        try:
            compressed_temp_image_array = compress_image_with_kmeans(temperature_image_array, k)
            compressed_temp_image_array = add_title_to_image(compressed_temp_image_array, f"(k={k})")
            compressed_room_image_array = compress_image_with_kmeans(room_image_array, k)
            compressed_room_image_array = add_title_to_image(compressed_room_image_array, f"(k={k})")
            output_temp_path = output_path.replace('.jpg', f'_temp_k{k}.jpg')
            save_and_show_image(compressed_temp_image_array, output_temp_path)
            output_room_path = output_path.replace('.jpg', f'_room_k{k}.jpg')
            save_and_show_image(compressed_room_image_array, output_room_path)
        except ValueError as e:
            print(e)
# Запуск публікації даних та MQTT клієнта
publish_data_to_mqtt()
start_mqtt_client()
# Виконання основної функції
temperature_data = [22, 23, 22, 23, 21, 23, 24, 24, 23, 21, 23, 22, 24, 22, 22, 22, 22, 21]
humidity_data = [37, 44, 49, 52, 57, 36, 48, 51, 39, 35, 42, 47, 50, 55, 38, 45, 53, 40]
image_path = r'C:\Users\viacheslav\project\images.jpg'
main(temperature_data, humidity_data, image_path)
```

A.7. LZMA.py

```
import numpy as np
from sklearn.cluster import KMeans
from PIL import Image, ImageDraw, ImageFont
import matplotlib.pyplot as plt
import paho.mqtt.client as mqtt
import mysql.connector
import random
import time
import lzma
import io
def create_temperature_image(temperature_map, min_temp, max_temp):
    """
    Створення зображення на основі карти температур.
    """
    height, width = temperature_map.shape
    normed_temp = (temperature_map - min_temp) / (max_temp - min_temp) * 255
    rgb_array = np.zeros((height, width, 3), dtype=np.uint8)
    rgb_array[:, :, 0] = normed_temp
    rgb_array[:, :, 1] = normed_temp
    rgb_array[:, :, 2] = normed_temp
    return rgb_array
def load_image(image_path):
    """
    Завантаження зображення з вказаного шляху.
    """
    img = Image.open(image_path)
    return np.array(img)
def compress_image_with_lzma(image_array, preset=lzma.PRESET_DEFAULT):
    """
    Стиснення зображення за допомогою алгоритму LZMA.
    """
    buffer = io.BytesIO()
    Image.fromarray(image_array).save(buffer, format='PNG')
    compressed_data = lzma.compress(buffer.getvalue(), preset=preset)
```

```
        return compressed_data
    def decompress_image_with_lzma(compressed_data):
        """
        Розпакування зображення, стиснутого за допомогою LZMA.
        """
        decompressed_data = lzma.decompress(compressed_data)
        buffer = io.BytesIO(decompressed_data)
        img = Image.open(buffer)
        return np.array(img)
    def save_and_show_image(image_array, output_path):
        """
        Збереження та відображення зображення.
        """
        img = Image.fromarray(image_array)
        img.save(output_path)
        plt.imshow(img)
        plt.axis('off')
        plt.title("Стиснуте зображення")
        plt.show()
    def add_title_to_image(image_array, title):
        """
        Додавання заголовка до зображення.
        """
        img = Image.fromarray(image_array)
        draw = ImageDraw.Draw(img)
        try:
            font = ImageFont.truetype("arial.ttf", size=150)
        except IOError:
            font = ImageFont.load_default()
        text_bbox = draw.textbbox((0, 0), title, font=font)
        text_width = text_bbox[2] - text_bbox[0]
        draw.text(((img.width - text_width) // 2, 10), title, fill="red", font=font)
        return np.array(img)
    def simulate_am2302_data():
        """
        Дані температури та вологості з сенсора AM2302.
        """
        temperature_data = [random.uniform(21.0, 25.0) for _ in range(80)]
        humidity_data = [random.uniform(35.0, 60.0) for _ in range(80)]
        return temperature_data, humidity_data
    def on_message(client, userdata, message):
        """
        Обробка отриманих повідомлень від MQTT брокера.
        """
        data = message.payload.decode('utf-8')
        temperature_data, humidity_data, ammonia_data, smoke_data = map(eval, data.split('|'))
        store_data_in_mysql(temperature_data, humidity_data, ammonia_data, smoke_data)
    def store_data_in_mysql(temperature_data, humidity_data, ammonia_data, smoke_data):
        """
        Збереження даних у базі даних MySQL.
        """
        conn = mysql.connector.connect(
            host="localhost",
            user="your_username",
            password="your_password",
            database="your_database"
        )
        cursor = conn.cursor()
        cursor.execute("CREATE TABLE IF NOT EXISTS sensor_data (timestamp TIMESTAMP DEFAULT
CURRENT_TIMESTAMP, temperature FLOAT, humidity FLOAT, ammonia FLOAT, smoke FLOAT)")
        for temp, hum, ammonia, smoke in zip(temperature_data, humidity_data, ammonia_data, smoke_data):
            cursor.execute("INSERT INTO sensor_data (temperature, humidity, ammonia, smoke) VALUES (%s,
%s, %s, %s)", (temp, hum, ammonia, smoke))
```

```
            conn.commit()
            conn.close()
        def publish_data_to_mqtt():
            """
            Публікація даних з сенсорів на MQTT брокер.
            """
            broker = "mqtt.eclipse.org"
            port = 1883
            topic = "sensor/data"
            client = mqtt.Client()
            client.connect(broker, port)
            temperature_data, humidity_data = simulate_am2302_data()
            ammonia_data = [30, 32, 29, 34, 31, 33, 28, 35, 30, 22, 25, 27, 23, 24, 26, 28, 21, 25]
            smoke_data = [1, 2, 1, 3, 1, 2, 2, 1, 0, 0, 1, 1, 2, 0, 1, 1, 0, 2]
            data_str = '|'.join([
                str(temperature_data),
                str(humidity_data),
                str(ammonia_data),
                str(smoke_data)
            ])
            client.publish(topic, data_str)
            client.disconnect()
        def start_mqtt_client():
            """
            Запуск MQTT клієнта для підписки та обробки повідомлень.
            """
            broker = "mqtt.eclipse.org"
            port = 1883
            client = mqtt.Client()
            client.on_message = on_message
            client.connect(broker, port)
            client.subscribe("sensor/data")
            client.loop_start()
        def main(temperature_data, humidity_data, image_path, output_path='compressed_image.xz', k_values=[2,
4, 8, 12, 16, 24, 32, 48, 64, 72, 80]):
            """
            Основна функція для обробки даних температури та зображень.
            """
            width, height = 20, 4
            temperature_map = np.array(temperature_data).reshape((height, width))
            temperature_image_array = create_temperature_image(temperature_map, min_temp=21, max_temp=25)
            room_image_array = load_image(image_path)
            for k in k_values:
                try:
                    compressed_temp_image_array = compress_image_with_lzma(temperature_image_array)
                    compressed_room_image_array = compress_image_with_lzma(room_image_array)
                    # Збереження та відображення зображень для порівняння
                    output_temp_path = output_path.replace('.xz', f'_temp_k{k}.xz')
                    with open(output_temp_path, 'wb') as f:
                        f.write(compressed_temp_image_array)
                    output_room_path = output_path.replace('.xz', f'_room_k{k}.xz')
                    with open(output_room_path, 'wb') as f:
                        f.write(compressed_room_image_array)

                    # Розпаковування та відображення
                    decompressed_temp_image_array                                                    =
decompress_image_with_lzma(compressed_temp_image_array)
                    decompressed_room_image_array                                                    =
decompress_image_with_lzma(compressed_room_image_array)
                    decompressed_temp_image_path = output_path.replace('.xz', f'_temp_decompressed_k{k}.jpg')
                    decompressed_room_image_path = output_path.replace('.xz', f'_room_decompressed_k{k}.jpg')
                    save_and_show_image(decompressed_temp_image_array, decompressed_temp_image_path)
                    save_and_show_image(decompressed_room_image_array, decompressed_room_image_path)
```

```
        except ValueError as e:
            print(e)
# Запуск публікації даних та MQTT клієнта
publish_data_to_mqtt()
start_mqtt_client()
# Виконання основної функції
temperature_data = [22, 23, 22, 23, 21, 23, 24, 24, 23, 21, 23, 22, 24, 22, 22, 22, 22, 21]
humidity_data = [37, 44, 49, 52, 57, 36, 48, 51, 39, 35, 42, 47, 50, 55, 38, 45, 53, 40]
image_path = r'C:\Users\viacheslav\project\images.jpg'
main(temperature_data, humidity_data, image_path)
```

Додаток Б

Допоміжні цифрові дані

Критерій Ст'юдента *t*

Число ступенів свободи f	Рівень значимості α			
	0,10	0,05	0,01	0,001
1	6,31	12,70	63,70	637,00
2	2,92	4,30	9,92	31,60
3	2,35	3,18	5,84	12,90
4	2,13	2,78	4,60	8,61
5	2,01	2,57	4,03	6,86
6	1,94	2,45	3,71	5,96
7	1,89	2,36	3,50	5,40
8	1,86	2,31	3,36	5,04
9	1,83	2,26	3,25	4,78
10	1,81	2,23	3,17	4,59
11	1,80	2,20	3,11	4,44
12	1,78	2,18	3,05	4,32
13	1,77	2,16	3,01	4,22
14	1,76	2,14	2,98	4,14
15	1,75	2,13	2,95	4,07
16	1,75	2,12	2,92	4,01
17	1,74	2,11	2,90	3,96
18	1,73	2,10	2,88	3,92
19	1,73	2,09	2,86	3,88
20	1,73	2,09	2,85	3,85
21	1,72	2,08	2,83	3,82
22	1,72	2,07	2,82	3,79
23	1,71	2,07	2,81	3,77
24	1,71	2,06	2,80	3,74
25	1,71	2,06	2,79	3,72
26	1,71	2,06	2,78	3,71
27	1,71	2,05	2,77	3,69
28	1,70	2,05	2,76	3,66
29	1,70	2,05	2,76	3,66
30	1,70	2,04	2,75	3,65
40	1,68	2,02	2,70	3,55
60	1,67	2,00	2,66	3,46
120	1,66	1,98	2,62	3,37
∞	1,64	1,96	2,58	3,29

Рис. Б.1 – Таблиця t-критерію Ст'юдента