МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет

імені Петра Могили

Факультет комп’ютерних наук

Кафедра комп'ютерної інженерії

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри,

д-р техн. наук, проф.

_____ І. М. Журавська

« __ » _____ 2024 р.

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

Система виявлення об’єктів на основі CSI мережі Wi-FI

Спеціальність 123 Комп’ютерна інженерія

123 – КБР.01 – 405.22010612

Студент _____ Д. О. Фрич

підпис

«__» _____ 2024 р.

Керівник д-р техн. наук, проф _____ І. М. Журавська

підпис

«__» _____ 2024 р.

Миколаїв – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет імені Петра Могили
Факультет комп'ютерних наук
Кафедра комп'ютерної інженерії

ЗАТВЕРДЖУЮ

Зав. кафедри _____ І. М. Журавська

«_____» _____ 2024 р.

ЗАВДАННЯ
на виконання кваліфікаційної бакалаврської роботи

Видано студенту групи 405 факультету комп'ютерних наук

Фричу Дану Олеговичу
(прізвище, ім'я, по батькові студента)

1. Тема кваліфікаційної роботи

Система виявлення об'єктів на основі CSI мережі Wi-Fi

Затверджена наказом по ЧНУ ім. Петра Могили від 30.01.2024 № 17.

2. Строк представлення кваліфікаційної роботи «____» _____ 2024 р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні
Очікуваним результатом роботи є апаратне та програмне забезпечення системи виявлення об'єктів на основі CSI мережі Wi-Fi

4. Перелік питань, що підлягають розробці
1) проаналізувати методи та засоби отримання, обробки та аналізу інформації про стан каналу (CSI) в мережах Wi-Fi за їх характеристиками;
2) дослідити існуючі підходи щодо використання CSI для виявлення присутності об'єктів у середовищі, їх класифікації та локалізації;
3) розробити апаратну та програмну платформу для реалізації системи аналізу CSI;
4) дослідити зміну CSI-даних в залежності від матеріалу об'єкту, що створює перешкоду;
5) провести тестування розробленого АПЗ;
6) навести приклади можливих покращень розробленого рішення.

5. Перелік графічних матеріалів
Концептуальна схема АПЗ
Алгоритм функції збору еталонних даних
Алгоритм функції порівняння вхідних даних з еталонними
Схема приміщення для експериментів
Графіки CSI-даних для 8 типів об’єктів
6. Завдання до спеціальної частини
Проведення інтегральної оцінки умов праці у виробничому приміщенні відповідно до чинних нормативних актів з охорони праці.

7. Консультанти:

Консультант	Кафедра (організація)	Частина роботи
Алексєєва А. О. канд. техн. наук, доцент	кафедра екології Медичного інституту ЧНУ імені Петра Могили	Спеціальна частина з охорони праці

Керівник роботи
д-р техн. наук, професор, Журавська Ірина Миколаївна
(посада, прізвище, ім’я, по батькові)

_____
(підпис)

Завдання прийнято до виконання

Фрич Дан Олегович
(прізвище, ім’я, по батькові студента)

_____
(підпис)

Дата видачі завдання «____» _____ 2024 р.

КАЛЕНДАРНИЙ ПЛАН

виконання кваліфікаційної роботи

Тема: Система виявлення об’єктів на основі CSI мережі Wi-Fi

№	Найменування роботи	Початок	Закінчення	Примітки
1	Розробка та затвердження завдання на виконання КР	05.02.2024	08.02.2024	Виконано
2	Огляд літератури за темою роботи	09.02.2024	12.03.2024	Виконано
3	Складання календарного плану КБР	26.02.2024	06.03.2024	Виконано
4	Аналіз предметної області	09.03.2024	15.03.2024	Виконано
5	Розробка проєктних рішень	15.03.2024	25.03.2024	Виконано
6	Вибір апаратної платформи	16.03.2024	27.03.2024	Виконано
7	Налаштування АПЗ	01.04.2024	07.04.2024	Виконано
8	Проведення досліджень	07.04.2024	10.05.2024	Виконано
9	Оформлення КРБ та презентації	18.04.2024	29.05.2024	Виконано
10	Перший передзахист	29.05.2024	29.05.2024	Виконано
11	Другий передзахист	06.06.2024	06.06.2024	Виконано
12	Рецензування	29.06.2024	05.06.2024	Виконано
13	Відгук керівника КР	10.06.2024	14.06.2024	Виконано
14	Завершення оформлення КР та презентації	06.06.2024	14.06.2024	Виконано
15	Захист кваліфікаційної роботи			

Розробив здобувач ВО Фрич Дан Олегович _____

(прізвище, ім’я, по батькові) *(підпис)*

«____» _____ 2024 р.

Керівник роботи проф. Журавська Ірина Миколаївна _____

(посада, прізвище, ім’я, по батькові) *(підпис)*

«____» _____ 2024 р.

АНОТАЦІЯ

до кваліфікаційної бакалаврської роботи
«Система виявлення об’єктів на основі CSI мережі Wi-Fi»
Студент гр. 405: Фрич Дан Олегович
Керівник: д-р техн. наук, проф. Журавська Ірина Миколаївна

Бакалаврська робота присвячена розвитку методів виявлення об’єктів на основі інформації про стан каналу (англ. Channel State Information, CSI) мережі Wi-Fi, а також створенню відповідного апаратно-програмного забезпечення (АПЗ). Актуальність цієї роботи обумовлена потенціалом використання систем, що ґрунтуються на аналізі CSI-даних, для неінвазивного виявлення небезпечних предметів в оточуючому середовищі.

Об'єктом дослідження є процес виявлення об'єктів на основі аналізу сигналів мережі Wi-Fi. Предметом дослідження є методи та алгоритми обробки CSI сигналів Wi-Fi для виявлення та класифікації об'єктів.

Робота пройшла апробацію на XXVI Всеукраїнській науково-практичній конференції «Могилянські читання–2023: досвід та тенденції розвитку суспільства в Україні: глобальний, національний та регіональний аспекти» (м. Миколаїв, листопад 2023 р.) та на фіналі Конкурсу стартапів «Startup BSNU 2024».

Пояснювальна записка бакалаврської роботи складається зі вступу, трьох розділів, висновків, переліку джерел посилання та чотирьох додатків. У вступі визначається актуальність теми, сформульовані мета, об’єкт, предмет роботи та завдання для досягнення поставленої мети. У першому розділі проводиться аналіз існуючих технологій потрібних для подальшої роботи. У другому розділі розглянуто та підібрано потрібні компоненти для розробки. У третьому розділі описано розробку АПЗ розроблюваної системи виявлення об’єктів на основі CSI мережі Wi-Fi. В цілому, бакалаврська робота містить 81 сторінку (без додатків), 33 рисунки, 12 таблиць, 26 джерел посилання.

Ключові слова: WiFi-мережа, інформація про стан каналу, RSSI, ESP32, виявлення об’єктів.

ABSTRACT
of the Bachelor's Thesis
"Object detection system based on CSI of Wi-Fi network"
Student: Dan Frych
Supervisor: Doctor of Technical Sciences, Professor Iryna Zhuravska

The bachelor's thesis is devoted to the creation of the object detection system based on the channel state information (CSI) of Wi-Fi network, as well as the development of the appropriate hardware and software. The relevance of this work is due to the potential of using systems based on the analysis of CSI data for non-invasive detection of dangerous objects in the environment.

The object of the work is the process of objects' detection based on the analysis of Wi-Fi network signal. The subject of the study is methods and algorithms for processing the CSI of Wi-Fi signals for detecting, classifying and determining the distance to the objects.

The work was tested at the XXVI All-Ukrainian Scientific and Practical Conference "Mohyla Readings-2023: Experience and Trends in the Development of Society in Ukraine: Global, National and Regional Aspects" (Mykolaiv, Nov.2023) and at the Final Pitch Day of Startup BSNU 2024 competition.

The explanatory note of the bachelor's thesis consists of an introduction, three chapters, conclusions, references and four appendices. The introduction defines the relevance of the topic, formulates the purpose, object, subject and objectives of the bachelor's work, and tasks that need to be performed to achieve the goal. The first chapter analyzes the existing technologies required for further work. The second section considers and selects the necessary components for development. The third chapter describes the development of hardware and software for an object detection system based on CSI of Wi-Fi network. In general, the bachelor's thesis contains 81 pages (without appendices), 33 figures, 12 tables, 26 sources of references, and 4 appendices.

Keywords: Wi-Fi network, channel state information, RSSI, ESP32, objects' detection.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

АПЗ – апаратно-програмне забезпечення
БД – база даних
СКБД – система керування базами даних

A-MPDU – Aggregated Mac Protocol Data Unit
ANN – Artificial Neural Network
AP – Access Point
CNN – Convolutional Neural Network
CSI – Channel State Information
GUI Graphical User Interface
IoT – Internet of Things
LOS – Line of Sight
MIMO – Multiple-Input and Multiple-Output
NIC – Network Interface Card
OFDM – Orthogonal Frequency Division Multiplexing
PCA – Principal Component Analysis
RF – Radio Frequency
RNN – Recurrent Neural Network
RSSI – Received Signal Strength Indication
SISO – Single-Input and Singe-Output
SNR – Signal to Noise Ratio
STBC – Space–Time Block Code
SSID – Service Set Identifier
UART – Universal Asynchronous Receiver-Transmitter
WEP – Wired Equivalent Privacy
WPA – Wi-Fi Protected Access

ВСТУП

Бездротові технології стали невід'ємною частиною сучасного світу, забезпечуючи зручний та швидкісний вихід у мережу Інтернет. Однак, окрім своєї основної функції – передачі даних – бездротові мережі мають значний потенціал для використання в інших галузях, наприклад, у таких як моніторинг середовища та виявлення об'єктів. Ця технологія відкриває нові можливості для створення інноваційних рішень, які можуть бути застосовані у сферах безпеки, розумних будинків, робототехніки та інших.

Актуальність дослідження системи виявлення об'єктів на основі інформації про стан каналу (англ. Channel State Information, CSI) мережі Wi-Fi обумовлена кількома чинниками. По-перше, наявність вже розгорнутої інфраструктури бездротових мереж робить таку систему економічно вигідною та зручною у впровадженні. По-друге, використання радіочастотних сигналів, замість традиційних датчиків або камер, забезпечує неінвазивний та безперешкодний моніторинг середовища, що є особливо важливим в певних ситуаціях, наприклад, в умовах конфлікту чи надзвичайних ситуацій. По-третє, така система може бути корисною для вирішення низки повсякденних завдань, таких як контроль доступу, виявлення вторгнень або відстеження переміщень.

В Україні, яка нині перебуває в стані війни, питання моніторингу та безпеки набувають особливої актуальності. Впровадження систем виявлення об'єктів на основі CSI мережі Wi-Fi може сприяти підвищенню рівня безпеки в місцях великого скупчення людей, таких як вокзали, торгові центри, театри та кінотеатри.

Метою роботи є розвиток методів неінвазивного моніторингу середовища для виявлення об'єктів на основі аналізу CSI-інформації про мережу Wi-Fi.

Для досягнення мети необхідно виконати такі ***завдання***:

– проаналізувати методи та засоби отримання, обробки та аналізу інформації про стан каналу (CSI) в мережах Wi-Fi за її характеристиками;

– дослідити існуючі підходи щодо використання CSI для виявлення присутності об'єктів у середовищі, їх класифікації та локалізації;

– розробити апаратну та програмну платформу для реалізації системи аналізу CSI;

– дослідити зміну CSI-даних в залежності від матеріалу об’єкту, що створює перешкоду;

– провести тестування розробленого апаратно-програмного забезпечення (АПЗ);

– навести приклади можливих покращень розробленого рішення.

Об'єктом дослідження є процес виявлення об'єктів на основі аналізу сигналів мережі Wi-Fi.

Предметом дослідження є методи та засоби виявлення об’єктів шляхом обробки інформації про стан каналу (CSI) сигналів Wi-Fi.

Практичне значення результатів дослідження та розробки полягає у можливості створення системи для моніторингу середовища, виявлення змін та визначення характеристик об'єктів без використання додаткових датчиків чи камер, що забезпечує економію ресурсів та розширює сфери застосування технології Wi-Fi. Крім того, ця робота може стати покроковою інструкцію з налаштування середовища для отримання CSI-даних для фахівців, що захочуть провести власні дослідження у цій сфері.

Апробація результатів кваліфікаційної роботи відбулася під час XXVI Всеукраїнської науково-практичної конференції «Могилянські читання – 2023» (Миколаїв, 06–10 листопада 2023 р.) та на фіналі Конкурсу стартапів «Startup BSNU 2024» (Миколаїв, 18 квітня 2024 р.) [27].

Публікації. Основні положення роботи опубліковані у збірнику матеріалів XXVI Всеукраїнської науково-практичної конференції «Могилянські читання – 2023» [28].

1 МЕТОДИ ТА ЗАСОБИ ОТРИМАННЯ, ОБРОБКИ ТА АНАЛІЗУ ІНФОРМАЦІЇ ПРО СТАН КАНАЛУ (CSI) В МЕРЕЖАХ WI-FI

1.1 Теоретичні основи виявлення об'єктів на основі показників CSI мережі Wi-Fi

1.1.1 CSI

CSI (Channel State Information) – це інформація про стан каналу, що описує властивості середовища поширення радіохвиль у безпровідних мережах. У мережах Wi-Fi CSI надає детальні відомості про амплітуду та фазу прийнятих сигналів на різних частотних діапазонах (підканалах) OFDM (Orthogonal Frequency Division Multiplexing) [1]. Ця інформація є ключовою для виявлення об'єктів та їх руху в межах зони покриття мережі Wi-Fi.

Структура CSI мережі Wi-Fi включає такі основні компоненти:

а) точки доступу (англ. Access Points, APs) пристрої, що забезпечують безпроводове з'єднання для клієнтських пристроїв та збирають CSI-дані;

б) клієнтські пристрої (наприклад, ноутбуки, смартфони) – пристрої, що підключаються до точок доступу для отримання доступу до мережі та можуть надавати додаткові CSI-дані;

в) канал зв'язку – середовище, через яке поширюються радіохвилі між точками доступу та клієнтськими пристроями.

Принцип роботи CSI мережі Wi-Fi базується на аналізі поширення радіохвиль у каналі зв'язку. Сигнали, що передаються від точки доступу, відбиваються від різних об'єктів (стін, меблів, людей тощо) і досягають клієнтського пристрою різними шляхами. Ці багатопроменеві компоненти сигналу інтерферують між собою, що призводить до змін амплітуди та фази сигналу на різних підканалах OFDM [2]. Ці зміни є унікальними для конкретного середовища поширення сигналу і залежать від розташування та характеристик об'єктів у цьому середовищі.

CSI містить інформацію про амплітуду та фазу для кожного підканалу OFDM, що дозволяє досліджувати характеристики багатопроменевого

середовища з високою роздільною здатністю. Ця інформація може бути використана для виявлення об'єктів та їх руху в межах зони покриття мережі Wi-Fi шляхом аналізу змін у CSI-даних [3].

Для збору CSI-даних використовуються спеціальні драйвери та інструменти, такі як Linux 802.11n CSI Tool [4], які дозволяють отримувати доступ до низькорівневої інформації про стан каналу в мережах Wi-Fi, що базуються на стандартах 802.11n та 802.11ac (рис. 1.1).

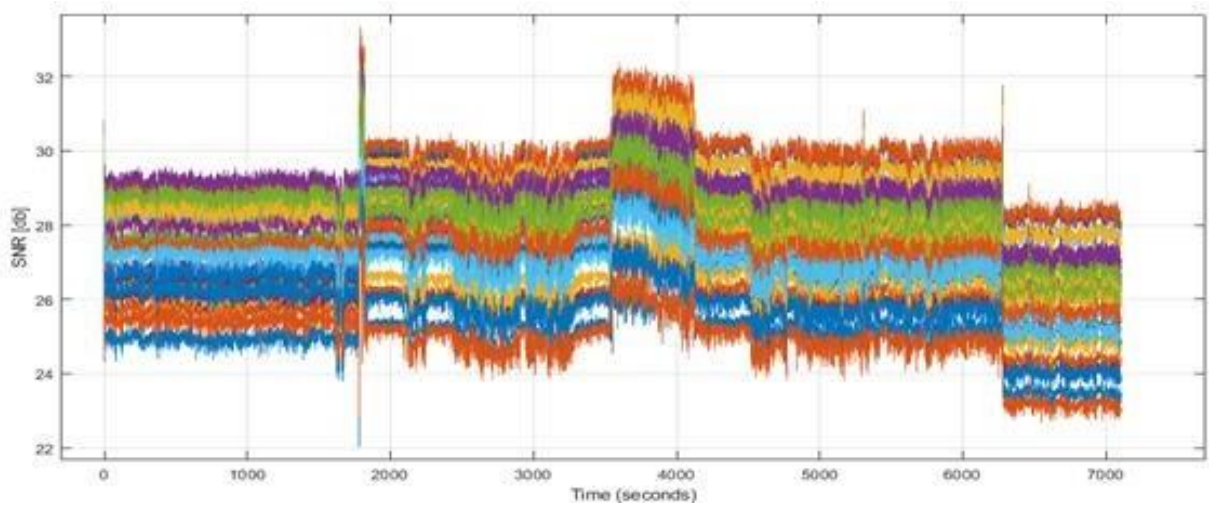


Рисунок 1.1 – Графік відношення сигнал/шум CSI [5]

Збір та аналіз CSI-даних відкриває нові можливості для розробки застосунків і систем на основі бездротових мереж Wi-Fi. Окрім виявлення об'єктів та руху, CSI-дані можуть бути використані для розпізнавання дій і активності людей [5], відстеження переміщень [6], локалізації пристроїв [7] тощо.

Однак, використання CSI-даних також має певні обмеження та виклики. CSI-дані можуть бути чутливими до перешкод та змін у середовищі поширення сигналу, що може призводити до неточностей та помилок у виявленні об'єктів. Крім того, збір та обробка великих обсягів CSI-даних вимагає потужних обчислювальних ресурсів.

1.1.2 RSSI

У бездротових мережах однією з ключових характеристик сигналу, яку можна легко виміряти, є показник потужності прийнятого сигналу (англ.

Received Signal Strength Indicator, RSSI). Цей параметр відображає рівень потужності сигналу, отриманого приймачем від передавача, і може бути корисним для різноманітних застосувань, пов'язаних із моніторингом та виявленням об'єктів.

RSSI є безодиничною величиною, що вимірюється у відносних одиницях або відсотках, а не в абсолютних фізичних одиницях потужності сигналу. Значення RSSI залежить від відстані між передавачем і приймачем, оскільки потужність сигналу зменшується зі збільшенням відстані через загасання у середовищі поширення. Крім того, RSSI чутливий до перешкод, таких як стіни, меблі, людські тіла або інші радіочастотні пристрої, що працюють на тій самій або близькій частоті.

Потужність сигналу, що поширюється в просторі, зменшується відповідно до законів поширення радіохвиль. Основними факторами, що впливають на згасання сигналу, є відстань, перешкоди на шляху поширення та багатопроменеве поширення. У вільному просторі потужність сигналу спадає обернено пропорційно до квадрата відстані (модель поширення у вільному просторі). Однак у реальних умовах присутні численні відбиття та розсіювання сигналу від перешкод, що призводить до більш складної моделі загасання, яка враховує додаткові втрати.

Більшість сучасних бездротових пристроїв, таких як точки доступу Wi-Fi та клієнтські пристрої, здатні вимірювати RSSI. Проте значення RSSI можуть відрізнятися для різних виробників або моделей через відмінності в апаратних реалізаціях та методах вимірювання. Деякі виробники можуть надавати калібровані значення RSSI в одиницях dBm (рівень потужності в децибелах щодо опорного рівня 1 мВт), що дозволяє порівнювати показники між різними пристроями.

Зміни в значеннях RSSI можуть вказувати на присутність або рух об'єктів у зоні покриття бездротової мережі. Це робить RSSI корисним для таких застосувань:

– виявлення присутності об'єктів. Раптові зміни у значеннях RSSI можуть свідчити про появу об'єкта в зоні покриття мережі;

– відстеження руху об'єктів. Аналіз змін RSSI в часі та просторі дозволяє відстежувати переміщення об'єктів у межах бездротової мережі;

– локалізація об'єктів. За допомогою методів, таких як трилатерація або фінгерпринтинг, можна оцінити приблизне місцезнаходження об'єкта на основі RSSI від кількох точок доступу;

– класифікація об'єктів. Різні об'єкти можуть по-різному впливати на RSSI, що дозволяє розрізняти їх за характерними «відбитками» змін сигналу.

Незважаючи на корисність RSSI для моніторингу об'єктів, існують певні обмеження та виклики, що потрібно враховувати:

– вплив середовища. Перешкоди, багатопроменеве поширення та інші фактори можуть значно спотворювати значення RSSI, ускладнюючи інтерпретацію даних;

– точність локалізації. Використання лише RSSI для локалізації об'єктів може забезпечити лише наближену оцінку місцезнаходження з обмеженою точністю. Тому у початковому експерименті доцільно попередньо визначити відстань до виявленого об’єкта;

– варіативність між пристроями. Різні виробники та моделі бездротових пристроїв можуть демонструвати різні характеристики RSSI, що ускладнює стандартизацію та порівняння;

– динамічність середовища. Зміни в оточенні, такі як переміщення людей або меблів, можуть вплинути на значення RSSI, що вимагає постійної калібрації або адаптації системи.

Для подолання цих обмежень часто використовуються додаткові методи та алгоритми, такі як машинне навчання, фільтрація даних, комбінування декількох показників (наприклад, RSSI та часових параметрів) та інтеграція з іншими датчиками.

Показник RSSI є доступною та корисною характеристикою для моніторингу та виявлення об'єктів у бездротових мережах, таких як Wi-Fi.

Хоча RSSI має певні обмеження, пов'язані з впливом середовища та варіативністю між пристроями, його можна ефективно використовувати для виявлення присутності, відстеження руху та локалізації об'єктів у поєднанні з відповідними алгоритмами обробки даних. Розуміння теоретичних основ та обмежень RSSI є важливим для розробки надійних систем моніторингу на основі бездротових мереж.

1.1.3 SISO та MIMO

У бездротових мережах Wi-Fi для передачі та прийому сигналів використовуються різні конфігурації антен. Існують дві основні технології, які застосовуються у цій галузі: SISO (Single Input Single Output) та MIMO (Multiple Input Multiple Output). Обидві ці технології відіграють важливу роль у системах виявлення об'єктів на основі даних CSI.

SISO є найпростішою конфігурацією антен, яка включає одну антену для передачі сигналу та одну антену для його прийому. Ця технологія широко використовується у системах виявлення об'єктів за допомогою CSI через кілька ключових переваг:

– спрощена апаратна реалізація: SISO не потребує складних антенних масивів чи обчислювально складних алгоритмів формування променів, що робить її простішою у впровадженні та економічно ефективнішою;

– достатність даних CSI: Для задач виявлення об'єктів та руху важливішою є якісна інформація про амплітуду та фазу сигналу, ніж підвищена пропускна здатність. SISO може надавати достатньо деталізовані дані CSI для цих цілей;

– зменшення багатопроменевого розсіювання: У деяких випадках багатопроменеве поширення сигналу, яке посилюється MIMO, може ускладнити інтерпретацію даних CSI. SISO забезпечує простішу модель поширення сигналу.

Однак, SISO має обмеження щодо діапазону та стійкості до перешкод, особливо у великих приміщеннях або середовищах з багатьма перешкодами.

MIMO, з іншого боку, є більш просунутою технологією, яка використовує кілька антен для передачі та прийому сигналів. Це дозволяє збільшити пропускну здатність та підвищити продуктивність бездротових сигналів за рахунок багатопроменевого поширення. У контексті виявлення об'єктів на основі CSI, MIMO має кілька переваг:

– покращена якість сигналу: Використання кількох антен забезпечує кращу якість сигналу та більш різноманітну інформацію про навколишнє середовище, що може підвищити точність виявлення об'єктів;

– збільшений діапазон та проникнення: MIMO може покращити діапазон та здатність сигналу проникати через перешкоди, що особливо корисно у великих приміщеннях або середовищах з багатьма перешкодами;

– підвищена стійкість до завад: Використання декількох потоків даних підвищує стійкість системи до завад та перешкод.

Однак, реалізація MIMO вимагає використання складніших апаратних компонентів, таких як антенні масиви та обчислювально складні алгоритми формування променів, що може збільшити вартість та складність системи.

Математично система MIMO може бути виражена наступним чином:

$$Y = Hx + n,$$

де Y – вектор прийому;

x – вектор передачі;

H – канальна матриця, що містить комплексні значення CSI;

n – вектор фонового шуму навколишнього середовища [3].

Вибір між SISO, MIMO чи гібридним підходом залежить від конкретних вимог застосунку, розмірів приміщення, наявних обчислювальних ресурсів та бюджету. Правильне поєднання цих технологій може забезпечити оптимальну продуктивність системи виявлення об'єктів на основі CSI мережі Wi-Fi.

1.1.4 OFDM

Основною причиною використання CSI є якість передачі даних. Стандарт 802.11n передає дані за допомогою ортогонального

мультиплексування з частотним розділенням каналів (OFDM). OFDM забезпечує передачу модульованих даних на декількох піднесучих з використанням декількох антен на передачі та на стороні приймача (рис. 1.2). На якість даних, що передаються між передавачем і приймачем, впливає багато факторів, таких як розсіювання, згасання і втрата сигналу на відстані. Для вимірювання якості використовується CSI, оскільки він може вимірювати властивості каналу на кожній окремій піднесучій через декілька антен [13].

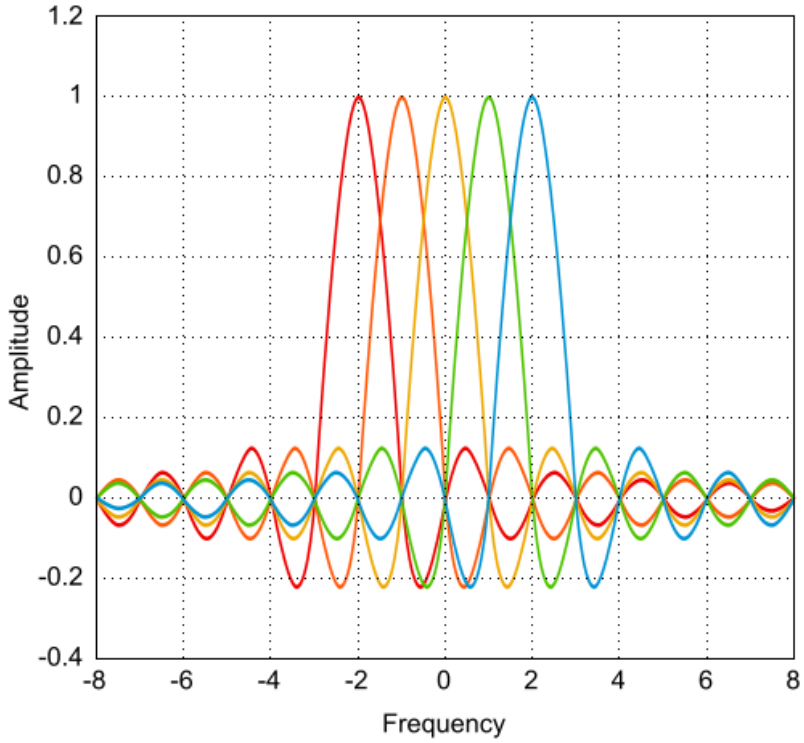


Рисунок 1.2 – Розподіл піднесучих OFDM сигналу відносно амплітудно-частотної характеристики

Матриця CSI або канальна матриця *H* містить комплексні числа, які описують амплітудні та фазові кути бездротового зв'язку [13]. Використовуючи MIMO, матриця CSI створюється шляхом передачі даних до приймача. Піднесучі, розсіюючись по різних шляхах, генерують унікальну CSI-сигнатуру, яка описується значеннями їх амплітуди і фази. Значення CSI визначаються для кожної піднесучої. Приймач, у свою чергу, будує матрицю даних CSI і повертає інформацію передавачу.

1.1.5 Функціональні особливості CSI мережі Wi-Fi

CSI-дані є надзвичайно чутливими до найменших змін у навколишньому середовищі. Це обумовлено природою поширення радіохвиль, які взаємодіють з різними об'єктами та матеріалами на своєму шляху. Навіть незначний рух або зміна положення об'єкта може призвести до помітних змін в амплітуді та фазі сигналу на різних підканалах.

Ступінь заломлення, відбиття та поглинання радіохвиль залежить від фізичних властивостей матеріалів, з якими вони взаємодіють. Наприклад, металеві поверхні сильно відбивають радіохвилі, тоді як об'єкти з води або біологічних тканин можуть спричиняти значне поглинання сигналу. Перешкоди з діелектричних матеріалів, таких як дерево, пластик або цегла, можуть викликати заломлення радіохвиль.

Ефекти заломлення та відбиття радіохвиль найбільш помітні на вищих частотах, адже довжина хвилі стає порівнянною з розмірами перешкод. Тому дані CSI, отримані на вищих частотних підканалах, зазвичай містять більше інформації про дрібні зміни в середовищі.

Крім того, поляризація радіохвиль також відіграє важливу роль. Вертикально поляризовані хвилі більш чутливі до вертикальних перешкод, таких як люди або меблі, тоді як горизонтально поляризовані хвилі краще взаємодіють з горизонтальними поверхнями, такими як підлога або стеля.

Одним з ключових викликів у використанні CSI є необхідність ефективної фільтрації та обробки величезного обсягу даних, що надходять від різних частотних підканалів. Для цього часто використовуються складні алгоритми машинного навчання, які можуть виявляти закономірності та тренди у даних CSI, пов'язані з рухом або присутністю об'єктів.

1.2 Переваги та недоліки використання CSI мережі Wi-Fi для виявлення об'єктів

Використання CSI мережі Wi-Fi для виявлення об'єктів є відносно новою та перспективною технологією, яка відкриває нові можливості для

безперешкодного та прихованого моніторингу середовища. На відміну від традиційних систем виявлення на основі камер, сканерів, датчиків руху чи інших спеціалізованих пристроїв, CSI мережа використовує наявну інфраструктуру Wi-Fi та аналізує зміни в характеристиках поширення радіохвиль для виявлення об'єктів.

Ця технологія має низку переваг, які роблять її привабливою для широкого кола застосувань, серед них:

– висока чутливість: на основі CSI-даних можна виявляти навіть незначні зміни в середовищі, спричинені присутністю чи рухом об'єктів, завдяки аналізу детальної інформації про амплітуду та фазу сигналу на різних піднесучих;

– широка зона покриття: оскільки CSI мережі використовує для збору даних стандартну інфраструктуру Wi-Fi, вона може забезпечувати виявлення об'єктів у великих приміщеннях або навіть на великих відкритих територіях;

– відсутність перешкод: на відміну від оптичних або інфрачервоних систем, радіохвилі Wi-Fi можуть проникати крізь непрозорі перешкоди, такі як стіни чи меблі, що робить систему виявлення об’єктів на основі CSI-даних більш гнучкою для використання в різноманітних середовищах;

– низька вартість: система може бути реалізована з використанням наявної інфраструктури Wi-Fi та бюджетних пристроїв, таких як ESP32, що робить її економічно вигідною.

Однак, незважаючи на свої переваги, використання системи виявлення об’єктів на основі CSI мережі Wi-Fi також має певні недоліки та обмеження, які необхідно враховувати:

– чутливість до перешкод: CSI-дані можуть бути спотворені різноманітними перешкодами, такими як інші безпроводові пристрої, електромагнітні завади чи динамічні зміни в середовищі поширення сигналу;

– обмежена роздільна здатність: хоча за допомогою CSI мережі можливо виявляти присутність об'єктів, здатність методу визначати точне

місцезнаходження або форму об'єктів є обмеженою порівняно з системами на основі камер або лідарів;

– складність обробки даних: аналіз великих обсягів CSI-даних вимагає обчислювальних ресурсів та ефективних алгоритмів машинного навчання;

– питання конфіденційності: оскільки подібні системи, що базуються на аналізі CSI-даних, можуть виявляти присутність людей та відстежувати їх рух, це може викликати занепокоєння щодо конфіденційності та необхідності забезпечення належного захисту даних;

– обмеження стандартів: не всі стандарти Wi-Fi підтримують збір CSI-даних, що може обмежувати застосування цієї технології в певних ситуаціях або вимагати використання додаткового обладнання.

Незважаючи на певні недоліки, використання CSI мережі Wi-Fi залишається перспективною технологією для виявлення об'єктів, особливо в сценаріях, де потрібен економічно вигідний, безперешкодний та негайний моніторинг великих зон покриття.

1.3 Сучасні інформаційні технології у сфері виявлення об'єктів на основі CSI мережі Wi-Fi

1.3.1 Технології штучного інтелекту та глибокого навчання

Згорткові нейронні мережі (англ. Convolutional Neural Networks, CNN), рекурентні нейронні мережі (англ. Artificial Neural Networks, RNN) і штучні нейронні мережі (англ. Artificial Neural Networks, ANN) є трьома найпотужнішими моделями машинного навчання, доступними сьогодні. Кожен тип мережі має свої переваги та недоліки та найкраще підходить для різних типів завдань. Такі методи глибокого навчання, як CNN та RNN, стали провідними технологіями для аналізу та інтерпретації CSI-даних. Ці алгоритми здатні автоматично виявляти складні закономірності та ознаки в багатовимірних CSI-даних, що робить їх ідеальними для завдань виявлення, класифікації та відстеження об'єктів.

Одним з прикладів успішного застосування CNN є система DeepSense [26], розроблена дослідниками з Університету Каліфорнії. DeepSense використовує згорткові нейронні мережі для розпізнавання різних видів діяльності людини, таких як ходьба, стрибки, присідання, на основі CSI-даних (рис. 1.3). Система продемонструвала високу точність понад 90 % під час тестування в різноманітних середовищах.

Перевагами використання глибокого навчання є висока точність, здатність обробляти великі обсяги даних та виявляти складні закономірності. Однак, ці переваги компенсуються високими обчислювальними вимогами та необхідністю великих обсягів розмічених даних для навчання моделей.

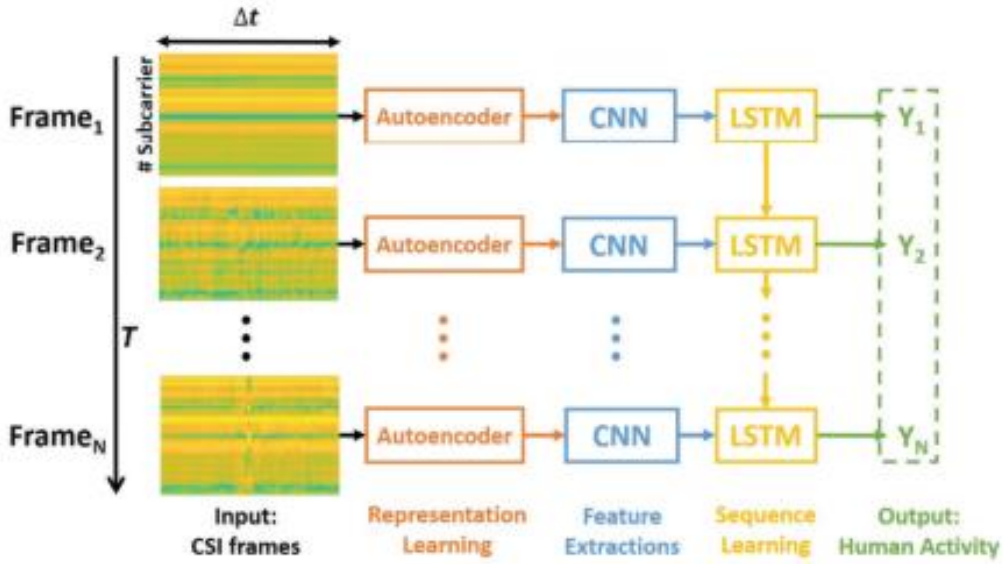


Рисунок 1.3 – Архітектура нейронної мережі у системі DeepSense [26]

Крім CNN, також застосовуються рекурентні нейронні мережі (RNN), які ефективно обробляють послідовні та часові CSI-дані. Наприклад, у роботі [8] запропоновано систему WiFi-U, яка використовує RNN для відстеження траєкторій руху людей у приміщенні з точністю до 0,5 м.

Також технології машинного навчання використовується у системі FreeDetector (рис. 1.4), яка використовує звичайні пристрої Wi-Fi для виявлення присутності людей у приміщенні без необхідності використовувати будь-які додаткові датчики чи пристрої, що носять на собі [25].

Основна ідея полягає у використанні даних CSI від декількох точок доступу Wi-Fi для відстеження змін у поширенні радіохвиль, спричинених рухом людей. FreeDetector аналізує ці зміни за допомогою машинного навчання для розрізнення різних типів рухів та визначення кількості людей у приміщенні.

Ключові особливості FreeDetector:

– система використовує модифіковані драйвери для мережевого обладнання для отримання даних CSI з декількох точок доступу в режимі реального часу;

– з сирих даних CSI виділяються корисні ознаки, такі як відхилення амплітуди та фази на різних підканалах;

– система використовує алгоритми машинного навчання, такі як random forest та штучні нейронні мережі, для класифікації виявлених рухів та підрахунку людей.

У своїх експериментах автори продемонстрували, що FreeDetector може досягати точності виявлення присутності людей понад 87 % у різних сценаріях. Система також може розрізняти різні типи рухів, такі як ходьба, сидіння та інші.

Рисунок 1.4 – Система FreeDetector [25]

Однак, варто зазначити, що точність системи може залежати від розташування точок доступу, розміру та форми приміщення, а також інших факторів навколишнього середовища. Крім того, система може мати труднощі з розрізненням окремих людей та визначенням їх точного місцезнаходження.

Загалом, робота FreeDetector демонструє потенціал використання CSI мережі Wi-Fi для задач виявлення та відстеження присутності людей, відкриваючи нові можливості для систем моніторингу та безпеки.

1.3.2 Технології з використанням мікроконтролерів

Одним із напрямків розвитку CSI-технологій є розробка рішень, що базуються на використанні бюджетних пристроїв, таких як ESP32 або Raspberry Pi. Це відкриває нові можливості для впровадження систем виявлення об'єктів на основі CSI у невеликих проєктах, домашніх середовищах та навчальних цілях.

Прикладом такої технології є проєкт ESP32-CSI-Tool [11], розроблений групою дослідників. Це програмне забезпечення дозволяє збирати CSI-дані за допомогою бюджетних пристроїв ESP32, що робить його доступним для широкого кола користувачів та ентузіастів.

Головною перевагою використання бюджетних пристроїв є їх низька вартість та енергоефективність, що робить їх придатними для застосувань з обмеженими ресурсами. Крім того, вони часто мають невеликі розміри, що полегшує їх інтеграцію в різноманітні середовища.

Однак слід зазначити, що бюджетні пристрої, як правило, мають обмежені обчислювальні можливості, що може стати перешкодою для запуску ресурсномістких алгоритмів обробки CSI-даних або глибокого навчання. Тому їх застосування найбільш доцільне для простих завдань виявлення об'єктів або у поєднанні з хмарними обчислювальними ресурсами.

Незважаючи на свою потужність, технології глибокого навчання також мають деякі недоліки. Крім високих обчислювальних вимог, вони можуть бути схильними до проблеми «чорного ящика», коли важко інтерпретувати

та пояснити, як модель приймає рішення. Це може бути проблемою у сценаріях, де потрібна прозорість процесу прийняття рішень.

1.3.3 Технології для інтелектуальних середовищ

CSI-дані можуть бути інтегровані з іншими датчиками та системами для створення інтелектуальних середовищ, таких як розумні будинки, розумні офіси чи розумні міста. Такі інтегровані рішення дозволяють збирати різноманітні дані про середовище та поведінку людей для забезпечення автоматизації, енергозбереження та підвищення комфорту.

Одним з провідних досліджень у цій галузі є система WiTrack [3] від Microsoft Research. WiTrack використовує CSI-дані для відстеження рухів людей у приміщенні та взаємодії з різноманітними пристроями розумного дому, такими як освітлення, кліматична техніка або мультимедійні системи (рис. 1.5). Наприклад, WiTrack може визначати, коли людина входить у кімнату, і автоматично ввімкнути світло та налаштувати комфортну температуру. Система також може розпізнавати певні жести рук для керування пристроями без фізичного контакту.

Рисунок 1.5 – Приклад роботи WiTrack [3]

Перевагою інтеграції CSI-технологій в інтелектуальні середовища – це можливість створення більш розумних, зручних та енергоефективних систем. CSI-дані надають цінну інформацію про присутність та поведінку людей, що дозволяє автоматизувати процеси та адаптувати середовище відповідно до їхніх потреб.

Однак реалізація таких інтегрованих рішень вимагає ретельного проєктування та координації різних компонентів системи між собою. Крім того, питання конфіденційності та безпеки також є критичними, оскільки ці системи можуть збирати чутливі дані про поведінку користувачів.

Незважаючи на ці виклики, технології для інтелектуальних середовищ з використанням CSI-даних є перспективним напрямком розвитку, який може сприяти створенню більш комфортних, ефективних та екологічних середовищ життя та роботи.

1.4 Формування вимог до АПЗ

Функціональні вимоги:

– отримання даних CSI: система повинна мати можливість безперебійно отримувати дані канальної інформації (CSI) від пристрою, підключеного до WiFi-мережі, з високою частотою вибірки для забезпечення достатньої деталізації даних;

– збереження даних: система повинна забезпечувати надійне та ефективне збереження отриманих CSI-даних у базі даних (БД) разом з відповідними мітками;

– управління еталонними зразками: система повинна надавати зручний інтерфейс для внесення, редагування та видалення еталонних CSI-зразків для різних типів об'єктів у БД;

– виявлення об'єктів: ключовою функцією системи є можливість порівнювати нові CSI-вимірювання з еталонними зразками в БД для виявлення присутності об'єктів із високою точністю та швидкістю;

– класифікація об'єктів: на основі порівняння з еталонними зразками CSI, система повинна визначати тип виявленого об'єкта;

– візуалізація результатів: система повинна забезпечувати графічне відображення CSI-даних.

Нефункціональні вимоги:

– продуктивність: система повинна забезпечувати швидке порівняння нових CSI-даних з еталонними зразками для забезпечення оперативного виявлення об'єктів у режимі реального часу або з мінімальною затримкою;

– масштабованість: система повинна бути масштабованою та здатною ефективно обробляти обсяги CSI-даних, що надходять від кількох пристроїв або джерел, без суттєвого зниження продуктивності;

– зручність використання: система повинна мати інтуїтивно зрозумілий та зручний для користувача інтерфейс, який дозволяє легко керувати системою, переглядати результати та налаштовувати параметри.

Вимоги до апаратного забезпечення:

– збір CSI-даних: система повинна отримувати та передавати CSI-дані через WiFi-мережу;

– підключення до комп’ютеру або серверу: система повинна мати змогу підключатися до комп'ютера або сервера з достатніми обчислювальними ресурсами для ефективної обробки великих обсягів CSI-даних та зберігання БД;

– мережеве з'єднання: система повинна мати стабільне та високошвидкісне з'єднання Wi-Fi для безперебійної передачі CSI-даних від пристрою збору даних до системи.

Вимоги до програмного забезпечення:

– система керування базами даних (СКБД): для зберігання CSI-даних та еталонних зразків необхідна СКБД, що здатна ефективно працювати з великими обсягами даних та забезпечувати швидкий пошук і доступ до даних;

– бібліотеки та інструменти: система повинна використовувати сучасні та ефективні бібліотеки та інструменти для обробки та аналізу CSI-даних, реалізації алгоритмів порівняння та класифікації, візуалізації даних та результатів виявлення об'єктів;

– безпека: для забезпечення безпеки та конфіденційності даних система має використовувати відповідні інструменти та бібліотеки для шифрування даних.

Висновки до розділу 1

Аналіз принципів роботи та структури CSI мережі Wi-Fi виявив, що її використання для виявлення об'єктів має низку переваг, серед яких: можливість працювати в умовах обмеженої видимості, низькі вимоги до апаратного забезпечення та невисока вартість розгортання системи. Однак слід враховувати і певні недоліки, такі як чутливість до перешкод та обмежений радіус дії.

Огляд існуючих методів обробки CSI-даних та алгоритмів машинного навчання засвідчив активний розвиток цієї галузі. Зокрема, перспективними є підходи, засновані на нейронних мережах, які демонструють вищу точність виявлення навіть для складних сцен.

Важливо зауважити, що більшість розробок націлені саме на виявлення людини та її переміщень у приміщенні. Це свідчить про значний потенціал технології CSI. Однак, дослідження також показують, що ця технологія може бути адаптована для виявлення інших типів об'єктів, зокрема небезпечних, що відкриває нові можливості в галузі безпеки. Такі системи можуть стати заміною рамок-сканерів для виявлення небезпечних предметів.

Таким чином, було виставлено вимоги до розробки системи виявлення об’єктів на основі CSI мережі Wi-Fi. Майбутня система повинна бути здатною у реальному часі збирати, обробляти та аналізувати CSI-дані для виявлення та класифікації об'єктів. Така система базуватиметься на достатньо потужному апаратному забезпеченні та ефективному програмному стеку.

2 ПРОЄКТУВАННЯ СИСТЕМИ ВИЯВЛЕННЯ ОБЄКТІВ НА ОСНОВІ CSI МЕРЕЖІ WI-FI

2.1 Вибір технологій та компонентів для реалізації АПЗ

2.1.1 Програмна платформа для збору CSI-даних

Для збору CSI-даних існує декілька найпопулярніших інструментів, що використовують охоплюють різне апаратне забезпечення (табл. 2.1):

- Linux 802.11n CSI Tool;
- Atheros CSI Tool;
- ESP32 CSI Tool.

Таблиця 2.1 – Порівняння інструментів для збору CSI-даних

Характеристика	**ESP32 CSI Tool**	**802.11n CSI Tool**	**Atheros CSI Tool**
Платформа	ESP32 (Espressif)	Intel WiFi Link	Atheros
Діапазон частот, ГГц	2,4	2,4/5	2,4/5
Кількість піднесучих	64	56/114	56/114
Підтримувані частоти, МГц	2412–2484	2412–2484, 5180–5825	2412–248, 5180–5825
Доступ до CSI	Повний	Частковий	Частковий
Вартість обладнання	Низька	Висока	Середня
Портативність	Висока	Низька	Низька
Енергоспоживання	Низьке	Високе	Середнє
Відкритий код	Так	Так	Так
Підтримка спільноти	Середня	Висока	Середня
Простота використання	Висока	Середня	Низька

Інструменти Linux 802.11n CSI Tool і Atheros CSI Tool [10] схожі в тому, що вони знімають CSI з мережевої карти (NIC) і використовують бездротові

драйвери з відкритим вихідним кодом. Однак є кілька основних відмінностей. Linux 802.11n Tool використовує мережеву карту Intel Wi-Fi Link 5300 та драйвер бездротового зв'язку *iwlwifi* [12]. Обмеженням інструменту Linux 802.11n є модифікована прошивка, яка використовується мережевою картою Intel Wi-Fi Link 5300 із закритим кодом. Іншим застереженням є те, що карта Intel обмежена захопленням 30 груп піднесучих CSI [12]. 30 груп піднесучих через декілька антен забезпечують значну кількість даних щодо CSI. Зібрані дані рівномірно розподілені між 56 піднесучими для каналу 20 МГц або 114 піднесучими для каналу 40 МГц [11].

Для порівняння, Atheros CSI Tool використовує ath9k – драйвер ядра Linux з відкритим вихідним кодом, – який стверджує, що підтримує всі чіпсети Atheros 802.11n [12]. Хоча ath9k підтримує ширший спектр бездротових пристроїв, обмеження на використання мережевої карти все ще залишається.

Функції інструменту мають відкритий вихідний код і розроблені виключно за допомогою програмного забезпечення [12]. Отже, немає необхідності вносити зміни до прошивки, і користувачі можуть модифікувати програмний інструмент відповідно до своїх потреб [16]. Однією з головних переваг Atheros CSI Tool є детальні інструкції щодо встановлення та використання, розміщені на вебсайті. До того ж, цей інструмент може захоплювати всі 56 піднесучих для каналу 20 МГц або 114 піднесучих для каналу 40 МГц [12].

Третьою програмною платформою для збору CSI-даних є ESP32 CSI Tool. Раніше згадані інструменти вимагають апаратного забезпечення для підтримки мережевої карти та додаткове обладнання для передачі. ESP32 – це автономний мікроконтролер «все-в-одному», який можна запрограмувати на роботу в якості точки доступу або активного бездротового клієнта, і він може збирати та одразу зберігати дані CSI, використовуючи вбудовану пам'ять. Інструмент ESP32 може захоплювати до 64 піднесучих, хоча через обмеження пропускної здатності для обробки CSI в реальному часі рекомендується зменшити розмір захоплення до 32 піднесучих або менше [10].

Серед доступних інструментів для збору та аналізу даних CSI бездротових мереж Wi-Fi, які були перераховані вище, в рамках даного дослідження було обрано ESP32-CSI-Tool через його високу портативність, низьку вартість обладнання та простоту використання.

2.1.2 Вибір мікроконтролера

ESP32 – це досить потужна серія мікроконтролерів від Espressif Systems, яка ідеально підходить для застосунків, пов'язаних з бездротовими мережами та обробкою сигналів. Вона базується на 32-бітному процесорі Tensilica Xtensa LX6 і призначена для широкого спектру застосувань, включаючи IoT (Internet of Things, укр. Інтернет речей) обробку даних від різноманітних датчиків тощо.

Серія мікроконтролерів ESP32 є відкритою апаратно-програмною платформою, що дозволяє розробникам створювати власні проєкти та модифікувати вихідний код відповідно до своїх потреб. Для застосунків використовується мова програмування C/C++. ESP32 підтримується великою спільнотою розробників та має багато бібліотек, прикладів коду та документації, що полегшує процес розробки.

Мікроконтролер ESP32 має набір вводу/виводу (GPIO) та підтримує декілька інтерфейсів, таких як UART, SPI, I2C, що дозволяє легко інтегрувати його з різноманітними датчиками, модулями та периферійними пристроями для збору додаткових даних.

Для розробки системи виявлення об'єктів на основі CSI мережі Wi-Fi, важливим є вибір моделі мікроконтролера, який забезпечить збір даних та ефективну обробку інформації. Тому доцільно детально розглянути декілька варіантів:

- ESP32;
- ESP32-S2;
- ESP32-S3;
- ESP32-C3;
- ESP32-C6.

ESP32 – це універсальний та бюджетний мікроконтролер (рис. 2.1), розроблений компанією Espressif Systems і представлений ще у 2016 році. Даний мікроконтролер має наступні характеристики (табл. 2.2).

Таблиця 2.2 – Характеристики ESP32

Характеристика	**Значення**
Процесор	Dual-core Xtensa LX7, до 240 МГц
Пам’ять	320 кбайт SRAM, до 4 Мбайт вбудованої flash-пам'яті
Wi-Fi	802.11b/g/n (2,4 ГГц)
Bluetooth	Bluetooth 5.0 LE
GPIO	До 43 програмованих виводи
Інтерфейси	USB OTG, SPI, I²C, I²S, UART
Ціна, грн	270

ESP32 є потужним та багатофункціональним рішенням, особливо для проєктів, які вимагають високої обчислювальної потужності, багатозадачності та широкого спектру інтерфейсів. Його двоядерний процесор, велика кількість GPIO та підтримка як Wi-Fi, так і Bluetooth роблять його ідеальним для складних IoT-застосувань, систем обробки даних у реальному часі та проєктів, що вимагають бездротового зв'язку.

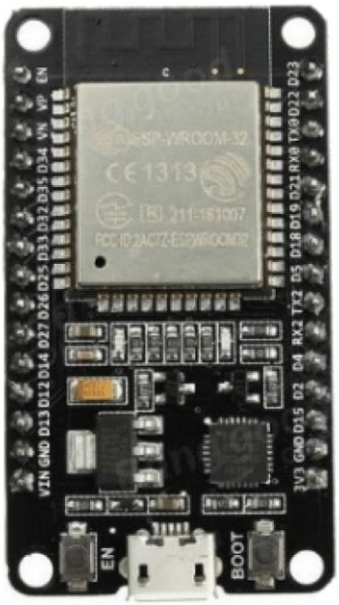


Рисунок 2.1 – ESP32-DevKit-V1

ESP32-S2 – одноядерний мікроконтролер (рис. 2.2), який має наступні характеристики, узагальнені до табл. 2.3 [2].

Таблиця 2.3 – Характеристики ESP32-S2

Характеристика	**Значення**
Процесор	Dual-core Xtensa LX7, до 240 МГц
Пам’ять	320 кбайт SRAM, до 4 Мбайт вбудованої flash-пам'яті
Wi-Fi	802.11b/g/n (2,4 ГГц)
Bluetooth	Bluetooth 5.0 LE
GPIO	До 43 програмованих виводи
Інтерфейси	USB OTG, SPI, I²C, I²S, UART
Ціна, грн	550

ESP32-S2 оптимізований для низького енергоспоживання та має вбудований USB інтерфейс, що може спростити процес програмування та взаємодії з комп'ютером. Однак, відсутність підтримки Bluetooth та наявність лише одного ядра процесора можуть обмежити його застосування в деяких сценаріях [3].

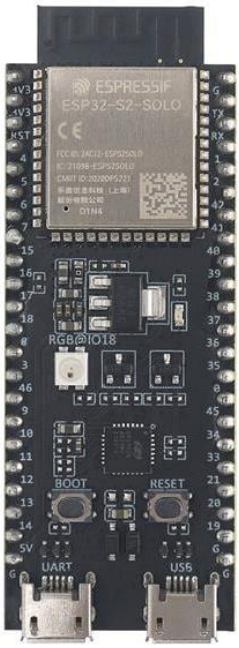


Рисунок 2.2 – ESP32-S2-DevKitC-1R

ESP32-S3 – це мікроконтролер є покращеною версією ESP32-S2 (рис. 2.3), з такими характеристиками, наведеними у табл. 2.4 [4].

Таблиця 2.4 – Характеристики ESP32-S3

Характеристика	Значення
Процесор	Dual-core Xtensa LX7, до 240 МГц
Пам’ять	512 кбайт SRAM, до 8 Мбайт вбудованої flash-пам'яті
Wi-Fi	802.11b/g/n (2,4 ГГц)
Bluetooth	Bluetooth 5.0 LE
GPIO	До 45 програмованих виводів
Інтерфейси	USB OTG, SPI, I²C, I²S, UART
Ціна, грн	600

ESP32-S3 має двоядерний процесор та підтримку Bluetooth 5.0 LE, що робить його більш універсальним для IoT-проєктів. Його збільшена пам'ять та кількість GPIO роблять його привабливим для більш складних застосувань, включаючи обробку CSI-даних [5].

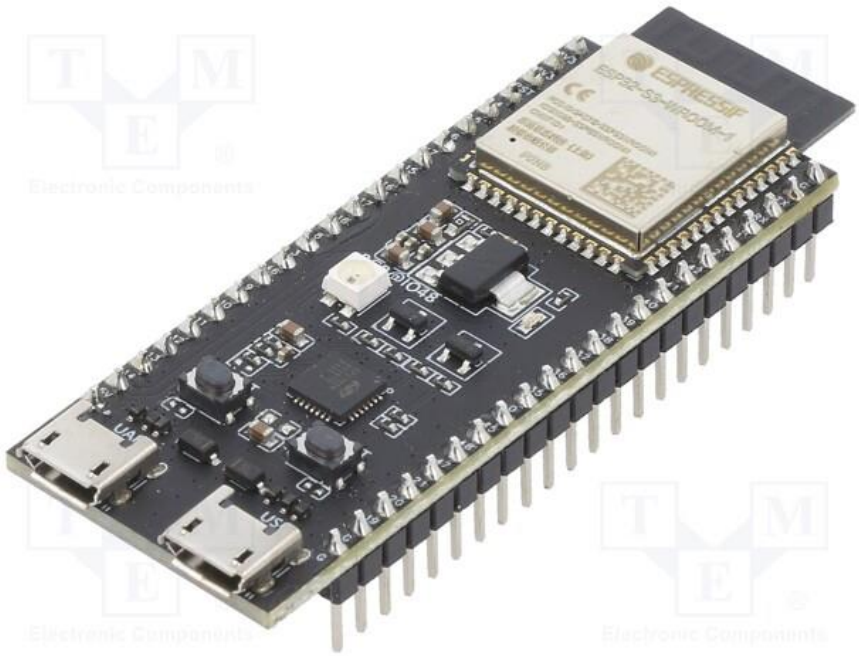


Рисунок 2.3 – ESP32-S3-DevKitC-8

Мікроконтролер ESP32-C3 побудований на архітектурі RISC-V та має характеристики, наведені у табл. 2.5 [6].

Таблиця 2.5 – Характеристики ESP32-C3

Характеристика	Значення
Процесор	32-bit single-core RISC-V, до 160 МГц
Пам’ять	400 кбайт SRAM, до 4 Мбайт вбудованої flash-пам'яті
Wi-Fi	802.11b/g/n (2,4 ГГц)
Bluetooth	Bluetooth 5.0 LE
GPIO	22 програмованих виводи
Інтерфейси	SPI, I²C, I²S, UART, RMT, TWAI
Ціна, грн	480

ESP32-C3 є компактним та економічно ефективним рішенням, особливо для проєктів, де важливі мініатюризація та низька вартість (рис. 2.4). Його архітектура RISC-V може забезпечити кращу енергоефективність, але менша кількість GPIO та нижча тактова частота можуть обмежити його застосування в складних системах обробки CSI [7].

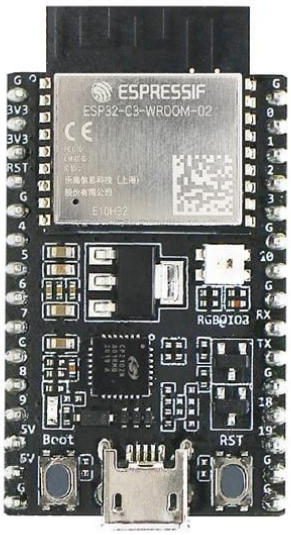


Рисунок 2.4 – ESP32-C3-DevKitC-02

ESP32-C6 – найновіший мікроконтролер в серії ESP32 (рис. 2.5), який має такі характеристики, наведені у табл. 2.6 [8].

Таблиця 2.6 – Характеристики ESP32-C6

Характеристика	**Значення**
Процесор	32-bit single-core RISC-V, до 160 МГц
Пам’ять	400 кбайт SRAM, до 4 Мбайт вбудованої flash-пам'яті
Wi-Fi	802.11a/b/g/n (2,4 та 5 ГГц)
Bluetooth	Bluetooth 5.0 LE
GPIO	30 програмованих виводів
Інтерфейси	SPI, I²C, I²S, UART, RMT, MCPWM
Ціна, грн	500

ESP32-C6 (рис. 2.5) відрізняється підтримкою Wi-Fi на частотах 2,4 ГГц та 5 ГГц, а також стандарту IEEE 802.15.4, що робить його ідеальним для систем, які потребують різноманітних протоколів зв'язку (табл. 2.7). Його збільшена кількість GPIO у порівнянні з ESP32-C3 також розширює можливості підключення датчиків та пристроїв [10].

Рисунок 2.5 – ESP32-C6-DevKit-C1

Таблиця 2.7 – Детальні характеристики мікроконтролерів ESP32 [10]

Параметри	**Моделі**			
	ESP32	**ESP32-S2**	**ESP32-S3**	**ESP32-C3**
Процесор	Tensilica Hitensa 32-bit LX6	Tensilica Hitensa 32-bit LX7	Tensilica Hitensa 32-bit LX7 dual core	RISC-V 32-bit
SRAM, Kbyte	520	320	512	400
ROM, Kbyte	448	128	384	384
JTAG	Так	Так	n/a	Так
Кеш, Kbyte	64	8/16 (configurable)	n/a	16
Версія Wi-Fi	Wi-Fi 4	Wi-Fi 4	Wi-Fi 4	Wi-Fi 4
Версія Bluetooth	BLE 4.2 (upgrade to 5.0, with limitations)	Hi	BLE 5.0	BLE 5.0
Ethernet	Так	Hi	n/a	Hi
RTC memory, Kbyte	16	16	16	8
PMU	Так	Так	n/a	Так
ULP coprocessor	Так	ULP-RISC-V	n/a	Hi
Cryptographic Accelerator	SHA, RSA, AES, RNG	SHA, RSA, AES, RNG, HMAC, Digital Signature	SHA, RSA, AES, RNG, HMAC, Digital Signature	SHA, RSA, AES, RNG, HMAC, Digital Signature
Secure boot	Так	Так	Так	Так
Flash encryption	Так	XTS-AES-128/256	Так	XTS-AES-128
SPI	4	4	n/a	3
I2C	2	2	n/a	1
I2S	2	1	n/a	1
UART	3	2	n/a	2
GPIO	34	43	44	22
LED PWM	16	8	n/a	6
MCPWM	6	0	2	0
Pulse counter	8	4	n/a	0
USB	Hi	USB OTG 1.1	n/a	Serial/JTAG

Параметри	Моделі			
	ESP32	**ESP32-S2**	**ESP32-S3**	**ESP32-C3**
ADC	2 × 12-bit SAR, up to 18 channels	2 × 13-bit SAR, up to 20 channels	n/a	2 ×12-bit SAR, up to 6 channels
DAC	2 × 8-bit	2 × 8-bit	n/a	Hi
RMT	8 × transmission/reception	4 × transmission/reception	n/a	2 × transmission + 2 ×reception
Таймер	4 × 64-bit	4 × 64-bit	n/a	2 × 54-bit + 1 × 52-bit
Temperature Sensor	Так	Так	n/a	Так
Hall Sensor	Так	Hi	n/a	Hi
Touch Sensor	10	14	n/a	Hi

У контексті розробки системи виявлення об'єктів на основі CSI мережі Wi-Fi ідеальним вибором стає ESP32 (плата ESP32-DevKit-V1) завдяки його двоядерному процесору, достатній кількості пам'яті та GPIO, а також підтримці Wi-Fi 802.11n. Крім того, ця версія мікроконтролеру найпопулярніша та найбюджетніша, через що пошук мікроконтролера в магазині не створює проблем. Також саме ESP32-DevKit-V1 точно підходить для роботи з інструментом для збору даних про стан каналу мережі Wi-Fi ESP32-CSI-Tool, це зазначено у таблиці протестованих пристроїв у документації [10].

Отже, вибір саме ESP32 (плати ESP32-DevKit-V1) є оптимальним для даного проєкту. Технічні характеристики мікроконтролера повністю відповідають висунутим вимогам до апаратного забезпечення.

2.1.3 Вибір мови програмування та бібліотек

Для розробки програмного забезпечення системи виявлення об'єктів на основі CSI мережі Wi-Fi доцільно використовувати дві мови програмування

C++ для роботи з мікроконтролером ESP32 та Python для обробки й візуалізації даних на комп'ютері.

Перевагою C++ є його низькорівневий доступ до апаратних ресурсів та система компіляції коду, що забезпечує високу продуктивність виконання програм. Для ESP32 код на C++ збирається за допомогою утиліти cmake з використанням ESP-IDF (Espressif IoT Development Framework) офіційного середовища розробки від Espressif. Скомпільована програма стає оптимізованою для конкретної мікроконтролерної архітектури та працює максимально ефективно.

Ще однією важливою перевагою C++ є наявність великої кількості бібліотек та фреймворків для роботи з різним апаратним забезпеченням, включаючи бібліотеки для ESP32 від Espressif. Весь вихідний код для взаємодії з WiFi-модулем та збору CSI-даних, який використовується в цьому проєкті, написаний саме на C++.

На відміну від C++, Python є інтерпретованою мовою з динамічною типізацією. Це робить Python більш гнучким та портативним: один і той же програмний код може запускатись на різних операційних системах без повторної компіляції.

Ключовою перевагою Python є багатий вибір бібліотек для візуалізації та (PyQtGraph, Matplotlib, Plotly), аналізу даних (NumPy, SciPy), та навіть машинного навчання (TensorFlow, Scikit-learn). Зокрема, для візуалізації даних та створення графічного інтерфейсу (англ. Graphical User Interface, GUI) застосунку буде використовуватися бібліотека PyQt5.

PyQt – це оболонка на мові програмування Python для Qt, потужної кросплатформної бібліотеки для створення GUI застосунків. Використання PyQt дозволяє об'єднати простоту та гнучкість Python з багатими можливостями Qt для розробки сучасних та інтерактивних користувацьких інтерфейсів.

Однією з ключових переваг PyQt є наявність PyQtGraph – спеціалізованого модуля для створення якісних графіків та візуалізації даних.

PyQtGraph дозволяє легко будувати різноманітні види графіків, включаючи лінійні, стовпчикові, розсіювання, 3D-поверхні та багато іншого. Ця функціональність є дуже корисною для візуалізації зібраних CSI-даних у різних форматах.

Крім того, PyQt поставляється з QtDesigner – потужним інструментом для створення GUI за допомогою зручного конструктора інтерфейсів. QtDesigner дозволяє візуально проєктувати вікна застосунків, розміщувати на них різні віджети (кнопки, поля вводу, діаграми тощо) та налаштовувати їх властивості. Це значно спрощує та пришвидшує процес розробки графічних інтерфейсів.

Порівняно з іншими популярними бібліотеками візуалізації даних для Python, такими як Matplotlib, PyQt має деякі переваги (табл. 2.8).

Таблиця 2.8 – Порівняння PyQt та Matplotlib

Критерій	**PyQt (PyQtGraph)**	**Matplotlib**
Платформи	Кросплатформна	Кросплатформна
GUI	Вбудовані віджети GUI	Окремі вікна
Інтерактивність	Висока	Обмежена
Продуктивність	Висока	Середня
3D графіка	Підтримується	Обмежена підтримка
Анімація	Вбудована	Вимагає додатковий код
Конструктор для розробки GUI	QtDesigner	Немає

Як видно з таблиці, PyQt забезпечує більшу інтерактивність, кращу продуктивність, підтримку 3D графіки та анімації, а також зручний інструмент QtDesigner для розробки GUI. З іншого боку, Matplotlib є більш спеціалізованим на візуалізації даних і може бути кращим вибором для простих випадків або вбудовування графіків у вебзастосунки.

Поєднання C++ для ефективної роботи з апаратним забезпеченням ESP32 та збору даних з Python для їх подальшої обробки та візуалізації на комп'ютері є оптимальним рішенням для даного проєкту. Кросплатформність

обох мов також забезпечує гнучкість та можливість переносу застосунку на будь-яку систему в майбутньому.

2.1.4 Вибір СКБД

У проєкті з виявлення об'єктів на основі CSI мережі Wi-Fi необхідно забезпечити надійне та ефективне зберігання великих обсягів зібраних даних. Очікується, що система буде генерувати кілька гігабайт CSI-даних щодня, які потрібно структуровано зберігати для подальшого аналізу та обробки. Тому вибір відповідної БД є критично важливим рішенням.

Розглядалися різні варіанти БД, серед яких популярні серверні рішення – MySQL та PostgreSQL, а також хмарні БД, такі як Google Cloud SQL чи Amazon RDS. Проте всі ці варіанти мають певні недоліки для використання в даному проєкті.

По-перше, клієнт-серверна архітектура серверних БД вимагає підключення до віддаленого сервера або налаштування окремого серверного процесу, що додає зайвої складності. По-друге, хмарні БД можуть бути дорогими у довгостроковій перспективі при зберіганні великих обсягів інформації.

Після ретельного аналізу було прийнято рішення використовувати локальну вбудовану БД SQLite. Порівняння з альтернативними варіантами продемонструвало низку переваг SQLite для цього проєкту (табл. 2.9).

Таблиця 2.9 – Порівняння БД

Критерій	**SQLite**	**MySQL**	**PostgreSQL**	**Хмарні БД**
Портативність	Висока, самодостатня	Низька	Низька	Низька
Вбудованість	Так, в застосунок	Ні	Ні	Ні
Надійність	Висока	Висока	Висока	Залежить від провайдера
Продуктивність	Достатня	Висока	Висока	Висока, масштабується

Критерій	SQLite	MySQL	PostgreSQL	Хмарні БД
Вартість	Безкоштовна	Безкоштовна, але потребує серверу	Безкоштовна, але потребує серверу	Платна
Підтримка SQL	Майже повна	Повна	Повна	Повна
Простота використання	Дуже проста	Складніша	Доволі складна	Залежить від провайдера

SQLite виявилася оптимальним вибором завдяки своїй можливості розгорнути БД локально, без додаткових вебсерверів, високій надійності та достатній продуктивності для обсягів даних проєкту. Вона безкоштовна, проста у використанні та підтримує більшість стандартів SQL, що спростить майбутню міграцію даних за потреби.

На відміну від серверних та хмарних БД, SQLite не потребує додаткових зусиль на налаштування та адміністрування. Це дозволить уникнути зайвих накладних витрат та зосередитися на основній функціональності системи виявлення об'єктів.

Таким чином, SQLite була обрана оптимальною БД для зберігання CSI-даних у даному проєкті завдяки своїй простоті, надійності, ефективності та безкоштовності при достатній функціональності для поставлених вимог.

2.2 Концепція та архітектура АПЗ

В попередньому підрозділі було визначено, що система виявлення об'єктів на основі CSI-даних WiFi-мережі складатиметься з мікроконтролера ESP32-DevKit-V1, комп'ютера з програмним забезпеченням на базі Python та PyQt, а також БД SQLite для зберігання еталонних CSI-даних. Взаємодія між цими компонентами зображена на рис. 2.6.

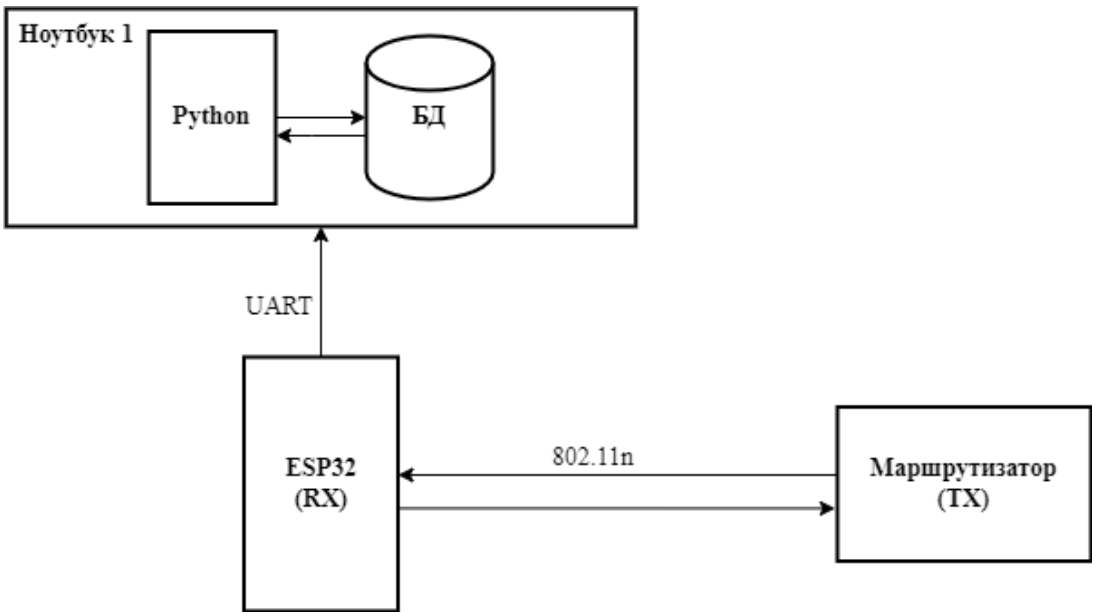


Рисунок 2.6 –Концептуальна схема АПЗ

На схемі зображено три основні модулі системи: TX (маршрутизатор), RX (ESP32) та ноутбук, на якому проводиться аналіз CSI-даних. Маршрутизатор виступає в ролі передавача пакетів даних через мережу Wi-Fi 802.11n. Мікроконтролер ESP32, підключений до цієї ж мережі Wi-Fi 802.11n і зчитує CSI-дані в режимі реального часу.

Система працюватиме наступним чином:

1. Мікроконтролер ESP32 безперервно зчитує CSI-дані з мережі Wi-Fi 802.11n і передає їх на комп'ютер через інтерфейс UART.

2. На комп'ютері працює застосунок, створений на мові Python, який отримує CSI-дані від мікроконтролера ESP32.

3. Застосунок виконує попередню обробку отриманих CSI-даних та порівнює їх з еталонними даними, що зберігаються в БД.

4. Результати порівняння та розпізнавання об'єктів відображаються в графічному інтерфейсі користувача, створеному за допомогою бібліотеки PyQt.

5. База даних SQLite використовується для зберігання еталонних CSI-даних для різних типів об'єктів, що дозволяє системі навчатися та розпізнавати нові об'єкти.

Робота системи поділятиметься на два основних етапи: збір еталонних даних та розпізнавання об'єктів. Такий поділ необхідний, оскільки для коректного розпізнавання об'єктів система повинна мати еталонні CSI-дані для різних типів об'єктів. Система зможе визначити, який саме об'єкт присутній в зоні дії WiFi-мережі, порівнюючи отримані в режимі реального часу CSI-дані з еталонними. Розглянемо тепер кожен етап детальніше.

Етап збору еталонних даних:

1. Для кожного типу об'єкта, який необхідно вміти розпізнавати, система проходить підготовчий етап збору еталонних CSI-даних.

2. Об'єкт розміщується в зоні дії WiFi-мережі, і мікроконтролер ESP32, підключений до цієї мережі, збирає відповідні CSI-дані.

3. Отримані еталонні CSI-дані зберігаються в БД SQLite разом з міткою, що ідентифікує тип об'єкта, для подальшого використання на етапі розпізнавання.

Етап виявлення об'єктів:

1. Мікроконтролер ESP32 безперервно зчитує CSI-дані з WiFi-мережі в режимі реального часу.

2. Отримані CSI-дані передаються через інтерфейс UART на комп'ютер для подальшої обробки.

3. Застосунок, створений на мові програмування Python, отримує потік даних від мікроконтролера ESP32.

4. Програма виконує попередню обробку даних, для відокремлення CSI-даних від інших характеристик.

5. Після попередньої обробки даних програма порівнює отримані CSI-дані з еталонними з БД.

6. Алгоритм порівняння шаблонів обчислює міру подібності між поточними CSI-даними та еталонними даними для кожного типу об'єкта.

7. Тип об'єкта визначається на основі найбільш схожого еталонного шаблону CSI-даних з БД.

8. Результати розпізнавання відображаються в графічному інтерфейсі користувача, створеному за допомогою бібліотеки PyQt.

Ключовим компонентом апаратного забезпечення системи буде мікроконтролер ESP32, який має вбудований модуль Wi-Fi та здатність зчитувати CSI-дані. На мікроконтролер ESP32 буде завантажено прошивку, написану мовою програмування C++, яка має виконувати такі функції:

а) підключення до заданої WiFi-мережі;

б) Безперервне зчитування CSI-даних у режимі реального часу;

в) Передача отриманих CSI-даних через інтерфейс UART на комп'ютер для подальшої обробки;

На комп'ютері буде розроблено застосунок з графічним інтерфейсом користувача (GUI) з використанням мови програмування Python та бібліотеки PyQt. Графічний інтерфейс користувача матиме такі основні компоненти та функціональність:

а) графік для візуалізації вхідних CSI-даних: буде розроблено компонент для відображення CSI-даних, отриманих від мікроконтролера ESP32 в режимі реального часу. Графік дозволить користувачеві наочно спостерігати зміни в характеристиках сигналу при наявності об'єктів у зоні дії WiFi-мережі;

б) інтерфейс для додавання еталонних даних: буде створено вікно для введення типу об'єкта, який використовуватиметься як еталон. Буде наявна кнопка для запуску процесу запису еталонних CSI-даних протягом певного періоду часу (наприклад, 1 хвилини). Під час запису еталонних даних об'єкт має знаходитися в зоні дії WiFi-мережі. Після завершення запису еталонні CSI-дані зберігатимуться в БД SQLite разом з введеним типом об'єкта.

в) інтерфейс для розпізнавання об'єктів: буде реалізована кнопка "Запустити детектор" для запуску режиму розпізнавання об'єктів. Після натискання на кнопку система безперервно отримуватиме CSI-дані від мікроконтролера ESP32 та порівнюватиме їх з еталонними даними з БД

за допомогою алгоритму порівняння шаблонів. Результати розпізнавання (тип об'єкта) відображатимуться в інтерфейсі в режимі реального часу.

Графічний інтерфейс користувача, створений за допомогою бібліотеки PyQt, забезпечить зручну взаємодію з системою, візуалізацію даних та керування процесами збору еталонних даних та розпізнавання об'єктів. Він дозволить користувачеві легко додавати нові типи об'єктів до БД еталонів та відстежувати роботу системи в режимі реального часу.

Висновки до розділу 2

У цьому розділі було визначено основні компоненти апаратно-програмного забезпечення системи виявлення об’єктів на основі CSI мережі Wi-Fi і описано архітектуру та принцип роботи АПЗ.

В якості платформи для збору CSI-даних, поміж усіх доступних варіантів було обрано мікроконтролер ESP32 доступний пристрій, який підтримує роботу з CSI-даними на низькому рівні. Для роботи з ESP32 та отримання CSI-даних буде використовуватися інструмент ESP32-CSI-Tool. Адже для використання інструментів таких як Linux 802.11n CSI Tool та Atheros CSI Tool необхідне більш дороговартісне мережеве обладнання, яке не так просто знайти.

Для програмної реалізації даної системи обрано кілька мов програмування:

– C++ для роботи з мікроконтролером ESP32:

– Python для візуалізації та аналізу отриманих даних з мікроконтролера.

В якості СКБД для зберігання даних буде використана SQLite – компактна локальна БД, що має усі необхідна інструменти для роботи.

Обрані апаратно-програмні компоненти та розроблена концепція архітектури створюють надійне підґрунтя для практичної реалізації системи виявлення об'єктів на основі CSI, деталі якої розглядатимуться в наступному розділі.

3 АПАРАТНО-ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ ТА ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ

3.1 Опис апаратної частини системи

Центральним компонентом апаратної частини системи виявлення об'єктів на основі CSI мережі Wi-Fi є мікроконтролер ESP32 від Espressif Systems. Зокрема, використовується плата розробки ESP32-DevKit-V1 на базі ESP32 (рис. 3.1). Цей мікроконтролер був обраний завдяки своїй високій продуктивності, низькому енергоспоживанню та наявності вбудованого модуля Wi-Fi, що робить його вдалим рішенням для збору CSI-даних.

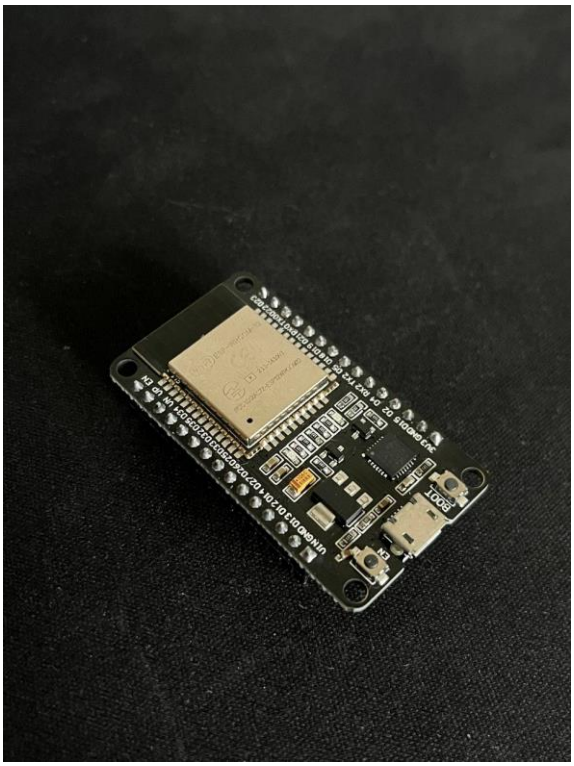


Рисунок 3.1 – ESP-devkit-V1

Використання окремого мікроконтролера для збору CSI-даних замість вбудованих мережевих адаптерів пристроїв, таких як IWL5300 або Atheros, має низку суттєвих переваг. По-перше, це забезпечує значно вищу гнучкість у налаштуванні та конфігурації системи збору даних, оскільки мікроконтролер не прив'язаний до певного пристрою чи операційної системи. По-друге, спеціалізовані мікроконтролери, такі як ESP32, розроблені для досягнення максимальної енергоефективності, що дозволяє системі тривалий час

працювати від батареї. Крім того, використання окремого мікроконтролера, як правило, є більш економічним рішенням порівняно з придбанням кількох різних пристроїв з вбудованими мережевими адаптерами. Нарешті, мікроконтролери легко доступні та можуть бути інтегровані у різноманітні системи, забезпечуючи високий рівень масштабованості.

ESP32 є потужною та водночас енергоефективною платформою, що поєднує двоядерний процесор Tensilica LX6 з тактовою частотою до 240 МГц, 520 КБ оперативної пам'яті SRAM та 4 МБ Flash-пам'яті. Вбудований модуль Wi-Fi забезпечує підтримку стандартів 802.11 b/g/n у режимах точки доступу (AP) та клієнта (STA), що є критично важливим для збору CSI-даних. Крім того, ESP32 підтримує Bluetooth v4.2 BR/EDR та BLE, що відкриває додаткові можливості для інтеграції з іншими пристроями та датчиками.

Одною з ключових переваг ESP32 є наявність широкого спектру периферійних інтерфейсів, таких як UART, SPI, I2C, I2S, RMII та PWM, а також підтримка LCD-дисплеїв. Це дозволяє легко інтегрувати мікроконтролер з різноманітним додатковим обладнанням та розширювати функціональність системи в майбутньому. Крім того, ESP32 підтримує живлення від широкого діапазону напруги – від 2,2 В до 3,6 В, що забезпечує додаткову гнучкість у виборі джерела живлення.

Важливо відзначити, що ESP32 має відкриту апаратну та програмну архітектуру, що сприяє легкій інтеграції та розробці застосунків. Наявність офіційної підтримки від Espressif Systems, а також активної спільноти розробників також є вагомою перевагою цієї платформи. Це забезпечує постійний розвиток екосистеми ESP32, регулярні оновлення програмного забезпечення та доступність численних бібліотек і прикладів коду.

Таким чином, використання плати розробки ESP32-DevKit-V1 на базі мікроконтролера ESP32 є оптимальним рішенням для реалізації апаратної частини системи виявлення об'єктів на основі CSI мережі Wi-Fi. Завдяки своїй високій продуктивності, енергоефективності, наявності вбудованого Wi-Fi модуля та широкому набору периферійних інтерфейсів, ESP32 забезпечує

ефективний, гнучкий та економічний спосіб збору CSI-даних, задовольняючи всі необхідні вимоги проєкту.

3.1.1 Підключення обладнання

Ключовим етапом налаштування апаратної частини системи є підключення плати розробки ESP32-DevKit-V1 до комп'ютера для конфігурації та завантаження на неї прошивки. Цей процес здійснюється за допомогою USB-інтерфейсу, що є стандартним рішенням для більшості сучасних мікроконтролерних платформ.

ESP32-DevKit-V1 оснащена роз'ємом micro USB, який дозволяє легко підключити її до комп'ютера за допомогою стандартного USB-кабелю. Однак, на відміну від багатьох інших пристроїв, ESP32 не підтримує прямий USB-інтерфейс для передачі даних. Замість цього, він використовує послідовний периферійний інтерфейс UART (Universal Asynchronous Receiver-Transmitter) для здійснення комунікації з комп'ютером.

Для забезпечення взаємодії між ESP32 та комп'ютером через USB, на платі ESP32-DevKit-V1 встановлено спеціалізований мікросхемний перетворювач інтерфейсів USB-UART. Цей перетворювач виконує конвертацію даних між USB та UART протоколами, дозволяючи комп'ютеру обмінюватися інформацією з ESP32 через віртуальний COM-порт (рис. 3.2).

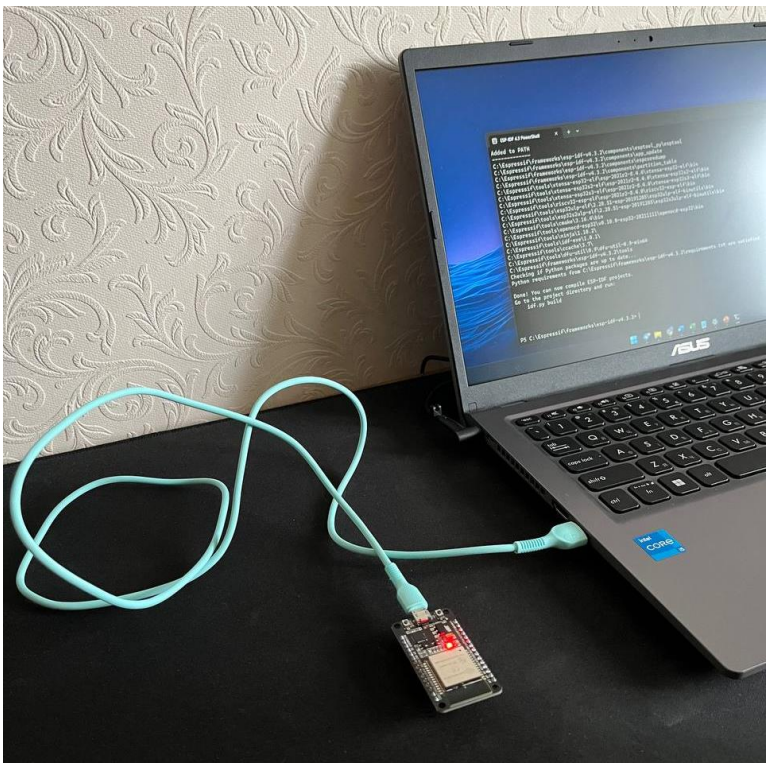


Рисунок 3.2 – Підключення ESP32 до ноутбука

Після підключення ESP32-DevKit-V1 до комп'ютера за допомогою USB-кабелю, операційна система автоматично встановить необхідні драйвери для віртуального COM-порту. Це дозволить програмному забезпеченню для розробки взаємодіяти з ESP32 через послідовний інтерфейс UART.

Використання UART для комунікації з ESP32 надає кілька переваг. По-перше, це дозволяє легко налагоджувати та відстежувати роботу мікроконтролера, виводячи діагностичні повідомлення та дані в термінал на комп'ютері. По-друге, UART забезпечує двосторонній зв'язок, що дозволяє не лише завантажувати програми на ESP32, але й надсилати команди та дані з комп'ютера під час виконання програми.

Крім того, UART є одним з найпоширеніших інтерфейсів для зв'язку з мікроконтролерами та вбудованими пристроями, що забезпечує сумісність ESP32 з широким колом програмного та апаратного забезпечення. Це робить процес розробки та налагодження більш зручним та гнучким.

Таким чином, підключення ESP32-DevKit-V1 до комп'ютера за допомогою micro-USB та використання інтерфейсу UART є ключовим етапом налаштування апаратної частини системи виявлення об'єктів на основі CSI

мережі Wi-Fi. Це забезпечує зручний та ефективний спосіб завантаження програмного забезпечення, налагодження та моніторингу роботи мікроконтролера, що є критично важливим для успішної реалізації проєкту.

3.2 Програмна реалізація системи

3.2.1 Встановлення середовища ESP-IDF

Першим важливим кроком для початку роботи з платою ESP32-DevKit-V1 є встановлення середовища розробки ESP-IDF від компанії виробника мікроконтролерів ESP Espressif Systems. ESP-IDF це середовище з відкритим вихідним кодом, що написане переважно на мові програмування C з використанням деяких компонентів на C++. Воно містить набір бібліотек ядра з низькорівневими абстракціями ESP32, таких як драйвери периферійних пристроїв, протоколи зв'язку (Wi-Fi, Bluetooth, Ethernet), файлові системи та інші основні функції. Крім того, ESP-IDF включає компоненти середовища виконання, які забезпечують планувальник завдань, потоки, синхронізацію, буфери, події та інші інструменти для підтримки багатозадачної обробки.

ESP-IDF забезпечує всі необхідні інструменти для створення, компіляції та завантаження прошивки на мікроконтролери ESP32, включаючи компілятор GCC, бібліотеки, утиліти прошивки та налагодження. Модульна структура ESP-IDF, названа файловою системою компонентів, дозволяє легко інтегрувати або видаляти функціональні компоненти залежно від вимог проєкту.

Середовище розробки ESP-IDF постачається з численними прикладами застосунків, що демонструють різноманітні функції та можливості ESP32, такі як Wi-Fi, Bluetooth, тощо. Також на офіційному сайті доступна детальна документація [9], в якій описано принципи роботи з окремими специфічними компонентами. Крім того, в цій документації описано принципи роботи з мікроконтролером для збору CSI-даних.

У цій кваліфікаційній роботі в якості операційної системи, на якій буде розроблятися програмне забезпечення для мікроконтролеру та аналізу CSI-даних виступає дистрибутив Linux Ubuntu 20.04.

Нижче наведена детальна інструкція з встановлення середовища ESP-IDF 5.0.2 на Linux Ubuntu 20.04:

Спершу варто підготувати необхідні інструменти для коректної роботи середовища розробки. Для цього варто оновити наявні пакети та встановити декілька додаткових залежностей, що вказані в офіційній документації [9]. Весь процес налаштування середовища розробки відбуватиметься через термінал, тому для оновлення пакетів та встановлення нових будемо використовувати команди *apt-get update* та *apt-get install* відповідно:

sudo apt-get update

sudo apt-get install git wget flex bison gperf python3 python3-pip python3-venv cmake ninja-build ccache libffi-dev libssl-dev dfu-util libusb-1.0-0

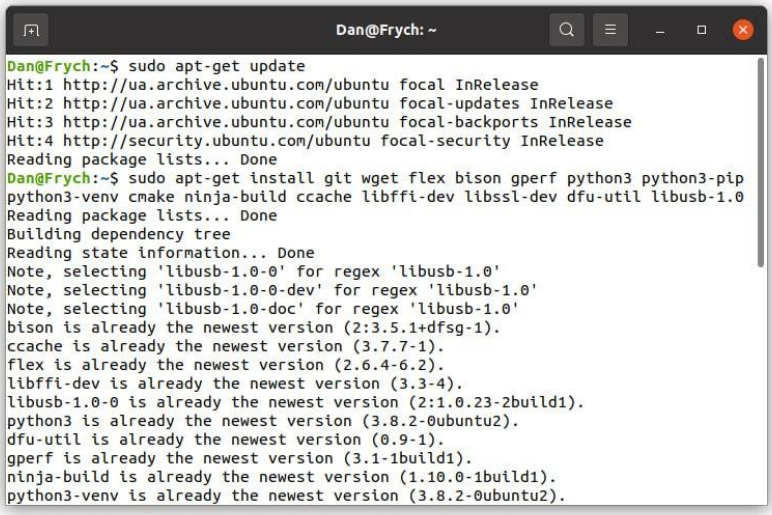


Рисунок 3.3 – Оновлення встановлених пакетів

Для завантаження ESP-IDF необхідно спочатку створити папку, в якій буде зберігатися увесь вихідний код. Після чого за допомогою git варто клонувати гілку віддаленого репозиторію з необхідною версією ESP-IDF (v5.0.2). Команди виглядатимуть наступним чином:

mkdir -p ~/esp32

cd ~/esp32

git clone -b v5.0.2 --recursive https://github.com/espressif/esp-idf.git

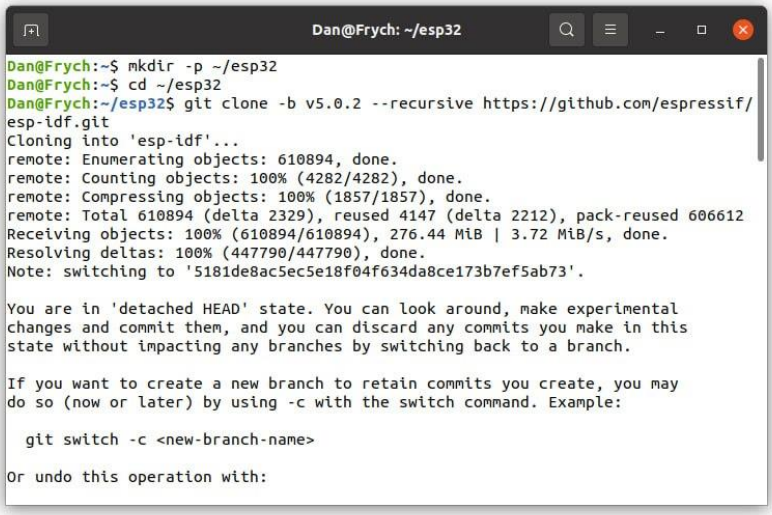


Рисунок 3.4 – Завантаження репозиторію ESP-IDF

Окрім самого ESP-IDF, для проєктів, що підтримують ESP32, також необхідно встановити інструменти, що використовуються ESP-IDF, такі як компілятор, відладчик, пакети Python тощо. Це можна зробити за допомогою наступних команд:

cd ~/esp32/esp-idf

./install.sh esp32

Цей крок може зайняти деякий час, оскільки він встановлює всі необхідні пакети.

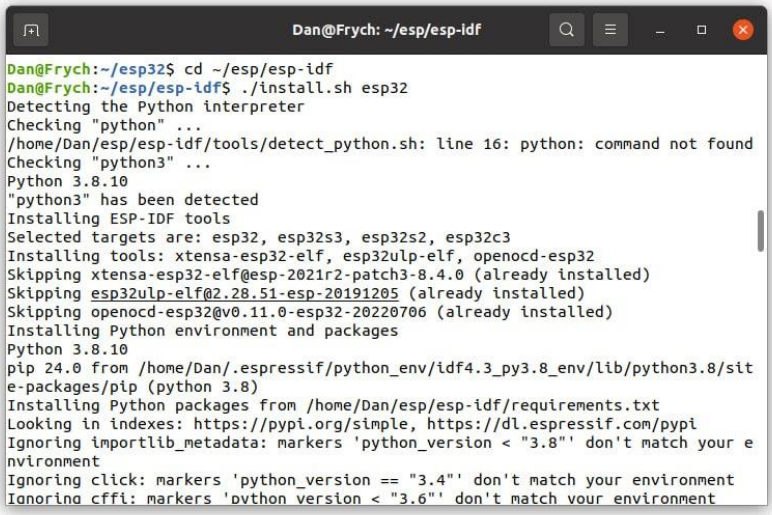


Рисунок 3.5 – Встановлення інструментів ESP-IDF

Встановлені інструменти ще не додано до змінної оточення PATH. Щоб зробити інструменти доступними для використання з терміналу, необхідно встановити деякі змінні оточення. У терміналі, де використовуватиметься ESP-IDF, треба ввести команду:

. $HOME/esp32/esp-idf/export.sh

Зверніть особливу увагу на крапку, що стоїть на початку команди. Вона необхідна для правильного налаштування змінних середовища в поточному сеансі.

Для зручності роботи створимо alias для запуску *export.sh*. Тобто замість того, щоб при вході у термінал виконувати команду *. $HOME/esp/esp-idf/export.sh* можна буде виконати просту команду *get_idf.*

Для цього необхідно відкрити конфігурацію профілю терміналу *.bashrc* та вставити туди команду *alias get_idf='. $HOME/esp32/esp-idf/export.sh'*. Відредагувати файл .bashrc можна за допомогою команди *nano ~/.bashrc* (рис. 3.6).

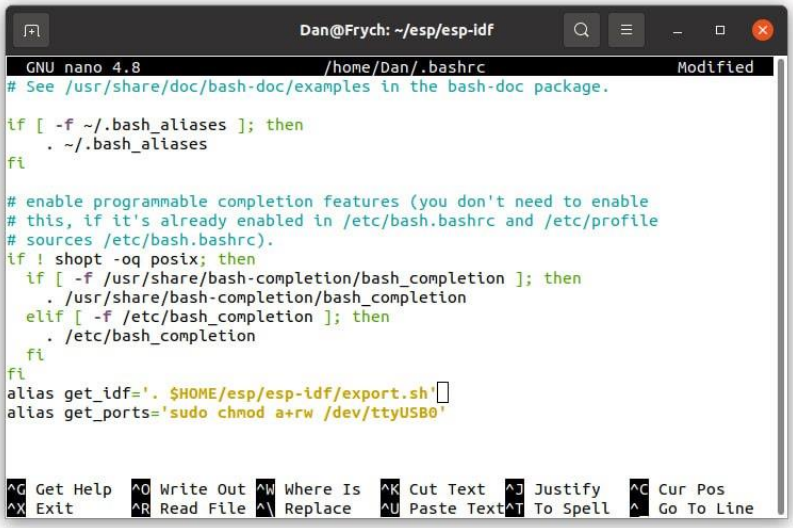


Рисунок 3.6 – Конфігурація .bashrc

Тепер можна легко використовувати *get_idf* для налаштування або оновлення середовища ESP-IDF у будь-якому сеансі роботи з терміналом.

Технічно, можна додати *export.sh* безпосередньо до профілю терміналу, але це робити не рекомендується. Це призведе до активації віртуального середовища IDF у кожному сеансі роботи з терміналом (включно з тими, де

IDF не потрібен), що суперечить призначенню віртуального середовища і, ймовірно, вплине на роботу іншого програмного забезпечення.

Після виконання всіх цих кроків середовище ESP-IDF 5.0.2 успішно встановлено на комп’ютер. Для того, щоб перевірити встановлення, можна виконати наступну команду *get_idf*, що запустить середовище розробки.

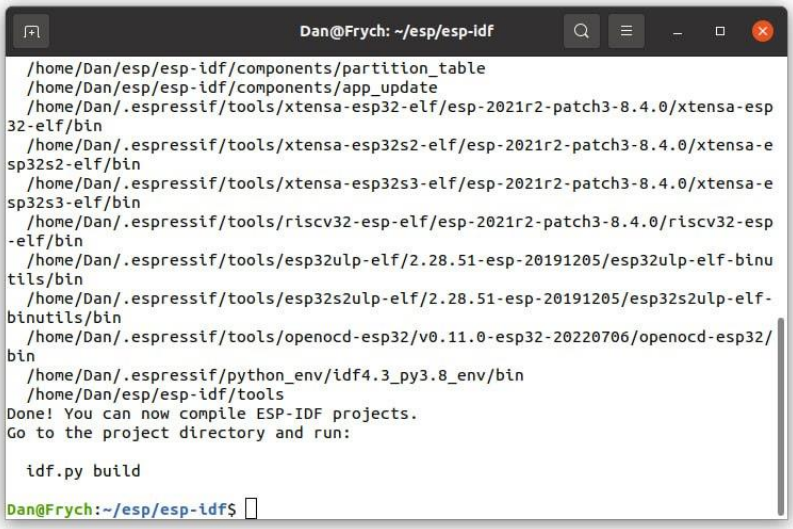


Рисунок 3.7 – Запуск середовища ESP-IDF

Після запуску середовища можна побачити пропозицію перейти до директорії з проєктом та запустити його. Отже на цьому етапі успішно встановлено середовище для розробки ESP-IDF.

3.2.2 Налаштування та конфігурація мікроконтролера

Як було зазначено у розділі 2 для збору CSI-даних з мікроконтролера ESP32 будуть використовуватися інструменти, що доступні в офіційному репозиторії esp-csi від Espressif Systems. У цьому репозиторії містяться приклади коду та бібліотеки для роботи з передачею та отриманням CSI-даних в мережах Wi-Fi з використанням мікроконтролерів ESP32.

Ключовими компонентами цього репозиторію є приклади прошивки для мікроконтролерів серії ESP32 для надсилання (csi_send) та отримання (csi_recv) CSI-даних.

Прошивка csi_send забезпечує функціональність для відправки тестових пакетів через канал Wi-Fi та збору відповідних CSI-даних від прийомного пристрою.

Прошивка csi_recv, у свою чергу, відповідає за отримання CSI-даних від відправника на стороні приймача. Мікроконтролер налаштовується на прослуховування визначеного каналу Wi-Fi та зчитує CSI-дані з кожного прийнятого пакета. Отримані дані можуть бути збережені у файлі або виведені у послідовний порт для подальшого аналізу.

Ці дві прошивки створені для роботи одна з одною, тобто на один мікроконтролер ESP32 завантажується csi_send, для відправки пакетів, а на інший завантажується cse_recv для отримання даних. Проте репозиторій містить прошивку csi_recv_router, яка є варіацією csi_recv, але адаптована для підключення до маршрутизатора Wi-Fi. Пристрій змушує маршрутизатор надсилати пакети даних через команду Ping, щоб отримати дані CSI між пристроєм і маршрутизатором

Усі прошивки написані на мові програмування C та оптимізовані для роботи на мікроконтролерах ESP32 з використанням бібліотек Espressif. Вони забезпечують зручний інтерфейс для налаштування параметрів збору даних, обробки та запису CSI-даних у файли або передачі їх у режимі реального часу для подальшого аналізу.

В ході цієї роботи для збору CSI-даних використовуватиметься прошивка csi_recv_router, тому для подальшої роботи необхідно розуміти як влаштований цей застосунок. Детальніше розглянемо основні компоненти вихідного коду:

1. Включення необхідних заголовочних файлів та бібліотек для роботи з Wi-Fi, CSI та іншими функціями ESP32.

2. Визначення функції *wifi_csi_rx_cb*, яка є обробником отриманих CSI-даних. Ця функція викликається кожного разу, коли ESP32 отримує CSI-дані від маршрутизатора. Функція форматує та виводить отримані дані у вигляді CSV-рядка.

3. Функція *wifi_csi_init* налаштовує конфігурацію CSI для ESP32, включаючи активацію фільтрації каналів, масштабування даних тощо. Вона також встановлює обробник *wifi_csi_rx_cb* для отримання CSI-даних і активує режим CSI на ESP32.

4. Функція *wifi_ping_router_start* налаштовує та запускає періодичну відправку пакетів до маршрутизатора. Ці пакети спричиняють відправлення маршрутизатором пакетів із CSI-даними, які ESP32 отримує та обробляє через встановлений обробник.

5. У функції *app_main* відбувається ініціалізація флеш-пам'яті NVS, мережевого інтерфейсу та події циклу. Після цього викликаються функції *wifi_csi_init* та *wifi_ping_router_start* для налаштування та запуску отримання CSI-даних від маршрутизатора.

Отже, цей код налаштовує ESP32 для отримання CSI-даних від підключеного маршрутизатора Wi-Fi шляхом періодичної відправки ping пакетів. Отримані CSI-дані форматуються і виводяться у вигляді CSV рядків для подальшого аналізу або збереження.

Розуміючи основні принципи роботи прошивки можна переходити до роботи з нею, конфігурацією та завантаження на плату ESP32-DevKit-V1. Перш за все запустимо термінал та активуємо середовище ESP-IDF за допомогою команди *get_idf*. Для роботи з мікроконтролером необхідно переконатися в тому, що плата підключена до комп’ютеру та до якого саме порту. Після чого варто надати права доступу для роботи з цим портом. Це можна зробити за допомогою наступних команд:

sudo dmesg | grep tty

sudo chmod a+rw /dev/ttyUSB0

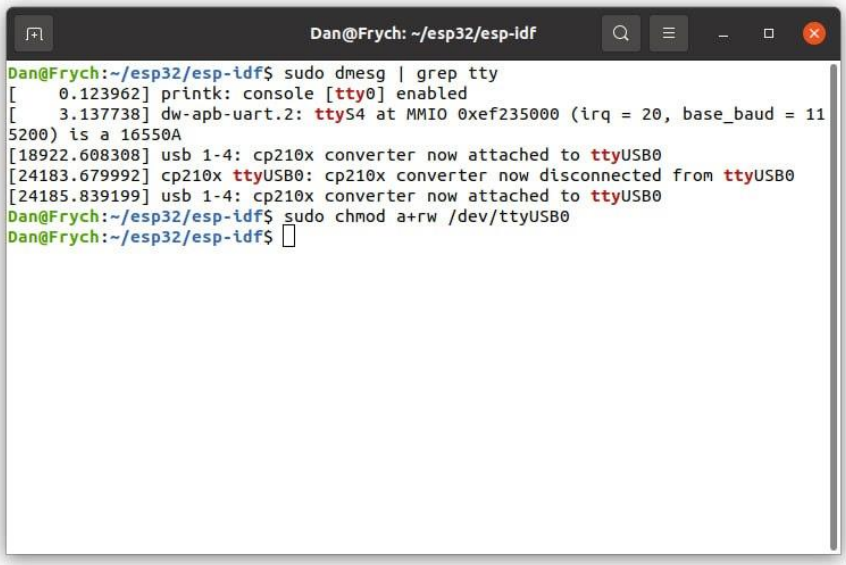
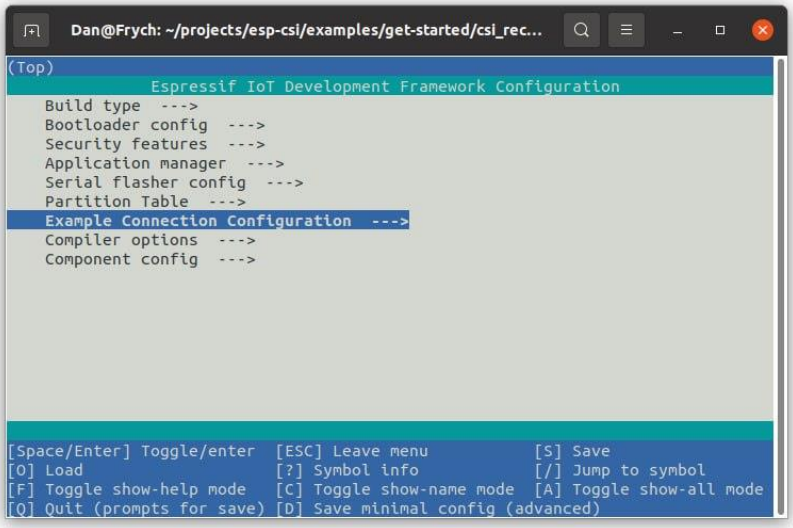
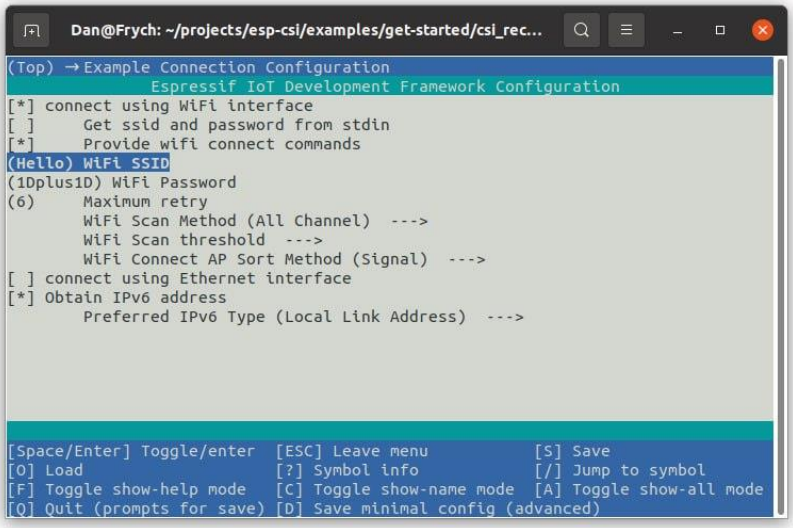


Рисунок 3.8 – Надання прав доступу для послідовного порту

Тепер перейдемо до самого проєкту *csi_recv_router* в репозиторії *esp-csi*, який було скачано в окрему директорію за допомогою команди *git clone https://github.com/espressif/esp-csi.git*. В даному випадку це *~/projects/esp-csi/examples/get-started/csi_recv_router*. Як вже було згадано вище, ця прошивка створена для отримання CSI-даних з маршрутизатору, тому для успішного підключення важливо вказати коректні дані мережі. Це можна зробити через конфігураційний файл (рис. 3.9), що запускається за допомогою команди *idf.py menuconfig*.



а)



б)

Рисунок 3.9 – Конфігурація прошивки для ESP32: а –головне меню; б – меню налаштувань мережі Wi-Fi

Для того, щоб вказати дані мережі, необхідно перейти до пункту Example Connection Configuration та ввести Wi-Fi SSID та Wi-Fi Password. Всі інші налаштування можна лишати за замовчуванням, вони підходять для подальшої роботи.

Заключним етапом роботи з мікроконтролером є завантаження прошивки на ESP32. В ESP-IDF для цього передбачено інструмент *idf.py flash*. Ця команда дозволяє завантажити скомпільований бінарний файл прошивки на ESP32 через послідовний порт. Процес відбувається в кілька етапів: спершу idf.py переводить ESP32 у режим прошивки, потім починається передача даних через послідовний порт, які записуються в флеш-пам'ять мікроконтролера, а після завершення – ESP32 автоматично перезавантажується з новою прошивкою.

Часто використовується додатковий параметр *–p* для вказівки порту, наприклад: *idf.py flash –p /dev/ttyUSB0*. Також можна комбінувати завантаження прошивки з моніторингом виводу ESP32 за допомогою команди *idf.py flash monitor*.

Для моніторингу виводу ESP32 після завантаження слугує команда *idf.py monitor*. Вона підключається до послідовного порту і відображає всі дані, які мікроконтролер надсилає через UART, включаючи вивід вашої прошивки, результати вимірювань, діагностичні повідомлення. Це дуже корисно для відлагодження та перевірки роботи нового коду. Команду *idf.py monitor* можна комбінувати з іншими параметрами: налаштовувати швидкість передачі, фільтри виводу, зберігати дані у файл журналу.

Отже спочатку скомпілюємо та завантажимо прошивку на ESP32-DevKit-V1 командою *idf.py flash*. Одразу почнеться компіляція усього вихідного ходу, після чого буде видно процес завантаження прошивки на ESP32 (рис. 3.10).

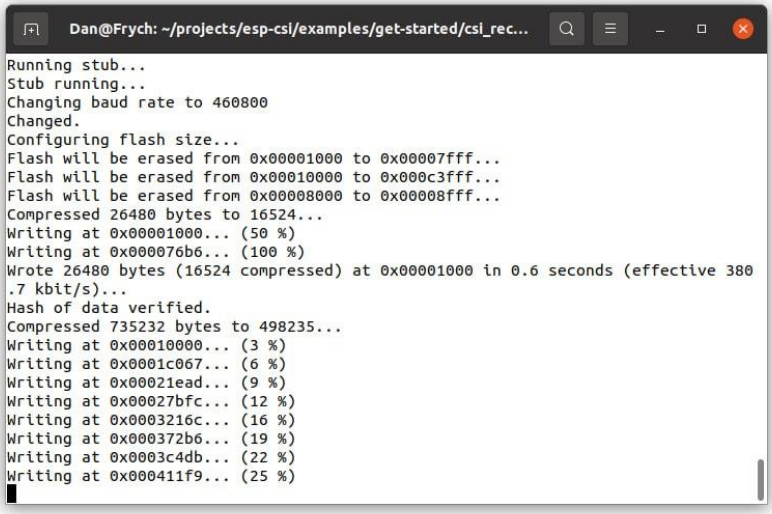


Рисунок 3.10 – Завантаження прошивки на мікроконтролер

Вже на цьому моменті мікроконтролер ESP32 підключився до вказаної мережі і почав збирати CSI-дані. Для того, щоб простежити за роботою ESP32 використаємо команду *idf.py monitor*, що буде виведе усі дані з послідовного порту у консоль та побачимо процес збору «сирих» даних (рис. 3.11).

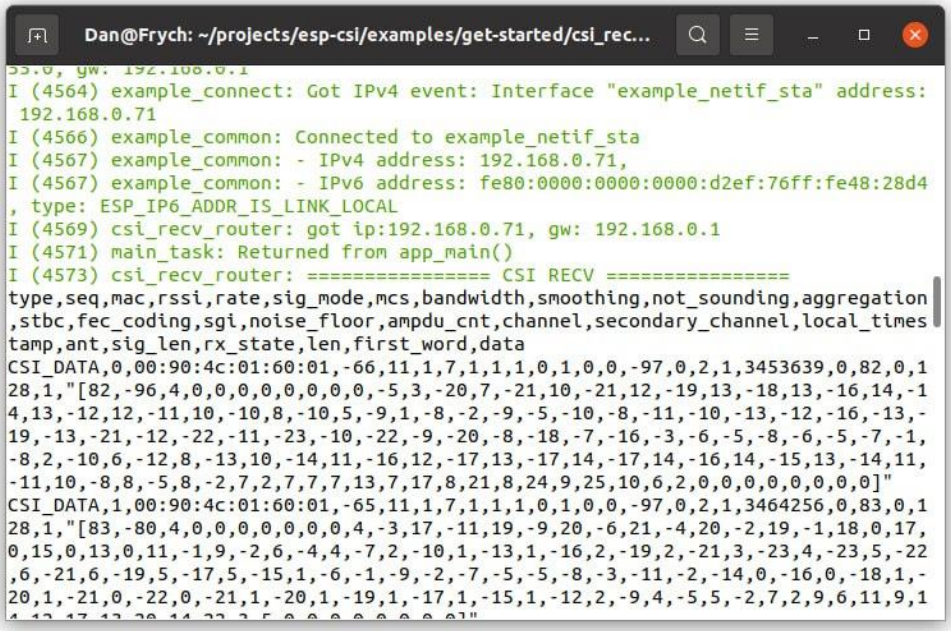


Рисунок 3.11 – Процес збору CSI-даних

Таким чином мікроконтролер ESP32 повністю підготовлено для збору CSI-даних мережі Wi-Fi, тепер треба створити БД для можливості зберігання зібраних даних та їх аналізу для виявлення об’єктів.

3.2.3 Створення бази даних

Для зберігання CSI-даних та можливості для подальшого їх аналізу у цій роботі буде використовуватися SQLite – реляційна БД, що зберігається у файлі прямо в репозиторії проєкту. Вона підтримує стандартний SQL-синтаксис і має багато корисних функцій, таких як індекси, тригери та транзакції.

Для зручної роботи з БД SQLite буде використовуватись DB Browser for SQLite – безкоштовна програма з відкритим кодом, яка надає зручний графічний інтерфейс для перегляду, редагування та керування БД SQLite (рис. 3.12). DB Browser for SQLite має такі переваги:

– зручний інтерфейс для перегляду та редагування даних у таблицях;

– можливість виконувати SQL-запити та переглядати їх результати;

– засоби для імпорту та експорту даних у різних форматах;

– візуальне створення та модифікація структури БД (таблиць, індексів, тригерів тощо).

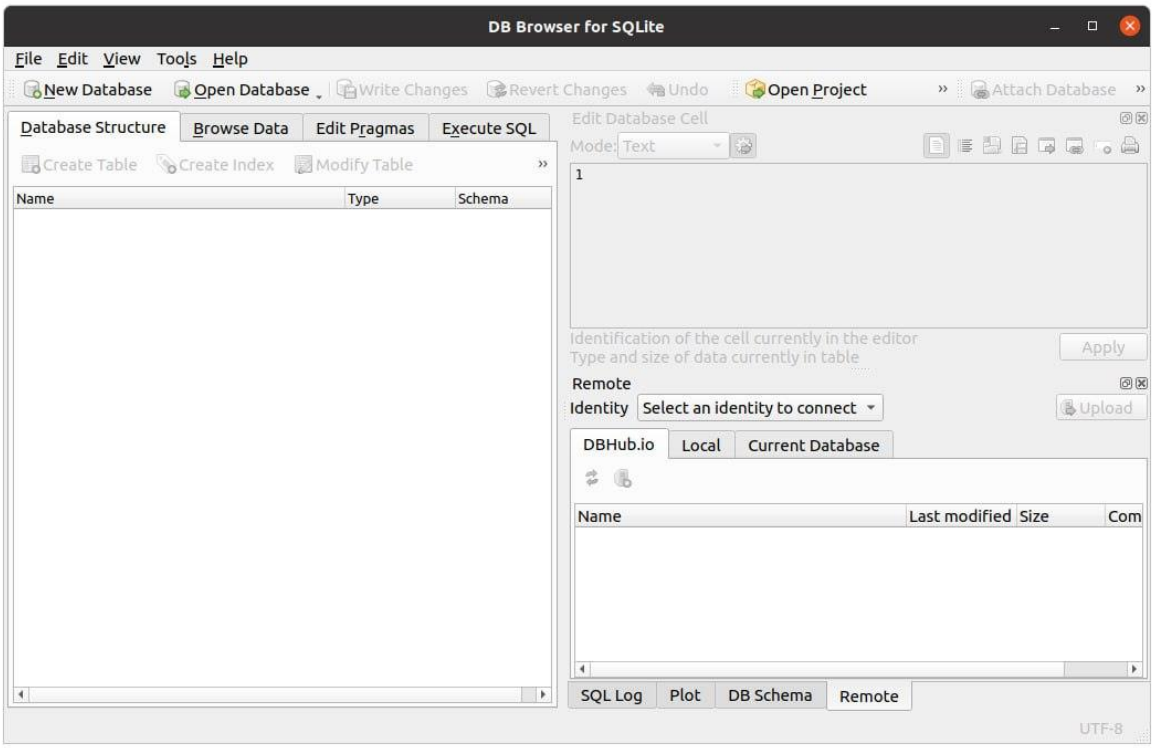


Рисунок 3.12 – DB Browser for SQLite

Вибір SQLite та DB Browser for SQLite обумовлений їх простотою, ефективністю та зручністю використання у невеликих проєктах. Ці технології дозволять зберігати та керувати даними, отриманими з пристрою ESP32-DevKit-V1, і забезпечать необхідну функціональність для роботи з БД.

Створення БД для ПЗ виявлення об'єктів на основі CSI мережі Wi-Fi проходило у кілька етапів:

1) встановлення та налаштування DB Browser for SQLite;
2) створення нової БД у форматі *.db*;
3) проєктування структури БД, визначення необхідних таблиць та їх полів;
4) створення таблиць у БД за допомогою візуального інтерфейсу DB Browser for SQLite.

Програма DB Browser for SQLite була завантажена з офіційного сайту та встановлена на комп'ютер. Після запуску програми було виконано базове налаштування, таке як вибір мови інтерфейсу та шляху для зберігання БД. Після чого, була створена нова БД *wfscan.db*.

На поточному етапі проєкту було визначено, що для зберігання CSI-даних достатньо однієї таблиці з полями усіх необхідних характеристик. Використання однієї таблиці для зберігання CSI-даних у цьому проєкті дозволить забезпечити більш швидку роботу з БД порівняно з використанням кількох пов'язаних таблиць.

У вкладці «Створити таблицю» було вказано назву таблиці та визначено структуру полів відповідно до проєктування (рис.3.13).

Name	Type	Schema
Tables (1)		
csi_data		CREATE TABLE "csi_data" ("object" T
object	TEXT	"object" TEXT
type	TEXT	"type" TEXT
id	INTEGER	"id" INTEGER
mac	NUMERIC	"mac" NUMERIC
rssi	INTEGER	"rssi" INTEGER
rate	INTEGER	"rate" INTEGER
sig_mode	INTEGER	"sig_mode" INTEGER
mcs	INTEGER	"mcs" INTEGER
bandwidth	INTEGER	"bandwidth" INTEGER
smoothing	INTEGER	"smoothing" INTEGER
not_sounding	INTEGER	"not_sounding" INTEGER
aggregation	INTEGER	"aggregation" INTEGER
stbc	INTEGER	"stbc" INTEGER
fec_coding	INTEGER	"fec_coding" INTEGER
sgi	INTEGER	"sgi" INTEGER
noise_floor	INTEGER	"noise_floor" INTEGER
ampdu_cnt	INTEGER	"ampdu_cnt" INTEGER
channel	INTEGER	"channel" INTEGER
secondary_channel	INTEGER	"secondary_channel" INTEGER
local_timestamp	INTEGER	"local_timestamp" INTEGER
ant	INTEGER	"ant" INTEGER
sig_len	INTEGER	"sig_len" INTEGER
rx_state	INTEGER	"rx_state" INTEGER
len	INTEGER	"len" INTEGER
first_word	INTEGER	"first_word" INTEGER
data	TEXT	"data" TEXT
Indices (0)		
Views (0)		
Triggers (0)		

Рисунок 3.13 – Таблиця *csi_data*

В таблиці було створено наступні поля:

– *object_type (INTEGER)* – тип об'єкта, для якого зберігаються CSI-дані;

– *id (INTEGER, PRIMARY KEY)* – унікальний ідентифікатор запису в таблиці;

– *mac (TEXT)* – MAC-адреса пристрою, з якого отримано CSI-дані;

– *rssi (INTEGER)* – показник рівня потужності прийнятого сигналу (RSSI);

– *rate (INTEGER)* – швидкість передачі даних, Мбіт/с;

– *sig_mode (TEXT)* – режим сигналізації (наприклад, *'legacy', 'ht', 'vht'*);

– *mcs (INTEGER)* – індекс схеми кодування модуляції (MCS);

– *bandwidth (INTEGER)* – ширина смуги пропускання каналу, МГц;

– *smoothing (INTEGER)* – параметр згладжування для фільтрації CSI-даних;

– *not_sounding (INTEGER)* – прапор відсутності зондування;

– *aggregation (INTEGER)* – прапор агрегації кадрів;

– *stbc (INTEGER)* – прапор використання технології просторово-часового блокового кодування (англ. Space–Time Block Code, STBC);

– *fec_coding (INTEGER*) – тип кодування з виправленням помилок (FEC);

– *sgi (INTEGER)* – прапор використання скороченого захисного інтервалу (SGI);

– *noise_floor (INTEGER)* – рівень шуму в середовищі;

– *ampdu_cnt (INTEGER)* – кількість агрегованих кадрів A-MPDU (Aggregated Mac Protocol Data Unit);

– *channel (INTEGER)* – номер основного каналу передачі;

– *secondary_channel (INTEGER)* – номер додаткового каналу (для режимів з шириною смуги 40 МГц або більше);

– *local_timestamp (INTEGER)* – локальна часова мітка отримання CSI-даних;

– *ant (INTEGER)* – номер антени, з якої отримано CSI-дані;

– *sig_len (INTEGER)* – довжина поля сигналізації в байтах;

– *rx_state (TEXT)* – стан прийому (*'ingress', 'ingress_ap'*);

– *len (INTEGER)* – загальна довжина CSI-даних в байтах;

– *first_word (INTEGER)* – перше слово CSI-даних;

– *data (BLOB)* – бінарні CSI-дані;

Після створення БД її можна буде використовувати для зберігання та керування даними, отриманими від пристрою ESP32-DevKit-V1, за допомогою відповідного програмного забезпечення, написаного на Python.

3.2.4 Створення програмного застосунку

Для реалізації графічного інтерфейсу користувача та візуалізації CSI-даних було розроблено програмний застосунок на мові Python з використанням бібліотеки PyQt.

На початковому етапі за допомогою вбудованого в PyQt інструменту Qt Designer (рис. 3.14) було розроблено макет інтерфейсу користувача. Цей інструмент дозволяє візуально розміщувати елементи інтерфейсу, такі як вікна, кнопки, поля введення та інші компоненти, з можливістю налаштування їх властивостей та функцій.

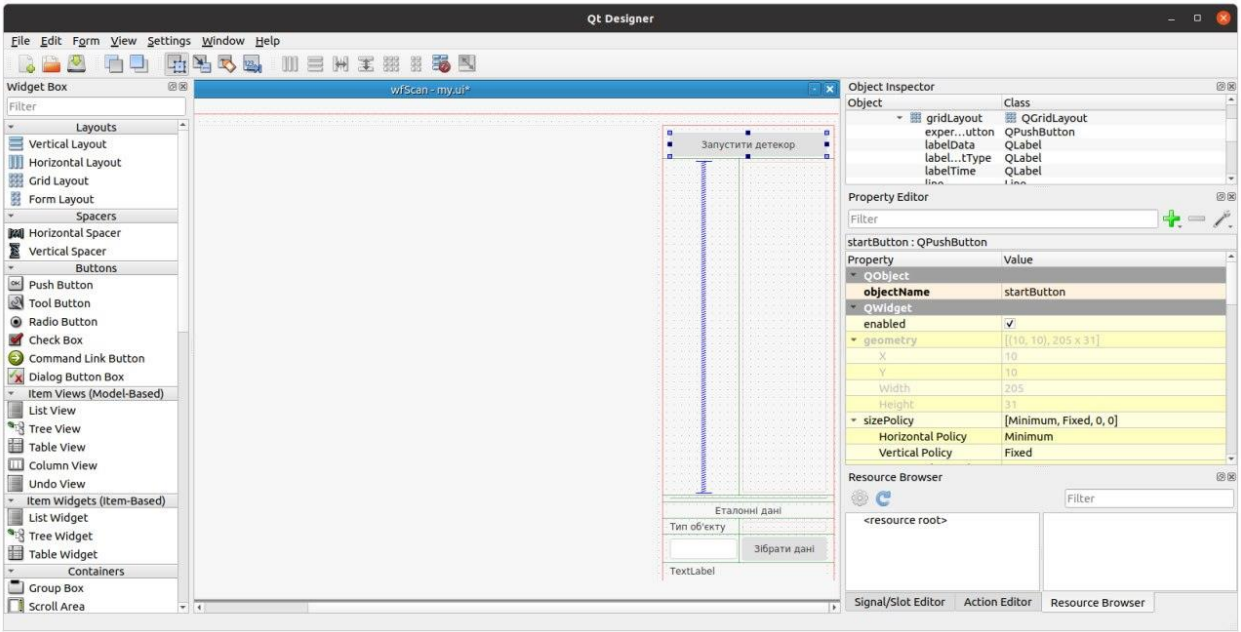


Рисунок 3.14 – Процес створення графічного інтерфейсу користувача в Qt Designer

Одним з ключових компонентів інтерфейсу став вбудований графік для візуалізації CSI-даних у режимі реального часу. Цей графік було інтегровано у вікно застосунку за допомогою спеціального віджету PlotWidget з бібліотеки

PyQtGraph Для забезпечення зручної роботи з графіком було реалізовано функціональність масштабування та прокручування графіку.

Крім графіка, інтерфейс користувача містить блок для збору еталонних даних у правій нижній частині екрану, в якій є поле для введення інформації про тип досліджуваного об'єкта та кнопки для запуску самого процесу збору даних (рис. 3.15). У верхній частині розміщено кнопку запуску застосунку для виявлення об’єктів.

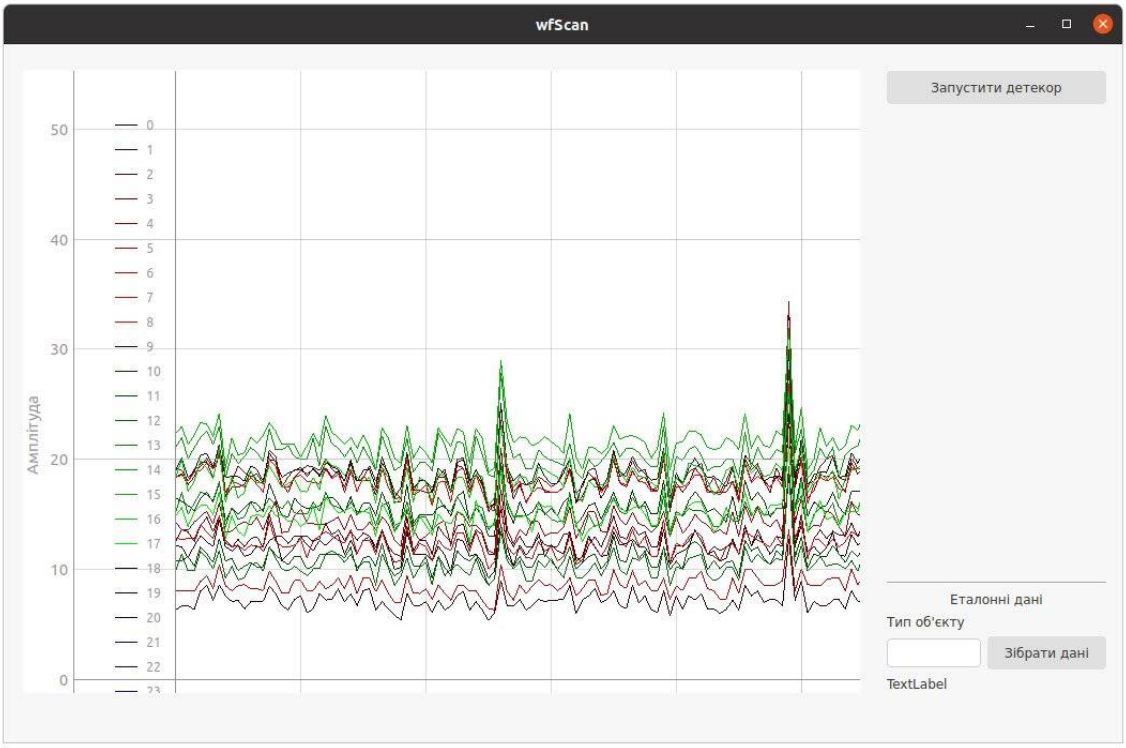


Рисунок 3.15 – Графічний інтерфейс застосунку

Після розробки макету інтерфейсу в Qt Designer був згенерований файл графічного інтерфейсу *.ui*. Після чого він був підключений до коду на Python, який включає класи та методи для створення та взаємодії з усіма елементами графічного інтерфейсу. Цей код було інтегровано в основний код застосунку, де були реалізовані логіка обробки CSI-даних, взаємодія з БД, алгоритми розпізнавання об'єктів та інші функціональні можливості.

Використання PyQt дозволило створити зручний та інтуїтивно зрозумілий графічний інтерфейс користувача, який забезпечує візуалізацію CSI-даних, керування процесами збору еталонних даних та розпізнавання типу об'єктів, а також налаштування різних параметрів системи. Завдяки

кросплатформній природі PyQt, застосунок може бути запущений на різних операційних системах без модифікацій коду.

Як було описано у розділі 2, робота системи виявлення об’єктів поділяється на два основних етапи: збір еталонних даних та розпізнавання об'єктів. Відповідно основних функцій у ПЗ можна виділити також дві:

1) функція запису еталонних даних у БД;
2) функція порівняння вхідних даних з еталонними.

Розглянемо принципи роботи основних функцій застосунку.

Функція збору еталонних CSI-даних реалізована наступним чином: вона прикріплена до обробника події кнопки «Зібрати дані» в графічному інтерфейсі, створеному за допомогою PyQt. Коли користувач хоче зібрати еталонні CSI-дані, він вводить назву об'єкта, який буде досліджуватися, у спеціальне текстове поле для вводу, а потім натискає кнопку «Зібрати дані».

Спочатку ця функція підключається до БД SQLite, де будуть зберігатися еталонні CSI-дані. Потім встановлюється таймер на 60 секунд для визначення проміжку часу збору даних. Після цього починається безперервне зчитування CSI-даних з послідовного порту комп'ютера, куди вони передаються з WiFi-мережі, в зоні дії якої знаходиться об'єкт дослідження.

Отримані CSI-дані проходять попередню обробку програмним забезпеченням для виділення корисної інформації. Усереднені еталонні CSI-дані разом з міткою, що ідентифікує досліджуваний об'єкт (введеною користувачем у текстове поле), зберігаються в БД SQLite. Після завершення 60-секундного таймера збір еталонних даних припиняється, і зібрані CSI-шаблони для даного об'єкта стають доступними для використання в режимі розпізнавання. Блок-схему алгоритму функції запису еталонних даних наведено на рис. 3.16.

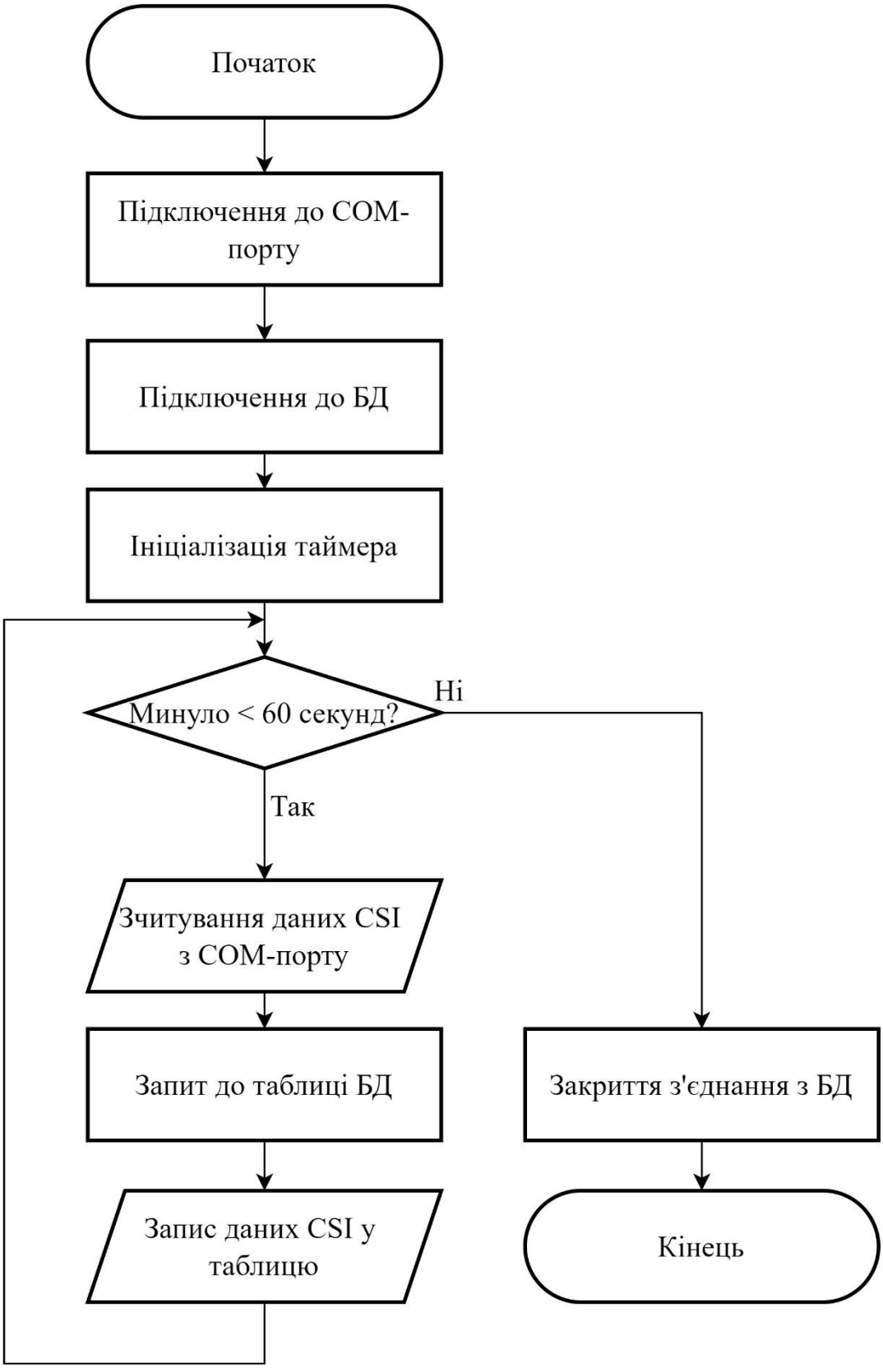


Рисунок 3.16 – Алгоритм функції запису еталонних даних

Функція порівняння вхідних даних з еталонними прикріплена до обробника подій кнопки «Запустити детектор» в графічному інтерфейсі. Після натискання на цю кнопку відбувається підключення до послідовного порту комп'ютера для зчитування вхідних CSI-даних та підключення до БД SQLite, де зберігаються еталонні CSI-дані.

Далі починається безперервне зчитування вхідних CSI-даних з послідовного порту в режимі реального часу. Отримані CSI-дані порівнюються з усіма еталонними даними, що зберігаються в БД. Для кожного типу об'єкта з БД підтягуються відповідні еталонні CSI-дані, позначені міткою (маркером) цього типу об'єкта.

Процес порівняння полягає в наступному: береться кожна піднесуча вхідних CSI-даних та кожна піднесуча еталонних CSI-даних, і обчислюється коефіцієнт схожості між ними. Якщо середній коефіцієнт схожості всіх піднесучих складає не менше ніж 95 %, вважається, що об'єкт виявлено.

Коефіцієнт схожості розраховується шляхом порівняння амплітуд та фаз піднесучих вхідних та еталонних CSI-даних. Чим ближче ці значення для кожної піднесучої, тим вищий коефіцієнт схожості. Якщо схожість всіх піднесучих перевищує поріг 95 %, система вважає, що поточні вхідні CSI-дані відповідають еталонним даним для певного типу об'єкта.

В такому випадку в графічному інтерфейсі з'являється діалогове вікно сповіщення (QMessageBox) від PyQt, в якому вказано, який саме об'єкт було виявлено, відповідно до мітки еталонних CSI-даних з найвищим коефіцієнтом схожості. Блок-схему алгоритму функції порівняння вхідних даних з еталонними наведено на рис. 3.17.

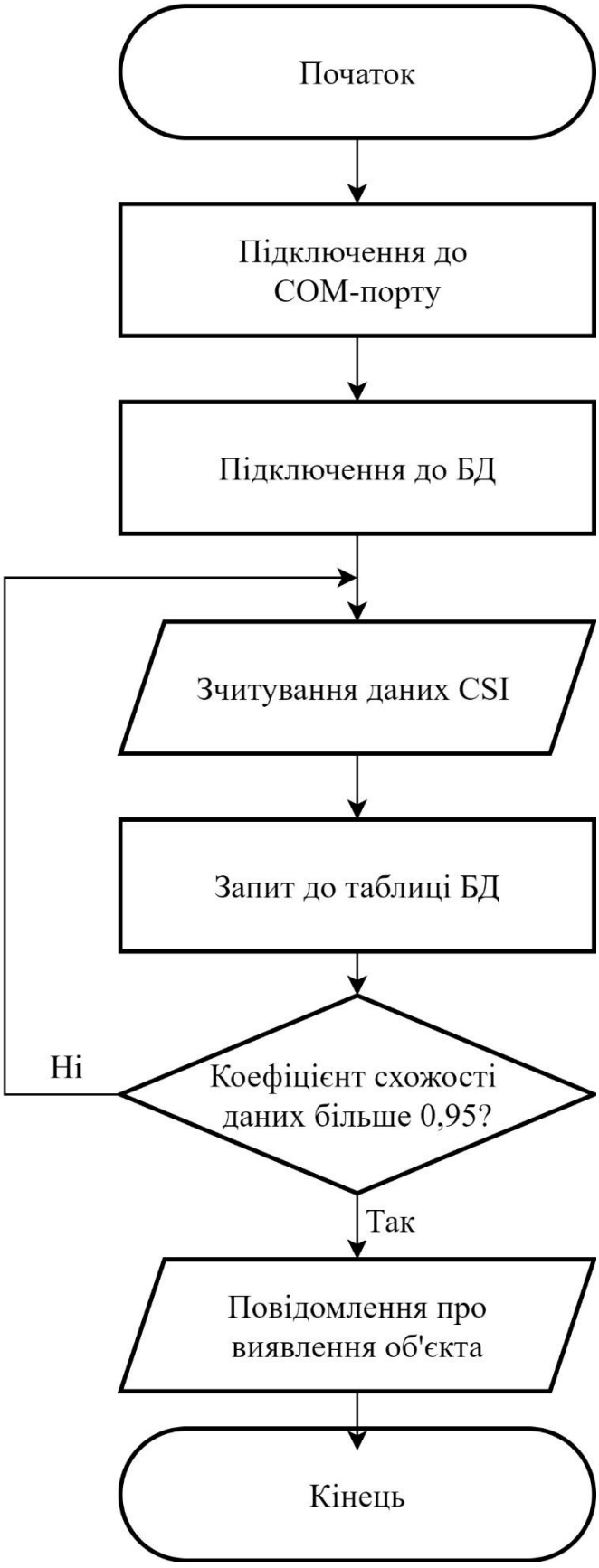


Рисунок 3.17 –Алгоритм функції порівняння вхідних даних з еталонними

У результаті було розроблено повнофункціональний застосунок з графічним інтерфейсом користувача для системи виявлення об'єктів

на основі аналізу CSI-даних WiFi-мережі. Застосунок створено з використанням мови програмування Python та бібліотеки PyQt.

Крім двох основних функцій – збору еталонних CSI-даних та розпізнавання об'єктів – у коді застосунку також присутні другорядні функції, які забезпечують коректну роботу системи. Наприклад, функція зчитування даних з послідовного порту комп'ютера та функція парсингу отриманих CSI-даних для виділення корисної інформації.

Реалізація всіх функцій – як основних, так і другорядних – наведена в додатку В. Вони забезпечують повноцінну роботу системи виявлення об'єктів – від отримання вхідних CSI-даних до порівняння їх з еталонними шаблонами та відображення результатів розпізнавання в графічному інтерфейсі.

3.3 Експериментальні дослідження та тестування застосунку

3.3.1 Проведення експериментальних досліджень

Для дослідження та тестування розробленої системи виявлення об'єктів на основі CSI-даних мережі Wi-Fi було обрано вісім різних типів матеріалів: метал, скло, тканина, пластик, сіль, борошно та селітра. Вибір саме цих матеріалів пов'язаний з тим, що в майбутньому дана система може використовуватися як засіб виявлення небезпечних предметів, зокрема саморобних вибухових пристроїв, компонентами яких можуть бути зазначені матеріали.

Експерименти проводилися у приміщенні, де були створені контрольовані умови для дослідження (рис. 3.18). В якості маршрутизатора використовувався другий ноутбук, на якому було ввімкнено точку доступу Wi-Fi. Дослідження проходили в однакових умовах, за відключення електроживлення, щоб уникнути будь-яких перешкод або додаткових сигналів від сусідніх маршрутизаторів. Таким чином, були забезпечені ідеальні умови для проведення експериментів.

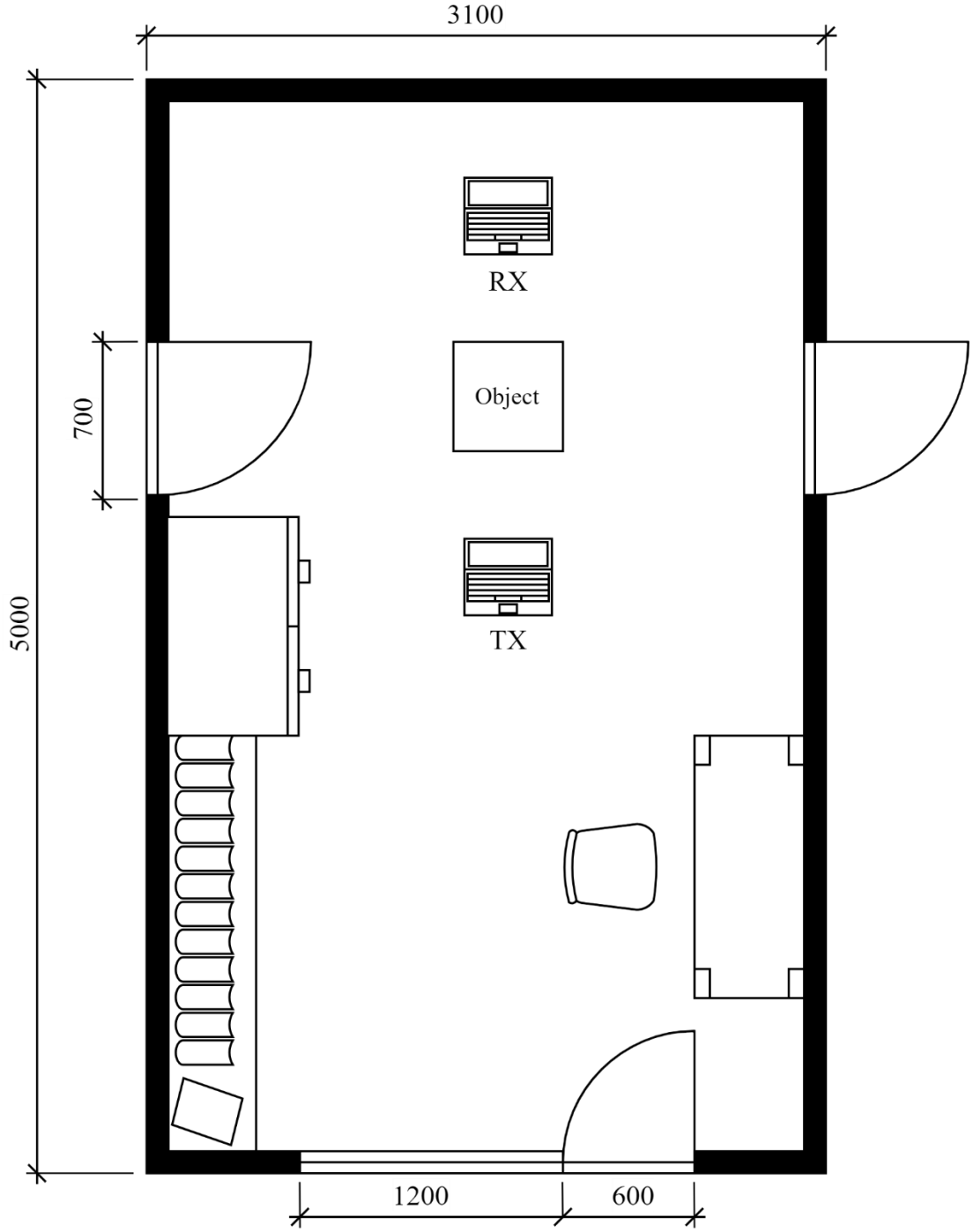
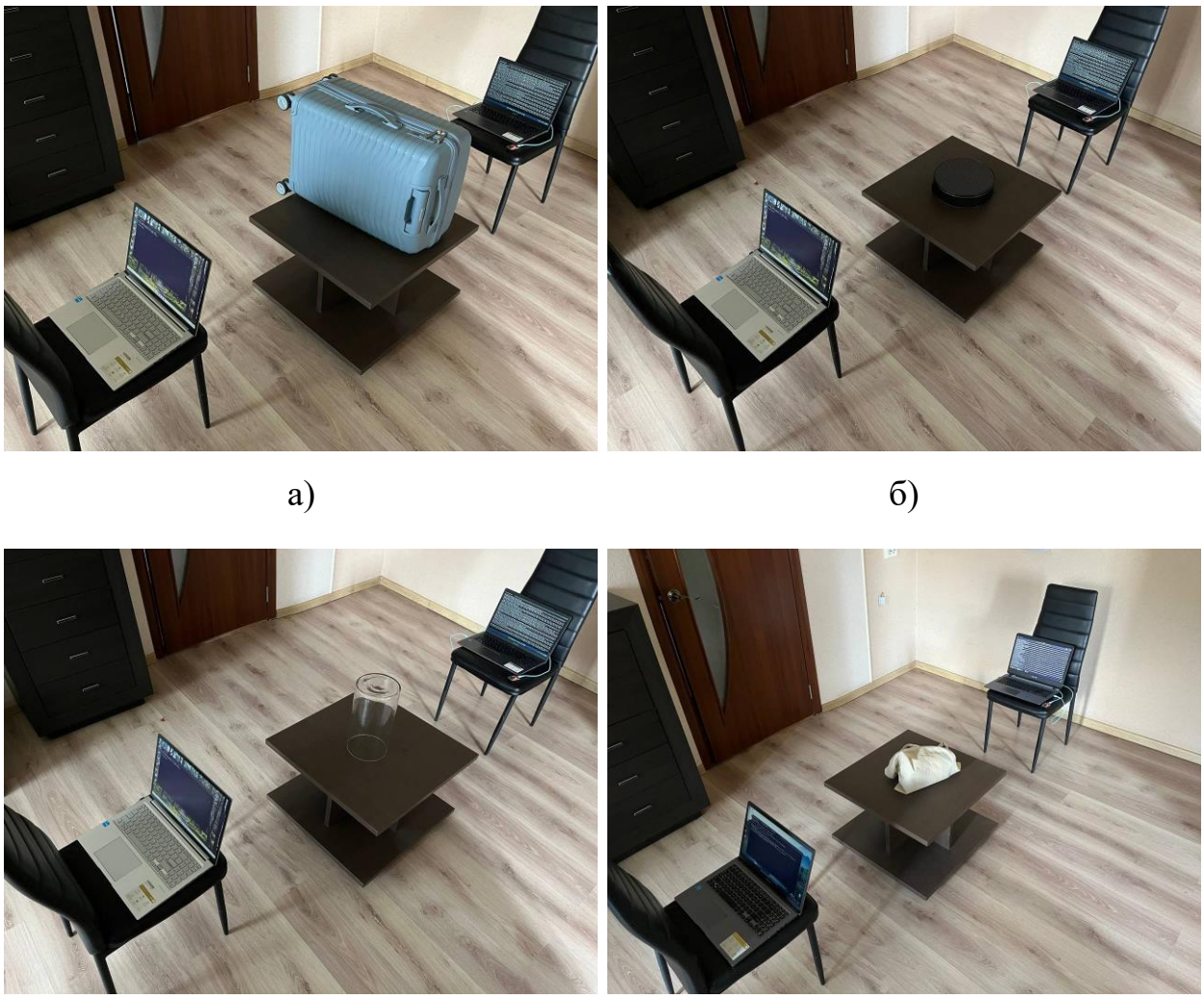


Рисунок 3.18 – Схема приміщення для експериментів

Під час експериментів кожен з матеріалів розміщувався в зоні дії WiFi-мережі, на відстані по 50 см між передавачем та приймачем на спеціальній дерев’яній підставці, висота якої складає 40 см. Приклад проведення експериментів зображено на рис. 3.19.

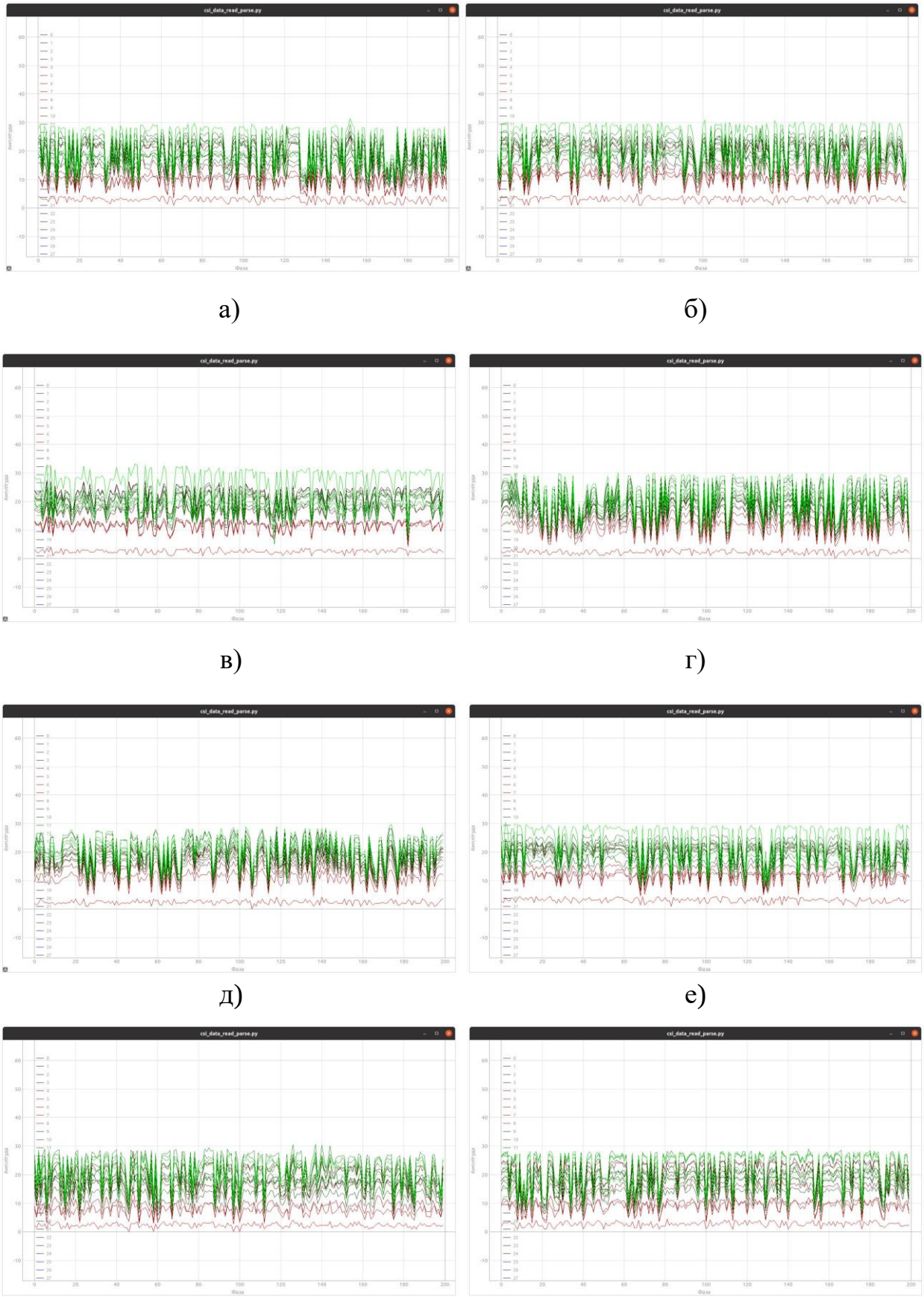


а) б)

в) г)

Рисунок 3.19 – Проведення досліджень з різними матеріалами: а – пластик; б – метал; в – скло; г – борошно

Система збирала CSI-дані, що характеризували вплив цього матеріалу на радіосигнал. Було проаналізовано зміну сили сигналу (англ. Received Signal Strength Indicator, RSSI) в залежності від типу об'єкта, а також досліджено і побудовано графіки CSI для різних типів об'єктів (рис. 3.20).



а) б)

в) г)

д) е)

ж) з)

Рисунок 3.20 – Візуалізація CSI-даних: а – метал; б – скло; в – тканина; г – пластик; д – картон; е – сіль; ж – селітра; з – борошно

Блок-схема алгоритму проведення експерименту та збору еталонних даних зображена на рис. 3.21.

Рисунок 3.21 – Алгоритм збору еталонних даних

Отримані експериментальні дані зберігалися як еталонні зразки для подальшого використання в алгоритмі розпізнавання об'єктів.

Результати експериментів, які наведено у табл. 3.1, були проаналізовані та використані для налаштування та оптимізації алгоритмів обробки та порівняння CSI-даних, що дозволило досягти вищої ефективності системи у виявленні потенційно небезпечних об'єктів на основі аналізу характеристик радіосигналу.

Таблиця 3.1 – Зміна RSSI в залежності від об’єкта

Об’єкт	**RSSI**	**Відмінність CSI-даних від ідеальних умов, %**
Метал	–51	27,5
Скло	–47	17,5
Пластик	–44	10,0
Тканина	–45	12,5
Картон	–48	20,0
Борошно	–42	5,0
Сіль	–43	7,5
Селітра	–44	10,0

Як і очікувалося, результати дослідження показують, що найбільший вплив на зміни в характеристиках CSI-даних мають металеві та скляні об'єкти. Це пояснюється тим, що метал та скло є хорошими відбивачами електромагнітних хвиль, що призводить до значних перешкод та спотворень сигналу. Також виявилося що картон теж має досить сильний вплив на зміну RSSI.

Рівень впливу пластикових та тканинних об'єктів на CSI-дані є приблизно однаковим і помірним. Це можна пояснити тим, що пластик та тканина є менш щільними матеріалами, що дозволяє частині сигналу проходити крізь них з меншими спотвореннями.

Що стосується солі, борошна та селітри, то їх вплив на CSI-дані є відносно незначним. Ці речовини є порошкоподібними та розсіюють сигнал, але не створюють суттєвих перешкод через свою низьку щільність та непровідні властивості. Однак, у випадку великих скупчень або висококонцентрованих розчинів, їх вплив може бути більш помітним.

3.3.2 Тестування АПЗ

Після успішного збору еталонних даних було здійснено комплексне тестування розробленої системи виявлення об'єктів на основі аналізу CSI-даних WiFi-мережі. Метою тестування була перевірка працездатності та точності виявлення різних типів об'єктів в реальних умовах.

Для кожного типу об'єкта (метал, скло, тканина, пластик, сіль, борошно та селітра) було проведено по 10 окремих тестувань. Під час тестування об'єкт розміщувався між ESP32 та ноутбуком на фіксованій відстані, після чого система збирала CSI-дані протягом однієї хвилини та намагалася визначити тип об'єкта шляхом порівняння з еталонними зразками.

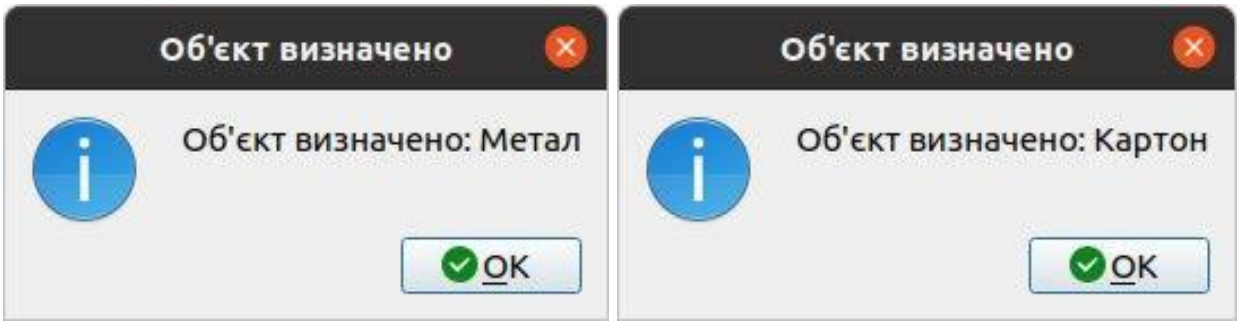
Результати тестування продемонстрували, що система виявляє об'єкти з високою, але не ідеальною точністю. У середньому, система успішно визначала тип об'єкта в 73,75 % випадків; результати тестувань наведені в табл. 3.2. Найвищі показники точності спостерігалися для металевих та скляних об'єктів, тоді як тканина та пластик викликали деякі труднощі у розпізнаванні через схожість їхніх характеристик CSI-даних.

Таблиця 3.2 – Кількість успішних фактів виявлення об'єктів

Об’єкт	**Кількість фактів успішного виявлення об’єктів, %**
Метал	90
Скло	80
Пластик	60
Тканина	60
Картон	80
Борошно	70
Сіль	70
Селітра	80

Під час тестування траплялися випадки, коли система невірно визначала тип об'єкта або не могла його ідентифікувати. Це могло бути викликано різними факторами, такими як специфічна форма або орієнтація об'єкта, а також вплив зовнішніх перешкод або завад.

Нижче наведено скріншоти з інтерфейсу застосунку, які демонструють приклади успішного виявлення різних типів об'єктів.



а) б)

Рисунок 3.22 – Повідомлення про визначення об'єкту: а – метал; б – картон

Загалом, проведені тести підтвердили працездатність розробленої системи та її здатність точно виявляти різні типи об'єктів на основі аналізу CSI-даних WiFi-мережі. Система продемонструвала стабільну роботу в контрольованих умовах та може бути використана для подальших досліджень та розробок у галузі виявлення об'єктів на основі CSI-даних.

3.4 Методи покращення системи виявлення об’єктів на основі CSI-даних мережі Wi-Fi

Одним з найбільш перспективних методів вдосконалення системи виявлення об'єктів на основі CSI мережі Wi-Fi є застосування алгоритмів машинного навчання, зокрема глибоких нейронних мереж. Використання алгоритмів машинного навчання дозволить значно підвищити точність розпізнавання та класифікації об'єктів, що знаходяться в зоні дії WiFi-мережі.

Застосування методів машинного навчання, таких як нейронні мережі, дозволить системі автоматично виявляти закономірності та ознаки в CSI-даних, які характеризують різні типи об'єктів. Це зменшить необхідність ручного збору та внесення еталонних даних для кожного нового типу об'єкта, що значно спростить процес налаштування та розширення можливостей системи.

Окрім загального виявлення об'єктів, одним з найбільш критично важливих напрямків застосування вдосконалених систем на основі CSI та глибокого навчання є ідентифікація потенційно небезпечних предметів, таких як вибухові пристрої.

Завдяки здатності глибоких нейронних мереж розпізнавати складні просторово-часові закономірності в даних CSI, вони зможуть не лише класифікувати окремі компоненти, але й об'єднувати їх у визначені категорії об'єктів. Наприклад, якщо виявлені компоненти, характерні для вибухових речовин, знаходяться в радіусі розміру чемодана або десь у скупченні людей, нейронна мережа зможе розпізнати їх як єдиний об'єкт та позначити як «можливу небезпеку».

Такі сигнали тривоги дозволять охоронцям швидко та ефективно перевіряти підозрілі речі без надмірних перевірок усіх пасажирів. У разі підтвердження загрози, така система зможе сповістити відповідні служби про місцезнаходження та характер виявленого об'єкту.

Безумовно, перед практичним впровадженням системи мають пройти ретельне тестування та сертифікацію відповідно до вимог безпеки. Однак потенціал виявлення небезпечних об'єктів на основі аналізу CSI за допомогою машинного навчання є надзвичайно великим і здатен підвищити рівень безпеки в аеропортах, на вокзалах та інших важливих місцях великого скупчення людей.

Висновки до розділу 3

У розділі 3 було детально розглянуто апаратно-програмне забезпечення розробленої системи виявлення об'єктів на основі CSI-даних WiFi-мережі, а також проведено експериментальні дослідження, тестування створеного застосунку та розглянуто можливі шляхи покращення системи.

Спочатку було описано апаратну частину системи на базі мікроконтролера ESP32-DevKit-V1 для збору CSI-даних та наведено схему підключення компонентів.

Далі представлено програмну реалізацію системи з деталізацією архітектури, основних модулів та компонентів. Детально розглянуто процес налаштування та конфігурації мікроконтролера ESP32 з використанням

ESP-IDF для збору CSI-даних за допомогою модифікованих кодів з репозиторію *csi-esp* від Espressif.

Окремо описано створення БД та розробка застосунку для комп'ютера на Python з GUI, реалізованим за допомогою PyQt5. Застосунок призначений для запису, візуалізації та аналізу отриманих CSI-даних.

Після розгляду апаратно-програмної реалізації, наведено результати експериментальних досліджень та тестування створеного застосунку в різних умовах для перевірки його працездатності та ефективності.

Наприкінці розділу запропоновано потенційні методи покращення розробленої системи виявлення об'єктів на основі CSI мережі Wi-Fi для підвищення точності, чутливості та швидкодії.

Таким чином, у розділі 3 комплексно розглянуто всі складові апаратно-програмного забезпечення системи, проведено її тестування та окреслено можливі шляхи вдосконалення в майбутньому.

ВИСНОВКИ

Порівняно з існуючими аналогами, розроблений підхід використання даних CSI для виявлення об'єктів у середовищі має такі переваги: безконтактність, відсутність потреби в додаткових датчиках, можливість виявлення прихованих об'єктів, низька вартість впровадження на базі стандартного обладнання Wi-Fi. Новизна полягає у розробці покращених алгоритмів обробки CSI для підвищення точності класифікації та визначення відстані до об'єктів.

Практична цінність результатів роботи полягає в тому, що вони дозволяють створювати економічно вигідні системи моніторингу навколишнього середовища, виявлення вторгнень, відстеження руху та інші подібні рішення, використовуючи лише наявну інфраструктуру бездротових мереж без додаткових витрат на спеціалізовані датчики. Така система може знайти широке застосування у сферах безпеки, промисловості, міського планування для розумних міст тощо.

Основні результати виконання кваліфікаційної роботи відповідають поставленим завданням:

– проаналізовано методи та засоби отримання, обробки та аналізу інформації про стан каналу (CSI) в мережах Wi-Fi за її характеристиками;

– досліджено існуючі підходи щодо використання CSI для виявлення присутності об'єктів у середовищі, їх класифікації та визначення відстані;

– розроблено апаратну та програмну платформу для реалізації системи аналізу CSI;

– досліджено зміну CSI-даних в залежності від матеріалу об’єкту, що створює перешкоду;

– проведено тестування розробленого АПЗ;

– наведено приклади можливих покращень розробленого рішення.

Подальший розвиток предмету дослідження може включати інтеграцію методів штучного інтелекту, зокрема глибокого навчання, для підвищення

точності та ефективності обробки даних CSI. Також перспективним є застосування методів посилення навчання для оптимізації процесів прийняття рішень щодо локалізації об'єктів на основі накопичених даних.

У майбутньому розроблену систему виявлення об'єктів за допомогою Wi-Fi можна впроваджувати у сферах безпеки, промисловості, «розумного» міського середовища для моніторингу, контролю та виявлення аномалій. Необхідно забезпечити навчання системи на великих обсягах реальних і змодельованих даних для різних сценаріїв застосування. Використання методів передового досвіду штучного інтелекту дозволить підвищити ефективність впровадженого рішення.

Результати кваліфікаційної роботи було представлено на XXVI Всеукраїнській науково-практичній конференції «Могилянські читання – 2023» (Миколаїв, 06–10 листопада 2023 р.) та на фіналі Конкурсу стартапів «Startup BSNU 2024» (Миколаїв, 18 квітня 2024 р.).

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. 60GHz mobile imaging radar / Y. Zhu et al. *Mobile Computing Systems and Applications (HotMobile'15) :* Proc. of the 16th Internat. Workshop, Santa Fe, New Mexico, USA. New York, NY, USA, 2015. DOI: 10.1145/2699343.2699363.

2. A survey on behavior recognition using WiFi channel state information / S. Yousefi et al. *IEEE Communications Magazine*. 2017. Vol. 55, no. 10. P. 98–104. DOI: 10.1109/MCOM.2017.1700082.

3. Abazorius A. New system allows for high-accuracy, through-wall, 3-D motion tracking. *Massachusetts Institute of Technology News* : web site. Publ. Dec. 11, 2013. URL: https://news.mit.edu/2013/new-system-allows-for-high-accuracy-through-wall-3-d-motion-tracking-1211 (Last accessed: 13.05.2024).

4. Adib F., Katabi D. See through walls with WiFi! *SIGCOMM'13:* Proc. of the ACM Conf., Hong Kong China. New York, NY, USA, 2013. DOI: 10.1145/2486001.2486039.

5. Capturing the human figure through a wall / F. Adib et al. *ACM Transactions on Graphics*. 2015. Vol. 34, no. 6. P. 1–13. DOI: 10.1145/2816795.2818072.

6. Channel state information from pure communication to sense and track human motion: A survey / M. A. A. Al-qaness et al. *Sensors*. 2019. Vol. 19, no. 15. P. 3329. DOI: 10.3390/S19153329 .

7. CSI-MIMO: Indoor Wi-Fi fingerprinting system / Y. Chapre et al. *Local Computer Networks (LCN) :* Proc. of the 2014 IEEE 39th Conf., Edmonton, AB, 08–11 September 2014. DOI: 10.1109/LCN.2014.6925773.

8. Ding J., Wang Y. WiFi CSI-based human activity recognition using Deep Recurrent Neural Network. *IEEE Access.* 2019. Vol. 7. P. 174257–174269. DOI: 10.1109/ACCESS.2019.2956952.

9. ESP-IDF Programming Guide v5.0.2 documentation. *Technical Documents | Espressif Systems.* URL: https://docs.espressif.com/projects/esp-idf/en/v5.0.2/esp32/get-started/index.html (Last accessed: 01.06.2024).

10. Espressif systems | esp-csi. *GitHub*. URL: https://github.com/espressif/esp-csi/tree/master/examples/get-started (Last accessed: 13.06.2024).

11. Hernandez S. M. Lightweight and standalone IoT based WiFi sensing for active repositioning and mobility. Richmond, VA, 2020.

12. Huang D., Nandakumar R., Gollakota S. Feasibility and limits of Wi-Fi imaging. *Embedded Network Sensor Systems, Memphis Tennessee (SenSys'14) :* Proc. of the 12th ACM Conf., New York, NY, USA, 2014. DOI: 10.1145/2668332.2668344.

13. Kim S., Kim T., Kim S. Human activity recognition and prediction based on Wi-Fi channel state information and machine learning. *Artificial Intelligence in Information and Communication (ICAIIC) :* Proc. of the 2019 International Conference, Okinawa, Japan, 2019. P. 418–422.

14. Krumbein A. Understanding the basics of MIMO communication technology. 2016. 12 p. URL: https://www.rfmw.com/data/swa-mimo-basics.pdf (Last accessed: 13.05.2024).

15. Locating rogue access point using fine-grained channel information / C. Wang et al. *IEEE Transactions on Mobile Computing*. 2017. Vol. 16, no. 9. P. 2560–2573. DOI: 10.1109/TMC.2016.2629473.

16. Ma Y. Improving WiFi sensing and networking with channel state information. 2019. 36 p. URL: https://scholarworks.wm.edu/etd/1593091976 (Last accessed: 13.05.2024).

17. Ma Y., Zhou G., Wang S. WiFi sensing with channel state information. *ACM Computing Surveys*. 2019. Vol. 52, no. 3. P. 1–36. DOI: 10.1145/3310194.

18. Object detection method and system based on WIFI and storage medium : patent CN110287774 China : G06K 9/00. No. 201910399370.X ; applied on 14.05.2019 ; published on 27.09.2019. URL: https://patentscope.wipo.int/search/ru/detail.jsf?docId=CN254163117&_fid=WO2020228324 (Last accessed: 13.05.2024).

19. Ralston T. S., Charvat G. L., Peabody J. E. Real-time through-wall imaging using an ultrawideband multiple-input multiple-output (MIMO) phased array radar system. *2010 IEEE Internat. Symp. on Phased Array Systems and Technology (ARRAY 2010)*, Waltham, MA, 12–15 Oct. 2010. 2010. DOI: 10.1109/ARRAY.2010.5613314.

20. Tool release / D. Halperin et al. *ACM SIGCOMM Computer Communication Review*. 2011. Vol. 41, no. 1. P. 53. DOI: 10.1145/1925861.1925870.

21. Using Channel State Information for Tamper Detection in the Internet of Things / I. E. Bagci et al. *The 31st Annual Computer Security Applications Conference,* Los Angeles, CA, USA, 07–11 December 2015. New York, USA, 2015. DOI: 10.1145/2818000.2818028.

22. Wi-Fi indoor radar: patent WO2017052875 : G01S 13/87 2006.1. Publ. 30.03.2017. URL: https://patentscope.wipo.int/search/en/detail.jsf?docId=WO2017052875&_cid=P10-LVVI9T-69168-1 (Last accessed: 13.05.2024).

23. Wi-Fi radar detection using synchronized wireless access point : patent EP3841397 : G01S 13/00. No. 19854164 ; applied on 25.08.2019 ; published on 30.06.2021. URL: https://patentscope.wipo.int/search/en/detail.jsf?docId=EP328951632&_cid=P10-LVVI9T-69168-1 (Last accessed: 13.05.2024).

24. Xie Y., Li Z., Li M. Precise power delay profiling with commodity WiFi. *Mobile Computing and Networking (MobiCom'15) :* Proc. of the 21th Annual Internat. Conf., Paris France. New York, NY, USA, 2015. DOI: 10.1145/2789168.2790124.

25. Zou H., Zhou Yu,, Yang J., Gu W., Xie L., Spanos C. J. FreeDetector: Device-free occupancy detection with commodity WiFi. *Proc. of the 2017 IEEE International Conference on Sensing, Communication and Networking (SECON Workshops),* San Diego, CA, USA, 2017. P. 1–5. DOI: 10.1109/SECONW.2017.8011040.

26. Zou H., Zhou Yu,, Yang J., Spanos C. J. Towards occupant activity driven smart buildings via WiFi-enabled IoT devices and deep learning. *Energy and Buildings*. 2018. Vol. 177. P. 12–22. DOI: 10.1016/j.enbuild.2018.08.010.

27. Конкурс стартапів «Startup BSNU 2024» : запис трансляції. Опубл. 18.04.2024. URL: https://www.youtube.com/watch?v=SWRwWHd3tNg (дата звернення: 01.06.2024).

28. Фрич Д. О., Журавська І. М. Виявлення небезпечних предметів за допомогою мережі Wi-Fi. *Могилянські читання – 2023* : тези доп. XXVI Всеукр. наук.-практ. конф. Миколаїв, 6–10 листоп. 2023 р. Миколаїв : Чорном. нац. ун-т ім. Петра Могили, 2023. С. 446–447.

ДОДАТОК А

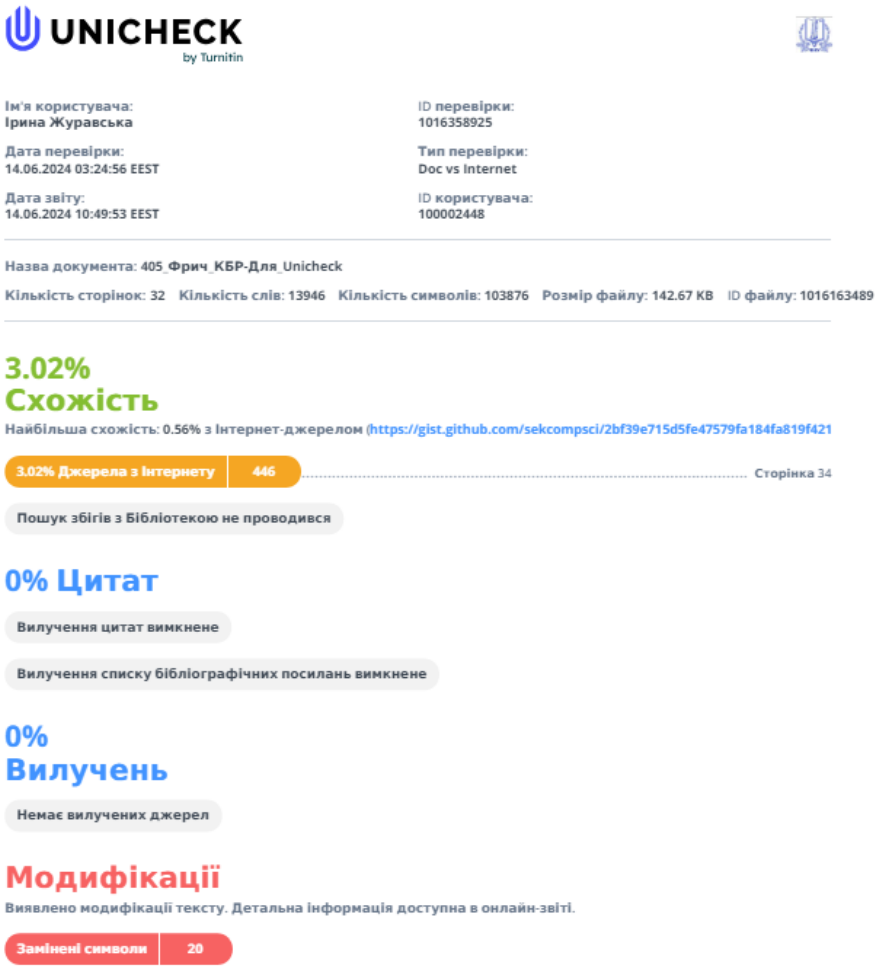
Довідка про перевірку на унікальність пояснювальної записки

бакалаврської кваліфікаційної роботи на тему:
«Система виявлення об'єктів на основі CSI мережі Wi-Fi»

студента спеціальності 123 «Комп'ютерна інженерія», 405 групи
Фрич Дан Олегович
прізвище, ім'я, по-батькові

Перевірку тексту здійснено сервісом: онлайн-сервіс Unicheck

Результат перевірки тексту бакалаврської кваліфікаційної роботи: схожість складає 3,02%.


UNICHECK
by Turnitin

Ім'я користувача: Ірина Журавська
ID перевірки: 1016358925
Дата перевірки: 14.06.2024 03:24:56 EEST
Тип перевірки: Doc vs Internet
Дата звіту: 14.06.2024 10:49:53 EEST
ID користувача: 100002448

Назва документа: 405_Фрич_КБР-Для_Unicheck
Кількість сторінок: 32 Кількість слів: 13946 Кількість символів: 103876 Розмір файлу: 142.67 KB ID файлу: 1016163489

3.02%
Схожість
Найбільша схожість: 0.56% з Інтернет-джерелом (https://gist.github.com/sekcompsci/2bf39e715d5fe47579fa184fa819f421

3.02% Джерела з Інтернету 446 Сторінка 34

Пошук збігів з Бібліотекою не проводився

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації
Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи 20

Здобувач:

_____ Д. О. Фрич
підпис ініціали, прізвище

Керівник:

д-р техн. наук, професор

_____ І. М. Журавська
підпис ініціали, прізвище

Дата: «___» _____ 2024 р.

ДОДАТОК Б

Код прошивки ESP32

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "freertos/event_groups.h"

#include "nvs_flash.h"

#include "esp_mac.h"
#include "rom/ets_sys.h"
#include "esp_log.h"
#include "esp_wifi.h"
#include "esp_netif.h"
#include "esp_now.h"

#include "lwip/inet.h"
#include "lwip/netdb.h"
#include "lwip/sockets.h"
#include "ping/ping_sock.h"

#include "protocol_examples_common.h"

#define CONFIG_SEND_FREQUENCY      100

static const char *TAG = "csi_recv_router";

static void wifi_csi_rx_cb(void *ctx, wifi_csi_info_t *info)
{
    if (!info || !info->buf) {
        ESP_LOGW(TAG, "<%s> wifi_csi_cb",
esp_err_to_name(ESP_ERR_INVALID_ARG));
        return;
    }

    if (memcmp(info->mac, ctx, 6)) {
        return;
    }

    static int s_count = 0;
    const wifi_pkt_rx_ctrl_t *rx_ctrl = &info->rx_ctrl;

    if (!s_count) {
        ESP_LOGI(TAG, "================ CSI RECV ================");

ets_printf("type,seq,mac,rssi,rate,sig_mode,mcs,bandwidth,smoothing,not_sound
ing,aggregation,stbc,fec_coding,sgi,noise_floor,ampdu_cnt,channel,secondary_c
hannel,local_timestamp,ant,sig_len,rx_state,len,first_word,data\n");
    }

    /** Only LLTF sub-carriers are selected. */
    info->len = 128;
```

```
    printf("CSI_DATA,%d," MACSTR
",%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d",
            s_count++, MAC2STR(info->mac), rx_ctrl->rssi, rx_ctrl->rate,
rx_ctrl->sig_mode,
            rx_ctrl->mcs, rx_ctrl->cwb, rx_ctrl->smoothing, rx_ctrl-
>not_sounding,
            rx_ctrl->aggregation, rx_ctrl->stbc, rx_ctrl->fec_coding,
rx_ctrl->sgi,
            rx_ctrl->noise_floor, rx_ctrl->ampdu_cnt, rx_ctrl->channel,
rx_ctrl->secondary_channel,
            rx_ctrl->timestamp, rx_ctrl->ant, rx_ctrl->sig_len, rx_ctrl-
>rx_state);

    printf(",%d,%d,\"[%d", info->len, info->first_word_invalid, info-
>buf[0]);

    for (int i = 1; i < info->len; i++) {
        printf(",%d", info->buf[i]);
    }

    printf("]\"\n");
}

static void wifi_csi_init()
{
    /**
     * @brief In order to ensure the compatibility of routers, only LLTF sub-
carriers are selected.
     */
    wifi_csi_config_t csi_config = {
        .lltf_en           = true,
        .htltf_en          = false,
        .stbc_htltf2_en    = false,
        .ltf_merge_en      = true,
        .channel_filter_en = true,
        .manu_scale        = true,
        .shift             = true,
    };

    static wifi_ap_record_t s_ap_info = {0};
    ESP_ERROR_CHECK(esp_wifi_sta_get_ap_info(&s_ap_info));
    ESP_ERROR_CHECK(esp_wifi_set_csi_config(&csi_config));
    ESP_ERROR_CHECK(esp_wifi_set_csi_rx_cb(wifi_csi_rx_cb, s_ap_info.bssid));
    ESP_ERROR_CHECK(esp_wifi_set_csi(true));
}

static esp_err_t wifi_ping_router_start()
{
    static esp_ping_handle_t ping_handle = NULL;

    esp_ping_config_t ping_config = ESP_PING_DEFAULT_CONFIG();
    ping_config.count             = 0;
    ping_config.interval_ms       = 1000 / CONFIG_SEND_FREQUENCY;
    ping_config.task_stack_size   = 3072;
    ping_config.data_size         = 1;
```

```
    esp_netif_ip_info_t local_ip;
    esp_netif_get_ip_info(esp_netif_get_handle_from_ifkey("WIFI_STA_DEF"),
&local_ip);
    ESP_LOGI(TAG, "got ip:" IPSTR ", gw: " IPSTR, IP2STR(&local_ip.ip),
IP2STR(&local_ip.gw));
    ping_config.target_addr.u_addr.ip4.addr = ip4_addr_get_u32(&local_ip.gw);
    ping_config.target_addr.type = ESP_IPADDR_TYPE_V4;

    esp_ping_callbacks_t cbs = { 0 };
    esp_ping_new_session(&ping_config, &cbs, &ping_handle);
    esp_ping_start(ping_handle);

    return ESP_OK;
}

void app_main()
{
    ESP_ERROR_CHECK(nvs_flash_init());
    ESP_ERROR_CHECK(esp_netif_init());
    ESP_ERROR_CHECK(esp_event_loop_create_default());

    ESP_ERROR_CHECK(example_connect());

    wifi_csi_init();
    wifi_ping_router_start();
}
```

ДОДАТОК В

Код програми аналізу CSI

```
import sqlite3
import sys
import csv
import json
import argparse
from datetime import datetime

import pandas as pd
import numpy as np

import serial
from os import path
from io import StringIO

from PyQt5.Qt import *
from PyQt5.QtWidgets import QApplication, QMainWindow, QDialog,
QVBoxLayout, QLabel, QPushButton, QWidget
from PyQt5 import QtWidgets
from PyQt5.QtCore import pyqtSlot, QThread
from pyqtgraph import PlotWidget
from PyQt5 import QtCore
import pyqtgraph as pq

import threading
import time

CSI_VAID_SUBCARRIER_INTERVAL = 3

# secondary channel : below, HT, 40 MHz, non STBC, v, HT-LFT: 0~63, -
64~-1, 384
csi_vaid_subcarrier_index = []
csi_vaid_subcarrier_color = []
color_step = 255 // (28 // CSI_VAID_SUBCARRIER_INTERVAL + 1)

# LLTF: 52
csi_vaid_subcarrier_index += [i for i in range(6, 32,
CSI_VAID_SUBCARRIER_INTERVAL)]  # 26  red
csi_vaid_subcarrier_color += [(i * color_step, 0, 0) for i in range(1,
26 // CSI_VAID_SUBCARRIER_INTERVAL + 2)]
csi_vaid_subcarrier_index += [i for i in range(33, 59,
CSI_VAID_SUBCARRIER_INTERVAL)]  # 26  green
csi_vaid_subcarrier_color += [(0, i * color_step, 0) for i in range(1,
26 // CSI_VAID_SUBCARRIER_INTERVAL + 2)]
CSI_DATA_LLFT_COLUMNS = len(csi_vaid_subcarrier_index)

# HT-LFT: 56 + 56
csi_vaid_subcarrier_index += [i for i in range(66, 94,
CSI_VAID_SUBCARRIER_INTERVAL)]  # 28  blue
```

```
csi_vaid_subcarrier_color += [(0, 0, i * color_step) for i in range(1,
28 // CSI_VAID_SUBCARRIER_INTERVAL + 2)]
csi_vaid_subcarrier_index += [i for i in range(95, 123,
CSI_VAID_SUBCARRIER_INTERVAL)]  # 28  White
csi_vaid_subcarrier_color += [(i * color_step, i * color_step, i *
color_step) for i in
                              range(1, 28 //
CSI_VAID_SUBCARRIER_INTERVAL + 2)]

CSI_DATA_INDEX = 200  # buffer size
CSI_DATA_COLUMNS = len(csi_vaid_subcarrier_index)
DATA_COLUMNS_NAMES = ["type", "id", "mac", "rssi", "rate", "sig_mode",
"mcs", "bandwidth", "smoothing", "not_sounding",
                      "aggregation", "stbc", "fec_coding",
                      "sgi", "noise_floor", "ampdu_cnt", "channel",
"secondary_channel", "local_timestamp", "ant",
                      "sig_len", "rx_state", "len", "first_word",
"data"]
csi_data_array = np.zeros(
    [CSI_DATA_INDEX, CSI_DATA_COLUMNS], dtype=np.complex64)

uiclass, baseclass = pq.Qt.loadUiType("my.ui")


def record_csi_data():
    conn = sqlite3.connect('wfscan.db')
    cursor = conn.cursor()

    start_time = time.time()
    end_time = start_time + 60

    while time.time() < end_time:
        csi_data = json.dumps(csi_data_array[-1].tolist())
        timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
        cursor.execute("INSERT INTO csi_data (timestamp, csi_data)
VALUES (?, ?)", (timestamp, csi_data))
        conn.commit()
        time.sleep(0.1)

    conn.close()
    print("Запис CSI даних завершено.")


def compare_csi_data(input_csi_data):
    conn = sqlite3.connect('wfscan.db')
    cursor = conn.cursor()

    cursor.execute("SELECT data FROM csi_data")
    db_csi_data = cursor.fetchall()

    for db_data in db_csi_data:
```

```
        db_data = json.loads(db_data[0])

        input_array = np.array(input_csi_data)
        db_array = np.array(db_data)

        if input_array.shape != db_array.shape:
            continue

        similarity = np.mean(np.abs(input_array - db_array) < 0.05) *
100

        if similarity >= 95:
            conn.close()
            return True

    conn.close()
    return False


class MainWindow(uiclass, baseclass):
    def __init__(self):
        super().__init__()
        self.setupUi(self)
        self.plot_placeholder = self.findChild(QtWidgets.QWidget,
'widget')

        # Create an instance of the csi_data_graphical_window
        self.csi_plot = csi_data_graphical_window()

        plot_layout = QtWidgets.QVBoxLayout(self.plot_placeholder)
        plot_layout.addWidget(self.csi_plot)
        self.experimentButton.clicked.connect(self.open_dialog)

        self.startButton = QPushButton("Запис CSI даних", self)
        self.startButton.clicked.connect(self.start_recording)

    @pyqtSlot()
    def on_experiment_button_clicked(self):
        text = self.lineEditObjectType.text()
        self.save_data(text)
        print(text)

    def open_dialog(self, object):
        box = QtWidgets.QMessageBox()
        box.setWindowTitle("Об'єкт визначено")
        box.setText(f"Об'єкт визначено: {object}")
        box.setIcon(QtWidgets.QMessageBox.Information)
        box.exec_()

    def start_recording(self):
        threading.Thread(target=record_csi_data).start()
```

```
    def save_data(self, data):
        with open("data.txt", "a") as file:
            file.write(data + "\n")


class csi_data_graphical_window(QWidget):
    def __init__(self):
        super().__init__()

        self.resize(1280, 720)
        self.plotWidget_ted = PlotWidget(self)
        self.plotWidget_ted.setGeometry(QtCore.QRect(0, 0, 1280, 720))

        self.plotWidget_ted.setYRange(-5, 50)
        self.plotWidget_ted.addLegend()
        self.plotWidget_ted.setBackground('white')
        self.plotWidget_ted.showGrid(x=True, y=True, alpha=1)
        self.plotWidget_ted.setLabel('left', "Амплітуда")
        self.plotWidget_ted.setLabel('bottom', "Фаза")
        self.csi_phase_array = np.abs(csi_data_array)
        self.curve_list = []

        for i in range(CSI_DATA_COLUMNS):
            curve = self.plotWidget_ted.plot(
                self.csi_phase_array[:, i], name=str(i),
pen=csi_vaid_subcarrier_color[i])
            self.curve_list.append(curve)

        self.timer = pq.QtCore.QTimer()
        self.timer.timeout.connect(self.update_data)
        self.timer.start(100)

    def update_data(self):
        self.csi_phase_array = np.abs(csi_data_array)

        for i in range(CSI_DATA_COLUMNS):
            self.curve_list[i].setData(self.csi_phase_array[:, i])

        if compare_csi_data(csi_data_array[-1]):
            self.parent().open_dialog("Знайдено схожий об'єкт")


def csi_data_read_parse(port: str, csv_writer):
    ser = serial.Serial(port=port, baudrate=921600,
                        bytesize=8, parity='N', stopbits=1)
    if ser.isOpen():
        print("open success")
    else:
        print("open failed")
        return
```

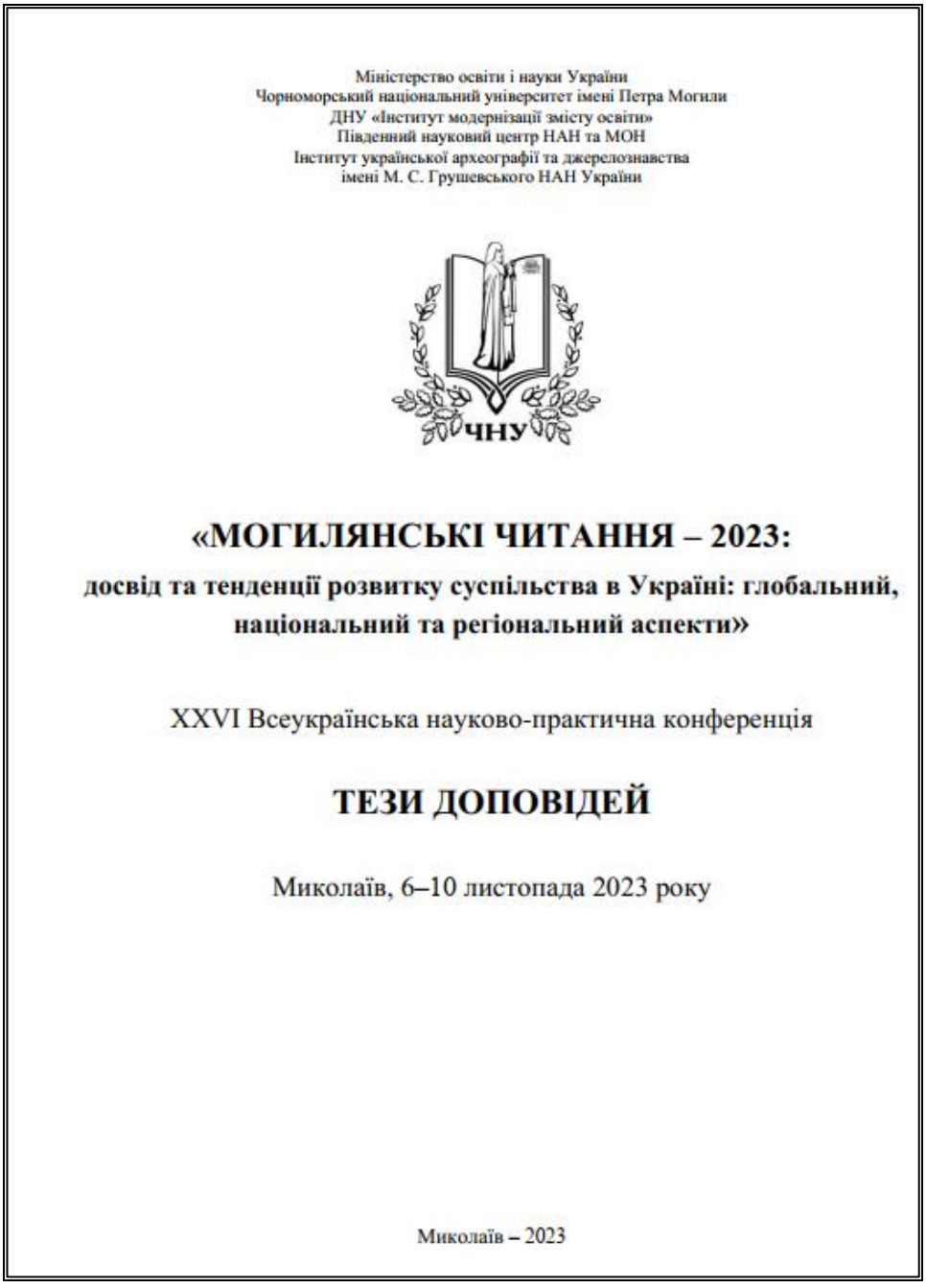
```
    while True:
        strings = str(ser.readline())
        if not strings:
            break

        strings = strings.lstrip('b\'').rstrip('\\r\\n\'')
        index = strings.find('CSI_DATA')

        if index == -1:
            continue

        csv_reader = csv.reader(StringIO(strings))
        csi_data = next(csv_reader)

        if len(csi_data) != len(DATA_COLUMNS_NAMES):
            print("element number is not equal")
            continue

        try:
            csi_raw_data = json.loads(csi_data[-1])
        except json.JSONDecodeError:
            print(f"data is incomplete")
            continue

        if len(csi_raw_data) != 128 and len(csi_raw_data) != 256 and
len(csi_raw_data) != 384:
            print(f"element number is not equal: {len(csi_raw_data)}")
            continue

        csv_writer.writerow(csi_data)

        # Rotate data to the left
        csi_data_array[:-1] = csi_data_array[1:]

        if len(csi_raw_data) == 128:
            csi_vaid_subcarrier_len = CSI_DATA_LLFT_COLUMNS
        else:
            csi_vaid_subcarrier_len = CSI_DATA_COLUMNS

        for i in range(csi_vaid_subcarrier_len):
            csi_data_array[-1][i] =
complex(csi_raw_data[csi_vaid_subcarrier_index[i] * 2 + 1],

csi_raw_data[csi_vaid_subcarrier_index[i] * 2])

    ser.close()
    return


class SubThread(QThread):
```

```
    def __init__(self, serial_port, save_file_name):
        super().__init__()
        self.serial_port = serial_port

        save_file_fd = open(save_file_name, 'w')
        self.csv_writer = csv.writer(save_file_fd)
        self.csv_writer.writerow(DATA_COLUMNS_NAMES)

    def run(self):
        csi_data_read_parse(self.serial_port, self.csv_writer)

    def __del__(self):
        self.wait()


if __name__ == '__main__':
    if sys.version_info < (3, 6):
        print(" Python version should >= 3.6")
        exit()

    parser = argparse.ArgumentParser(
        description="Read CSI data from serial port and display it
graphically")
    parser.add_argument('-p', '--port', dest='port', action='store',
required=True,
                        help="Serial port number of csv_recv device")
    parser.add_argument('-s', '--store', dest='store_file',
action='store', default='./csi_data.csv',
                        help="Save the data printed by the serial port
to a file")

    args = parser.parse_args()
    serial_port = args.port
    file_name = args.store_file

    app = QApplication(sys.argv)

    subthread = SubThread(serial_port, file_name)
    subthread.start()

    window = MainWindow()
    window.show()

    sys.exit(app.exec())
```

ДОДАТОК Г

Матеріали апробації роботи

В.1 XXVI Всеукраїнська науково-практична конференція «Могилянські читання – 2023»



Міністерство освіти і науки України
Чорноморський національний університет імені Петра Могили
ДНУ «Інститут модернізації змісту освіти»
Південний науковий центр НАН та МОН
Інститут української археографії та джерелознавства
імені М. С. Грушевського НАН України

ЧНУ

«МОГИЛЯНСЬКІ ЧИТАННЯ – 2023:
досвід та тенденції розвитку суспільства в Україні: глобальний, національний та регіональний аспекти»

XXVI Всеукраїнська науково-практична конференція

ТЕЗИ ДОПОВІДЕЙ

Миколаїв, 6–10 листопада 2023 року

Миколаїв – 2023

Тези доповідей

14

УДК 004.93

Фрич Д. О.,
студент 4-го курсу,
Журавська І. М.,
д-р техн. наук, проф., зав. кафедри комп'ютерної інженерії,
ЧНУ імені Петра Могили, м. Миколаїв, Україна

ВИЯВЛЕННЯ НЕБЕЗПЕЧНИХ ПРЕДМЕТІВ ЗА ДОПОМОГОЮ МЕРЕЖІ WI-FI

Під час повномасштабного вторгнення в Україні підвищився ризик терактів та пронесення летальної зброї в публічні місця. Переносні небезпечні предмети, такі як летальна зброя, саморобні бомби та вибухонебезпечні хімічні речовини, становлять дедалі більшу загрозу громадській безпеці. Зазвичай, для терористичних атаках зловмисникам достатньо просто сховати небезпечні предмети в невеликому багажі, не привертаючи до себе уваги в громадських місцях.

Щоб зменшити такі загрози, зберігаючи особисте життя, необхідно широко впроваджувати неінтрузивні перевірки безпеки в громадських місцях (наприклад, у закладах освіти, торгівельних центрах, музеях).

Традиційне виявлення підозрілих предметів у багажі передбачає:

- ручний огляд (наприклад, встановлення контрольно-пропускного пункту на кожному вході);
- спеціальне обладнання: нп., камера спостереження, рентгенівський апарат, надширокосмуговий сканер тощо [1–2].

Такі методи вимагають високих витрат на розгортання, а також складні в масштабуванні.

Нещодавно радіочастотні сигнали (наприклад, Wi-Fi і радари 60 ГГц) продемонстрували свій великий потенціал у багатьох сферах. Наприклад, сигнали Wi-Fi можна використовувати для розпізнавання людської активності за стіною [3]. Радіолокатор 60 ГГц можна використовувати для диференціації об'єктів, хоча й без можливості класифікувати об'єкти за типами матеріалів. Однак, існуючі радіолокаційні підходи пов'язані з високими накладними витратами, оскільки вимагають великої антенної решітки або спеціалізованих сигналів.

Основна ідея полягає в тому, щоб комплексно дослідити інформацію про стан каналу (англ. Channel State Information, CSI), яка включає як амплітуду, так і фазову характеристику бездротових сигналів для аналізу перешкод, викликаних певними об'єктами. Найбільш небезпечні об'єкти, такі як зброя, саморобні бомби та вибухові речовини, зазвичай виготовлені з металу або рідини, які створюють значні перешкоди (наприклад, поглинання, заломлення та рефракцію) для бездротових сигналів, в той час, як чемодани зазвичай виготовляються з тканини, пластику або паперу, які дозволяють бездротовим сигналам проходити крізь них. Такий різний вплив на бездротові сигнали дозволяє припустити, що бездротові сигнали можна використовувати для виявлення та ідентифікації підозрілих предметів, захованих у багажі.

Для ідентифікації різних матеріалів використовуються сигнали Wi-Fi, що проходять через об'єкт або оминають його, що призводить до зміни значень CSI-характеристик. Крім того, можливо виділити сигнал, відбитий об'єктом, щоб оцінити його форму (наприклад, ширину і висоту) або об'єм, виходячи з того, що сила відбитого сигналу пропорційна площі відбиття об'єкта. Система вимагає лише WiFi-пристрою з 2–3 антенами і може бути інтегрована в існуючі мережі Wi-Fi з низькими витратами і зусиллями по розгортанню.

Тобто, технологія використовує звичайний WiFi-сигнал для створення недорогого і простого в масштабуванні рішення, яке забезпечує першу лінію захисту для виявлення прихованих підозрілих об'єктів. Така система може бути легко розгорнута в багатьох місцях, які ще не мають попередньо встановленої інфраструктури перевірки безпеки і вимагають великої потужності для проведення перевірки безпеки, таких як стадіони, станції метро/залізниці, парки, музеї.

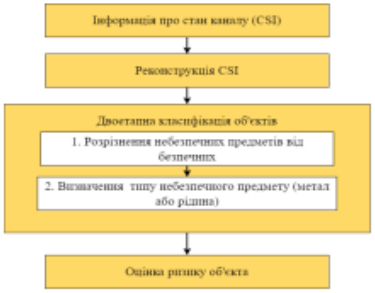


Рисунок 1 – Принцип роботи системи виявлення небезпечних предметів на основі CSI

Щоб полегшити виявлення та ідентифікацію підозрілих об'єктів, система використовує інформацію CSI, доступну в існуючих пристроях Wi-Fi. Як показано на рис. 1, система приймає CSI від пари WiFi-передавач та WiFi-приймач як вхідні дані. Потім виконується реконструкція CSI, тобто видаляються шуми Після цього попередньо оброблені дані проходять через два основних компоненти:

1) двоетапна класифікація матеріалів фокусується на аналізі типу матеріалу для виявлення підозрілих об'єктів;

2) оцінка ризику об'єкта на основі візуалізації форми для металу та оцінки об'єму для рідини в контейнерах. Це пов'язано з тим, що рідина має вищий рівень ризику, якщо її об'єм перевищує допустиму межу, а металевий предмет є більш підозрілим, якщо він має форму, схожу на форму зброї.

Таким чином, звичайну мережу Wi-Fi можна додатково використовувати в якості системи безпеки – детектора підозрілих предметів. Система не потребує додаткового обладнання і може бути легко розвернути у вже існуючій мережі; також не має виникнути проблем при її масштабуванні.

Список використаних джерел

1. S. Chi and C. H. Caldas, «Automated object identification using optical video cameras on construction sites,» *Computer-Aided Civil and Infrastructure Engineering*, vol. 26, no. 5, pp. 368–380, 2011.

2. D. Turcsany, A. Mouton, and T. P. Breckon, «Improving feature- based object recognition for x-ray baggage security screening using primed visualwords,» in *IEEE International Conference on Industrial Technology*, 2013, pp. 1140–1145.

3. T. S. Ralston, G. L. Charvat, and J. E. Peabody, «Real-time through- wall imaging using an ultrawideband multiple-input multiple-output (mimo) phased array radar system,» in *IEEE International Symposium on Phased Array Systems and Technology*, 2010, pp. 551–558.

УДК 004.89

Швайко В. К.,
студентка 4-го курсу бакалаврату,
Ільчишина Ю. В.,
студентка 4-го курсу бакалаврату
Павлова О. О.,
д-р філософії, доцент,
Хмельницький нац. ун-т, м. Хмельницький, Україна

ВИЗНАЧЕННЯ ІНДИКАТОРІВ ДЛЯ МОРФОФУНКЦІОНАЛЬНИХ ПОКАЗНИКІВ ЛЮДИНИ ДЛЯ АВТОМАТИЗОВАНОГО ПІДБОРУ ВИДУ СПОРТУ

У міру того як людство оговтується від наслідків глобальної пандемії COVID-19 та намагається повернутися до активного ритму життя. Статистика зайнятості спортом дітей шкільного віку та підлітків в Україні за останні роки свідчить про позитивну тенденцію до зростання кількості дітей, які займаються спортом. Так, у 2021 році 80% дітей шкільного віку та підлітків займалися спортом, що на 10% більше, ніж у 2020 році. Найпопулярнішими видами спорту серед дітей шкільного віку та підлітків є футбол, баскетбол, волейбол, легка атлетика та плавання. Зростання

В.2 Фінал Конкурсу стартапів «Startup BSNU 2024»