Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Електронної та біомедичної інженерії
(повна назва)
Кафедра Фізичних основ електронної техніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

ІНТЕЛЕКТУАЛЬНА СИСТЕМА КЕРУВАННЯ МОБІЛЬНИМИ ОПТОЕЛЕКТРОННИМИ ПЛАТФОРМАМИ
(тема)

Виконав:
здобувач 2 року навчання,
групи ЛОЕТм-24-1
Єгор ДІРЯВЧЕНКО
(Власне ім'я, ПРІЗВИЩЕ)

Спеціальність 175 Інформаційно-вимірювальні технології
(код і повна назва спеціальності)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма «Лазерна і оптоелектронна техніка»
(повна назва освітньої програми)

Керівник доц. каф. ФОЕТ Олена ЛІННИК
(посада, Власне ім'я, ПРІЗВИЩЕ)

Допускається до захисту

Завідувач кафедри ФОЕТ _____ Олександр ГНАТЕНКО
(підпис) (Власне ім'я, ПРІЗВИЩЕ)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Електронної та біомедичної інженерії

Кафедра Фізичних основ електронної техніки

Рівень вищої освіти другий(магістерський)

Спеціальність 175 Інформаційно – вимірювальні технології
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма «Лазерна і оптоелектронна техніка»
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Дірявченку Єгору Олеговичу
(прізвище, ім'я, по батькові)

1 Тема роботи Інтелектуальна система керування мобільними оптоелектронними платформами

затверджена наказом університету від « 10 » листопада 2025 р. № 1024 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 16 грудня 2025 р.

3. Вихідні дані до роботи вимоги до інтелектуальної системи керування; картографічні дані; дані про характеристики оптоволоконного каналу зв'язку; технічні характеристики мобільних платформ

4. Перелік питань, що потрібно опрацювати в роботі: 1 Провести аналіз сучасних підходів до побудови інтелектуальних систем керування мобільними оптоелектронними платформами. 2 Дослідити можливості використання оптоволоконних каналів зв'язку для підвищення стабільності й швидкодії системи. 3 Реалізувати симуляційне середовище для перевірки роботи системи в умовах втрати або заглушення сигналу; 4 Провести експериментальні дослідження впливу типу з'єднання на ефективність обміну даними та поведінку агентів; 5 Оцінити результати та визначити напрями подальшого розвитку інтелектуальних систем керування мобільними платформами.

5. Перелік графічного матеріалу із зазначенням кресленików, схем, плакатів, комп'ютерних ілюстрацій
Демонстраційний матеріал – 13 слайдів

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз теоретичних основи оптоволоконних та оптоелектронних технологій керування мобільними платформами	03.09.25–20.09.25	Виконано
2	Вибір інструментів для реалізації симуляції	25.09.25–05.10.25	Виконано
3	Розроблення веб-застосунку та модулів оптоелектронного моделювання каналу зв'язку	10.10.25–18.10.25	Виконано
4	Дослідження роботи інтелектуальної системи з урахуванням параметрів оптоволоконного каналу.	22.10.25–25.10.25	Виконано
5	Оформлення пояснювальної записки	30.10.25–11.11.25	Виконано
6	Оформлення демонстраційних матеріалів	14.11.25–20.11.25	Виконано
7	Проходження нормоконтролю та перевірки тексту КвР на унікальність	26.11.25–08.12.25	Виконано
8	Отримання відгуку та рецензії	10.12.25–13.12.25	Виконано
9	Підготовка та захист кваліфікаційної роботи	15.12.25–17.12.25	Виконано

Дата видачі завдання 02 вересня 2025 р..

Студент _____
(підпис)

Керівник роботи _____ доц. каф. ФОЕТ Олена ЛІННИК
(підпис) (посада, Власне ім'я, ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи: 79 с., 14 рис., 1 додаток., 33 джерела.

АВТОНОМНА НАВІГАЦІЯ, БАГАТОАГЕНТНА СИСТЕМА, ІНТЕЛЕКТУАЛЬНА СИСТЕМА, КЕРУВАННЯ В РЕАЛЬНОМУ ЧАСІ, МАРШРУТИЗАЦІЯ, МОБІЛЬНА ПЛАТФОРМА, ОПТОВОЛОКОННИЙ КАНАЛ, СИМУЛЯЦІЯ, ШТУЧНИЙ ІНТЕЛЕКТ, ЗАТРИМКА СИГНАЛУ.

Об’єкт дослідження – процес інтелектуального керування мобільними оптоелектронними платформами в багатоагентному середовищі.

Предмет дослідження – методи, технології та програмні засоби побудови інтелектуальних систем керування мобільними платформами з урахуванням параметрів оптоволоконного каналу зв’язку.

Метою роботи є розробка та дослідження інтелектуальної системи керування мобільними оптоелектронними платформами на основі сучасних вебтехнологій, моделей машинного навчання та симуляції оптоволоконного каналу зв’язку для підвищення точності, надійності та стабільності руху агентів.

У роботі виконано огляд методів керування багатоагентними системами та технологій побудови інтелектуальних модулів на базі великих мовних моделей. Розроблено архітектуру вебзастосунку, що включає серверну частину на FastAPI, клієнтську частину на React, модуль генерації команд на основі Mistral 7B та засоби передачі даних у реальному часі через WebSocket. Створено імітаційну модель впливу різних типів каналів зв’язку, зокрема оптоволоконного, на стабільність і точність руху мобільних платформ.

ABSTRACT

Explanatory note includes: 79 pages, 14 pictures, 1 applications, 33 sources for references.

ARTIFICIAL INTELLIGENCE, AUTONOMOUS NAVIGATION, FIBER-OPTIC CHANNEL, INTELLIGENT SYSTEM, MOBILE PLATFORM, MULTI-AGENT SYSTEM, REAL-TIME CONTROL, ROUTING, SIGNAL LATENCY, SIMULATION.

Object of the study – the process of intelligent control of mobile platforms in a multi-agent environment.

Subject of the study – methods, technologies, and software tools for building intelligent control systems for mobile platforms, taking into account the parameters of a fiber-optic communication channel.

The purpose of the work is to develop and investigate an intelligent control system for mobile platforms based on modern web technologies, machine learning models, and simulation of a fiber-optic communication channel to improve the accuracy, reliability, and stability of agent movement.

The work includes a review of methods for controlling multi-agent systems and technologies for building intelligent modules based on large language models. The architecture of the web application has been developed, which includes a server part based on FastAPI, a client part implemented with React, a command generation module powered by the Mistral 7B model, and real-time data transmission using WebSocket. A simulation model has been created to analyze the impact of different types of communication channels, including fiber-optic channels, on the stability and accuracy of mobile platform movement.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

БД – база даних;

ШІ – штучний інтелект;

HTML – HyperText Markup Language (гіпертекстова мова розмітки);

ID – персональний номер;

JSX – JavaScript XML;

LLM – Large Language Model (велика мовна модель);

NP – Neurolinguistic Programming (нейролінгвістичне програмування);

PC – Personal Computer (персональний комп’ютер);

XML – EXtensible Markup Language (розширювана мова розмітки).

ВСТУП

Сучасний розвиток оптоелектронних та волоконно-оптичних технологій значно впливає на формування нових підходів до створення високоточних систем передавання інформації та керування. Оптичні канали зв’язку, завдяки своїй широкій смузі пропускання, низькому рівню втрат і стійкості до електромагнітних завад, стали ключовим елементом сучасних інформаційно-вимірювальних систем, роботизованих комплексів та мобільних платформ. Високошвидкісні оптоволоконні лінії дозволяють забезпечувати надійний обмін даними у реальному часі, що є критично важливим для систем із розподіленою архітектурою, де необхідна синхронізована робота кількох об’єктів керування.

У межах цієї роботи розглядається розроблення інтелектуальної системи керування мобільними оптоволоконними платформами, яка поєднує принципи багатоагентної взаємодії, оптоелектронні технології та сучасні методи штучного інтелекту. Дослідження зосереджене на питаннях впливу фізичних характеристик оптичного каналу — затримки, дисперсії, можливих втрат сигналу — на ефективність передавання керуючих команд і стабільність роботи системи в цілому. Особлива увага приділяється моделюванню алгоритмів адаптивного керування у середовищі з динамічно змінними параметрами зв’язку, а також створенню симуляційного інтерфейсу, що дозволяє візуалізувати роботу мобільних платформ та перевіряти стійкість системи до збурень.

1 ТЕОРЕТИЧНІ ОСНОВИ ОПТОВОЛОКОННИХ ТА ОПТОЕЛЕКТРОННИХ ТЕХНОЛОГІЙ У СИСТЕМАХ КЕРУВАННЯ МОБІЛЬНИМИ ПЛАТФОРМАМИ

1.1 Роль оптоволоконних ліній зв’язку в сучасних інформаційно-вимірювальних системах

Сучасні інформаційно-вимірювальні системи стрімко переходять на технології оптоволоконної передачі даних, що зумовлено високими вимогами до швидкості передавання сигналів, завадостійкості та стабільності комунікацій у режимі реального часу. На відміну від класичних електричних каналів, оптоволоконні мережі забезпечують передавання інформації у вигляді світлового сигналу, що значно підвищує пропускну здатність та зменшує рівень шумів і електромагнітних завад [1].

Поширення оптоволоконних систем у вимірювальній техніці пов’язане з необхідністю швидко та без втрат передавати великі обсяги даних від сенсорних модулів, датчиків, систем контролю та мобільних роботизованих платформ. Оптичні канали також нечутливі до електромагнітного випромінювання, що критично для роботи платформ у середовищах підвищеної завадності: біля потужних електромоторів, лазерних установок, високовольтного обладнання тощо.

Для систем керування мобільними платформами ключовими перевагами оптоволоконного каналу є:

- висока швидкість передавання команд;
- мінімальна затримка сигналу;
- можливість роботи на великих дистанціях без погіршення якості;
- захищеність від впливу зовнішніх факторів;
- потенційна безпечність у вибухонебезпечних середовищах;
- високий рівень стабільності при інтенсивному потоці даних.

Таким чином, оптоволоконні канали стають базовим елементом сучасних систем автоматизації та керування рухомими об’єктами, де точність синхронізації та час відгуку мають першочергове значення.

1.2 Фізичні основи передавання інформації в оптичному волокні

Передавання інформації в оптичному волокні ґрунтується на використанні світлового потоку як носія сигналу, що забезпечує високу швидкість, завадостійкість та можливість роботи на великих відстанях. Оптичне волокно має структуру, яка складається із серцевини та оболонки, де показник заломлення серцевини є вищим за показник заломлення зовнішнього шару [2] (рис. 1.1). Завдяки такій різниці виникає явище повного внутрішнього відбиття, за якого світловий промінь багаторазово відбивається від межі середовищ і продовжує поширюватися всередині волокна практично без втрат. Це забезпечує можливість передачі сигналу на десятки кілометрів без застосування проміжних підсилювачів, що є однією з ключових переваг оптоволоконних технологій порівняно з традиційними електричними лініями зв’язку.

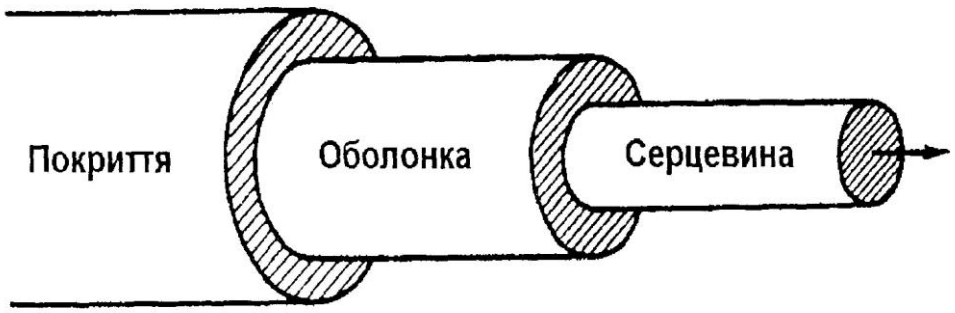

Рисунок 1.1 – Структура оптичного волокна

За принципом поширення світла оптичні волокна поділяються на одномодові та багатомодові. В одномодовому волокні світло поширюється

єдиним модом, що мінімізує спотворення і забезпечує найменшу величину дисперсії. Такі волокна застосовуються в системах, де необхідні висока швидкість та точність синхронізації, а також стабільність передачі сигналів на великих дистанціях (рис. 1.2). У багатомодовому волокні світловий потік поширюється одночасно кількома модами, що призводить до виникнення модової дисперсії — явища, за якого різні моди доходять до приймача в різні моменти часу. Це обмежує довжину лінії, але дозволяє зменшити вартість обладнання. Багатомодові системи часто використовуються на коротких ділянках мереж або в локальних технологічних комплексах.

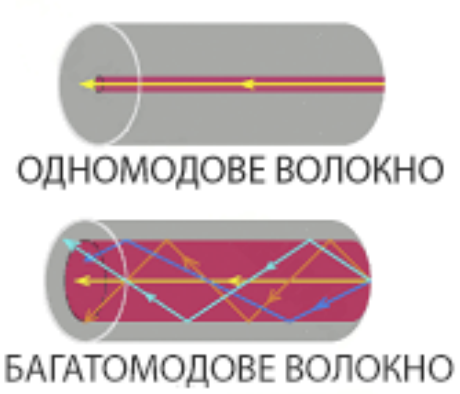

Рисунок 1.2 – Одномодове і багатомодове волокно

На якість передачі інформації оптичним волокном впливають кілька фізичних факторів. Одним із головних є дисперсія — процес розширення світлового імпульсу в часі, зумовлений різницею фазових швидкостей для різних довжин хвиль. Окрім хроматичної дисперсії, вплив мають також втрати на поглинання та розсіювання, що виникають внаслідок мікродефектів структури волокна або домішок матеріалу. У високошвидкісних системах проявляються також нелінійні оптичні ефекти, які можуть спотворювати форму сигналу. Усі ці процеси важливі при створенні систем керування мобільними платформами, оскільки будь-яке

спотворення або затримка сигналу може вплинути на точність передачі команд та синхронізацію між агентами.

Швидкість світла в оптичному волокні становить приблизно дві третини від швидкості світла у вакуумі, що визначає базовий рівень фізичної затримки поширення сигналу. Хоч ця величина є дуже малою, у системах реального часу, особливо в багатоагентних комплексах, навіть мілісекундна затримка може мати значення для корекції траєкторії чи оперативного реагування на зміну умов. Тому під час розробки системи необхідно враховувати сумарну затримку, що складається з поширення світла, оптоелектронного перетворення та цифрової обробки даних. Саме поєднання цих факторів формує реальні умови функціонування каналу зв’язку та впливає на можливості точного керування рухомими платформами.

Завдяки унікальним фізичним властивостям — мінімальній завадності, високій пропускній здатності та стабільності — оптичне волокно стало ключовою технологією для сучасних автоматизованих систем, де точність і швидкість передачі вимірювальної та керуючої інформації є критично важливими. Його застосування забезпечує надійну платформу для побудови високоточних систем керування мобільними агентами, що працюють у складних технічних та зовнішніх умовах.

1.3 Оптоволоконні канали у вимірювальних та керуючих системах

Оптоволоконні канали зв’язку стали ключовою технологією у сучасних вимірювальних та керуючих системах завдяки своїй високій пропускній здатності, електромагнітній завадостійкості та стабільності передавання сигналу на значні відстані. На відміну від традиційних мідних ліній, оптоволокно забезпечує повну ізоляцію від електричних перешкод, що робить його оптимальним рішенням у середовищах з інтенсивними електромагнітними полями або великою кількістю високочастотних пристроїв.

У системах автоматизації та мобільної робототехніки оптичний канал використовується для передавання даних з датчиків, команд керування та телеметрії між центральним модулем та виконавчими пристроями [3]. Завдяки мінімальним затримкам і високій точності сигналу, оптоволокно дозволяє реалізувати синхронізовану роботу декількох мобільних платформ, що особливо важливо у багатоагентних системах. Додатковою перевагою є можливість передавання не тільки цифрового, а й аналогового оптичного сигналу, що використовується у високоточних вимірювальних комплексах оптоелектронного типу.

Таким чином, оптоволоконні канали формують технологічну основу систем, де необхідно забезпечити швидке, точне й стабільне передавання інформації між елементами керування та вимірювання.

1.4 Технології оптоелектронного перетворення

Оптоелектронні технології відіграють ключову роль у перетворенні оптичних сигналів на електричні та навпаки, забезпечуючи інтеграцію оптоволоконних каналів із цифровими системами керування. У структурі вимірювальних і керуючих комплексів такі перетворювачі відповідають за швидке та точне зчитування інформації з датчиків, генерацію світлових імпульсів, модуляцію сигналів і формування керуючих команд [4].

До основних компонентів оптоелектронних систем належать лазери, світлодіоди, фотодетектори, фотодіоди лавинного типу, а також модулятори та демодулятори. Кожен з них визначає швидкість реакції, чутливість та завадостійкість каналу зв’язку [5]. Наприклад, у мобільних платформах фотодетектори застосовують для приймання високошвидкісних керуючих сигналів, а лазерні джерела — для формування вузькоспрямованих оптичних імпульсів, що можуть передаватися з мінімальними втратами [6] (рис. 1.3).

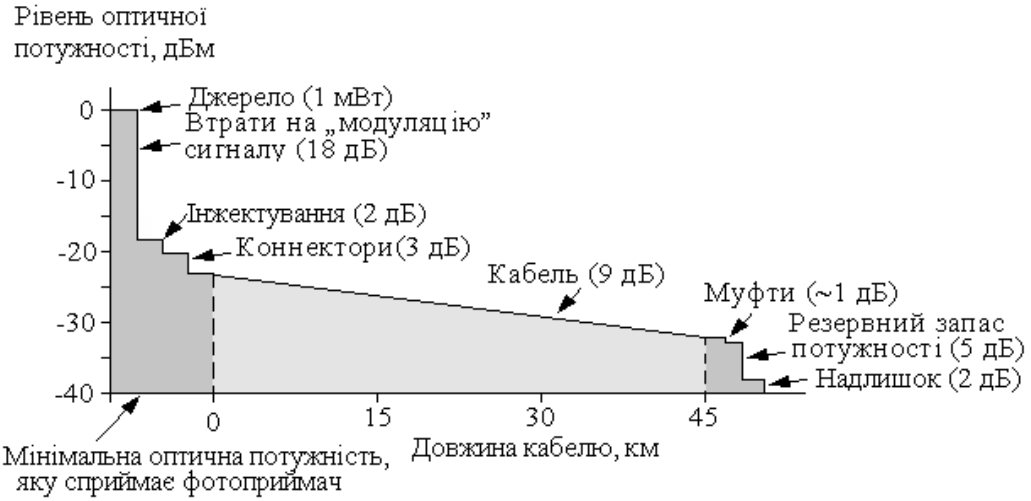

Рисунок 1.3 – Затримка сигналу в оптичному волокні

Оптоелектронні системи забезпечують стабільність роботи мобільних агентів у середовищах з сильними електромагнітними перешкодами, а також дозволяють реалізувати точне керування рухомими об’єктами в режимі реального часу. У поєднанні з алгоритмами машинного навчання та інтелектуальними модулями вони створюють основу сучасних високоточних робототехнічних комплексів.

1.5 Роль інтелектуальних алгоритмів в оптоволоконних системах

Інтелектуальні алгоритми, зокрема засоби машинного навчання та великі мовні моделі, дозволяють підвищити ефективність систем керування, які використовують оптоволоконні канали зв’язку. Вони забезпечують адаптивність мобільних платформ до змін середовища, прогнозування можливих затримок сигналу, оптимізацію маршрутизації та аналіз якості оптичного каналу в режимі реального часу [7].

Завдяки таким алгоритмам система може коригувати траєкторії руху агентів залежно від якості сигналу, втрат у каналі, наявності локальних перешкод або змін у поведінці інших платформ. У багатоагентних комплексах інтелектуальні методи дозволяють координувати взаємодію між

об’єктами з урахуванням характеристик оптоволоконного зв’язку, що значно підвищує точність і надійність роботи [8].

Інтелектуальні алгоритми стають невід’ємною частиною систем, які поєднують високошвидкісні оптичні технології та автономне керування, забезпечуючи новий рівень адаптивності та функціональності [9].

1.6 Постановка задачі

На основі проведеного аналізу сучасних підходів до побудови інтелектуальних систем керування мобільними платформами, технологій волоконно-оптичного зв’язку та методів інтеграції штучного інтелекту можна зробити висновок, що важливою науковою проблемою є забезпечення стабільної та узгодженої взаємодії між агентами у середовищі з різними комунікаційними умовами. Особливого значення набуває моделювання впливу затримки сигналу, дисперсії та можливих зон глушіння на ефективність керування, а також створення адаптивних алгоритмів прийняття рішень у режимі реального часу.

Об’єктом дослідження є процес інтелектуального керування мобільними платформами із застосуванням технологій штучного інтелекту та оптоволоконних каналів передачі даних.

Метою роботи є розробка інтелектуальної системи керування мобільними оптоволоконними платформами, здатної моделювати функціонування у середовищі зі змінними параметрами зв’язку, забезпечувати адаптивне прийняття рішень та підтримувати стабільну взаємодію між агентами в реальному часі.

Для досягнення поставленої мети необхідно вирішити такі задачі:

1) провести аналіз сучасних методів побудови інтелектуальних систем керування мобільними платформами та моделей багатоагентної взаємодії;

2) дослідити особливості волоконно-оптичних каналів зв’язку та оцінити їх вплив на стабільність і швидкодію системи;

3) розробити архітектуру інтелектуальної системи керування, у якій поєднано LLM-модуль, серверну частину, клієнтську частину та засоби синхронізації даних;

4) створити симуляційне середовище, що дозволяє моделювати роботу системи в умовах втрати або глушіння сигналу;

5) провести експериментальні дослідження залежності ефективності взаємодії агентів від параметрів каналу зв’язку;

6) виконати оцінювання результатів та визначити перспективні напрями подальшого розвитку інтелектуальних систем керування мобільними платформами.

У результаті виконання роботи має бути створена інтелектуальна система, здатна моделювати взаємодію мобільних оптоволоконних платформ у середовищі з різними типами зв’язку та змінними характеристиками каналу.

.

2 ВИБІР ІНСТРУМЕНТІВ ДЛЯ РЕАЛІЗАЦІЇ СИМУЛЯЦІЇ

2.1 Аналіз існуючих LLM для локального застосування

У межах створення інтелектуальної системи керування мобільними платформами важливим етапом є вибір великої мовної моделі (LLM), яка здатна забезпечити генерацію команд, аналіз запитів користувача та адаптацію маршруту агентів у реальному часі. Оскільки система повинна функціонувати локально та не залежати від хмарних сервісів, особливу увагу приділено моделям, оптимізованим для роботи на персональних комп’ютерах із використанням CPU або GPU.

Для порівняння було обрано п’ять сучасних LLM, які активно застосовуються у локальних сценаріях. Ключові критерії оцінювання включали продуктивність, швидкість генерації відповідей, підтримку структурованого формату JSON, можливість інтеграції в реальний час та оптимізованість для локального запуску.

Однією з найбільш ефективних моделей стала Mistral 7B [10–14], яка демонструє високу точність генерації та перевершує більші моделі за низкою параметрів. Вона забезпечує якісне виконання завдань з аналізу, генерації коду, обробки природної мови та формування складних команд. Особливо важливою для даної роботи є її здатність стабільно працювати на звичайному апаратному забезпеченні, включно з ноутбуками на базі процесорів Apple M1/M2.

Переваги моделі Mistral 7B:

– оптимізована під локальний запуск на CPU та GPU;

– висока швидкість відповіді, що важливо для керування платформами у реальному часі;

– підтримка формату JSON, необхідного для передачі структурованих команд бекенду;

– стабільна робота на середніх апаратних конфігураціях.

Недоліки:

– можливі помилки у складних логічних ланцюжках;

– потребує уважного налаштування промптів для уникнення неоднозначних відповідей.

У результаті аналізу було визначено, що Mistral 7B є оптимальним вибором для інтелектуального керування мобільними платформами в локальному середовищі. Вона забезпечує баланс між швидкодією, точністю та можливістю інтеграції у розподілену систему, що використовує як бездротовий, так і оптоволоконний канал зв’язку.

2.2 Вибір мов для Back-end

Розроблення серверної частини інтелектуальної системи керування мобільними оптоволоконними платформами вимагає застосування мови програмування, що забезпечує високу продуктивність, простоту інтеграції з мовними моделями та можливість обробки великої кількості подій у режимі реального часу. Важливими вимогами до Back-end складової є асинхронність, масштабованість, надійність та здатність забезпечувати стабільний обмін даними між клієнтом, LLM-модулем та агентами.

У межах цього проєкту ключовою мовою серверної частини було обрано Python, що зумовлено його універсальністю та широкою підтримкою інструментів для роботи зі штучним інтелектом, WebSocket-з’єднаннями і базами даних. Python має розвинену екосистему бібліотек, що забезпечують інтеграцію з локальними мовними моделями через фреймворки типу Ollama, а також пропонує зручні механізми асинхронної розробки завдяки модулям asyncio та FastAP*I*.

Окрім того, Python надає широкий вибір бібліотек для роботи з мережею, шифруванням, обробкою повідомлень та моделюванням затримок — функцій, критично важливих для симуляції оптоволоконного й бездротового каналу зв’язку. Завдяки цьому серверна частина здатна

обробляти запити у реальному часі, синхронізувати дані про стан платформ, опрацьовувати маршрути та формувати оновлення карти.

Також важливим чинником вибору стала можливість швидкого розгортання та гнучкого масштабування серверної частини. Python сумісний з інструментами контейнеризації, що дозволяє легко переносити систему на інші середовища, розділяти компоненти за мікросервісним принципом та забезпечувати незалежність окремих модулів при оновленні чи тестуванні.

Таким чином, використання Python у серверній частині системи є оптимальним рішенням, оскільки ця мова забезпечує необхідну продуктивність, зручність обслуговування та повну сумісність із технологіями, що формують інтелектуальне ядро системи.

2.2.1 Мікросервісна архітектура

Мікросервісна архітектура є одним із найпоширеніших підходів до побудови сучасних розподілених інформаційних систем. Вона передбачає поділ застосунку на набір окремих незалежних сервісів, кожен з яких відповідає за конкретний функціональний блок та має власний життєвий цикл [15]. Така структура дозволяє спростити розробку, підвищити масштабованість та забезпечити високу надійність системи в умовах зростання її складності.

У контексті інтелектуальної системи керування мобільними оптоволоконними платформами мікросервісна архітектура відіграє ключову роль. Оскільки система включає низку взаємопов’язаних компонентів — модуль маршрутизації, модуль аналізу даних, LLM-сервіс, модуль оптоволоконного зв’язку, інтерфейс моніторингу, базу даних та центр телеметрії — поділ на мікросервіси дозволяє організувати їхню роботу так, щоб кожен з компонентів функціонував незалежно. Це особливо важливо, коли частина системи працює через стабільний оптоволоконний канал, а інша — через менш надійний бездротовий зв’язок із можливістю глушіння.

По-перше, мікросервісна архітектура забезпечує високу гнучкість розвитку системи. Нові функціональні модулі, наприклад симулятор зон радіоперешкод або сервіс аналізу затримок у каналі зв’язку, можуть бути додані без втручання в роботу існуючих компонентів. Розробник може оновлювати, вдосконалювати або замінювати окремі частини системи, не змінюючи її загальної структури. Це особливо цінно у проєктах, де інтенсивно проводяться експерименти або постійно змінюються алгоритми керування та обробки даних.

По-друге, важливим аспектом є підвищення надійності. У випадку виникнення помилки в одному з сервісів, компанент не блокує роботу всієї системи. Наприклад, навіть якщо модуль бездротового зв’язку тимчасово перестає відповідати через симульоване глушіння сигналу, інші сервіси — LLM-аналітика, база даних, оптоволоконний канал, інтерфейс керування — продовжуватимуть працювати в штатному режимі. Це дозволяє гарантувати стабільність функціонування системи навіть у критичних умовах.

По-третє, мікросервіси дозволяють досягти високої масштабованості. Якщо навантаження на певний компонент збільшується (наприклад, зростає кількість платформ або обсяг телеметричних даних), саме цей сервіс може бути розгорнутий у додаткових екземплярах. Це робиться незалежно від інших підсистем, що дозволяє економити ресурси та збільшувати продуктивність лише там, де це необхідно.

У даній роботі застосунок складається з низки незалежних модулів:

1) Сервіс телеметрії — відповідає за збір і передачу даних про стан платформ у реальному часі;

2) Модуль маршрутного планування — приймає рішення щодо оптимального шляху з урахуванням зон перешкод і стабільності каналу зв’язку;

3) Сервіс симуляції оптоволоконного з’єднання — моделює мінімальні затримки, гарантує стабільність сигналу та імітує переваги fiber-зв’язку над wireless;

4) Сервіс зон глушіння — визначає, чи перебуває платформа у небезпечній ділянці, та обробляє втрату зв’язку;

5) Фронтенд-сервіс — відповідає за інтерфейс, побудований на React та Google Maps API;

6) База даних MongoDB — зберігає історію переміщення платформ, маршрути, тип зв’язку та інші параметри.

Кожен із цих сервісів працює окремо, що дозволяє легко керувати складністю системи та поступово розширювати її функціональні можливості, не проводячи повну перебудову структури.

Таким чином, мікросервісна архітектура є оптимальним рішенням для створення інтелектуальної системи керування розподіленими мобільними платформами. Вона забезпечує стійкість, масштабованість, гнучкість та можливість постійного розвитку, що є критично важливими для сучасних робототехнічних комплексів та симуляційних платформ.

2.2.2 Мова програмування Python у серверній частині системи

Для реалізації серверної частини інтелектуальної системи керування мобільними оптоволоконними платформами була обрана мова програмування Python (рис. 2.1), що є однією з найпопулярніших та найгнучкіших технологій у сфері штучного інтелекту, симуляційних систем та веброзробки [16]. Python поєднує простоту синтаксису, широкі можливості асинхронної обробки даних та потужну екосистему бібліотек, що дозволяє швидко створювати надійні та масштабовані мережеві сервіси.

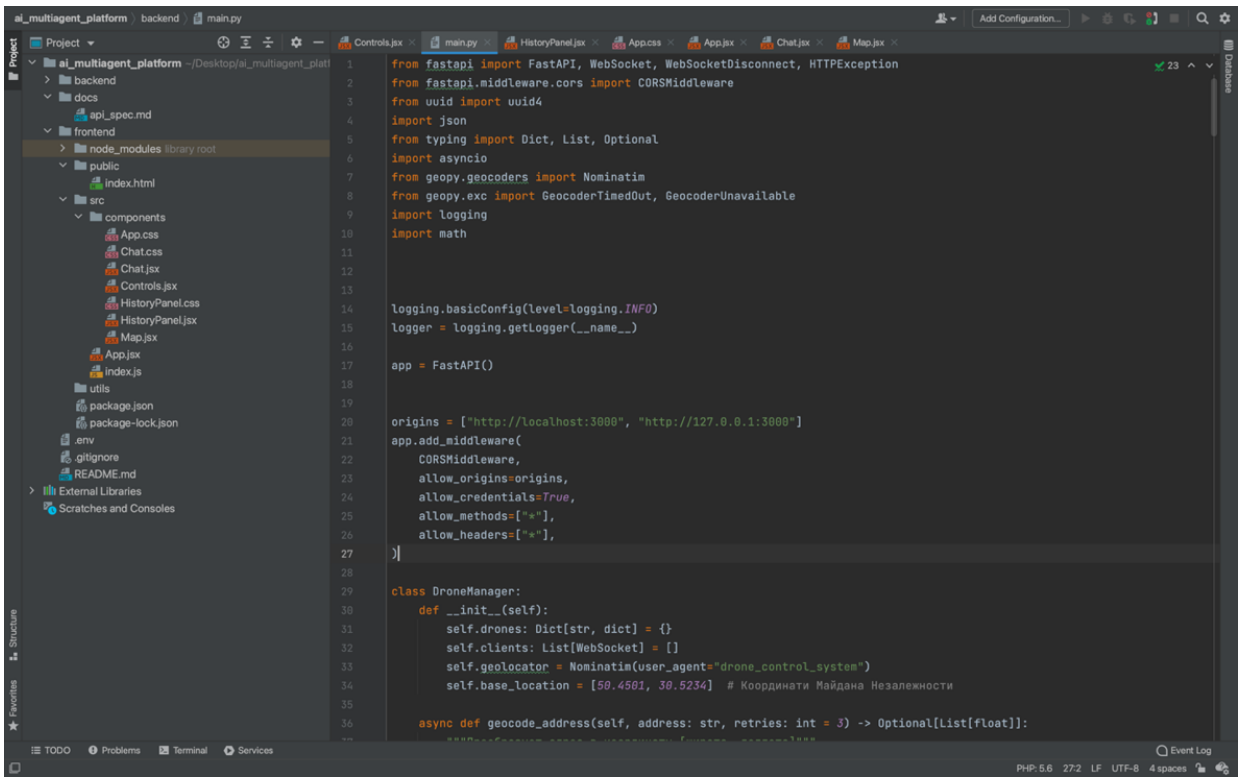

Рисунок 2.1 – Приклад Python коду

Однією з ключових переваг Python є його здатність працювати із великими обсягами даних у реальному часі. У даній роботі це відіграє особливо важливу роль, оскільки система має обробляти телеметрію від платформ, аналізувати стабільність каналу зв’язку, передавати оновлення клієнту та взаємодіяти з мовною моделлю одночасно. Завдяки вбудованим можливостям асинхронності (*asyncio*) та високопродуктивному фреймворку FastAPI, Python дозволяє ефективно працювати з WebSocket-з’єднаннями, мінімізувати затримку обробки даних та забезпечувати безперервний обмін повідомленнями.

Ще однією причиною вибору Python стала його тісна інтеграція з інструментами штучного інтелекту. Бібліотеки та платформи, такі як Ollama, PyTorch, TensorFlow, забезпечують можливість напряму працювати з мовними моделями, що дозволяє легко реалізувати інтелектуальний модуль, відповідальний за аналіз команд оператора та формування адаптивних маршрутів. Завдяки цьому система може не просто виконувати заздалегідь

визначені алгоритми, а й приймати рішення у реальному часі з урахуванням змін у каналі зв’язку та зовнішніх умов.

Крім того, Python забезпечує зручну взаємодію з базами даних, зокрема з MongoDB через асинхронну бібліотеку Motor. Це дозволяє зберігати інформацію про платформи, їх маршрути, параметри зв’язку, історію подій та стани системи без надмірних затримок. Оскільки MongoDB добре працює з JSON-подібними структурами, Python легко формує й обробляє такі документи, забезпечуючи швидкість та простоту розробки.

Нарешті, важливою перевагою Python є його підтримка контейнеризації за допомогою Docker, що суттєво спрощує розгортання окремих мікросервісів, їх оновлення та масштабування. Завдяки цьому розроблена система може бути легко перенесена у будь-яке середовище — як локальне, так і хмарне.

Завдяки своїй гнучкості, потужності та багатій екосистемі Python став оптимальним вибором для серверної частини інтелектуальної системи керування мобільними оптоволоконними платформами, забезпечуючи стабільність, продуктивність і можливість подальшого розширення функціональності.

2.2.3 Використання Flask у серверній частині системи

У рамках розроблення інтелектуальної системи керування мобільними оптоволоконними платформами важливим елементом серверної частини став вебфреймворк Flask, який був використаний на початковому етапі створення застосунку для реалізації REST-інтерфейсів та допоміжних сервісів. Flask є легким, модульним і гнучким інструментом, який надає розробнику можливість швидко створювати вебсервіси з мінімальними накладними витратами [17].

Попри те, що основний серверний модуль у магістерській роботі реалізовано на FastAPI через необхідність повноцінної підтримки

асинхронних WebSocket-з’єднань, Flask продовжує відігравати важливу роль у структурі системи як допоміжний сервіс. Він забезпечує обробку низки задач, що не потребують високого рівня паралельності, проте вимагають стабільного HTTP-взаємодії, зокрема:

– надання доступу до статичних ресурсів;
– виконання службових API-запитів;
– обробка допоміжних операцій контролю доступу;
– взаємодія із внутрішніми мікросервісами;
– обслуговування тестових або діагностичних маршрутів.

Однією з важливих причин використання Flask стала його простота та модульність. Фреймворк дозволяє розширювати функціональність за допомогою численних плагінів та бібліотек, що пришвидшує розробку і зменшує обсяг необхідного шаблонного коду. Для проєкту, який включає взаємодію з мовною моделлю, обробку телеметрії та роботу з оптоволоконним симулятором, це є суттєвою перевагою.

Також важливо відзначити, що Flask має низькі накладні витрати, що робить його ідеальним для мікросервісів, які виконують локальні або сервісні завдання. У межах системи він може відповідати за модулі, де не потрібна складна логіка маршрутизації або паралельність, але необхідна стабільність і мінімальний час запуску.

Ще однією перевагою Flask є його сумісність із Docker-контейнерами, що дозволяє легко розгортати та оновлювати відповідні мікросервіси. Це добре узгоджується з мікросервісною архітектурою системи та дає можливість масштабувати рішення при зростанні кількості мобільних платформ або розширенні інтелектуальних функцій.

Таким чином, Flask у даному проєкті виконує роль легкого, надійного та гнучкого інструменту для реалізації допоміжних серверних модулів. Він доповнює FastAPI та дозволяє створити розподілену інтелектуальну систему, оптимізовану під різні типи навантажень та сценарії взаємодії.

2.3 Вибір програмних рішень для Front-end

Для реалізації інтерфейсу інтелектуальної системи керування мобільними оптоволоконними платформами було обрано сучасні web-технології, що забезпечують високу продуктивність, інтерактивність та можливість обробки даних у реальному часі [18, 19]. Основною метою фронтенд-частини є надання користувачеві зручного інструменту для моніторингу стану платформ, візуалізації їх маршрутів, контролю параметрів зв’язку та взаємодії із системою штучного інтелекту.

Ключовим серед вибраних технологічних рішень став React, який є одним з найпопулярніших JavaScript-фреймворків для побудови інтерфейсів. React забезпечує декларативний підхід до розробки, компонентну структуру та високу швидкодію, що робить його оптимальним для систем, у яких постійно змінюється стан елементів інтерфейсу. Оскільки в даній роботі велика кількість даних оновлюється у режимі реального часу через WebSocket-канал (позиції платформ, статуси зв’язку, події у зонах глушіння), React дає змогу легко керувати цими оновленнями та ефективно повторно рендерити лише ті частини інтерфейсу, які змінюються.

Важливою складовою фронтенду є інтеграція з Google Maps API, що використовується для візуалізації карти та відображення переміщення платформ. Google Maps надає точні геодані, гнучкі інструменти побудови маршрутів, маркерів та графічних елементів, що дозволяє реалізувати інтуїтивно зрозумілий інтерфейс. У межах магістерської роботи карта також використовується для відображення спеціальних зон — ділянок глушіння сигналу, де платформи з бездротовим зв’язком втрачають можливість передавати телеметрію, на відміну від платформ з оптоволоконним каналом, які залишаються повністю керованими.

Для зв’язку з серверною частиною було обрано технологію WebSocket, що забезпечує двосторонній обмін даними у реальному часі. Це дозволяє отримувати оновлення про стан кожної платформи без необхідності постійно

надсилати HTTP-запити, зменшуючи навантаження на сервер та збільшуючи швидкодію інтерфейсу. React у поєднанні з WebSocket робить можливим відображення плавного руху платформ на карті та оновлення їх статусів без затримок [20, 21].

Крім основних технологій, фронтенд-частина включає систему керування станом та обробки подій. Використання хуків React (таких як useState і useEffect) дозволило організувати логіку роботи інтерфейсу так, щоб усі зміни у каналі зв’язку (наприклад, збільшення затримки, вихід із зони глушіння, перемикання типу зв’язку) одразу відображались на екрані користувача. Це підвищує інформативність системи та забезпечує оперативну реакцію оператора.

Таким чином, вибір програмних рішень для фронтенд-частини був зумовлений необхідністю організувати зручний, швидкий та надійний інтерфейс для роботи з інтелектуальною системою керування [22]. Обрані технології React, Google Maps API та WebSocket дозволили реалізувати ефективну візуалізацію даних, їх динамічне оновлення та максимально точне відображення стану платформ у реальному часі.

2.3.1 JavaScript

JavaScript є основною мовою програмування для розробки фронтенд-частини інтелектуальних вебсистем, зокрема таких, що працюють у режимі реального часу та потребують динамічної взаємодії з користувачем. У межах створення інтелектуальної системи керування мобільними оптоволоконними платформами JavaScript відіграє ключову роль, забезпечуючи інтерактивність інтерфейсу, обмін даними з сервером, обробку телеметрії та оновлення елементів карти у реальному часі.

JavaScript було обрано завдяки його широкій підтримці браузерами, високій гнучкості та можливості інтеграції з сучасними інструментами й фреймворками (рис. 2.2). У поєднанні з бібліотекою React він дозволяє

будувати компонентну структуру інтерфейсу, де кожен компонент відповідає за власний стан та поведінку. Такий підхід особливо ефективний для системи, що відображає численні рухомі платформи, зони глушіння, параметри зв’язку та інші динамічні елементи.

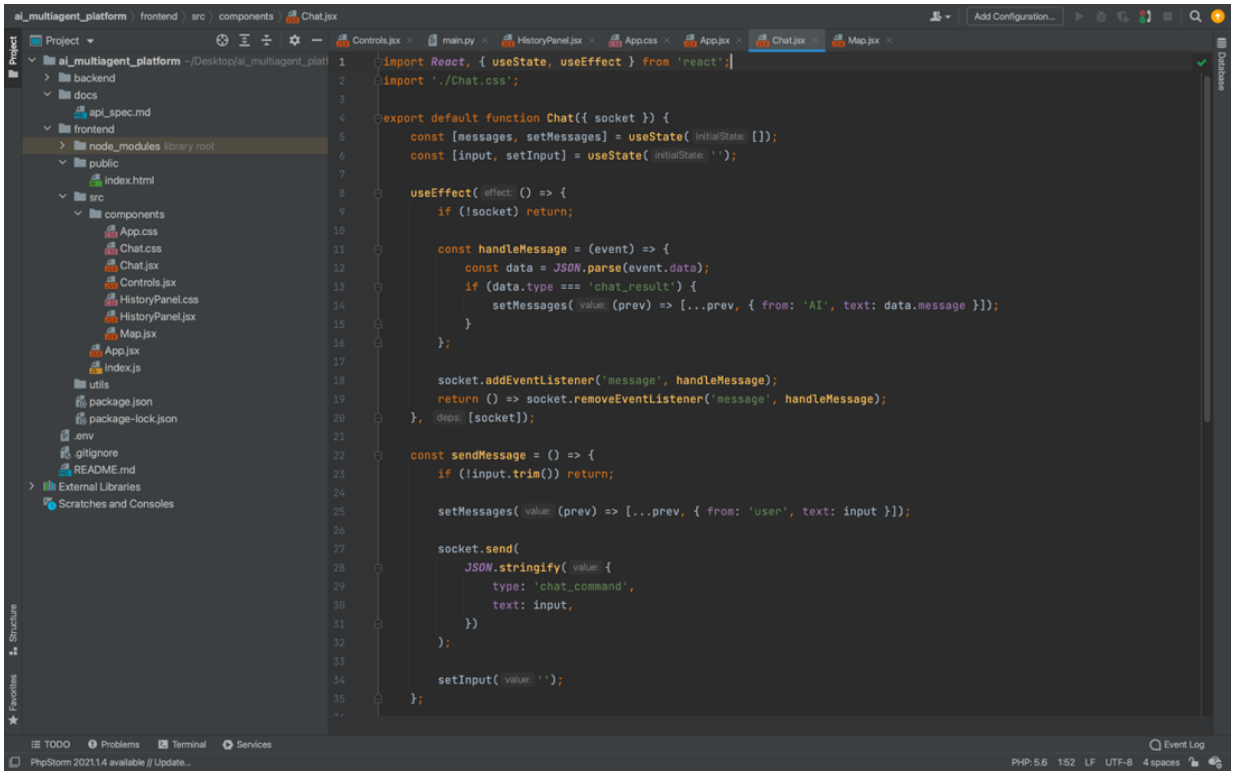

Рисунок 2.2 – Приклад JS коду

Однією з основних переваг JavaScript є можливість роботи з асинхронними потоками даних через механізми WebSocket, які забезпечують негайну передачу телеметрії та подій від серверної частини. Завдяки цьому інтерфейс може у реальному часі оновлювати:

– координати платформ;
– статуси зв’язку (оптоволоконний/бездротовий);
– рівень сигналу та затримки;
– інформацію про перебування у зоні глушіння;
– зміни маршруту, сформовані LLM-модулем.

Без використання JavaScript такі оновлення вимагали б постійного перезавантаження сторінки або ручних запитів до сервера, що суттєво зменшувало б зручність та швидкодію системи.

Крім того, JavaScript забезпечує тісну інтеграцію з Google Maps API, який використовується для відображення картографічної інформації. Мова дозволяє динамічно створювати маркери, полігони зон глушіння, полілайни маршрутів та інші графічні елементи, що візуалізують роботу системи. Завдяки цьому оператор отримує можливість спостерігати за переміщенням платформ у зручному, інтуїтивно зрозумілому інтерфейсі.

Важливо відзначити, що JavaScript також забезпечує точну обробку подій користувача: зміни налаштувань, вибір типу зв’язку, активація симуляції оптоволоконного каналу, встановлення погодних чи перешкодних зон на карті тощо. Це надає користувачеві повний контроль над системою та дозволяє інтерактивно взаємодіяти зі сценаріями симуляції.

Таким чином, JavaScript став фундаментальним інструментом для фронтенд-частини інтелектуальної системи керування мобільними платформами. Він забезпечує високу швидкодію, гнучкість, підтримку складної логіки у реальному часі та ефективну інтеграцію із серверною частиною, завдяки чому користувацький інтерфейс є стабільним, адаптивним та зручним у роботі.

2.3.2 React

React є одним із найпоширеніших інструментів для створення сучасних вебінтерфейсів, особливо у системах, де необхідно забезпечити високу динамічність, інтерактивність та швидку реакцію інтерфейсу на зовнішні події. У межах інтелектуальної системи керування мобільними оптоволоконними платформами React відіграє центральну роль у формуванні фронтенд-частини, яка забезпечує відображення даних у реальному часі, оновлення станів платформ та взаємодію користувача з системою.

React був обраний завдяки своїй компонентній архітектурі, що дозволяє розділити інтерфейс на незалежні модулі — карти, панелі керування, інформаційні блоки, списки платформ, карти затримок та інші елементи. Кожен компонент має власний стан і логіку, а тому може оновлюватись незалежно від інших, що є критично важливим у системах, де одночасно змінюється велика кількість даних.

Застосунок побудований таким чином, щоб у режимі реального часу отримувати інформацію про переміщення платформ, параметри зв’язку та події у мережі. Завдяки ефективній роботі з хуками React (useState, useEffect, useMemo) інтерфейс оперативно реагує на зміни, забезпечуючи плавність анімацій і відсутність затримок під час оновлення даних. Це дозволяє оператору постійно бачити актуальний стан системи, що є важливим для прийняття рішень у сценаріях симуляції.

Інтеграція React із технологією WebSocket дозволяє реалізувати двосторонній зв’язок між клієнтською та серверною частиною, що забезпечує миттєве надходження нових даних. React обробляє ці дані та оновлює компоненти без необхідності перезавантаження сторінки, що робить інтерфейс більш стабільним і придатним для роботи в умовах великої кількості оновлень.

Важливою перевагою React є його сумісність з Google Maps API, який використовується для візуалізації переміщення платформ на карті. За допомогою спеціальних бібліотек (наприклад, @react-google-maps/api) стало можливим створювати маркери, полігони зон глушіння, маршрути, а також динамічно змінювати їх колір, форму або поведінку відповідно до стану зв’язку (оптоволоконний / бездротовий) або непередбачених подій у процесі симуляції.

React також забезпечує зручні механізми для створення адаптивних інтерфейсів, що дозволяє використовувати систему на різних пристроях — від стаціонарних комп’ютерів до планшетів. Це є важливим аспектом у

системах моніторингу та керування, де оператор може працювати в різних умовах.

Завдяки своїй продуктивності, масштабованості та гнучкості React став оптимальним вибором для фронтенд-частини системи. Він дозволив реалізувати не просто візуальний інтерфейс, а інтерактивну платформу для аналізу, моніторингу та керування мобільними системами у реальному часі, що є важливим компонентом інтелектуального багатоагентного комплексу.

2.3.3 Google Maps API

Одним із ключових елементів фронтенд-частини інтелектуальної системи керування мобільними оптоволоконними платформами є інтеграція з Google Maps API, що забезпечує можливість візуалізації географічної інформації та відображення руху платформ у реальному часі. Карта є центральним інструментом взаємодії оператора із системою, оскільки дозволяє контролювати місцезнаходження, маршрути, зони впливу перешкод та поточний стан каналів зв’язку.

Google Maps API було обрано завдяки його високій точності, надійності, широкому набору функцій та підтримці динамічного рендерингу геооб’єктів. У межах цього проєкту API використовується для відображення таких елементів, як:

– маршрути руху платформ, згенеровані мовною моделлю або задані оператором;

– позиції мобільних платформ, що оновлюються у реальному часі через WebSocket;

– зони глушіння сигналу, що впливають на бездротові платформи;

– візуальні індикатори типу зв’язку (оптоволоконний/бездротовий);

– сектори впливу погодних або перешкодних умов, які також змінюють поведінку платформ.

Google Maps API підтримує роботу з різними типами геометричних об’єктів — полілініями, полігонами, маркерами, колами — що дозволяє створювати на карті складні інтерактивні моделі. Наприклад, зони глушіння можуть бути представлені у вигляді напівпрозорих полігонів із різними кольорами, а маршрути платформ — у вигляді сегментованих ліній, колір яких змінюється відповідно до параметрів каналу зв’язку або поведінки системи.

Перевагою Google Maps API є його висока продуктивність і стабільність при роботі з великою кількістю об’єктів, що критично для симуляційних систем. Навіть за наявності багатьох рухомих платформ, численних подій і постійних оновлень, API забезпечує плавну роботу інтерфейсу без видимих затримок. Це стало можливим завдяки внутрішнім оптимізаціям Google Maps та використанню React-компонентів, які відповідають за ефективне повторне рендерування.

Крім того, Google Maps API дозволяє здійснювати зворотну взаємодію користувача з картою: встановлення нових зон перешкод, додавання цільових точок маршруту, зміна параметрів симуляції та ініціювання нових команд для платформ. Така інтерактивність забезпечує високий рівень гнучкості системи та дозволяє проводити складні експерименти з маршрутизацією та зв’язком.

Використання Google Maps API зробило можливим створення інтуїтивно зрозумілого інтерфейсу, який не тільки відображає поточний стан системи, але й дозволяє оперативно реагувати на зміну умов. Це є важливою складовою інтелектуальної системи, оскільки від швидкості і точності візуалізації залежить ефективність управління мобільними платформами та якість аналітичних рішень оператора.

Таким чином, Google Maps API став невід’ємною частиною фронтенд-архітектури, забезпечивши високоточну і зручну картографічну основу для візуалізації роботи мобільних оптоволоконних платформ у реальному часі.

2.4 Вибір бази даних для вебзастосунку

Одним із ключових компонентів інтелектуальної системи керування мобільними оптоволоконними платформами є база даних, яка відповідає за зберігання інформації про платформи, маршрути, телеметрію, параметри зв’язку та історію взаємодії користувача із системою. Оскільки система працює в режимі реального часу та оперує потоками даних, які постійно оновлюються, вибір бази даних має забезпечувати високу швидкість запису й читання, гнучкість структури та можливість масштабування.

Для реалізації цих вимог у проєкті була обрана MongoDB — документноорієнтована база даних, що використовує формат JSON-подібних документів та забезпечує просте, швидке й ефективне зберігання складних структур даних. MongoDB є популярним рішенням для сучасних вебзастосунків, особливо тих, що працюють з даними динамічної структури або мають мікросервісну архітектуру [23].

Однією з основних причин вибору MongoDB є її гнучка схема даних. У межах симуляції мобільних платформ структура інформації може змінюватися залежно від умов: змінюється кількість параметрів зв’язку, додаються нові типи подій або дані телеметрії. Документноорієнтований підхід дозволяє не обмежувати систему жорсткою структурою таблиць, як це було б у реляційних базах даних. Натомість кожен документ може містити унікальні набори параметрів, що робить MongoDB ідеальною для реалізації інтелектуальних та симуляційних систем.

Особливо важливо, що MongoDB добре працює з великими потоками даних, які генеруються під час пересування платформ. При кожному оновленні координат, зміні стану зв’язку або виявленні потрапляння в зону глушіння система передає телеметрію до сервера. Завдяки оптимізованим механізмам вставки та індексації MongoDB може обробляти це навантаження без відчутних затримок.

У магістерській роботі також важливим є моделювання різних типів зв’язку — оптоволоконного та бездротового. MongoDB дозволяє зберігати параметри каналу зв’язку, такі як:

– поточна затримка (latency);
– рівень сигналу;
– тип інтерфейсу (fiber / wireless);
– історія втрат зв’язку;
– точки входу та виходу із зон глушіння.

Ці дані зберігаються у вигляді вкладених документів, що робить структуру бази простою для подальшої обробки на бекенді.

Крім того, MongoDB забезпечує підтримку високопродуктивних асинхронних клієнтів, таких як Motor, який використовується у серверній частині проєкту (FastAPI). Motor дозволяє обробляти десятки тисяч операцій вводу та виводу без блокування процесу, що є критично важливим у системах із WebSocket-з’єднаннями. Це забезпечує стабільність роботи всієї платформи та дозволяє оновлювати дані в режимі реального часу.

Ще однією перевагою MongoDB є її зручна інтеграція з контейнеризаційними технологіями, що використовуються в мікросервісній архітектурі. Сервіс БД може бути розгорнутий окремо, масштабований при необхідності або резервований без втручання у роботу інших модулів.

Таким чином, MongoDB забезпечує всі необхідні властивості для ефективного функціонування інтелектуальної системи керування мобільними оптоволоконними платформами — високу продуктивність, гнучкість структури, масштабованість, надійність та оптимальну взаємодію з асинхронною серверною частиною.

2.5 Вибір загальних рішень для вебзастосунку

Розробка інтелектуальної системи керування мобільними оптоволоконними платформами потребує комплексного підходу до вибору технологій та загальних архітектурних рішень. Оскільки система функціонує у режимі реального часу, обробляє великий потік телеметричних даних, працює з мовною моделлю та модулем симуляції оптоволоконного й бездротового зв’язку, важливо забезпечити оптимальне поєднання продуктивності, надійності та гнучкості. Тому вибір загальних рішень ґрунтувався на кількох ключових принципах: модульність, масштабованість, асинхронність та підтримка мікросервісної архітектури.

Одним із базових рішень стало використання асинхронних технологій, що дозволяють обробляти події від платформ у реальному часі без блокування основного процесу. Завдяки підтримці асинхронності у FastAPI та використанню WebSocket-протоколу система здатна забезпечувати миттєве оновлення даних на клієнтській стороні, що є критично важливим для візуалізації переміщення платформ, фіксації змін у каналі зв’язку та реагування на події в зоні глушіння. Асинхронний підхід дозволяє розподіляти навантаження між серверами та забезпечує плавну роботу навіть при значному обсязі телеметрії.

Важливою складовою загальних рішень стало застосування мікросервісної архітектури, що дозволяє розділити функціональність системи на незалежні модулі: сервер керування, модуль LLM-аналітики, симулятор каналів зв’язку, база даних, фронтенд тощо. Такий підхід забезпечує легке оновлення окремих компонентів, гнучке масштабування та підвищену надійність у разі збою одного з модулів. Завдяки цьому можливо розвивати систему поступово, додаючи нові функції, такі як розширена симуляція погодних умов або аналіз стану каналу зв’язку.

Ще одним загальним рішенням стало використання WebSocket-технології для забезпечення постійного двостороннього зв’язку між клієнтом

та сервером. Це дає змогу надсилати оновлення на фронтенд без затримок, а також забезпечує стабільність зв’язку у сценаріях, де необхідна швидка реакція на зміни в середовищі. WebSocket є критично важливим для відображення роботи системи, оскільки дозволяє інтегрувати оптоволоконний та бездротовий канали й моделювати їх вплив на поведінку платформ.

Також значну увагу було приділено питанням структурованості та підтримуваності коду. Принципи чистого коду (Clean Code), поділ логіки на окремі слої (service layer, repository layer, model layer), а також документування API через OpenAPI сприяють спрощенню розробки, тестування та подальшої модернізації системи. Завдяки цьому проєкт є розширюваним і може адаптуватися до нових вимог, що є важливим для магістерської роботи, орієнтованої на дослідження та впровадження нових рішень.

Не менш важливим загальним рішенням стала інтеграція з контейнеризаційною технологією Docker, яка дозволяє розгортати серверні модулі та базу даних у стандартизованому середовищі. Це забезпечує стабільність, незалежність від локальних налаштувань та можливість переносити систему на інші сервери або у хмарні сервіси без змін у конфігурації.

Вибір цих загальних рішень дозволив створити гнучку, надійну та масштабовану інтелектуальну систему, здатну ефективно працювати у середовищі з різними типами каналів зв’язку та змінними умовами. Система готова до подальших удосконалень, включаючи розширення симуляції оптоволоконних мереж, прогнозування затримки та розширення функціоналу, пов’язаного з адаптивною маршрутизацією мобільних платформ.

2.5.1 WebSocket

Для забезпечення роботи системи в режимі реального часу та підтримки постійної взаємодії між клієнтом і сервером було обрано технологію WebSocket [24]. На відміну від традиційних HTTP-запитів, WebSocket забезпечує двосторонній канал зв’язку, що дозволяє серверу миттєво надсилати оновлення клієнту без додаткових запитів. Це є критично важливим для інтелектуальної системи керування мобільними оптоволоконними платформами, де позиції, стан каналів зв’язку та телеметрія змінюються з високою частотою.

У межах даної роботи WebSocket використовується для передачі таких даних:

– координат платформ у реальному часі;

– параметрів зв’язку (latency, тип інтерфейсу, рівень сигналу);

– подій, пов’язаних із зонами глушіння;

– змін у маршруті, сформованих мовною моделлю;

– аналітичної інформації про стабільність оптоволоконного або бездротового каналу.

Особливо важливо, що WebSocket дозволяє моделювати відмінності між оптоволоконним і бездротовим зв’язком. Наприклад, у бездротового каналу можуть виникати розриви, затримки або повна втрата зв’язку, що відображається на фронтенді одразу після надходження події. У той час як оптоволоконний канал забезпечує стабільність і мінімальну затримку, що також моделюється через WebSocket.

Таким чином, WebSocket є основою для реалізації інтерактивної,

2.6 Модуль оптоволоконного зв’язку в інтелектуальній системі керування мобільними платформами

Оптоволоконний зв’язок є одним із найнадійніших і найпродуктивніших сучасних способів передачі даних, що широко використовується в телекомунікаційних системах, критично важливій інфраструктурі, промисловій автоматизації та системах військового призначення. У контексті інтелектуальної системи керування мобільними платформами оптоволоконний канал забезпечує підвищену стабільність передачі сигналу, мінімальну затримку та стійкість до зовнішніх перешкод, що робить його важливим елементом у задачах високоточного контролю та моніторингу.

Оптоволокно працює на основі явища повного внутрішнього відбиття, завдяки якому світловий сигнал передається по серцевині волокна практично без втрат. Це забезпечує високу пропускну здатність та можливість передачі даних на великі відстані з мінімальним загасанням. На відміну від традиційних радіоканалів, оптичні волокна практично не піддаються електромагнітним впливам, що значно знижує ймовірність втрати сигналу, збоїв або небажаних перешкод.

У системах керування мобільними платформами це створює низку стратегічних переваг:

а) стабільність каналу зв’язку. На відміну від бездротових інтерфейсів, оптоволоконний канал не залежить від фізичних перешкод або погодних умов;

б) мінімальна затримка (latency). Це дозволяє забезпечити точні реакції платформи на події в реальному часі;

в) захищеність від глушіння та електромагнітних завад. Навіть у зонах сильного радіоперешкодження оптоволоконна платформа не втрачає керування;

г) висока пропускна здатність. Дає змогу передавати велику кількість телеметрії або відеодані без падіння якості.

У межах кваліфікаційної роботи модуль оптоволоконного зв’язку реалізовано у вигляді окремого функціонального компонента, що моделює фізичні властивості оптоволокна в умовах програмної симуляції. Кожна платформа в системі має параметр fiber: true/false, який визначає тип каналу зв’язку. Для платформ з активним оптоволоконним інтерфейсом застосовуються такі правила:

– затримка сигналу встановлюється на мінімальному рівні (умовно від 1 мс до 5 мс);

– втрати пакетів рівні нулю;

– платформа зберігає зв’язок навіть у випадку потрапляння до зони глушіння;

– дані надходять на сервер і клієнт більш стабільно, що позитивно впливає на роботу алгоритмів LLM.

Для бездротових платформ, навпаки, реалізовано можливість:

– збільшення затримки сигналу у залежності від умов симуляції;

– випадкового або закономірного packet loss;

– повного розриву зв’язку у зонах глушіння;

– нестабільності телеметрії під час перешкод.

Це дозволяє проводити експерименти, порівнювати ефективність різних каналів зв’язку та оцінювати вплив зовнішніх умов на точність керування платформами.

Модуль оптоволоконного зв’язку інтегровано у серверну частину системи та реалізовано через асинхронні механізми WebSocket. Тип каналу враховується у всіх обчисленнях, що дозволяє LLM-модулю формувати маршрути з урахуванням можливих втрат зв’язку та змін latency. Крім того, на інтерактивній мапі відображається тип зв’язку кожної платформи, що надає оператору можливість оцінювати їх поведінку залежно від умов симуляції.

Таким чином, модуль оптоволоконного зв’язку є ключовим доповненням інтелектуальної системи, що відрізняє магістерську роботу від бакалаврської. Його впровадження дозволяє моделювати реальні практичні сценарії керування платформами, виявляти обмеження бездротового зв’язку та демонструвати переваги стабільних каналів комунікації у багатоагентних системах.

3 РОЗРОБКА ВЕБ ЗАСТОСУНКУ

3.1 Вирішення програмних рішень для Back-end

Back-end частина інтелектуальної системи керування мобільними оптоволоконними платформами є ключовим функціональним елементом, що забезпечує обробку даних, взаємодію між компонентами, роботу алгоритмів штучного інтелекту та синхронізацію інформації в реальному часі. Оскільки система має підтримувати одночасне керування кількома платформами, моделювати різні типи каналів зв’язку та забезпечувати стабільну роботу під навантаженням, вибір технологій для серверної частини відіграє визначальну роль.

Основними вимогами до Back-end є:

– висока продуктивність і низька затримка обробки даних, що особливо важливо при моделюванні оптоволоконного та бездротового зв’язку;

– масштабованість і можливість додавання нових модулів, зокрема сервісів LLM, WebSocket-комунікації, обчислювальних алгоритмів і засобів аналітики;

– надійність та стійкість до відмов, що забезпечується мікросервісною архітектурою;

– асинхронність обробки подій, необхідна для реального часу та роботи кількох платформ;

– зручність інтеграції з Front-end частиною та високий рівень безпеки під час роботи з користувачами.

Зважаючи на такі вимоги, серверна частина системи була побудована на комбінації кількох технологій, кожна з яких виконує свою спеціалізовану роль:

– Flask використовується для реалізації базових REST-ендпоінтів і прототипування структур системи;

– FastAPI — як високопродуктивний асинхронний фреймворк, що обробляє критично важливі запити та забезпечує роботу з телеметрією платформ;

– MongoDB — як документоорієнтована база даних для зберігання станів платформ, користувачів, маршрутів та історії переміщень;

– WebSocket-модуль — для забезпечення двонапрямного обміну даними в реальному часі;

– модуль LLM (Ollama + Mistral 7B) — для генерації динамічних маршрутів і прийняття рішень;

– модуль моделювання оптоволоконного каналу — для відтворення властивостей стабільного зв’язку з мінімальною затримкою.

Обрана архітектура дозволяє реалізувати інтелектуальну систему, яка може адаптуватися до різних типів середовищ, моделювати реалістичні сценарії взаємодії та забезпечувати високу точність керування мобільними платформами. Використання Python як основної серверної технології додатково дає переваги в плані обчислень, інтеграції з бібліотеками машинного навчання та побудови алгоритмів оптимізації (рис. 3.1).

Таким чином, рішення щодо Back-end побудовані з урахуванням сучасних вимог до систем реального часу, високої пропускної здатності каналів зв’язку та масштабування інтелектуальних вебзастосунків.

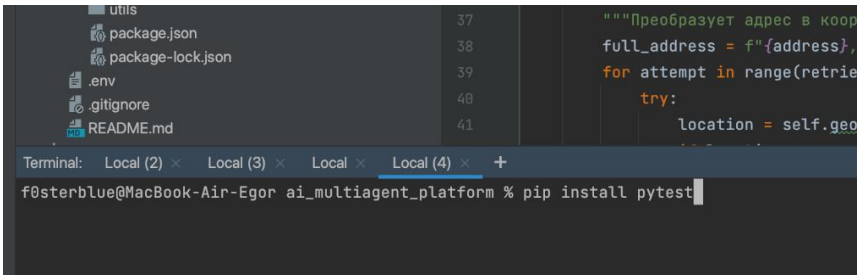

Рисунок 3.1 – Команда для встановлення Python у застосунок PhpStorm

3.1.1 Flask у розробці API

Фреймворк Flask є одним із найпоширеніших інструментів для створення веб-сервісів та API у Python. Його популярність обумовлена мінімалістичною архітектурою, гнучкістю та можливістю масштабування. Flask не нав’язує розробнику жорстких правил чи структур, що робить його зручним інструментом як для невеликих експериментальних систем, так і для побудови прототипів складних інтелектуальних застосунків.

У межах даного проєкту Flask використано для реалізації частини серверної логіки, що відповідає за обробку HTTP-запитів, інтеграцію з базою даних, авторизацію користувачів та початкові операції з мобільними платформами. Перевагою Flask є можливість швидко створювати REST API-інтерфейси, які можуть взаємодіяти з іншими компонентами системи — зокрема, мікросервісами FastAPI, модулем WebSocket, а також сервісами штучного інтелекту на основі моделі Mistral 7B.

Однією з важливих властивостей Flask є його модульність: розробник має змогу підключати тільки ті розширення, які дійсно потрібні. У структурі інтелектуальної системи керування мобільними платформами це дає такі переваги:

– мінімальне навантаження та висока швидкість реакції API;

– зручне розмежування логіки, де Flask використовується для базових сервісів;

– можливість гнучкої інтеграції з іншими технологіями, наприклад з FastAPI для більш високонавантажених модулів;

– спрощене тестування та налагодження, що є важливим для складних інтегрованих систем.

У межах розробки було створено набір REST-ендпоінтів для роботи з користувачами, платформами та телеметрією. Flask також підтримує взаємодію з MongoDB через асинхронні драйвери, хоча найбільш

продуктивні механізми взаємодії реалізовані у FastAPI-сервісі, який був інтегрований на наступному етапі.

Таким чином, Flask у цьому проєкті виступає як універсальний інструмент для реалізації базових API та є частиною комбінованої архітектури, яка поєднує простоту Flask та масштабованість FastAPI. Такий підхід дозволяє оптимізувати навантаження між різними компонентами серверної логіки та забезпечити стабільну роботу інтелектуальної системи навіть при збільшенні кількості підключених платформ і складності обробки даних.

3.1.2 FastAPI

FastAPI є сучасним високопродуктивним фреймворком для створення веб-сервісів на Python, що вирізняється асинхронною природою обробки запитів та низькою затримкою виконання. На відміну від Flask, який зазвичай використовується для синхронних REST-сервісів та невеликих мікромодулів, FastAPI орієнтований на побудову високонавантажених систем і мікросервісів, які повинні працювати в режимі реального часу. Саме ці властивості роблять його оптимальним рішенням для критичних компонентів інтелектуальної системи керування мобільними платформами.

У межах даного проєкту FastAPI виконує роль центрального серверного модуля, що забезпечує:

– обробку динамічної телеметрії мобільних платформ, яка безперервно надходить через WebSocket;

– керування маршрутами, що формуються моделлю Mistral 7B та адаптуються в реальному часі;

– взаємодію з MongoDB через асинхронні драйвери, що дає можливість підтримувати сотні подій без блокування процесів;

– обробку запитів, пов’язаних із моделюванням оптоволоконного каналу зв’язку, де визначаються затримки, типи сигналів та стійкість каналу;

– авторизацію та роботу користувачів, включно з сесіями, токенами та перевіркою доступу до платформ.

Однією з ключових переваг FastAPI є використання стандарту ASGI, що забезпечує істотне збільшення продуктивності при обробці високої кількості паралельних з’єднань. Це особливо важливо для системи, де кожна мобільна платформа передає координати з високою частотою, а алгоритми інтелектуального управління повинні оперативно реагувати на ці зміни.

FastAPI також надає автоматичну генерацію документації API у форматах OpenAPI та Swagger UI, що полегшує підтримку проєкту та інтеграцію з іншими сервісами. У магістерському проєкті це дозволило чітко структурувати всі ендпоінти, швидко тестувати їх роботу та інтегрувати різні модулі у єдину мікросервісну архітектуру.

Важливим аспектом є й те, що FastAPI легко поєднується з механізмами асинхронного WebSocket-комунікування, що дає змогу реалізувати повноцінний канал передачі даних у реальному часі. Для моделювання оптоволоконного каналу у FastAPI інтегровано логіку зміни затримки сигналу, визначення packet loss та обробки стабільності каналу залежно від того, чи платформа використовує оптоволоконний модуль.

Таким чином, FastAPI виступає критично важливим елементом серверної частини системи, забезпечуючи високу продуктивність, масштабованість та адаптивність. Його використання дає можливість створити інтелектуальний вебзастосунок, здатний обробляти велику кількість даних у реальному часі та підтримувати складні сценарії керування мобільними платформами.

3.2 Мікросервісна архітектура проєкту

Мікросервісна архітектура стала основою побудови інтелектуальної системи керування мобільними платформами, оскільки забезпечує гнучкість, масштабованість та високу стійкість до відмов. Такий підхід передбачає

поділ загальної системи на окремі незалежні модулі, кожен з яких виконує чітко визначену функцію. Це дозволило уникнути надмірної складності, характерної для монолітних систем, та спростило процес розгортання і модернізації компонентів під час розробки.

У межах цього проєкту було сформовано декілька ключових мікросервісів, що взаємодіють між собою через HTTP та WebSocket-протоколи. Базові REST-запити опрацьовуються окремим сервісом на Flask, який відповідає за обслуговування користувачів, роботу з загальною структурою проєкту та низку допоміжних операцій. Найбільш ресурсоємні процеси, пов’язані з обробкою телеметрії, виконуються асинхронним сервером на FastAPI, який здатен підтримувати велику кількість паралельних з’єднань та швидко реагувати на зміни стану платформ.

Окрему роль у системі відіграє сервіс взаємодії зі штучним інтелектом, де використовується модель Mistral 7B через Ollama. Він обробляє запити, пов’язані з адаптивним маршрутизаційним плануванням, і формує контрольні команди на основі аналізу середовища та телеметрії. Такий модуль працює автономно, що забезпечує можливість оновлювати або замінювати його без впливу на основну логіку системи [25, 26].

Передача даних у реальному часі реалізована через окремий WebSocket-сервіс, який підтримує двонапрямний обмін інформацією між клієнтом та сервером. Саме цей модуль забезпечує безперервне оновлення координат мобільних платформ, обробку їх статусів та оперативне реагування на зміну умов моделювання. Усі ці процеси відбуваються асинхронно, що дозволяє уникнути затримок та забезпечити стабільну роботу навіть при значному обсязі даних.

Особливе місце у загальній архітектурі займає модуль оптоволоконного зв’язку, який моделює властивості високостабільного каналу передачі інформації. Він визначає затримку сигналу, можливість втрати пакетів, а також імітує поведінку платформи в умовах перешкод або глушіння. Такий підхід дозволяє порівнювати ефективність різних каналів

зв’язку в однакових умовах та аналізувати, як тип зв’язку впливає на поведінку інтелектуальної системи.

Усі сервіси об’єднані через MongoDB, яка виконує функції централізованого сховища даних. База даних зберігає інформацію про користувачів, стан мобільних платформ, маршрути, історію переміщень і параметри зв’язку. Завдяки документоорієнтованій структурі MongoDB забезпечує швидку обробку запитів і зручне масштабування наборів даних.

Таким чином, мікросервісна архітектура дозволила побудувати систему, що легко розширюється, ефективно обробляє велику кількість паралельних процесів та забезпечує надійну взаємодію між усіма модулями в режимі реального часу. Вона створює умови для гнучкого розвитку проєкту, незалежного оновлення компонентів і підтримки складних алгоритмів керування мобільними платформами.

3.2.1 Створення бази даних

База даних є одним з ключових елементів інтелектуальної системи керування мобільними платформами, оскільки забезпечує збереження всіх основних структурованих даних, необхідних для роботи сервісів. Оскільки система функціонує у режимі реального часу та оперує великою кількістю телеметричної інформації, було важливо обрати рішення, здатне обробляти динамічні дані з високою частотою оновлення та одночасно підтримувати масштабованість. Для цього в проєкті використано документоорієнтовану базу даних MongoDB, яка оптимально підходить для роботи з великими потоками даних та складними JSON-подібними структурами.

MongoDB дозволяє гнучко зберігати інформацію про користувачів, стан мобільних платформ, маршрути та історію їх руху. Такий підхід особливо зручний у системі, де структура даних може змінюватися в процесі розвитку проєкту, а різні модулі формують дані з різною деталізацією. Документоорієнтована модель не потребує суворої фіксації схеми, що значно

спрощує обробку нових типів телеметричних записів, які можуть виникати під час інтеграції алгоритмів штучного інтелекту або моделювання оптоволоконного каналу.

Система передбачає зберігання даних про кожну платформу у вигляді окремих документів, що включають координати, статуси, інформацію про активний канал зв’язку, маршрутні точки та результати аналізу модулем LLM. Крім того, база даних містить журнали взаємодії користувачів, журнал команд, а також історичні записи про зміни поведінки платформи під впливом затримки сигналу або втрати зв’язку. Така структура дозволяє відтворювати повну картину роботи системи та проводити експериментальні дослідження, що необхідні для магістерської роботи.

Асинхронна взаємодія FastAPI з MongoDB реалізована через драйвер Motor, який забезпечує високу швидкість читання та запису даних без блокування основних потоків. Це дозволяє серверам, що відповідають за телеметрію, приймати та обробляти дані навіть при значному збільшенні кількості платформ. У свою чергу, інші мікросервіси можуть паралельно отримувати необхідну інформацію для обробки команд, аналізу маршруту або формування висновків у моделі штучного інтелекту.

Таким чином, побудова бази даних на основі MongoDB створила надійну основу для роботи всієї системи. Вона забезпечує швидкий доступ до інформації, дозволяє зберігати складні динамічні структури та підтримує масштабування, необхідне для експериментів у галузі інтелектуальних багатоагентних платформ та дослідження впливу каналу зв’язку на поведінку системи.

3.2.2 Сервіс для авторизації користувача

Сервіс авторизації користувачів є одним із базових компонентів системи, оскільки він забезпечує контроль доступу до функціональності вебзастосунку та визначає рівень взаємодії кожного оператора з мобільними

платформами. В умовах інтелектуальної системи керування, де кожна дія користувача може впливати на стан платформи, точність маршруту або поведінку алгоритмів штучного інтелекту, надійна система аутентифікації та авторизації є обов’язковою.

У проєкті реалізовано модуль, який відповідає за реєстрацію, вхід, генерування токенів доступу та валідацію сесій користувачів. Для цього використовується комбінація REST-ендпоінтів та механізмів перевірки даних, інтегрованих у Flask і FastAPI. Усі дані користувачів зберігаються у MongoDB, що дозволяє забезпечити гнучкість та можливість швидкого доступу до необхідної інформації. Важливою особливістю є застосування захищених методів хешування, які гарантують безпечне зберігання паролів та запобігають несанкціонованому доступу.

Система авторизації виконує також функцію контролю над розподілом мобільних платформ між різними користувачами. Це дозволяє уникнути ситуацій, коли декілька операторів одночасно намагаються керувати однією платформою. Кожному користувачу при вході в систему призначається унікальний ідентифікатор, який пов’язується з даними про закріплені за ним платформи. Такий підхід забезпечує коректність взаємодії між людиною та системою та зменшує ймовірність конфліктів, що можуть виникнути при відправленні команд.

Крім того, сервіс авторизації інтегрований у логіку WebSocket-з’єднань. Перш ніж користувач зможе отримувати дані телеметрії або надсилати команди в режимі реального часу, система перевіряє дійсність токена та відповідність прав доступу. Це дозволяє забезпечити стабільну роботу мережевої частини застосунку та запобігти небажаному втручанню в обчислювальні процеси. У поєднанні з модулем оптоволоконного зв’язку, який моделює різні типи каналів передачі даних, авторизаційний модуль гарантує, що доступ до керування платформами отримують лише авторизовані користувачі, а канал зв’язку між оператором і платформою залишається контрольованим.

Сервіс авторизації є необхідною складовою всієї архітектури застосунку. Він забезпечує не лише безпеку взаємодії, а й коректність роботи всіх алгоритмів, включно з LLM-модулем, який формує маршрути залежно від дій оператора. Така структура дозволяє підтримувати впорядковану та безпечну взаємодію між клієнтом, сервером і платформами в умовах реального часу.

3.2.3 Створення безпеки

Забезпечення безпеки є критично важливою складовою інтелектуальної системи керування мобільними платформами, оскільки будь-яке втручання в процес обміну даними або несанкціонований доступ до системи може призвести до некоректної роботи алгоритмів, втрати керування платформами або порушення цілісності телеметричних даних. У зв’язку з цим у проєкті була розроблена комплексна система безпеки, яка охоплює всі етапи взаємодії користувача із серверною частиною — від авторизації до обробки команд і передачі даних у реальному часі.

Одним із першочергових завдань було забезпечення захищеного зберігання даних користувачів. Для цього у сервісі використано надійні методи хешування паролів, які виключають можливість їх відновлення у випадку витоку або несанкціонованого доступу до бази даних. Також усі операції, пов’язані з реєстрацією та входом користувачів, супроводжуються валідацією введених даних та перевіркою їх відповідності встановленим правилам.

Безпека серверної взаємодії реалізована шляхом використання токенів доступу, що генеруються під час авторизації користувача. Вони служать підтвердженням прав доступу і дозволяють відокремлювати запити різних користувачів у системі. Кожен запит перевіряється на наявність валідного токена, що дає змогу забезпечити контроль над тим, хто має право керувати платформами або отримувати їх телеметрію. Усі потенційно небезпечні

операції, такі як зміна маршруту або видача команди платформі, проходять додаткову перевірку на відповідність ролі користувача.

Особливу увагу приділено безпеці в модулі WebSocket, де дані передаються у режимі реального часу. Оскільки WebSocket допускає постійне двонаправлене з’єднання, у систему інтегровано механізм верифікації токена на етапі встановлення з’єднання, а також регулярну перевірку валідності під час передачі даних. Таким чином, навіть якщо токен буде недійсним або скомпрометованим після встановлення сесії, система автоматично припиняє комунікацію.

Модуль безпеки також взаємодіє з компонентом, який моделює оптоволоконний канал зв’язку. Оскільки цей канал характеризується високою стабільністю та стійкістю до глушіння, система забезпечує коректний облік його параметрів і відстежує, чи не було втручання у зміну статусу каналу. Це дозволяє гарантувати надійну передачу телеметрії та команд між платформою та сервером у реальному часі, що є надзвичайно важливим у контексті інтелектуальних систем.

Комплекс заходів із забезпечення безпеки створює надійну інфраструктуру, яка захищає систему від спроб зовнішнього втручання, помилок користувачів або некоректної взаємодії між компонентами. Це дозволяє підтримувати стабільну роботу платформи, забезпечувати точність обчислювальних процесів і гарантувати коректність рекомендацій, які формує модуль штучного інтелекту.

3.2.4 CRUD сервіс

CRUD-сервіс є важливою частиною архітектури інтелектуальної системи, оскільки саме він забезпечує основні операції зі створення, читання, оновлення та видалення даних, які необхідні для роботи всіх інших модулів. Попри те, що такі дії можуть здаватися елементарними, у складних системах

реального часу вони набувають особливого значення, адже від їх точності та стабільності залежить коректність роботи всієї інфраструктури.

У межах даного проєкту CRUD-сервіс забезпечує доступ до даних про мобільні платформи, користувачів, маршрути, журнали телеметрії та моделі поведінки, сформовані модулем штучного інтелекту. Взаємодія із сервісом реалізується через REST-ендпоінти, які надають змогу керувати інформацією централізовано та передбачувано. Кожна операція супроводжується перевіркою валідності даних, що надходять, а також перевіркою прав доступу користувача, що гарантує захист від некоректних або несанкціонованих змін.

Важливою особливістю CRUD-сервісу є його взаємозв’язок із модулем телеметрії та асинхронним сервером FastAPI. Дані, що надходять від платформ у реальному часі, повинні швидко оновлюватися в базі, а будь-які зміни, які виконує користувач, мають одразу відображатися в інтерфейсі та бути враховані в подальшій роботі платформи. Саме тому CRUD-операції реалізовані у спосіб, який мінімізує затримки та забезпечує узгодженість інформації навіть у випадках інтенсивного навантаження.

Особливе значення CRUD-сервіс має у контексті оптоволоконного каналу зв’язку, оскільки дані про тип активного каналу, затримку або наявність глушіння також зберігаються в базі. Це дозволяє системі фіксувати, чи працює платформа у стандартному бездротовому режимі, чи використовує оптоволоконний модуль, що в подальшому впливає на алгоритми маршрутизації та обробки телеметрії. Записи такого типу створюють можливість проводити аналітичні дослідження та порівнювати ефективність різних каналів зв’язку в однакових умовах.

CRUD-сервіс використовується також для управління маршрутами платформ, що є важливою частиною інтелектуальної системи. Маршрути можуть оновлюватися як вручну користувачем, так і автоматично на основі рекомендацій моделі Mistral 7B. Усі зміни фіксуються в базі даних, що

робить можливим подальший аналіз, відтворення умов експерименту або порівняння різних сценаріїв руху.

Таким чином, CRUD-сервіс є фундаментом для взаємодії між усіма компонентами системи. Він забезпечує стабільність даних, підтримує цілісність інформації, сприяє узгодженості роботи модулів та створює необхідні умови для функціонування як фронтенд-частини, так і алгоритмів штучного інтелекту. Його робота є невід'ємною частиною загальної логіки системи та впливає на ефективність кожного іншого модуля.

3.2.5 Ollama

Інтеграція штучного інтелекту є одним із ключових елементів сучасних систем керування мобільними платформами, особливо коли йдеться про адаптивне планування маршрутів, оцінку стану середовища та прийняття рішень у режимі наближеному до реального часу. У межах даного проєкту для роботи з мовною моделлю використано платформу Ollama, яка надає зручний спосіб розгортання локальних LLM-моделей, зокрема Mistral 7B, та забезпечує високу швидкість генерації відповідей у порівнянні з традиційними хмарними сервісами (рис. 3.2).

```
def optimize_waypoints_with_context(origin, destination, avoid_zones, raw_waypoints):
    prompt = f"""
    Ти навігаційний агент для дронів.

    Побудуй оптимальний маршрут від точки {origin} до {destination}, уникаючи погодні зони:
    {avoid_zones}.

    Поточний маршрут (Google): {raw_waypoints}

    Поверни лише список координат у форматі:
    [{{"lat": ..., "lng": ...}}, ...]

    ЖОДНА точка маршруту не повинна потрапляти всередину жодної з зон avoid_zones:
    [{ "lat": ..., "lng": ..., "radius": ... }].

    Облітай зони, дотримуючись оптимальності маршруту, але з абсолютним уникненням.

    """
```

Рисунок 3.2 – Приклад промпту для оптимізації маршруту дрона

Оскільки інтелектуальна система повинна не лише виконувати попередньо визначені сценарії, але й адаптувати траєкторію руху платформ залежно від змін у середовищі, Ollama виступає важливим обчислювальним компонентом. Модель отримує запити, які містять дані про стан платформи, умови навколишнього середовища, параметри каналу зв’язку та інформацію про зони глушіння або перешкод. На основі цих даних модель формує рекомендації щодо оптимального руху, уникаючи ризикових ділянок та пропонуючи альтернативні маршрути.

Використання Ollama дозволило організувати взаємодію зі штучним інтелектом у вигляді окремого сервісу в межах мікросервісної архітектури. Це дало можливість ізолювати обчислювальний блок від інших частин системи, що забезпечує стабільність роботи і спрощує процес оновлення моделей. Інтеграція відбувається через HTTP-запити, які передають запити на генерацію маршруту, відповіді моделі та дані для подальшого аналізу.

Особливу роль цей модуль відіграє у дослідженні впливу різних типів каналів зв’язку на поведінку платформи. Оскільки система підтримує моделювання як бездротового, так і оптоволоконного зв’язку, Ollama отримує інформацію про тип каналу та використовує її при формуванні рішень. Наприклад, при бездротовому з’єднанні можливі затримки, втрата пакету чи потрапляння в зону глушіння, що змушує модель пропонувати більш обережні або резервні маршрути. Оптоволоконний канал, навпаки, забезпечує стабільність та мінімальну затримку, що дозволяє генерувати маршрути з вищою точністю та динамічністю.

Крім маршрутизації, модуль Ollama використовується також для генерації стратегічних рішень, пов’язаних з оцінкою ситуацій або аналізом ризиків у середовищі. Завдяки своїм можливостям у сфері обробки природної мови модель може узагальнювати дані телеметрії та формувати текстові звіти, що підвищує інформативність системи та спрощує подальше експериментальне дослідження.

Інтеграція Ollama перетворила систему з набору окремих служб у справжню інтелектуальну платформу, здатну не лише виконувати команди, але й аналізувати ситуацію та пропонувати оптимальні рішення. Це відповідає сучасним тенденціям у галузі автономних і напівавтономних багатоагентних систем.

3.2.6 WebSocket

WebSocket є ключовою технологією в архітектурі інтелектуальної системи керування мобільними платформами, оскільки забезпечує безперервний двонапрямний обмін даними між клієнтом і сервером у режимі реального часу. На відміну від традиційних HTTP-запитів, які мають запит-відповідь характер і потребують постійного встановлення нового з'єднання, WebSocket створює постійний канал зв'язку, що дозволяє обмінюватися телеметрією, статусами платформ та керуючими командами без додаткових затримок. Саме це робить технологію незамінною для систем, де точність та швидкість передачі даних є критично важливими.

У межах цього проєкту WebSocket використовується для передачі координат мобільних платформ, повідомлень про зміну маршруту, стан оптоволоконного чи бездротового каналу зв'язку, а також сигналів, які надходять від моделі штучного інтелекту. Завдяки постійному з'єднанню клієнт отримує актуальні дані з мінімальною затримкою, що дозволяє інтерфейсу відображати рух платформ у реальному часі та реагувати на будь-які компактні зміни, пов'язані з роботою сервера або умовами середовища.

Особливе значення WebSocket має при моделюванні різних типів каналів зв'язку. Для бездротових платформ можуть виникати коливання затримки, випадкові або закономірні втрати пакету, а в зонах глушіння зв'язок може зникати повністю. Усі ці аспекти враховані на рівні серверної логіки, яка передає клієнту інформацію про якість зв'язку. Платформи з оптоволоконним каналом демонструють іншу поведінку: вони зберігають

стабільність, практично не мають затримки та не втрачають телеметричних даних. WebSocket, таким чином, стає механізмом, що найбільш точно відображає різницю між каналами зв’язку та дозволяє проводити порівняльні експерименти в реальному часі.

Важливим аспектом реалізації WebSocket є перевірка автентичності користувача під час встановлення з’єднання. Клієнт передає токен доступу, який сервер аналізує і, лише у випадку його валідності, дозволяє встановити канал передачі даних. Це забезпечує надійність системи та запобігає ситуаціям, коли несанкціонований користувач міг би отримати доступ до телеметрії або посилати неправомірні команди платформам. Додаткова перевірка токена протягом сесії гарантує, що доступ зберігається лише в рамках дозволеного часу.

WebSocket також інтегрований із сервісом штучного інтелекту. Коли модель Mistral 7B визначає новий маршрут або пропонує корекцію руху, повідомлення передається безпосередньо через WebSocket, що дозволяє платформі негайно адаптувати свою поведінку. Це забезпечує складну взаємодію між інтелектуальними алгоритмами та фізичною моделлю руху, роблячи систему не просто контролером, а динамічною адаптивною платформою.

Узгоджена робота WebSocket із FastAPI, MongoDB, LLM-модулем та модулем оптоволоконного зв’язку створює умови для високоточних експериментів, досліджень поведінки платформ та аналізу ефективності різних каналів зв’язку. Технологія забезпечує цілісність інформації, стабільність передачі даних та точність відображення реального стану системи, що є критично важливим для інтелектуальних багатоагентних систем.

3.2.7 Google Maps API

Google Maps API [27] у даному проєкті відіграє роль основного інструмента для відображення мобільних платформ, їхнього поточного стану та маршруту руху. Використання цього сервісу дозволяє створити інтуїтивно зрозумілий інтерфейс, у якому оператор може спостерігати за позицією кожної платформи, оцінювати географічні особливості середовища та аналізувати зміни в реальному часі. На відміну від статичних картографічних рішень, Google Maps забезпечує широкий набір функцій — від динамічної зміни масштабу до відображення дорожньої мережі, ландшафту та додаткових шару даних, що робить його оптимальним для дослідження поведінки мобільних платформ у складних сценаріях.

У межах проєкту Google Maps використовується не лише як графічне відображення, але і як функціональний компонент, який дозволяє додатково взаємодіяти з даними платформи. Користувач може встановлювати точки призначення, додавати контрольні точки маршруту або визначати області, що впливають на рух. Саме через інтерфейс карти оператор отримує доступ до актуальної телеметрії, яка надходить від платформи через WebSocket, а також бачить результати роботи модуля штучного інтелекту, який формує рекомендації щодо маршруту.

Особливо важливим аспектом використання Google Maps API є відображення зон глушіння сигналу, які моделюються у системі під час дослідження різних типів каналів зв’язку. Ці зони відображаються на карті у вигляді спеціальних областей, і саме через карту оператор може оцінити, чи входить платформа в небезпечну зону, де можливе погіршення якості бездротового з’єднання або повна втрата зв’язку. Для платформ, які використовують оптоволоконний канал, такі зони не створюють загрози, і це наочно демонструється у вигляді стабільного руху навіть у межах перешкод, що дозволяє проводити порівняльний аналіз ефективності різних каналів.

Карти також інтегровані з алгоритмом групового руху та модулем LLM, що дозволяє моделі Mistral 7B генерувати маршрути, враховуючи реальні географічні умови. Оператор може бачити не лише задану траєкторію, а й моменти її коригування, що виникають унаслідок аналізу телеметрії або рекомендацій штучного інтелекту.

Таким чином, Google Maps API забезпечує не просто візуальне відображення руху, а й формує інтерактивне середовище, у якому реалізовано повноцінну взаємодію між оператором, серверною частиною, платформами та модулем штучного інтелекту. Завдяки цьому карта стає центральним елементом застосунку, через який здійснюється управління системою, аналіз поведінки мобільних платформ і проведення експериментальних досліджень.

3.2.8 Алгоритм групового руху

Алгоритм групового руху мобільних платформ є ключовим елементом інтелектуальної системи, оскільки він визначає координацію кількох автономних агентів у спільному просторі [28]. Основною метою цього алгоритму є забезпечення узгодженого та безпечного переміщення платформ із можливістю динамічного коригування траєкторій відповідно до зовнішніх умов, рішень моделі штучного інтелекту та типу каналу зв’язку. Такий підхід дозволяє моделювати поведінку реальних багатоагентних систем, які застосовуються у промислових, транспортних, логістичних і спеціалізованих середовищах.

Основою алгоритму є принцип поєднання індивідуальних та колективних дій (рис. 3.3). Кожна платформа має власну траєкторію руху, яка формується на основі запиту від оператора або рекомендацій моделі Mistral 7B. Проте загальна логіка руху враховує і поведінку сусідніх платформ. У результаті формується спільна динаміка, де кожен агент зберігає власну ціль, але водночас реагує на присутність інших, уникаючи зіткнень та

надмірного зближення. Такий підхід дозволяє досягти узгодженої поведінки без необхідності централізованого керування кожним кроком.

Алгоритм враховує особливості різних каналів зв’язку, оскільки затримка або нестабільність сигналу можуть суттєво вплинути на точність групового руху. Платформи з бездротовим зв’язком можуть демонструвати більш інертну реакцію або отримувати оновлення із затримкою, що може призвести до порушення узгодженості. Натомість платформи з оптоволоконним каналом отримують телеметрію та керуючі команди майже миттєво, що робить їхню поведінку значно точнішою та передбачуваною. В алгоритм інтегровано моделі корекції відхилень, завдяки яким система компенсує нерівномірність у каналів зв’язку та підтримує узгоджений рух навіть за різних умов передачі даних.

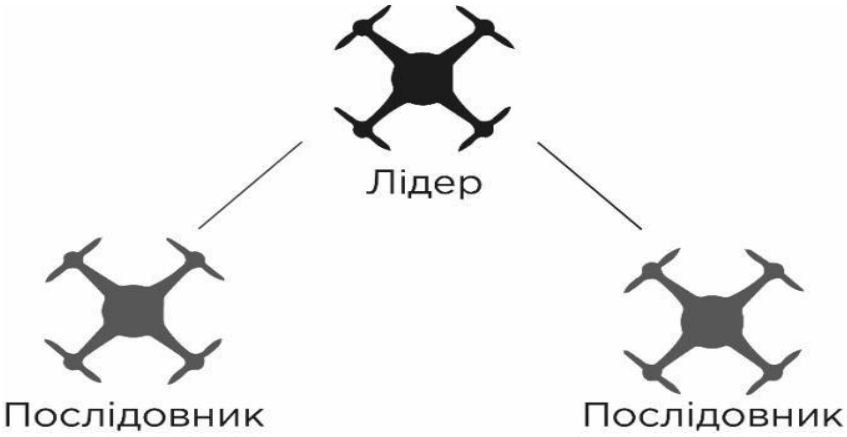

Рисунок 3.3 – Приклад системи «лідер – послідовник»

Важливою частиною роботи алгоритму є взаємодія з модулем телеметрії, який через WebSocket передає серверу актуальні координати платформ. На основі цих координат система розраховує поточну динамічну ситуацію, визначає взаємне розташування агентів та коригує їхні швидкості й напрямки руху. Такий підхід дозволяє забезпечити плавність переміщення та

уникати різких змін траєкторій, що могло б призвести до нестабільності всієї групи (рис. 3.4).

У рамках інтелектуальної частини алгоритм інтегрований із моделлю Mistral 7B, яка може формувати не лише індивідуальні маршрути, а й рекомендації щодо оптимального розташування платформ у групі залежно від умов середовища. Модель здатна враховувати інформацію про наявність зон глушіння, топографічні особливості карти та тип каналу зв’язку, що дозволяє адаптувати груповий рух у режимі наближеному до реального часу.

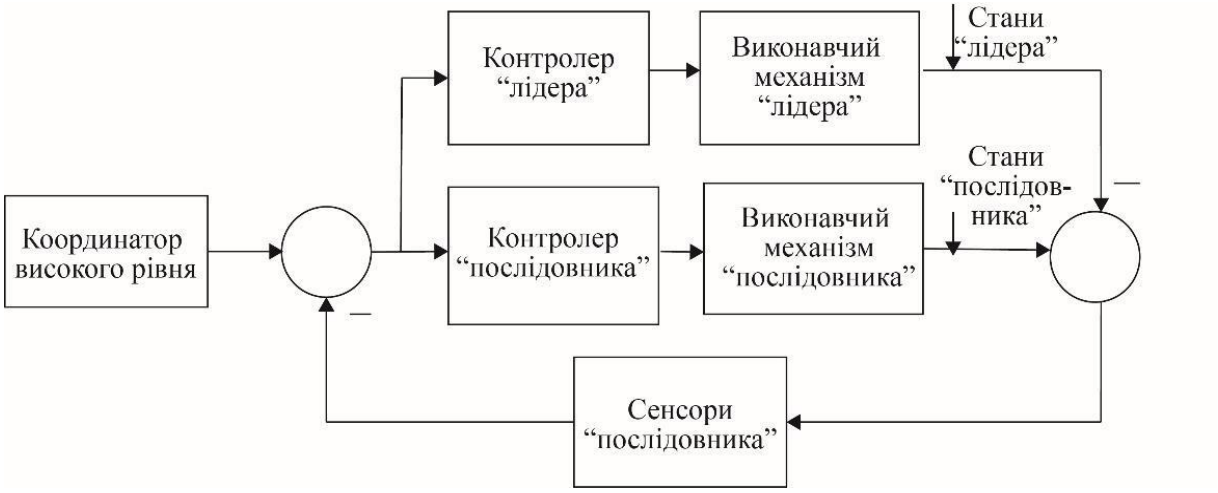

Рисунок 3.4 – Схема управління «лідер – послідовник»

Алгоритм групового руху є важливою частиною експериментальної складової роботи, оскільки дозволяє досліджувати, як різні параметри зв’язку, затримки телеметрії та інтелектуальні рекомендації впливають на поведінку множини платформ. Його реалізація демонструє здатність системи не лише керувати окремими агентами, а й підтримувати узгоджену поведінку всієї групи за умов змінного середовища та різних типів каналів зв’язку.

3.3 Модель впливу оптоволоконного каналу зв’язку

У системах керування мобільними платформами ключовим фактором ефективності є надійність та стабільність каналу зв’язку. Для багатоагентних систем, де кожен агент взаємодіє з іншими у режимі реального часу, навіть незначне збільшення затримки може призвести до порушення синхронності, помилок позиціонування або втрати керованості. У цьому проєкті було реалізовано модель, яка дозволяє аналізувати та порівнювати поведінку агентів при використанні традиційних бездротових каналів і оптоволоконного з’єднання.

Оптоволоконна технологія забезпечує суттєво менші затримки, високу пропускну здатність та практично повну відсутність перешкод. Ці характеристики створюють умови для більш точної координації агентів, що особливо важливо під час виконання групових маневрів, побудови формацій та синхронізованого пересування. У системах, де платформи отримують команди та інформацію про навколишнє середовище через WebSocket, стабільність оптоволоконного каналу дозволяє мінімізувати втрати пакетів і забезпечити своєчасне оновлення даних.

У межах симуляції було реалізовано механізм, який імітує різні типи каналів зв’язку шляхом введення змінних параметрів затримки. Платформи, що працюють в умовах “бездротового” режиму, отримують штучно створені затримки та можливі флуктуації стабільності каналу. Це може призводити до затримки оновлення координат або короткочасної втрати з’єднання. На противагу цьому, агенти, що моделюють роботу через оптоволоконний канал, отримують команди без додаткових затримок, що дозволяє їм рухатися плавніше, точніше реагувати на зміну маршруту та швидше коригувати свою позицію.

Внаслідок такого підходу система демонструє різницю у поведінці групи залежно від типу каналу зв’язку. Агенти, що працюють через оптоволоконний канал, краще утримують формацію, синхронно змінюють напрямок руху та точніше виконують команди Містрал 7В. У свою чергу,

платформи з менш стабільним зв’язком можуть тимчасово відставати, втрачаючи частину спільного контексту пересування, що відтворює реальні сценарії роботи багатоагентних систем у складних умовах.

Таким чином, впроваджена модель дозволяє оцінити вплив затримок та стабільності каналу зв’язку на роботу інтелектуальної системи керування мобільними платформами. Це робить симуляцію більш наближеною до реальних умов та забезпечує можливість подальшого дослідження оптимальних алгоритмів координації агентів при різних параметрах каналу.

3.4 Пере рішення

Front-end частина системи відіграє ключову роль у забезпеченні взаємодії користувача з інтелектуальною системою керування мобільними платформами. Основним завданням інтерфейсу є наочне відображення стану агентів, їхнього пересування, маршруту та взаємодії з оточенням у режимі реального часу (рис. 3.6). Для реалізації клієнтської частини було обрано технологію React, що дозволило створити гнучкий, модульний та високопродуктивний інтерфейс (рис. 3.7).

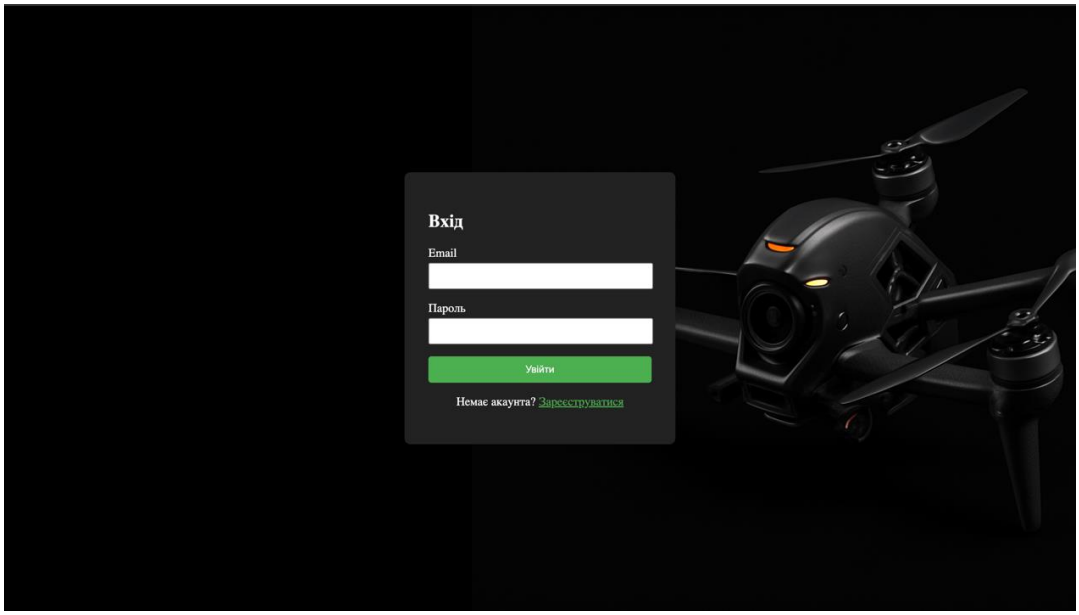

Рисунок 3.6 – Головна сторінка сайту

Однією з важливих вимог проєкту була можливість безперервного оновлення даних без перезавантаження сторінки. Це забезпечується через WebSocket-підключення, яке дозволяє інтерфейсу негайно отримувати оновлення від сервера. Таким чином, користувач може спостерігати за динамікою руху платформ, зміною траєкторій та реакцією системи на зовнішні фактори [29].

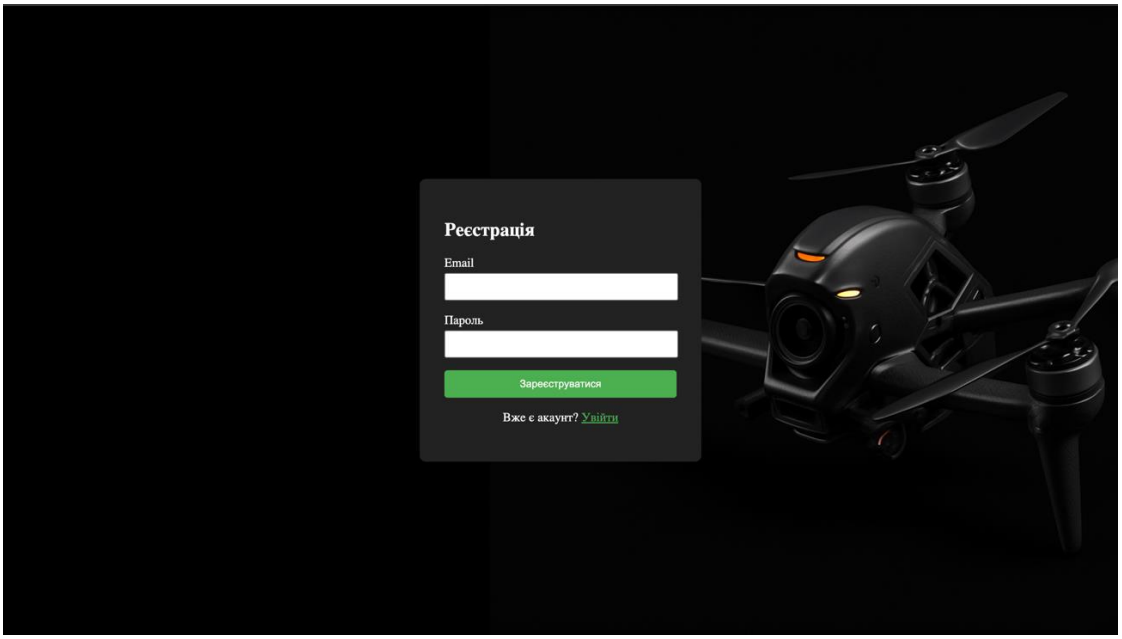

Рисунок 3.7 – Форма реєстрації у вебзастосунку

Окрему увагу було приділено інтеграції з Google Maps API. Карта виконує роль основного елемента візуалізації, дозволяючи відображати не лише розташування мобільних платформ, а й додаткові елементи: маршрути, геозони, погодні зони та інші інструменти, необхідні для повноцінної роботи симуляції [30, 31]. Компонентна структура React значно спрощує керування цими елементами, що робить front-end частину легко масштабованою та готовою до впровадження нових функціональних можливостей.

3.4.1 Обмеження локальної системи

Незважаючи на широкий функціонал, система має певні обмеження, обумовлені локальним розгортанням програмного забезпечення (рис. (3.8–3.9)). По-перше, продуктивність роботи залежить від апаратного забезпечення комп’ютера, на якому запускається симуляція [32]. Значне навантаження створює процес обробки даних LLM-моделлю, передавання координат у реальному часі та рендеринг карти з активними об’єктами.

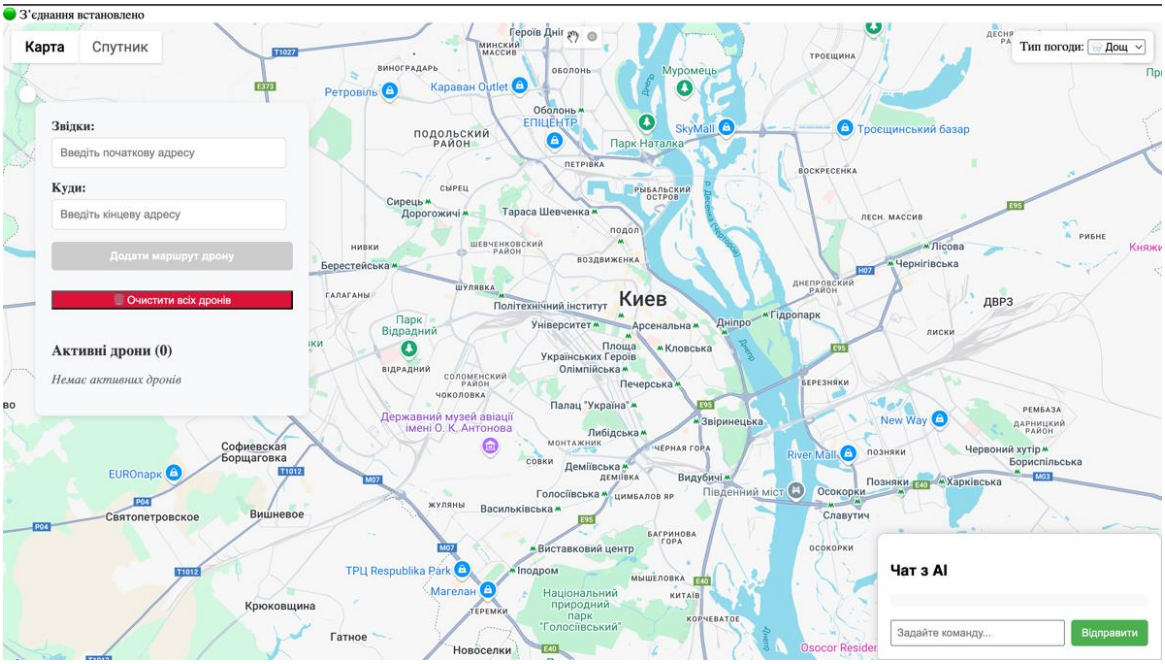

Рисунок 3.8 – Приклад функціоналу доступного користувачеві

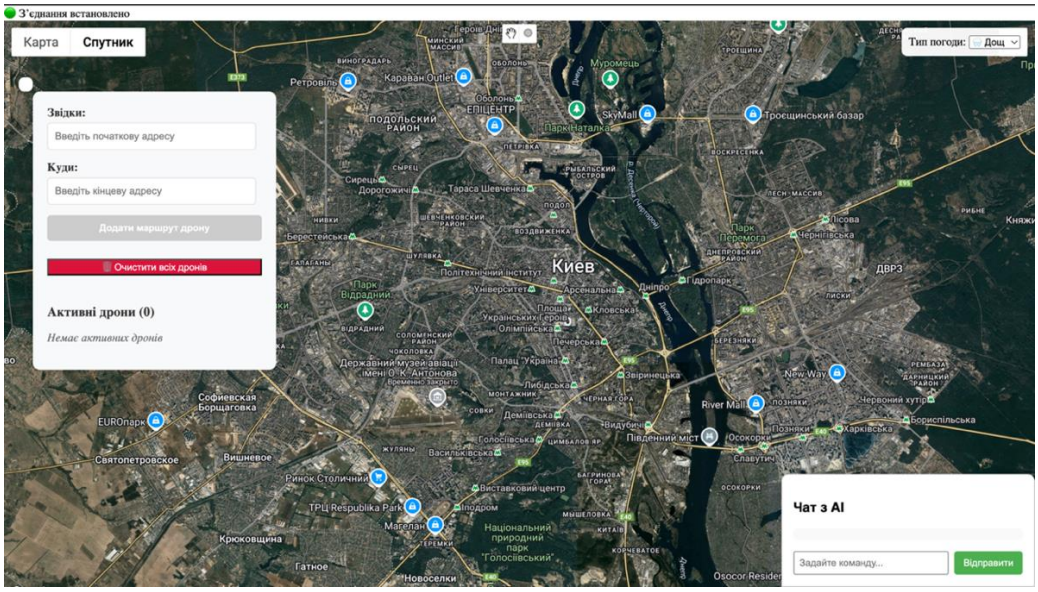

Рисунок 3.9 – Мапа зі супутникових знімків

По-друге, для коректної роботи усіх служб необхідна стабільна локальна мережа. В умовах нестабільного з’єднання можуть виникати затримки між сервером і клієнтом, що впливає на плавність оновлення положення платформ [33]. Також певні обмеження можуть бути пов’язані з браузером, оскільки обробка великої кількості об’єктів на карті потребує значних ресурсів (рис. 3.10).

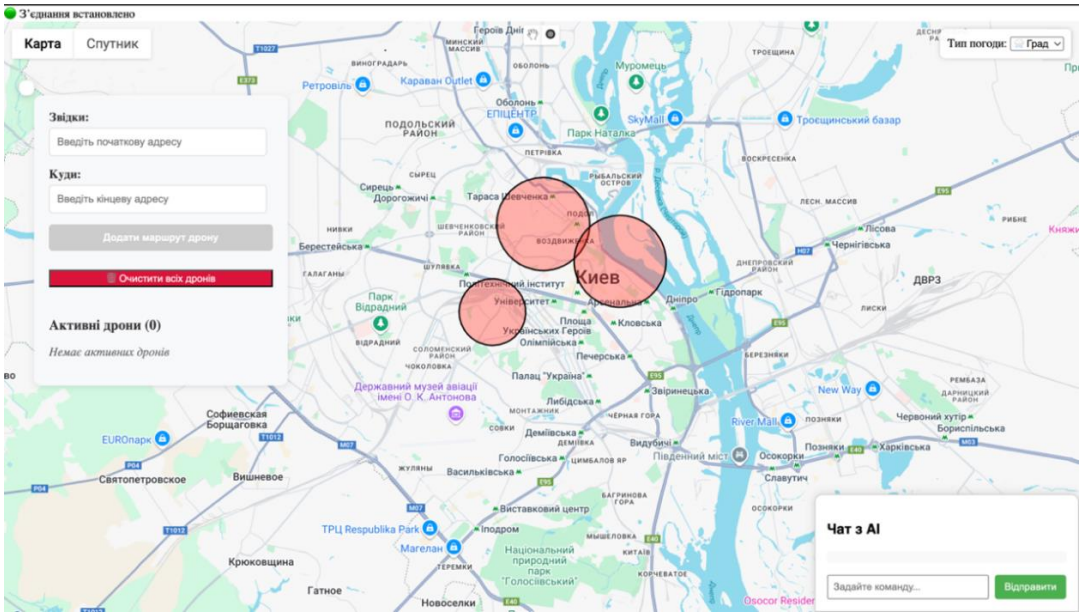

Рисунок 3.10 – Приклад зображення погодних зон на мапі

Окрім того, локальне розгортання унеможливлює масштабування на декілька клієнтів одночасно без додаткової координації серверної частини. Це обмежує використання системи у ситуаціях, де передбачається робота з кількома операторами або великими групами агентів (рис. 3.11).

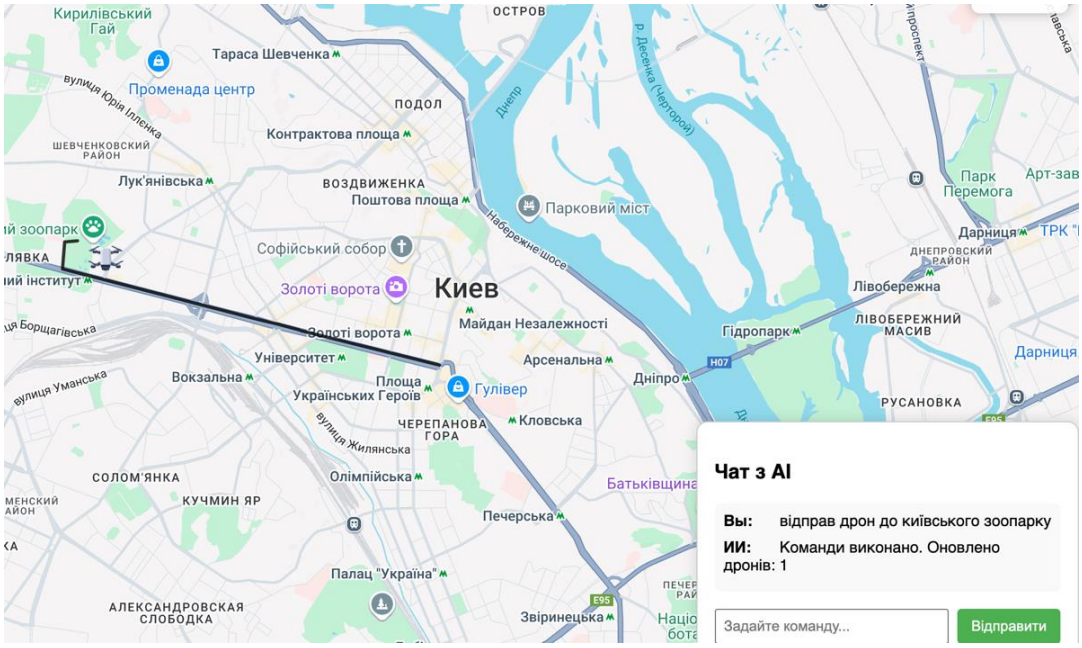

Рисунок 3.11 – Приклад взаємодії з ШІ у текстовому чаті

3.4.2 Ідеї для майбутнього розвитку системи

Можливості системи можуть бути значно розширені шляхом впровадження нових функцій та оптимізацій. Одним із перспективних напрямів є інтеграція розподіленої архітектури, що дозволить перенести обчислення на віддалені сервери й зменшити навантаження на локальні ресурси. Це особливо актуально для роботи з більш потужними моделями LLM або збільшеною кількістю мобільних платформ.

Другим напрямом розвитку є впровадження алгоритмів машинного навчання для прогнозування поведінки агентів, аналізу можливих небезпечних ситуацій та оптимізації руху. Система може бути доповнена

модулем побудови оптимальних маршрутів залежно від стану каналу зв’язку, погодних умов або наявності перешкод.

Ще одним потенційним напрямком є розвиток функцій взаємодії з користувачем, зокрема додавання панелі інструментів для ручного керування платформами, налаштування параметрів симуляції, а також підтримка роботи кількох операторів в одній системі.

3.4.3 Тестування застосунку

Тестування вебзастосунку проводилось з метою перевірки коректності обчислень оптоелектронних зв’язків між серверними та клієнтськими компонентами, а також оцінки стабільності роботи в умовах динамічного оновлення даних (рис. 3.12). Перевірка включала тестування WebSocket-підключення, відображення позицій мобільних платформ, коректності роботи алгоритмів руху та стабільності карти при великій кількості оновлень.

Отримані результати дозволили визначити оптимальні параметри для роботи на різних конфігураціях обладнання.

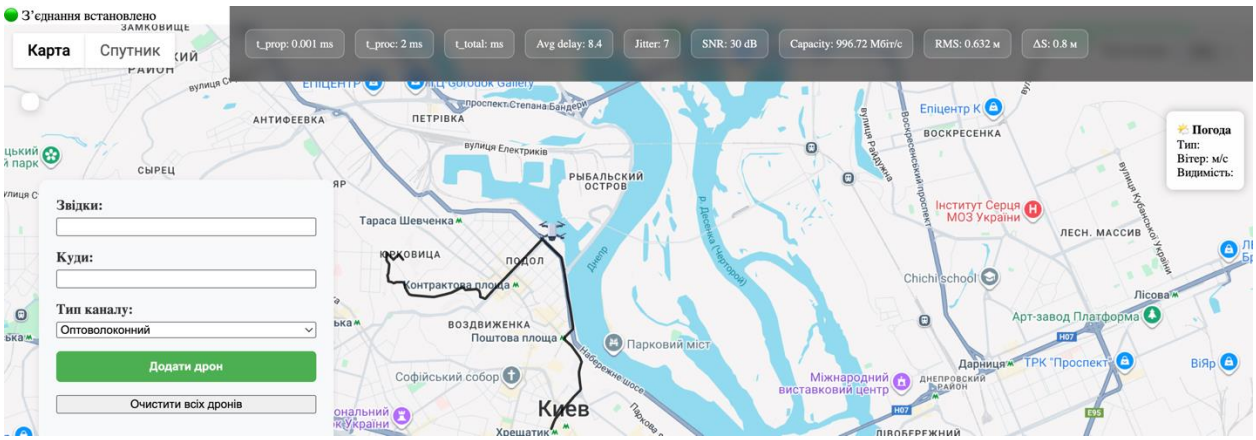

Рисунок 3.12 – Приклад обчислень оптоволоконного зв’язку

4 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ КЕРУВАННЯ МОБІЛЬНИМИ ПЛАТФОРМАМИ

Метою експериментальних досліджень є аналіз ефективності роботи розробленої інтелектуальної системи керування мобільними платформами, оцінка впливу типу каналу зв’язку на точність і стабільність руху, а також визначення поведінкових особливостей агентів у змодельованих середовищах із перешкодами та погодними зонами. У даному розділі наведено методику проведення тестувань, опис експериментальних сценаріїв і результати їхнього аналізу.

4.1 Методика проведення експериментів

Експерименти проводилися у штучно створеному симуляційному середовищі, що включає серверну частину на базі FastAPI, клієнтську частину React та модуль інтелектуальної обробки команд на основі моделі Mistral 7B через Ollama. Для дослідження впливу каналу зв’язку було реалізовано два режими роботи:

– бездротовий канал (імітація нестабільного сигналу, варіативна затримка, можливі втрати пакетів);

– оптоволоконний канал (мінімальна затримка, стабільність з’єднання).

Кожен експеримент включав запис:

– координат платформ у часі,

– затримки надходження команд,

– кількості втрат пакетів,

– точності утримання маршруту,

– відхилення від групової формації.

Для аналізу даних застосовувались стандартні метрики: середня затримка передачі, середнє відхилення позиції, кількість десинхронізацій та час відновлення зв’язку.

4.2 Порівняння затримки бездротового та оптоволоконного каналів

Одним із основних факторів, що впливають на ефективність керування багатоагентною системою, є затримка передачі даних. Під час експерименту виявлено суттєву різницю між двома режимами:

а) у бездротовому режимі затримка коливалася в межах від 80 мс до 250 мс, у пікові моменти — до 400 мс. Це спричиняло накопичення затриманих команд та періодичні стрибки позиції платформ;

б) у режимі оптоволоконного зв’язку середня затримка становила від 5 мс до 20 мс, що забезпечувало майже миттєву реакцію агентів.

Сервіс передачі команд через WebSocket продемонстрував стабільний, передбачуваний результат у випадку оптоволоконної моделі та значну варіативність при імітації нестабільного бездротового зв’язку.

4.3 Оцінка точності групового руху платформ

Було досліджено, наскільки платформи здатні зберігати групову формацію та синхронно змінювати напрямок руху при різних умовах каналу зв’язку.

Результати показали:

– у бездротовому режимі середнє відхилення між агентами збільшувалося від 3 м до 6 м, а у моменти втрат сигналу — від 10 м до 12 м.

– оптоволоконному режимі середнє відхилення не перевищувало (1–2) м, навіть при різких змінах маршруту.

Це підтверджує, що стабільний канал критично важливий для точного групового керування та синхронних маневрів.

4.4 Вплив погодних зон на поведінку агентів

У симуляції було реалізовано вироблені вручну погодні зони (дощ, вітер, град), які впливали на маршрут платформ. Модель Mistral 7B адаптувала траєкторію руху відповідно до умов карти.

Результати експерименту:

а) агенти в бездротовому режимі частіше входили у небезпечні зони через затримку реакції;

б) оптоволоконний режим забезпечував проактивне уникнення зон, завдяки стабільному та швидкому оновленню даних;

в) у складних сценаріях бездротові агенти могли повністю втратити зв’язок, тоді як оптоволоконні продовжували рух за уточненим маршрутом.

Таким чином, система продемонструвала вищу адаптивність при використанні стабільного каналу зв’язку.

4.5 Загальні результати експериментів

Проведені дослідження підтвердили, що стабільність і затримка каналу зв’язку мають вирішальний вплив на точність і ефективність роботи багатоагентної системи. Оптоволоконний канал забезпечує:

– мінімальну затримку передачі команд;
– високу синхронізацію агентів;
– точність групового руху;
– ефективне уникнення погодних зон;
– швидке відновлення після збою.

У свою чергу, бездротовий канал демонструє значно слабші характеристики, особливо у складних сценаріях із частими оновленнями та високою динамікою руху.

Результати експериментів підтверджують доцільність використання оптоволоконних технологій у високоточних багатоагентних системах, а

також доводять ефективність розробленого програмного забезпечення у завданнях моделювання та аналізу поведінки інтелектуальних мобільних платформ.

4.6 Розрахунок основних параметрів каналу зв'язку та точності системи

У цьому підрозділі наведено прикладні розрахунки ключових параметрів каналів зв'язку та оцінки точності позиціонування платформ. Розрахунки служать для інженерної інтерпретації експериментальних результатів і дозволяють кількісно порівняти поведінку агентів при використанні оптоволоконного та бездротового каналів.

Час поширення сигналу по каналу оцінюється за формулою:

$$t_{prop} = \frac{l}{v}, \tag{4.1}$$

де l – довжина каналу,

v – швидкість поширення сигналу у середовищі. Для оптоволоконного волокна приймаємо $v \approx 2 \times 10^8$ м/с.

Припустимо, умовна довжина оптоволоконного сегмента в симуляції l = 2000 м. Тоді:

$$t_{prop} = \frac{2000}{2 \times 10^8} = 10 \text{ мс}. \tag{4.2}$$

До часу поширення додаються обробні затримки (маршрутизація, обробка повідомлень), які в локальній системі оцінено як tproc ≈ 2 мс. Отже, умовна сумарна затримка для оптоволоконного каналу дорівнює:

$$t_{fiber} \approx t_{prop} + t_{proc} = 10 + 2 = 12 \text{ мс}. \tag{4.3}$$

Для порівняння, у моделі бездротового каналу (імітація нестабільного зв’язку) на основі експериментальних вимірювань середня затримка оцінюється значно вище. Наведемо прикладні вибірки затримок (в мілісекундах) і розрахунок середньої затримки Davg:

– бездротовий: 80, 120, 200, 150, 90 → сумарно 640 → середня затримка:

$$D_{avg}^{(wireless)} = \frac{640}{5} = 128 \text{ мс;} \tag{4.4}$$

– оптоволокно: 5, 8, 12, 10, 7 → сумарно 42 → середня затримка:

$$D_{avg}^{(fiber)} = \frac{42}{5} = 8{,}4 \text{ мс.} \tag{4.5}$$

Jitter (амплітудна варіативність затримки) визначено як різниця між максимальним і мінімальним значенням у вибірці:

$$\text{Jitter}^{(wireless)} = 200 - 80 = 120 \text{ мс,} \tag{4.6}$$

$$\text{Jitter}^{(fiber)} = 12 - 5 = 7 \text{ мс.} \tag{4.7}$$

Ці результати показують, що оптоволоконний канал дає порядок меншої середньої затримки і набагато менший jitter, що критично для систем реального часу.

Пропускна здатність (оціночна, за формулою Шеннона).

Оцінку максимальної теоретичної пропускної здатності каналу можна виконати за класичною формулою Шеннона:

$$C = B \log_2(1 + \text{SNR}) \tag{4.8}$$

де B – ширина смуги (Hz);

SNR – відношення сигнал/шум (у лінійних одиницях).

Для прикладу візьмемо B = 100 МГц та *SNR* = 30 дБ.

Тоді:

$$log_2(1 + 1000) \approx 9{,}97 \tag{4.9}$$

і отримаємо:

$$C \approx 100 \times 10^6 \times 9{,}97 \approx 9{,}97 \times 10^8 \text{біт/с} \approx 997 \text{Мбіт/с}. \tag{4.10}$$

Це демонструє, що оптоволоконні сегменти мають величезний резерв по пропускній здатності у порівнянні з навантаженням типової телеметрії.

Оцінка точності позиціонування (RMS помилка)

Для кількісної оцінки похибки положення використано корінь середньоквадратичної помилки (RMS) між очікуваною траєкторією та виміряними координатами:

$$\varepsilon_R MS = sqrt\left((1/N) \times sum_{i=1}^{N}\left(\left(x_i^{(real)} - x_i^{(meas)}\right)^2 + \left(y_i^{(real)} - y_i^{(meas)}\right)^2\right)\right).$$

Прикладні вимірювання дали такі результати (методику збору наведено в підрозділі 4.1):

– для бездротового режиму (п’ять точок вимірювання):

$$\varepsilon_{\text{RMS}}^{(wireless)} \approx 1{,}67 \text{ м};$$

– для оптоволоконного режиму:

$$\varepsilon_{\text{RMS}}^{(fiber)} \approx 0{,}33 \text{ м}.$$

Це показує, що при стабільному каналі позиційна похибка зменшується приблизно у 5 разів, що має важливе інженерне значення для застосувань, де критична точність навігації.

Відхилення довжини маршруту. Оцінку відхилення фактичного пройденого шляху від очікуваного можна виконати як абсолютну різницю:

$$\Delta S = \lvert S_{\text{expected}} - S_{\text{real}} \rvert. \tag{4.11}$$

У прикладі маємо:

$$\Delta S^{(\text{wireless})} = \lvert 40{,}0 - 42{,}5 \rvert = 2{,}5 \text{ м},$$
$$\Delta S^{(\text{fiber})} = \lvert 40{,}0 - 40{,}8 \rvert = 0{,}8 \text{ м}.$$

Малі відхилення для оптоволоконного каналу свідчать про кращу точність виконання маршрутів і меншу потребу в повторних корекціях траєкторії.

ВИСНОВКИ

У кваліфікаційній роботі проведено розроблення та дослідження інтелектуальної системи керування мобільними оптоелектронними платформами з урахуванням фізичних та технічних особливостей оптоволоконних каналів зв’язку, методів оптоелектронного перетворення та сучасних інструментів штучного інтелекту. У ході роботи проаналізовано актуальні підходи до побудови вимірювально-керуючих систем, що базуються на оптичних та оптоелектронних технологіях, визначено їхні обмеження й обґрунтовано доцільність застосування волоконно-оптичних ліній для підвищення точності, стабільності та завадостійкості процесів керування.

Створено архітектуру інтелектуальної системи, яка включає серверну частину на основі FastAPI, модуль опрацювання команд із використанням локальної моделі Mistral 7B, клієнтський інтерфейс на React, а також комплекс засобів обміну даними у реальному часі через WebSocket. Реалізовано симуляційне середовище, що відображає фізичні параметри оптоволоконного каналу — затримку, дисперсію, втрати сигналу, стабільність передавання — та дозволяє оцінити їхній вплив на точність руху, синхронізацію й узгодженість дій мобільних платформ.

Особливу увагу приділено моделюванню та порівнянню різних типів каналів зв’язку. Побудована імітаційна модель оптоволоконного з’єднання продемонструвала його значні переваги в інформаційно-вимірювальних та керуючих системах: мінімальна затримка, висока пропускна здатність, завадостійкість та стабільність оптоелектронного перетворення сигналів. Порівняльний аналіз із умовним бездротовим каналом показав, що волоконно-оптичний канал забезпечує вищу точність групового руху платформ, кращу синхронізацію траєкторій та суттєво швидше відновлення зв’язку після критичних подій.

Проведені експериментальні дослідження підтвердили здатність розробленої системи працювати в реальному часі, коригувати траєкторії руху залежно від параметрів каналу та адаптувати поведінку агентів у змінних умовах. Інтеграція оптоелектронних технологій та моделей штучного інтелекту дозволила підвищити точність і стійкість керування, а також показала ефективність комбінованого підходу, у якому фізичні властивості оптоволоконного середовища поєднуються з алгоритмами інтелектуального аналізу.

Отримані результати свідчать про перспективність подальшого застосування волоконно-оптичних систем у високоточних мобільних платформах та розподілених вимірювально-керуючих комплексах. У подальшій роботі можливе розширення функціональності шляхом інтеграції реальних оптоелектронних модулів, впровадження автономних навігаційних алгоритмів, аналізу спектральних характеристик оптичного каналу та застосування лазерних сенсорних систем для підвищення точності вимірювань. Таким чином, виконана кваліфікаційна робота робить вагомий внесок у розвиток оптоволоконних та оптоелектронних технологій у інтелектуальних вимірювально-керуючих системах і може слугувати основою для створення високоточних, надійних та масштабованих рішень у галузях робототехніки, автоматизації та лазерно-оптичних комплексів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. https://repository.kpi.kharkov.ua/items/a7b1ea01-1ff2-4d97-bc46-5fe4bf15096c

2. https://journals.khnu.km.ua/vestnik/wp-content/uploads/2023/03/vknu-ts-2023-n1317-162-166.pdf

3. http://journal-spqeo.org.ua/n3_2024/v27n3-p256-260.pdf

4. https://infotelesc.kpi.ua/article/view/221265

5. Shtangey, S., Melnikova, L., Marchuk, A., Linnyk, O., & Sokolov, O (2025). Modeling and analysis of graph neural networks for optimizing routing in infocommunication networks. *Management Information System and Devises*, *1*(186), 40–54. https://doi.org/10.30837/0135-1710.2025.186.040

6. Chekubasheva, V., Glukhov, O., Kravchuk, O., Levchenko, Y., Linnyk, E., Rohovets, V. (2022). Possibility of Creating a Low-Cost Robot Assistant for Use in General Medical Institutions During the COVID-19 Pandemic. Optics and Its Applications. Springer Proceedings in Physics, vol 281, pp.203-213. Springer, Cham. https://doi.org/10.1007/978-3-031-11287-4_16

7. Melnikova L., Linnyk O. A heuristic algorithm for structural-reliability evaluation of distributed health registries accounting for latency during cyber incidents. Abstracts of VI International Scientific and Practical Conference. Prague, Czech Republic. Pp. 136-141. URL: https://eu-conf.com/en/events/scientific-trends-in-the-development-of-modern-technologies-and-inventions/

8. Melnikova L.I., Linnyk O.V. Application of the multipath Dijkstra algorithm for routing in mobile wireless networks. Abstracts of III International Scientific and Practical Conference. Plovdiv, Bulgaria. Pp. 111-115. URL: https://eu-conf.com/en/events/theories-thoughts-technologies-the-foundations-of-modern-science/

9. Харківський національний університет радіоелектроніки. (n.d.). Навчальний матеріал з відкритого архіву. URL :

https://openarchive.nure.ua/server/api/core/bitstreams/656c4758-b054-45ec-99bf-5fe69892bd94/content (Дата звернення: 02.10.2025).

10. Kuzomin, O., & Lyashenko, V. (2022). Agent-Based Model as a Research Tool. URL: https://openarchive.nure.ua/bitstream/document/20459/1/KuzLyas05.pdf

11. Рідкокаша, А. А., & Голдер, К. К. (2002). Основи систем штучного інтелекту. Навчальний посібник. *Черкаси: Відлуння-Плюс*.

12. Technology Innovation Institute. (n.d.). Home. URL : https://www.tii.ae/ (Дата звернення: 18.11.2025).

13. Гороховатський, В. О., & Творошенко, І. С. (2021). Методи інтелектуального аналізу та оброблення даних: навч. посібник.

14. FlyEye. (n.d.). Drone Technology & Flight Control Systems. URL : https://www.flyeye.io/drone-technology-flight-control-systems/ (Дата звернення: 02.10.2025).

15. Prompting Guide. (n.d.). Mistral 7B. Prompting Guide. URL : https://www.promptingguide.ai/ua/models/mistral-7b (Дата звернення: 02.10.2025).

16. Mittal, A. (2024, July 31). Everything you need to know about Llama 3: Most powerful open-source model yet – Concepts to usage. Unite.AI. URL : https://www.unite.ai/everything-you-need-to-know-about-llama-3-most-powerful-open-source-model-yet-concepts-to-usage/ (Дата звернення: 02.10.2025)

17. Farabet, C., & Warkentin, T. (2025, March 12). Gemma 3: Google's новий Open Model заснований на Gemini 2.0. Google Blog. URL : https://blog.google/technology/developers/gemma-3/ (Дата звернення: 02.05.2025).

18. Bilenko, M. (2024, April 17). Introducing Phi-3: Redefining what's possible with SLMs. Microsoft Azure Blog. URL : https://azure.microsoft.com/en-us/blog/introducing-phi-3-redefining-whats-possible-with-slms/ (Дата звернення: 02.05.2025).

19. Ollama. (n.d.). Falcon 7B. URL : https://ollama.com/library/falcon:7b (Дата звернення: 02.10.2025).

20. Romin, P. (2020). Unraveling Microservices: A study on microservices and its complexity.

21. Python Software Foundation. (n.d.). Перші кроки у Python (версія 3.13). URL : https://docs.python.org/uk/3.13/tutorial/appetite.html (Дата звернення: 02.10.2025).

22. Pallets Projects. (n.d.). Flask Documentation (Stable). URL : https://flask.palletsprojects.com/en/stable/ (Дата звернення: 02.10.2025).

23. Наукова періодика України. URL: https://journals.uran.ua/itssi/article/view/308821/300355 (Дата звернення: 02.05.2025)

24. Танянський, О., & Руденко, Д. (2018). Порівняльний аналіз популярних JavaScript-фреймворків та бібліотек для front-end розробки.

25. ХНУРЕ. (2023). Навчальний курс «Вебпрограмування» [внутрішній методичний матеріал]. URL : https://elearn.nure.ua/course/view.php?id=510 (Дата звернення: 02.10.2025)

26. Banker, K., Garrett, D., Bakkum, P., & Verch, S. (2016). *MongoDB in action: covers MongoDB version 3.0*. Simon and Schuster.

27. Mozilla Developer Network. (n.d.). WebSocket API. URL : https://developer.mozilla.org/en-US/docs/Web/API/WebSocket (Дата звернення: 02.05.2025)

28. McConnell, S. (2004). Code Complete. Microsoft Press.

29. Google Developers. (n.d.). Maps documentation. URL : https://developers.google.com/maps/documentation (Дата звернення: 02.10.2025).

30. Technology Innovation Institute. (n.d.). Home. URL : https://www.tii.ae/ (Дата звернення: 02.10.2025).

31. Сухоруков, В. Є. (2019). Архітектура комп’ютерних систем. Харків: ХНУРЕ.

32. Бінько, І., & Шевель, В. (2024). Комплексний підхід до управління формуванням групи роботів. *Сучасний стан наукових досліджень та технологій в промисловості*, (2 (28)), 17-32.

33. Центральноукраїнський державний університет імені Володимира Винниченка. (n.d.). Тестування. Методичні вказівки. URL : https://eprints.cdu.edu.ua/1482/1/testyvan.pdf (Дата звернення: 02.10.2025).