МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління

БУДАКОВСЬКИЙ-КАПАНЮК Артем Русланович

Програмний модуль управління рухом робота на основі нечіткої логіки / Software module for robot movement based on fuzzy logic

Спеціальність 122 – Комп'ютерні науки
Освітньо-професійна програма – Штучний інтелект

Кваліфікаційна робота

Виконав студент групи КНШІ-41
А.Р. Будаковський-Капанюк

Науковий керівник:
к.т.н., доцент О. П. Адамів

Кваліфікаційну роботу допущено до захисту
«___»_____ 2025 р.

В.о. завідувача кафедри
_____ Н.М. Васильків

Тернопіль – 2025

Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «бакалавр»
спеціальність 122 – Комп'ютерні науки
освітньо-професійна програма – Штучний інтелект

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Н.М. Васильків
«____» _____2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
БУДАКОВСЬКИЙ-КАПАНЮК Артем Русланович
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: Програмний модуль управління рухом робота на основі нечіткої логіки / Software module for robot movement based on fuzzy logic
керівник роботи Адамів О.П., к.т.н, доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від 20 грудня 2024 р. № 938.
2. Строк подання студентом закінченої кваліфікаційної роботи 25 травня 2025 р.
3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.
4. Основні питання, які потрібно розробити:
– провести аналіз предметної області управління рухом мобільних роботів;
– проаналізувати відомі рішення, які реалізують методи навігації з використанням нечіткої логіки та нейронних мереж;
– сформулювати постановку задачі дослідження;
– спроєктувати архітектуру програмного модуля;
– розробити алгоритми функціонування окремих компонентів програмного модуля;
– інтегрувати розроблений модуль у загальну систему керування мобільного робота;
– реалізувати програмний модуль навігації;
– провести серію експериментів у різних сценаріях;
– проаналізувати результати експериментів.
5. Перелік графічного матеріалу в роботі:
– архітектура програмного модуля для автономного управління рухом мобільного робота;
– структурна схема реалізації деліберативної навігації мобільного робота на основі симулятора V-REP;
– структурна схема реалізації реактивної навігації мобільного робота на основі симулятора V-REP.

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 20 грудня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Затвердження теми кваліфікаційної роботи, ознайомлення з літературними джерелами та складання плану роботи	до 01.01. 2025 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.02. 2025 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 01.04.2025 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 01.05. 2025 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 15.05.2025 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів	до 25.05.2025 р.	
7	Перевірка кваліфікаційної роботи на оригінальність тексту	до 30.05.2025 р.	
8	Оформлення кваліфікаційної роботи та отримання допуску до захисту	до 10.06.2025 р.	
9	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії	до 10.06. 2025 р.	

Студент_____ А.Р. Будаковський-Капанюк
(підпис) (прізвище та ініціали)

Керівник кваліфікаційної роботи_____ О. П. Адамів
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи: 59 с., 7 рис., 1 табл., 2 додатки, 36 джерел.

Об’єктом дослідження є процес автономної навігації мобільного робота в умовах змінного і частково невідомого середовища.

Метою кваліфікаційної роботи є розробка, програмна реалізація та експериментальна перевірка програмного модуля для навігації мобільного робота, яка поєднує деліберативні та реактивні методи управління з використанням нейро-нечіткої системи прийняття рішень.

Методи дослідження включають аналіз і систематизацію наукових джерел у сфері автономної навігації мобільних роботів, моделювання архітектури програмного модуля, застосування алгоритмів деліберативного планування на основі методів вибіркового семплювання, побудову нейро-нечіткої системи прийняття рішень на основі функцій належності та нечітких правил, а також експериментальне тестування і порівняльний аналіз ефективності в умовах змінного середовища.

Результати дослідження полягають у створенні програмного модуля навігації мобільного робота з використанням нейро-нечіткої логіки, який поєднує деліберативне та реактивне управління. Реалізовано архітектуру на основі ROS, розроблено відповідні алгоритми та проведено симуляційне тестування, що підтвердило ефективність запропонованого рішення.

Результати роботи можуть бути використані для розробки інтелектуальних систем автономного керування мобільними роботами, що функціонують у динамічному або частково невідомому середовищі; впровадження навчальних симуляторів у сфері робототехніки.

Ключові слова: МОБІЛЬНИЙ РОБОТ, АВТОНОМНА НАВІГАЦІЯ, НЕЧІТКА ЛОГІКА, НЕЙРО-НЕЧІТКА СИСТЕМА, ДЕЛІБЕРАТИВНЕ УПРАВЛІННЯ, РЕАКТИВНЕ УПРАВЛІННЯ, ПЛАНУВАННЯ ТРАЄКТОРІЇ.

ANNOTATION

The bachelor’s thesis report: 59 pages, 7 figures, 1 table, 2 appendices, 36 sources.

The object of the research is the process of autonomous navigation of a mobile robot in a dynamic and partially unknown environment.

The purpose of this qualification work is the development, software implementation, and experimental evaluation of a navigation module for a mobile robot that combines deliberative and reactive control methods using a neuro-fuzzy decision-making system.

The research methods include analysis and systematization of scientific literature in the field of autonomous mobile robot navigation, modeling of the software module architecture, application of deliberative planning algorithms based on sampling methods, construction of a neuro-fuzzy decision-making system using membership functions and fuzzy rules, as well as experimental testing and comparative analysis of effectiveness in a dynamic environment.

The research results consist in the creation of a navigation software module for a mobile robot using neuro-fuzzy logic that integrates deliberative and reactive control. The architecture based on ROS has been implemented, relevant algorithms have been developed, and simulation testing has confirmed the effectiveness of the proposed solution.

The results of this work can be used for the development of intelligent systems for autonomous control of mobile robots operating in dynamic or partially unknown environments, as well as for the implementation of educational simulators in the field of robotics.

Keywords: MOBILE ROBOT, AUTONOMOUS NAVIGATION, FUZZY LOGIC, NEURO-FUZZY SYSTEM, DELIBERATIVE CONTROL, REACTIVE CONTROL, PATH PLANNING.

ЗМІСТ

ВСТУП

Робототехніка об’єднує досягнення різних наук – електроніки, механіки та інформаційних технологій – для розробки та створення роботизованих систем. Автономний мобільний робот призначений для виконання фізичних завдань без постійної участі людини шляхом впровадження основних положень теорії керування та розробки штучного «інтелекту» [1].

Одним із найбільш критичних аспектів управління роботами є забезпечення плавного і безпечного пересування у визначеному середовищі. Важливою складовою системи навігації є розв’язання задачі виявлення перешкод та планування траєкторії руху з урахуванням безпечної взаємодії робота із навколишнім середовищем [2].

У системах автономної навігації все ширше застосовуються методи штучного інтелекту, серед яких особливе місце займають нечіткі логічні системи [3] та нейронні мережі [4–6]. Нечітка логіка дозволяє моделювати процес прийняття рішень у невизначених умовах, наближаючись до людського способу мислення, тоді як нейронні мережі забезпечують здатність до навчання на основі сенсорних даних.

Використання нечітких систем керування для мобільної навігації дозволяє об’єднати переваги реактивного та деліберативного управління. Реактивна навігація краще підходить для динамічних середовищ, де рішення приймаються на основі актуальних сенсорних даних, тоді як деліберативна навігація орієнтована на побудову оптимального шляху за наявності часткової або повної інформації про середовище [7].

Дослідження показують, що поєднання нейронних мереж та нечіткої логіки у рамках нейро-нечітких систем дозволяє істотно підвищити ефективність управління рухом мобільних роботів у складних умовах [5].

Метою кваліфікаційної роботи є розробка, програмна реалізація та експериментальна перевірка програмного модуля для навігації мобільного робота, яка поєднує деліберативні та реактивні методи управління з

використанням нейро-нечіткої системи прийняття рішень.

Для досягнення поставленої мети необхідно виконати такі основні завдання:

– провести аналіз предметної області управління рухом мобільних роботів;

– проаналізувати відомі рішення, які реалізують методи навігації з використанням нечіткої логіки та нейронних мереж;

– сформулювати постановку задачі дослідження;

– спроєктувати архітектуру програмного модуля;

– розробити алгоритми функціонування окремих компонентів програмного модуля;

– інтегрувати розроблений модуль у загальну систему керування мобільного робота;

– реалізувати програмний модуль навігації;

– провести серію експериментів у різних сценаріях;

– проаналізувати результати експериментів.

Об’єкт дослідження – процес автономної навігації мобільного робота в умовах змінного і частково невідомого середовища.

Предмет дослідження – методи та алгоритми реалізації програмного модуля навігації мобільного робота на основі інтеграції деліберативної та реактивної навігації з нейро-нечіткою системою прийняття рішень.

Результати кваліфікаційної роботи апробовані на студентській науково-практичній конференції “Інтелектуальні інформаційні технології в прикладних дослідженнях” (IITAR – 2025), Тернопіль, ЗУНУ, 27–29 травня 2025 р.

Кваліфікаційна робота складається із вступу, трьох розділів, висновків, списку використаних джерел та додатків.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ УПРАВЛІННЯ РУХОМ МОБІЛЬНИХ РОБОТІВ

1.1 Аналіз сучасних підходів до управління рухом мобільних роботів

Управління рухом мобільних роботів є ключовим завданням у галузі робототехніки. Залежно від характеристик середовища, вимог до точності навігації та рівня автономності, розроблено кілька підходів до організації процесу переміщення мобільних платформ. Серед них виділяють традиційні алгоритми навігації, реактивні методи управління та деліберативні стратегії планування руху.

Традиційні алгоритми навігації ґрунтуються на заздалегідь сформованих картах середовища та використовують алгоритми пошуку найкоротшого шляху, такі як Dijkstra та A*. Алгоритм Dijkstra забезпечує знаходження найкоротшого шляху в графі, але має високу обчислювальну складність, що обмежує його застосування в реальному часі [8]. Алгоритм A* використовує евристичні функції для спрямування пошуку, що дозволяє зменшити обчислювальні витрати порівняно з Dijkstra [9]. Проте обидва алгоритми мають обмеження в динамічних або невідомих середовищах, де необхідна адаптація до змін у реальному часі.

Реактивні методи управління базуються на безпосередньому аналізі поточних даних сенсорів і прийнятті рішень без побудови глобальної карти. Прикладами таких методів є алгоритми потенційних полів та Bug-алгоритми. Алгоритм потенційних полів моделює ціль як притягуюче поле, а перешкоди – як відштовхуючі поля, що дозволяє роботу рухатися до цілі, уникаючи перешкод [10]. Bug-алгоритми, такі як Bug1 та Bug2, дозволяють роботу обходити перешкоди, слідуючи їх контуру, поки не буде знайдено шлях до цілі [11]. Основними перевагами реактивних методів є простота реалізації та здатність працювати в невідомих або змінюваних середовищах. Однак вони можуть призводити до застрягання в локальних мінімумах і не гарантують знаходження оптимального шляху.

Деліберативні стратегії управління передбачають побудову моделі

середовища, оцінку можливих варіантів руху та вибір оптимальної траєкторії. Такі системи використовують методи одночасної локалізації та побудови карти (SLAM) для створення карти середовища та визначення положення робота в ній [12]. Деліберативне управління дозволяє ефективно вирішувати складні навігаційні задачі, передбачати небезпеки та планувати дії на кілька кроків уперед. Проте ці системи мають високу обчислювальну складність, що обмежує їх застосування в реальному часі для роботів із обмеженими ресурсами.

У сучасних робототехнічних системах дедалі частіше використовуються гібридні підходи, які поєднують переваги реактивного та деліберативного управління. У таких системах деліберативний рівень відповідає за загальне планування руху, тоді як реактивний рівень – за локальну адаптацію до перешкод, які з'являються несподівано [13]. Прикладом такої інтеграції є використання нейро-нечітких систем навігації, де нечітка логіка моделює гнучке прийняття рішень, а нейронні мережі забезпечують навчання оптимальній поведінці на основі досвіду.

Таким чином, сучасні підходи до управління рухом мобільних роботів розвиваються у напрямі підвищення адаптивності, надійності та автономності. Вибір конкретної стратегії або їх комбінації залежить від характеристик середовища, цільових задач робота та обмежень апаратного забезпечення.

1.2 Аналіз відомих рішень у сфері навігації мобільних роботів на основі нечіткої логіки та нейронних мереж

Одним із важливих аспектів забезпечення автономної навігації мобільних роботів є розробка ефективних методів виявлення перешкод та планування траєкторії руху у складних середовищах. У межах огляду були проаналізовані декілька ключових наукових праць, що стосуються розвитку методів навігації мобільних роботів.

Одним із критичних завдань, які необхідно вирішити під час організації автономної навігації мобільного робота, є виявлення об'єктів навколо та

уникнення зіткнень. У сучасних системах для цього часто використовуються візуальні сенсори, зокрема RGB-камери або глибинні камери, які дозволяють формувати уявлення про навколишнє середовище у вигляді цифрового зображення або глибинної карти. Саме таку стратегію аналізу середовища обґрунтовано у праці [14].

Автори відзначають, що хоча сенсори візуального типу забезпечують велику кількість детальної інформації про перешкоди, високий обсяг вхідних даних створює додаткове обчислювальне навантаження на систему. Обробка таких об’ємів інформації вимагає застосування швидких і ефективних алгоритмів попередньої обробки, які здатні виокремити лише ті деталі, що є критичними для ухвалення рішень про рух.

У зв’язку з цим у дослідженні запропоновано метод сегментації зображень, спрямований на виділення зон перешкод у вхідних візуальних даних. Зміст методу полягає в автоматичному аналізі вхідного зображення з метою ідентифікації ділянок, які потенційно можуть бути небезпечними для руху робота. Така сегментація дозволяє локалізувати перешкоди на зображенні, відокремивши їх від фону та інших об’єктів, що не становлять загрози.

Перевагою цього підходу є те, що система отримує не повне зображення, яке потребує детального аналізу, а виділені фрагменти з важливою інформацією, які можуть бути використані для побудови карти перешкод, формування потенційних полів або ухвалення рішень у рамках реактивної навігації.

Завдяки зменшенню обсягу даних для подальшої обробки, такий метод підвищує швидкодію і точність реагування системи на появу перешкод, особливо в умовах обмежених ресурсів обчислювальної платформи. Додатково, сегментовані зображення можуть використовуватися для навчання систем комп’ютерного зору, що дозволяє створювати більш інтелектуальні та адаптивні системи керування.

Таким чином, аналізована робота демонструє ефективний підхід до реалізації візуального сприйняття у системах мобільної навігації. Метод сегментації зображень слугує основою для подальших дій робота щодо

уникнення перешкод, особливо у випадках, коли використання глибоких моделей або повноцінного SLAM-підходу є недоцільним або надто ресурсоємним.

Одним із відомих підходів до управління рухом мобільного робота є вдосконалений метод локальної навігації на основі потенційних полів із урахуванням градієнта напрямку до цілі [15]. Аналіз глобальних і локальних методів навігації, виконаний у цій праці, дозволив авторам виокремити ключові недоліки традиційних підходів, зокрема такі як застрягання робота у локальних мінімумах, відсутність механізму обходу перешкод, та складність реагування на динамічні зміни в середовищі.

Запропонований авторами вдосконалений підхід базується на методі потенційних полів, який доповнено врахуванням градієнта до цільової точки. На практиці це означає, що робот не лише притягується до цілі й відштовхується від перешкод, а й додатково орієнтується за напрямом найбільшого зменшення штучного потенціалу – тобто рухається по траєкторії, яка є найбільш перспективною з точки зору досягнення цілі. Такий підхід дозволяє мінімізувати ризик застрягання у «пастках» потенційного поля, де результуюча сила може дорівнювати нулю, але ціль не досягнута.

Крім того, у структурі алгоритму передбачено механізм обходу заблокованих перешкод із контролем повернення на попередню траєкторію. Це означає, що робот має змогу тимчасово відійти від запланованого шляху, щоб оминути перешкоду, а потім продовжити рух у напрямку початково запланованої цілі. Така поведінка наближає систему до принципів адаптивної навігації, що особливо важливо в умовах частково відомого або динамічного середовища.

Метод було протестовано за допомогою імітаційного моделювання на реальній базі мобільного робота Amigo, що дозволило отримати реалістичні результати щодо ефективності та надійності запропонованого алгоритму. Практичні експерименти підтвердили здатність системи долати локальні мінімуми та забезпечувати цілеспрямований рух навіть за наявності складних конфігурацій перешкод. Таким чином, дана робота демонструє один із ефективних підходів до подолання традиційних обмежень локальної навігації.

Використання градієнтного методу в поєднанні з потенційними полями та механізмами контролю дозволяє створити гнучку систему ухилення від перешкод, яка може бути застосована в реальних робототехнічних платформах.

Робота [16] продовжує тему вдосконалення локальної навігації мобільних роботів. У ній описується метод побудови траєкторії руху з використанням потенційних полів та градієнта напряму до цілі. Основна увага приділена підвищенню стійкості навігації у складних умовах середовища шляхом мінімізації ризику застрягання робота у локальних мінімальних зонах.

Проаналізовані праці мають спільний фокус на удосконаленні механізмів локальної навігації мобільних роботів, забезпеченні адаптивного обходу перешкод та зниженні ризику втрати керування при русі у складних або динамічно змінних умовах. Вони демонструють еволюцію підходів: від оптимізації обробки сенсорної інформації до розробки складних стратегій планування руху, що поєднують методи потенційних полів із використанням градієнтних напрямків. Результати досліджень цих авторів є важливим підґрунтям для подальшого розвитку програмних модулів навігації, що базуються на сучасних методах нечіткої логіки та інтелектуальних систем прийняття рішень.

На сьогодні, одним із найпоширеніших підходів до створення систем автономної навігації мобільних роботів є використання нечіткої логіки. Розроблена ще на початку розвитку теорії нечітких множин, нечітка логіка сьогодні широко застосовується у багатьох галузях, включно з управлінням рухом мобільних роботів та обробкою зображень [17]. Технології автономних мобільних платформ значною мірою спираються на здатність нечітких систем адаптуватися до змінних умов середовища, що робить їх критичним елементом сучасних навігаційних рішень.

У багатьох дослідженнях запропоновані підходи із застосуванням інтелектуальних нечітких контролерів, які дозволяють мобільним роботам ефективно орієнтуватися у непередбачуваних або динамічно змінюваних середовищах [18]. Відзначається, що методи нечіткої логіки значною мірою

натхненні особливостями людського мислення, зокрема візуальним сприйняттям, яке визначає спосіб інтерпретації просторової інформації. У задачах навігації та уникнення перешкод нечіткі контролери дозволяють адаптивно модифікувати функції належності параметрів відповідно до змінних даних із сенсорів [19].

Одним із важливих напрямів є застосування оптимізованих моделей нечітких систем, таких як контролери на основі підходу Такагі-Суґено, що дозволяють зменшити похибки навігації шляхом використання градієнтних методів оптимізації. Наприклад, у роботах Qing-Yong та співавторів запропоновано поведінкову модель навігації мобільних роботів, яка передбачає реалізацію чотирьох основних поведінкових стратегій: досягнення цілі, уникнення перешкод, моніторинг середовища та допоміжні дії [20]. Побудова нечітких контролерів для задач уникнення перешкод і досягнення основної мети здійснюється за допомогою визначення восьми основних принципів поведінки [21].

Існують також розробки, що базуються на використанні мікроконтролерів типу Atmega, де нечіткий логічний контролер дозволяє мобільному роботу самостійно пересуватися у просторі без необхідності безпосереднього управління оператором [22]. Система керування отримує дані про відстань до перешкод із набору сенсорів і на їх основі керує приводами лівого та правого коліс робота.

Окрім нечіткої логіки, важливу роль у задачах автономної навігації відіграють штучні нейронні мережі, натхненні принципами роботи людського мозку. Нейронні мережі активно застосовуються у задачах аналізу сигналів, розпізнавання образів, планування маршрутів руху мобільних роботів та інших сферах [23]. Одним із відомих підходів є створення систем планування траєкторії руху на основі багатошарових прямопередавальних нейронних мереж у поєднанні з методами навчання з підкріпленням типу Q-навчання [24].

У системах регулювання швидкості руху використовуються нейронні мережеві контролери разом із класичними PID-регуляторами, що дозволяє

забезпечити більш плавний і точний рух мобільних роботів, зокрема на базі мікроконтролерів Arduino Uno [25]. Також розробляються стратегічні підходи на основі нейронних мереж для автоматизованого управління навігацією транспортних засобів у змінних середовищах [26].

У нестаціонарних середовищах, де можливі непередбачувані зміни розташування перешкод, застосовуються біоінспіровані нейронні мережі, що дозволяють будувати траєкторії з уникненням зіткнень [27, 28]. Зокрема, використання нейронних мереж із підкріпленням для реалізації поведінки «переслідування цілі з уникненням перешкод» показало свою ефективність у плануванні маршруту в умовах обмеженої інформації про середовище [29].

Одним із ефективних підходів є використання змішаних нейронних мереж для управління навігацією мобільних роботів у статичних і динамічних умовах [30]. У таких системах, зокрема, використовується чотиришарова нейронна мережа, яка на основі даних про відстані до перешкод формує керуючий вплив на кут повороту робота.

Суттєвий прогрес демонструють нейро-нечіткі системи навігації, які поєднують адаптивні можливості нейронних мереж із гнучкістю нечіткої логіки. Такі системи дозволяють мобільним роботам ефективно орієнтуватися у незнайомому середовищі, одночасно реалізуючи поведінки досягнення цілі та уникнення перешкод. Для підвищення ефективності навігації використовується навчання нейронної мережі з метою оптимізації параметрів функціональної належності нечітких множин, що дозволяє мінімізувати довжину шляху від стартової точки до цілі [31, 32].

Також перспективним є застосування комбінованих моделей на основі нечітких контролерів типу Такагі-Суґено та нейронних мереж із радіальними базисними функціями (RBFNN), що забезпечує високу адаптивність системи навігації до змін середовища [32]. У таких рішеннях нечітка логіка дозволяє враховувати невизначеність середовища, а нейронна мережа – динамічно адаптувати параметри системи.

Нарешті, для автономної навігації в неструктурованому середовищі

запропоновано поведінкові нейро-нечіткі системи, що регулюють швидкість і траєкторію руху мобільного робота на основі сенсорної інформації та моделей адаптивного навчання [33].

Таким чином, аналіз існуючих підходів свідчить про ефективність застосування нечітких систем та нейронних мереж як окремо, так і в їх інтегрованому використанні для задач автономної навігації мобільних роботів у складних динамічних середовищах.

1.3 Постановка задачі дослідження

Створення ефективної системи управління рухом мобільного робота в умовах змінного або невідомого середовища залишається складною науково-прикладною задачею. Традиційні алгоритми навігації демонструють високу ефективність лише за умови повного знання карти простору та статичності обстановки. Натомість у реальних умовах робот часто стикається з непередбачуваними перешкодами, невизначеністю розташування об’єктів та змінами в конфігурації середовища. Це вимагає розробки більш гнучких і адаптивних систем керування, здатних приймати рішення на основі неповної, нечіткої або суперечливої інформації.

У зв’язку з цим актуальним є завдання створення програмного модуля, який забезпечить автономну навігацію мобільного робота за допомогою механізмів нечіткої логіки. Використання нечітких систем дозволяє описувати знання та приймати рішення у формах, наближених до людської логіки, що особливо важливо при обробці неоднозначних або динамічних даних сенсорів. Нечітка логіка надає можливість не тільки обробляти нечітку вхідну інформацію, а й гнучко формувати правила поведінки робота в різноманітних ситуаціях.

Аналіз існуючих рішень показав, що поєднання методів нечіткої логіки із сенсорними технологіями (зокрема з використанням пристроїв типу Kinect) та симуляційними середовищами на кшталт V-REP забезпечує можливість створення високоефективних навігаційних систем. Використання ROS як базової

програмної платформи дозволяє організувати модульну архітектуру системи керування, що спрощує розробку, тестування та подальше вдосконалення програмного забезпечення.

Отже, метою кваліфікаційної роботи є розробка, програмна реалізація та експериментальна перевірка програмного модуля для навігації мобільного робота, яка поєднує деліберативні та реактивні методи управління з використанням нейро-нечіткої системи прийняття рішень.

Для досягнення поставленої мети необхідно виконати такі основні завдання:

– провести аналіз предметної області управління рухом мобільних роботів;

– проаналізувати відомі рішення, які реалізують методи навігації з використанням нечіткої логіки та нейронних мереж;

– сформулювати постановку задачі дослідження;

– спроєктувати архітектуру програмного модуля;

– розробити алгоритми функціонування окремих компонентів програмного модуля;

– інтегрувати розроблений модуль у загальну систему керування мобільного робота;

– реалізувати програмний модуль навігації;

– провести серію експериментів у різних сценаріях;

– проаналізувати результати експериментів.

Поставлена задача передбачає побудову системи, здатної:

– автономно переміщатися до заданої цілі;

– виявляти та обходити статичні і динамічні перешкоди;

– адаптувати свою поведінку у відповідь на непередбачені зміни у середовищі;

– забезпечувати безпечний рух без зіткнень.

Особливу увагу у рамках дослідження буде приділено формуванню бази нечітких правил, визначенню параметрів функцій належності для вхідних

змінних та розробці механізму узагальненого нечіткого висновку, що дозволить забезпечити високу гнучкість і надійність системи керування рухом мобільного робота.

Таким чином, успішне вирішення поставленої задачі дозволить розробити ефективний інтелектуальний програмний модуль, який підвищить рівень автономності мобільних роботів і забезпечить їхню здатність функціонувати в реальних, змінних середовищах.

Отже, у першому розділі проведено огляд сучасних підходів до управління рухом мобільних роботів, включно з традиційними, реактивними, деліберативними та гібридними стратегіями. Проаналізовано основні існуючі рішення із застосуванням нечіткої логіки та нейронних мереж у задачах автономної навігації, а також розглянуто основні інструментальні засоби для побудови таких систем, зокрема ROS, V-REP та сенсор Kinect.

На основі аналізу літературних джерел сформульовано постановку задачі дослідження, яка полягає у створенні програмного модуля керування рухом мобільного робота з використанням методів нечіткої логіки. Це рішення має забезпечити автономну навігацію в умовах невизначеності та змінного середовища, адаптивне ухилення від перешкод і досягнення заданих цілей.

2 ПРОЄКТУВАННЯ ПРОГРАМНОГО МОДУЛЯ УПРАВЛІННЯ РУХОМ МОБІЛЬНОГО РОБОТА

2.1 Архітектура програмного модуля для автономного управління рухом мобільного робота

Проєктування архітектури програмного модуля для автономного управління рухом мобільного робота базується на принципах модульності, гнучкості та адаптивності. Основу архітектури становить багаторівнева система, яка інтегрує деліберативне та реактивне управління із нейро-нечітким контролером для забезпечення адаптації до змін у середовищі. Взаємодія між компонентами модуля реалізована у середовищі ROS (Robot Operating System), що дозволяє здійснювати паралельну обробку даних, обмін повідомленнями між вузлами та контроль за роботою кожного модуля в реальному часі.

Загалом архітектура складається з трьох логічних рівнів (рисунок 2.1): сенсорного рівня, рівня прийняття рішень та виконавчого рівня.

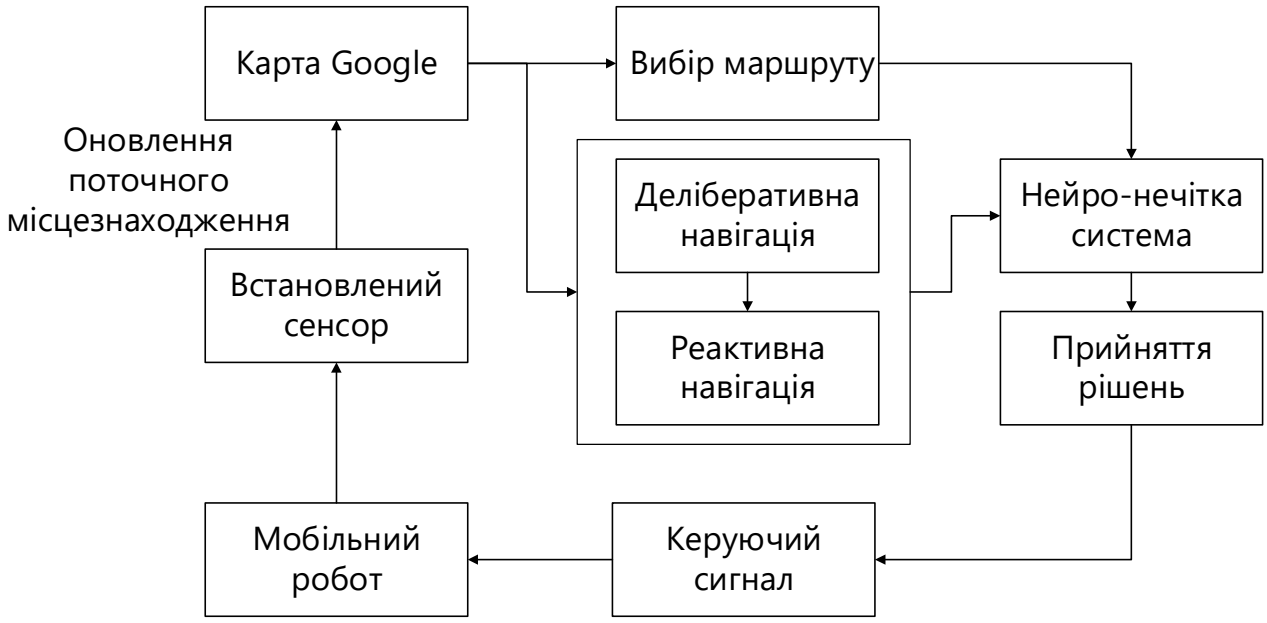

Рисунок 2.1 – Архітектура програмного модуля для автономного управління рухом мобільного робота

Кожен рівень виконує визначені функції, а взаємозв’язки між ними

забезпечують замкнутий цикл управління: від отримання даних про середовище до формування команд для виконавчих механізмів робота.

На сенсорному рівні встановлені пристрої для збору даних про навколишнє середовище – зокрема, глибинний сенсор Kinect, який генерує потоки інформації у форматі RGB-D (зображення та глибина). Ці дані надходять до вузла формування карти, де відбувається реконструкція 2D-площини з відображенням перешкод і вільних зон. Зібрана інформація також використовується для оновлення поточного положення робота на карті (локалізація) та фіксації змін у середовищі.

Наступним є рівень прийняття рішень, який містить декілька основних модулів:

1. Модуль деліберативної навігації, який будує глобальний маршрут від поточного положення до цілі, використовуючи алгоритми семплінгу. Найчастіше використовуються варіанти алгоритмів типу RRT (Rapidly-exploring Random Trees). Побудова маршруту базується на множині станів $S \subseteq \mathbb{R}^n$, між якими будується граф $G = (V, E)$, де V – множина вершин (допустимих положень), E – ребра, що з'єднують ці вершини.

2. Модуль реактивної навігації, що формує локальні траєкторії обходу перешкод, використовуючи метод потенційних полів. Реактивний рух ґрунтується на обчисленні результуючого векторного поля $\vec{F}(x, y)$, що включає притягуюче поле до цілі $\vec{F}_{attr}$ і відштовхуюче поле від перешкод $\vec{F}_{rep}$:

$$\vec{F}(x, y) = \vec{F}_{attr}(x, y) + \sum_{i=1}^{n} \vec{F}_{rep}^{(i)}(x, y) \quad (2.1)$$

Притягуюче поле зазвичай визначається як:

$$\vec{F}_{attr} = -k_{att} \cdot \nabla \left(\frac{1}{2} \cdot d^2(x, y) \right), \quad (2.2)$$

де $d(x, y)$ – відстань до цілі;

k_{att} – коефіцієнт притягання.

Відштовхуюче поле моделюється за допомогою гаусового ядра для кожної перешкоди:

$$V_{rep}(x,y) = A \cdot \exp\left(-\left(\frac{(x-x_0)^2}{2\sigma_x^2} + \frac{(y-y_0)^2}{2\sigma_y^2}\right)\right), \quad (2.3)$$

де A – максимальна інтенсивність поля;

σ_x, σ_y– параметри ширини по осях;

(x_0, y_0) – центр перешкоди.

3. Нейро-нечітка система прийняття рішень, яка здійснює узгодження між деліберативним і реактивним режимами на основі двох параметрів.

Припустимо, що маємо два вхідні параметри:

$p \in [0,1]$ – ступінь валідності маршруту;

$s \in [0,1]$ – рівень безпеки руху.

На виході формується коефіцієнт ухваленого рішення $f \in [0,1]$, за допомогою якого обчислюється результуюча швидкість:

$$V = f \cdot V_{delib} + (1-f) \cdot V_{react} \quad (2.4)$$

де V_{delib} – швидкість за глобальним планом;

V_{react} – швидкість за реактивною моделлю.

Кінцевим є виконавчий рівень, який містить модуль керування приводами. На основі обчислених швидкісних векторів формується керуючий сигнал $u(t)$, що подається на актуатори. Керування здійснюється за допомогою ПІ-регулятора, який задається рівнянням:

$$u(t) = K_p \cdot e(t) + K_i \int_0^t e(\tau)\, d\tau \quad (2.5)$$

де $e(t)$ – поточна похибка між бажаним і фактичним положенням;

K_p, K_i – коефіцієнти регулювання.

Усі компоненти програмного модуля функціонують в єдиному інформаційному середовищі, що дозволяє організувати асинхронну обробку даних, забезпечити паралельну роботу модулів і гнучко масштабувати систему при додаванні нових функцій або сенсорів. Завдяки використанню ROS, система підтримує розширення та інтеграцію з іншими платформами, а її компоненти можуть бути повторно використані у подальших проєктах.

Таким чином, архітектура програмного модуля управління забезпечує високу ступінь автономності мобільного робота, дозволяє ефективно поєднувати глобальне планування з локальною адаптацією та створює основу для реалізації поведінкових стратегій з використанням штучного інтелекту. Вона також забезпечує модульність, прозорість внутрішніх процесів і можливість тестування в симуляційному середовищі V-REP перед переходом до реального апаратного прототипу.

2.2 Розробка алгоритмів функціонування програмного модуля

2.2.1 Алгоритм деліберативної навігації

Деліберативна навігація – це один із ключових етапів у реалізації автономного управління рухом мобільного робота. Її основна мета полягає в побудові оптимального маршруту до цільової точки, виходячи з попередньо зібраної або частково відомої карти середовища. Цей підхід ґрунтується на плануванні руху з урахуванням глобальної інформації про простір, що забезпечує цілеспрямоване переміщення робота між стартовою і цільовою позиціями при мінімізації довжини або небезпеки траєкторії.

На початковому етапі алгоритм деліберативної навігації отримує дані про поточне положення мобільного робота, координати цільової точки, а також карту середовища, яка може бути сформована із сенсорних вимірювань або за допомогою синхронної локалізації та побудови карти (SLAM). Як зазначено у схемі, показаній на рисунку 2.2, карта додатково узгоджується з глобальними

навігаційними даними, наприклад, із Google Maps, що дозволяє враховувати просторове розміщення об’єктів за межами локального поля зору робота.

Рисунок 2.2 – Структурна схема реалізації деліберативної навігації мобільного робота на основі симулятора V-REP

Основним інструментом у деліберативному плануванні виступає алгоритм побудови траєкторії на основі вибіркового семплювання, наприклад, RRT (Rapidly-exploring Random Trees) або його модифіковані варіанти. Завдяки здатності ефективно працювати у складних і розгалужених середовищах, ці алгоритми дозволяють створити дерево допустимих шляхів, з якого вибирається найкоротший або найнадійніший маршрут до цілі. У процесі побудови враховується ймовірність зайнятості окремих ділянок карти, що дозволяє уникати ризикованих зон.

Після побудови маршруту координати ключових контрольних точок передаються у блок реактивної навігації та нейро-нечіткої системи для подальшої перевірки валідності шляху в умовах зміни середовища. Якщо план втрачає актуальність, наприклад, через появу нових перешкод, він автоматично скасовується і генерується повторно з урахуванням оновленої карти.

Псевдокод алгоритму деліберативної навігації мобільного робота:

```
ІНІЦІАЛІЗАЦІЯ:
  Завантажити карту місцевості
  Встановити початкову та цільову позиції робота
  Ініціалізувати модулі Move-it та V-REP
  Запустити симулятор

ЦИКЛ НАВІГАЦІЇ (поки ціль не досягнута):

  // КРОК 1: Отримання та оновлення даних
  SENSORS ← зчитати сенсорні дані
  Оновити карту середовища у V-REP
  Локалізувати позицію робота
  Узгодити координати з картою Google

  // КРОК 2: Планування маршруту
  PLAN ← побудувати маршрут до цілі за допомогою Move-it
  if PLAN == NULL:
      Повідомити: "Неможливо побудувати маршрут"
      break

  // КРОК 3: Вибір траєкторії
  PATH ← вибрати оптимальну траєкторію з PLAN
  ПЕРЕВІРИТИ валідність PATH (враховуючи оновлення карти)

  if PATH невалідна:
      Запустити перепланування
      continue

  // КРОК 4: Формування керуючого сигналу
  VELOCITY ← оцінити швидкість для поточної ділянки PATH
  SIGNAL ← сформувати керуючий сигнал на основі VELOCITY
  Надіслати SIGNAL до приводу робота через ROS

  // КРОК 5: Перевірка досягнення цілі
  if distance(робот, ціль) < ε:
      Зупинити рух
      break

КІНЕЦЬ ЦИКЛУ
```

Таким чином, алгоритм деліберативної навігації виконує роль "стратега", який визначає загальний напрямок руху та основні проміжні цілі. У поєднанні з

адаптивними реактивними методами цей підхід забезпечує надійне досягнення цілі навіть у частково або повністю невідомих середовищах.

2.2.2 Алгоритм реактивної навігації

Реактивна навігація відіграє ключову роль у забезпеченні здатності мобільного робота своєчасно реагувати на локальні зміни середовища, зокрема на появу нових перешкод, які не були враховані під час глобального планування. На відміну від деліберативного рівня, що працює з повною або частково відомою картою простору, реактивна навігація функціонує на основі даних, отриманих безпосередньо із сенсорів у режимі реального часу.

На рисунку 2.3 подано послідовність схему реалізації реактивної навігації, який реалізується у середовищі симуляції V-REP (CoppeliaSim).

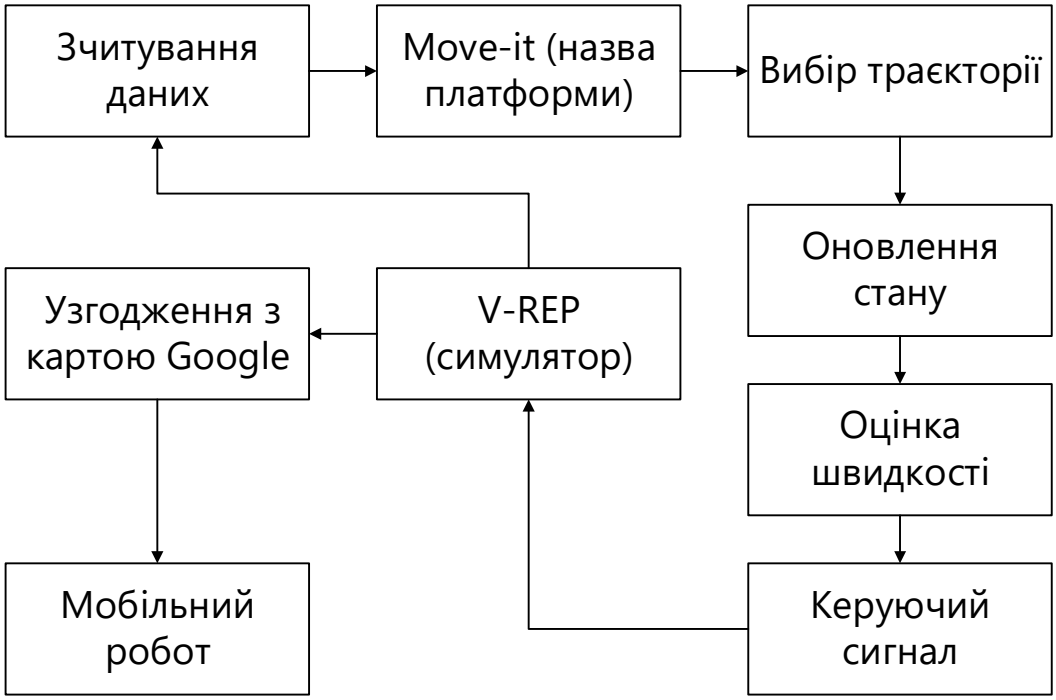

Рисунок 2.3 – Структурна схема реалізації реактивної навігації мобільного робота на основі симулятора V-REP

Мобільний робот постійно отримує інформацію про оточення за допомогою сенсорів – наприклад, глибинного сенсора чи лазерного далекоміра. Ці дані передаються у вузол Sensing data, де формується поточна картина

середовища.

Далі у вузлі Set the Potential point визначається набір привабливих та відштовхуючих точок, які моделюють вплив цілі та перешкод на навігацію. В основі цього етапу лежить концепція потенційних полів. Для опису відштовхуючих полів від перешкод використовується гаусове ядро, представлене формулою:

$$Z(x,y) = A \cdot exp\left(-\left(\frac{(x-x_a)^2}{2\sigma_x^2} + \frac{(y-y_a)^2}{2\sigma_y^2}\right)\right), \quad (2.6)$$

де A – максимальна інтенсивність поля;

σ_x, σ_y – ширини гаусової функції по відповідних координатах;

x_a, y_a – координати центру перешкоди.

Після цього на основі відомих значень потенційних полів у вузлі Gradient vector обчислюється напрямок найшвидшого зменшення потенціалу, тобто напрямок руху до цілі з одночасним обходом перешкод:

$$\vec{F}(x,y) = -\nabla Z(x,y). \quad (2.7)$$

Отриманий градієнт використовується у вузлі Estimate velocity для визначення швидкості робота у відповідному напрямку. Після цього у вузлі Control Signal формується команда на виконавчі механізми, яка направляється до приводу мобільного робота.

Крім того, реактивна навігація включає динамічну оцінку рівня безпеки та відстані до перешкод. Ці параметри передаються до нейро-нечіткої системи, яка приймає рішення щодо переходу між деліберативною та реактивною стратегією або об’єднання обох підходів для досягнення кращої адаптивності. У межах нейро-нечіткого контролера використовуються сенсорні дані: довжина до перешкод спереду d, зліва d_1, справа d_2 та кут до цілі α, що надходять у вхідний шар із чотирма нейронами.

На наступних прихованих шарах реалізується функціональна нечітка класифікація. Вихід формується на основі правил нечіткої логіки та виражається як:

$$f_i - S_i(p_i d + q_i d_1 + r_i d_2 + s_i \alpha), \tag{2.8}$$

де f_i – відповідь i-го нейрона,

p_i, q_i, r_i, s_i – вагові коефіцієнти, що навчаються.

Попередньо кожне S_i нормалізується:

$$S_i = \frac{W_i^T}{\sum_{i=1}^{48} W_i^T}, \tag{2.9}$$

а остаточний вихід системи обчислюється як:

$$output = \sum_{i=1}^{48} f_i. \tag{2.10}$$

Для формування функцій належності у нейро-нечіткій системі використовується узагальнене колоколоподібне ядро:

$$F_n(x; a, b, c) = \frac{1}{1+\left|\frac{x-c}{a}\right|^{2b}}. \tag{2.11}$$

Така побудова забезпечує гнучке, нечітке подання простору рішень, що дозволяє системі автономно вибирати стратегію руху з урахуванням змін у середовищі.

Псевдокод алгоритму реактивної навігації мобільного робота:

```
// КРОК 1: Збір даних
Отримати сенсорні дані з навколишнього середовища
Оновити карту середовища у V-REP

// КРОК 2: Встановлення потенційної точки
Встановити координати цілі як атрактор (потенційна точка
притягування)

// КРОК 3: Обчислення потенційного поля
Для кожної перешкоди:
    Обчислити репульсивне поле за допомогою Гаусового ядра:
    Z(x, y) = A * exp(-(((x - x₀)^2) / (2σx^2) + ((y - y₀)^2) /
    (2σy^2)))
Сумарне потенційне поле = поле цілі + поля перешкод

// КРОК 4: Обчислення градієнта
∇Z ← обчислити градієнт потенційного поля
F ← -∇Z (вектор сили руху до цілі)

// КРОК 5: Оцінка швидкості
V_linear ← модуль вектора F
V_angular ← напрямок вектора F

// КРОК 6: Формування керуючого сигналу
Створити керуючий сигнал Twist:
    cmd_vel.linear ← V_linear
    cmd_vel.angular ← V_angular

// КРОК 7: Надсилання команд
Надіслати cmd_vel до робота через ROS

// КРОК 8: Перевірка завершення
Якщо відстань до цілі < ε:
    Зупинити робота
    Завершити цикл
```

У підсумку, алгоритм реактивної навігації дозволяє роботу:

– у режимі реального часу уникати зіткнень;
– адаптувати швидкість і напрямок руху;
– передавати параметри до загального прийняття рішень;
– залишатися стабільним навіть при зміні конфігурації середовища.

2.3 Інтеграція програмного модуля в систему керування мобільного робота

Інтеграція програмного модуля управління навігацією в загальну систему керування мобільного робота є критичним етапом для забезпечення узгодженої та стабільної взаємодії між апаратним та програмним забезпеченням. Впровадження інтелектуального модуля на базі деліберативної, реактивної та нейро-нечіткої навігації вимагає створення єдиного інформаційного середовища, де кожен функціональний блок працює в режимі реального часу, обмінюючись даними через спільні канали.

Центральною платформою, яка забезпечує взаємозв'язок між усіма модулями, є Robot Operating System (ROS) – сучасна відкрита програмна екосистема, орієнтована на створення розподілених робототехнічних систем. ROS дозволяє реалізувати архітектуру у вигляді набору незалежних вузлів (nodes), які виконують конкретні завдання: обробку сенсорних даних, побудову карти, навігаційне планування, ухвалення рішень та формування керуючих сигналів.

Програмний модуль інтегрується з фізичним або симульованим роботом за допомогою таких основних інтерфейсів:

1. Інтерфейс сенсорів – включає отримання даних із глибинних камер, ультразвукових сенсорів або лідарів. Зібрана інформація надходить у модулі побудови карти та локалізації. У разі використання симулятора (V-REP або CoppeliaSim) дані передаються через спеціальні ROS-топіки типу /camera/depth, /scan, або /odom.

2. Інтерфейс глобальної навігації – відповідає за побудову траєкторії до цілі. Алгоритми деліберативного планування, реалізовані на базі MoveIt! або navfn, генерують глобальний маршрут у форматі списку точок, що передаються до реактивного шару для локальної адаптації.

3. Інтерфейс локальної навігації – реалізує реактивну поведінку на основі поточних сенсорних вимірювань та потенційних полів. У цьому модулі обчислюються градієнтні вектори та швидкості уникнення перешкод, які

динамічно коригують глобальний маршрут.

4. Нейро-нечіткий контролер – функціонує як окремий ROS-вузол, який отримує дані з навігаційних підсистем та повертає узгоджене рішення. Це рішення формалізується у вигляді коефіцієнта змішування між реактивною та деліберативною стратегіями, на основі якого формується остаточна команда швидкості руху:

$$V = f \cdot V_{delib} + (1 - f) \cdot V_{react}. \quad (2.12)$$

5. Інтерфейс актуаторів – виконує передачу сформованих керуючих сигналів до низькорівневих пристроїв: коліс, сервоприводів або двигунів. У ROS це може реалізовуватись через контролери типу ros_control, diff_drive_controller, які інтерпретують команди швидкості у форматі geometry_msgs/Twist.

Для забезпечення надійності й ефективності комунікацій використовуються топіки, сервіси та екшн-сервери ROS, що дозволяє організувати як одноразові, так і циклічні обміни даними. Наприклад, передача команди руху здійснюється через топік /cmd_vel, а спостереження за виконанням траєкторії – через /feedback.

Важливим аспектом інтеграції є синхронізація даних між модулями. Для цього застосовується message_filters або механізми таймстемпів, які дозволяють уникнути затримок або втрати актуальності під час обробки швидкоплинних сенсорних подій.

Під час тестування у симульованому середовищі V-REP було забезпечено візуалізацію кожного етапу обробки: побудова карти, генерація потенційного поля, обчислення траєкторії, прийняття рішень та зміна положення робота. Це дозволило не лише налагодити логіку взаємодії, а й оцінити часові затримки між модулями, що критично важливо для застосування системи в реальних умовах.

У результаті інтеграції програмного модуля до системи керування мобільного робота вдалося досягти таких властивостей:

– автономність – здатність приймати рішення без зовнішнього

втручання;

– гнучкість – можливість адаптації до змін навколишнього середовища;

– модульність – розділення функцій на незалежні блоки;

– масштабованість – можливість розширення функціоналу без зміни базової структури.

Таким чином, інтеграція програмного модуля у систему керування забезпечує цілісну і злагоджену роботу мобільного робота, де сенсори, навігаційні алгоритми, нейро-нечітке прийняття рішень і механізми керування приводами утворюють єдиний адаптивний цикл дії, придатний для реалізації у реальних або змодельованих сценаріях.

3 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ПРОГРАМНОГО МОДУЛЯ

3.1 Організація експериментальної бази та симуляційного середовища

Для створення практичної експериментальної бази в рамках цієї кваліфікаційної роботи було обрано комбінацію мобільного робота Amigobot та глибинного сенсора Kinect. Такий вибір зумовлений низкою важливих технічних і практичних переваг, зокрема доступністю, відкритістю апаратної платформи, підтримкою у ROS, а також поширеним використанням у дослідницькому середовищі.

Amigobot – це компактна автономна платформа з колісним приводом, яка широко використовується у навчанні та дослідженнях з мобільної робототехніки. Його основні характеристики:

– колісна база – диференційне керування з двома незалежними приводами для повороту та пересування вперед/назад;

– вбудована система управління – контролер, який дозволяє взаємодіяти з датчиками та моторами через послідовний інтерфейс або бездротово;

– підтримка ROS – існують готові драйвери та пакети, які дозволяють інтегрувати Amigobot у ROS, що забезпечує спрощене підключення до навігаційної системи та симуляторів;

– можливість модифікації – до платформи легко додаються додаткові сенсори (наприклад, камери, ультразвукові чи глибинні сенсори).

Amigobot має невеликі габарити, що дозволяє йому безпечно переміщатися в обмежених просторах, таких як лабораторії або модульні симуляційні середовища. Робот добре пристосований до тестування алгоритмів руху, навігації, обходу перешкод, адаптивного планування траєкторій.

Kinect – це пристрій глибинного бачення, спочатку розроблений компанією Microsoft для ігрової консолі Xbox, який згодом став популярним у сфері комп’ютерного зору та робототехніки. Основні переваги сенсора Kinect:

– отримання зображень у форматі RGB-D – сенсор надає одночасно звичайне зображення (кольорове) та карту глибини, що дозволяє відображати об’єкти у 3D-просторі;

– ПОЛЕ зору та точність – забезпечує глибину в межах 0.5–4 метрів із достатньою деталізацією для задач навігації;

– легка інтеграція з ROS – доступні драйвери (freenect, iai_kinect2, openni) дозволяють підключити Kinect до ROS та використовувати його як джерело сенсорних даних;

– використання для SLAM – сенсор часто застосовується для побудови карти середовища за допомогою SLAM-алгоритмів (Simultaneous Localization and Mapping).

У даній системі Kinect виконує функцію первинного джерела інформації про навколишнє середовище. Дані з сенсора використовуються для:

– побудови карти простору (вузол Mapping);

– виявлення та класифікації перешкод;

– оновлення позиції робота на карті (локалізація);

– аналізу потенційного поля та векторів руху.

Поєднання Amigobot і Kinect дозволяє реалізувати повноцінну мобільну платформу для дослідження алгоритмів навігації в реальних та симульованих умовах. Серед переваг такої конфігурації:

– вартість – обидва компоненти є економічно доступними, що дозволяє створити дослідницьку платформу навіть у рамках обмеженого бюджету;

– гнучкість – на платформу можна встановити додаткові модулі (наприклад, Raspberry Pi, Arduino, інші датчики);

– сумісність – наявність драйверів та документації для інтеграції з MoveIt!, ROS і V-REP спрощує розгортання архітектури;

– можливість моделювання – на основі CAD-моделі робота та сенсора можливо створити повну віртуальну сцену в симуляторі V-REP (CoppeliaSim).

Таким чином, вибір Amigobot та Kinect як основи експериментальної платформи забезпечив практичну реалізацію всіх функцій запропонованої

навігаційної системи, з можливістю гнучкого налаштування, симуляції та подальшого масштабування. Це рішення є збалансованим як з погляду технічної складності, так і з точки зору вартості та доступності апаратних компонентів.

Для реалізації програмного модуля навігації мобільного робота використовувалась операційна система Ubuntu 12.04.5 LTS разом із версією ROS Groovy. Усі програмні вузли були реалізовані за допомогою цього середовища, і система успішно функціонувала у симуляційному середовищі.

У процесі реалізації програмного модуля управління рухом мобільного робота було використано низку сучасних фреймворків та інструментів, які забезпечують підтримку симуляцій, планування руху, взаємодії між модулями, а також моделювання реального середовища.

Robot Operating System (ROS) – це відкритий середовищний фреймворк, призначений для розробки робототехнічних систем. Він не є операційною системою в класичному розумінні, а радше middleware – програмною платформою, яка забезпечує обмін повідомленнями між окремими компонентами системи. Серед основних можливостей ROS:

– мережева архітектура – дозволяє розділити робототехнічну систему на незалежні вузли (nodes), які можуть працювати паралельно, обмінюючись інформацією через топіки (topics) і сервісні виклики;

– бібліотеки і пакети – ROS включає велику кількість готових до використання бібліотек для роботи із сенсорами, керуванням, плануванням, SLAM, локалізацією тощо;

– інструменти налагодження – такі як rqt, rviz, rosbag, що дозволяють візуалізувати, записувати й аналізувати дані під час роботи робота;

– підтримка симуляторів – повна інтеграція з V-REP (тепер CoppeliaSim), Gazebo та іншими середовищами моделювання.

У реалізації даного програмного модуля ROS виступає як базовий «зв’язуючий шар», який дозволяє організувати взаємодію між деліберативною, реактивною та нейро-нечіткою підсистемами.

MoveIt! – це надбудова над ROS, орієнтована на задачі планування руху,

кінематики та динаміки роботів. Спочатку вона створювалася для керування маніпуляторами, але надалі стала застосовуватись і для мобільних платформ. Її ключові функціональні можливості:

– планування траєкторій – генерація оптимальних маршрутів від початкової до цільової точки з урахуванням перешкод;

– облік кінематичних обмежень – підтримка зворотної та прямої кінематики для різних типів роботів;

– колізійне моделювання – система перевіряє, чи не перетинає траєкторія зони перешкод;

– інтеграція з сенсорними системами – дозволяє враховувати карту середовища, побудовану на основі сенсорних даних;

– інтерактивне середовище – у поєднанні з RViz MoveIt! дозволяє тестувати траєкторії в реальному часі.

У даній роботі MoveIt! використовується як основний інструмент деліберативної навігації, тобто глобального планування маршруту в просторі, побудованому на основі сенсорної інформації та Google Maps.

V-REP (Virtual Robot Experimentation Platform) – це симуляційне середовище для тривимірного моделювання робототехнічних систем, яке з 2019 року має нову назву CoppeliaSim. Воно дозволяє:

– моделювати поведінку роботів у складних середовищах – створювати сцени з перешкодами, нерівностями, рухомими об’єктами.

– імітувати сенсори та приводи – наприклад, камери, лідари, ультразвукові датчики, мотори коліс.

– реалізовувати алгоритми управління у віртуальному середовищі – V-REP має вбудовані API для взаємодії з ROS через ROS Interface.

– підтримує зворотний зв’язок – V-REP може надсилати дані з симуляції до ROS, де відбувається обчислення керуючих сигналів, які потім повертаються в симуляцію.

У цій кваліфікаційній роботі V-REP використовується для створення симуляційної сцени з мобільним роботом (Amigobot) та його взаємодії з

перешкодами й навколишнім середовищем. Це дозволяє безпечно перевірити алгоритми до їх реального впровадження.

Комбінація ROS, MoveIt! та V-REP дозволяє реалізувати повноцінну програмну архітектуру, яка поєднує симуляцію, управління, навігацію та планування в одному середовищі. Такий підхід дозволяє ефективно розробляти, тестувати та масштабувати системи автономного керування мобільними роботами, з урахуванням факторів навколишнього середовища та непередбачуваних змін.

3.2 Програмна реалізація модуля автономної навігації мобільного робота

Опис програмної реалізації модуля автономної навігації мобільного робота здійснено шляхом побудови архітектури, що поєднує реактивне та деліберативне управління з використанням нейро-нечіткої системи ухвалення рішень. Для забезпечення функціонування всіх компонентів було реалізовано декілька окремих програмних модулів у середовищі ROS, що дозволило організувати взаємодію між ними та підтримку реального часу. Програмна реалізація модуля автономної навігації мобільного робота включає (додаток А):

– reactive_navigation.py – для реактивної навігації за допомогою потенційних полів;

– deliberative_navigation.py – для глобального планування маршруту на основі MoveIt!;

– decision_fuzzy_controller.py – нейро-нечіткий модуль прийняття рішень;

– navigation.launch – файл запуску всіх модулів у ROS.

Перший модуль – реактивна навігація, реалізована у файлі reactive_navigation.py. Цей компонент відповідає за безпосереднє управління рухом робота у локальному середовищі, використовуючи метод потенційних полів. Робот реагує на перешкоди та притягується до цільової точки, яка задається у вигляді координат. Потенціальне поле розраховується на основі

гаусового ядра, а рух робота визначається за градієнтом цього поля. Таким чином, формується швидкісний вектор, який дозволяє роботу уникати зіткнень і поступово прямувати до цілі.

Другий модуль – деліберативна навігація, розміщений у файлі deliberative_navigation.py, відповідає за побудову глобального маршруту від стартової точки до цільової, з урахуванням карти середовища та доступної інформації від сенсорів. Для цього використовується планувальник з бібліотеки MoveIt!, що дозволяє генерувати траєкторії руху з урахуванням конфігурації робота та обмежень. У випадку зміни середовища модуль здійснює перепланування маршруту в режимі онлайн.

Третім ключовим компонентом є нейро-нечітка система прийняття рішень, яка реалізується у файлі decision_fuzzy_controller.py. Цей модуль координує взаємодію між деліберативною і реактивною навігацією. Залежно від поточної ситуації – валідності маршруту, наявності перешкод, рівня безпеки – система ухвалює рішення, яке управління є більш доцільним: планове чи реактивне. Використання нечіткої логіки дозволяє не тільки перемикатись між стратегіями, але й об’єднувати їх, формуючи інтегрований керуючий сигнал.

Для об’єднання всіх вузлів у єдину систему використовується файл запуску ROS (наприклад, navigation.launch), який ініціалізує роботу всіх компонентів – від симуляції в середовищі V-REP до запуску керуючих алгоритмів. Усі модулі взаємодіють між собою через топіки, сервіси та повідомлення ROS, що забезпечує модульність та гнучкість системи.

Загалом реалізований програмний модуль є комплексним, адаптивним рішенням для автономної навігації мобільного робота, здатним працювати в умовах невизначеного середовища та здійснювати управління рухом з урахуванням мінливих обставин. Система підтримує масштабування та інтеграцію з додатковими модулями або сенсорами, що дозволяє використовувати її у реальних сценаріях, у тому числі й на апаратному рівні з роботами типу Amigobot.

3.3 Аналіз результатів та оцінка ефективності запропонованого рішення

В процесі оцінювання ефективності було проаналізовано, наскільки успішно робот може досягти заданої цілі, не потрапляючи у небезпечні ситуації. Виявилося, що при використанні лише реактивної навігації, без глобального планування маршруту, робот не завжди зміг дістатися до цілі. Це пояснюється тим, що в складному середовищі з великою кількістю перешкод реактивна система не здатна обчислити найкращий шлях, особливо якщо довкола є зони з обмеженими можливостями для маневрування.

Деліберативна навігація, яка використовує попереднє планування маршруту з можливістю його оновлення під час руху, показала кращі результати щодо досягнення цілі. Проте, навіть при її використанні іноді виникали ситуації, коли робот торкався стін або потрапляв у зіткнення. Основні причини цього – неточність у керуванні рухом, недостатній час для перепланування шляху або занадто вузькі проходи для заданих розмірів платформи робота. Один зі способів покращення такої ситуації – це збільшення радіусу безпеки навколо перешкод під час планування або зниження максимальної швидкості робота. Проте такі зміни вимагають адаптації системи під конкретні сценарії використання.

Для оцінки роботи запропонованої комбінованої навігаційної системи були створені різні симуляційні середовища. У таблиці 3.1 наведено результати експериментів у чотирьох типових сценаріях, які враховували рівень знань про навколишнє середовище.

Таблиця 3.1 – Різні варіанти сценаріїв середовища

Середовище	Сценарій 1	Сценарій 2	Сценарій 3	Сценарій 4
Точно відоме	82%	35%	15%	10%
Повністю невідоме	20%	40%	83%	96%
Частково або помилково відоме	8%	37%	14%	0%

Ці результати свідчать, що у випадках, коли середовище було повністю невідоме або змінювалося в процесі руху, нейро-нечітка система показувала кращу здатність до адаптації. Це досягалося завдяки гнучкому поєднанню реактивної та деліберативної навігації.

На основі даних, отриманих від сенсорів у режимі реального часу, система формує вектор швидкості для кожного положення робота. Як видно з графіків на рисунках 3.1–3.4, поведінка робота змінюється залежно від ситуації у навколишньому середовищі. У деяких випадках домінує реактивна поведінка, коли робот швидко уникає перешкод, а в інших – деліберативна, коли рух відбувається відповідно до заздалегідь запланованого маршруту.

Графік на рисунку 3.1 демонструє побудову поля швидкостей у реактивному режимі, тоді як рисунок 3.2 ілюструє траєкторію руху мобільного робота в одному з експериментів. На рисунку 3.3 представлено аналіз вектора швидкості за модулем (тобто величиною), а на рисунку 3.4 – за кутом (напрямком).

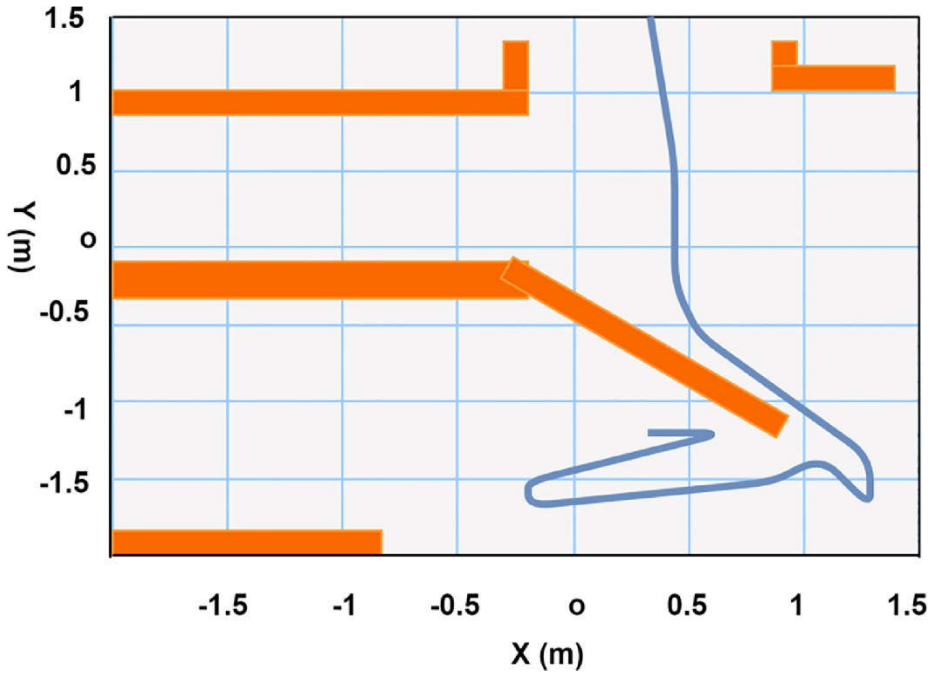

Рисунок 3.1 – Поле реактивних швидкостей

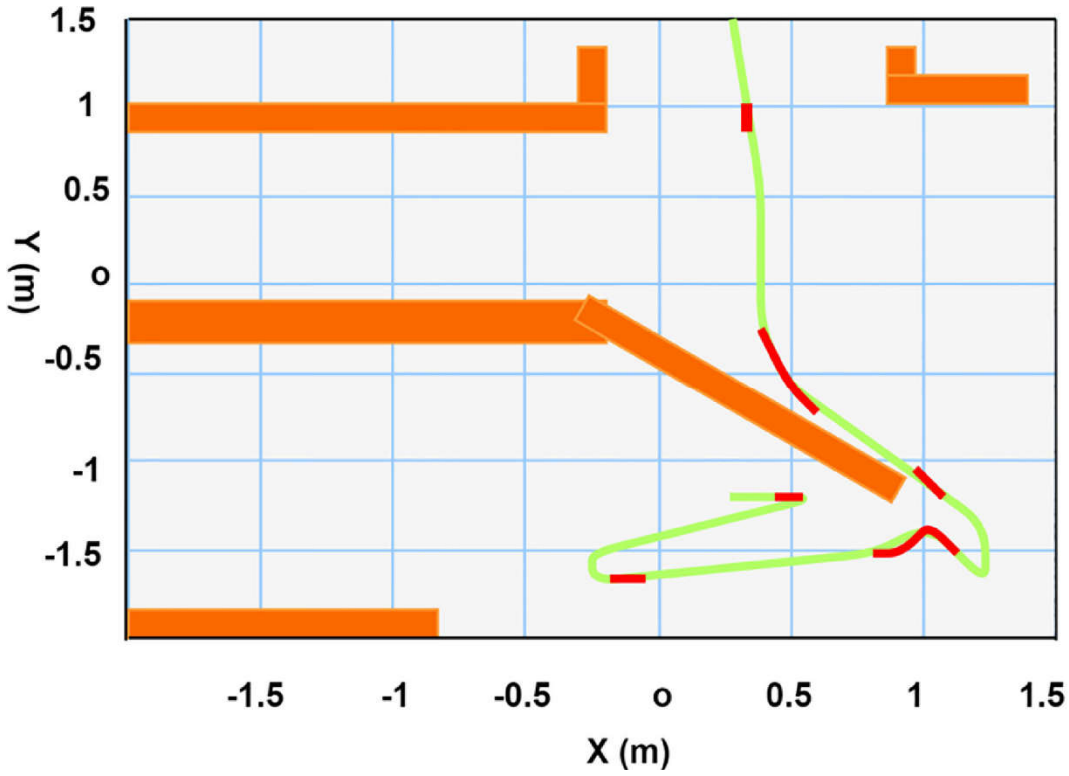

Рисунок 3.2 – Траєкторія навігації мобільного робота

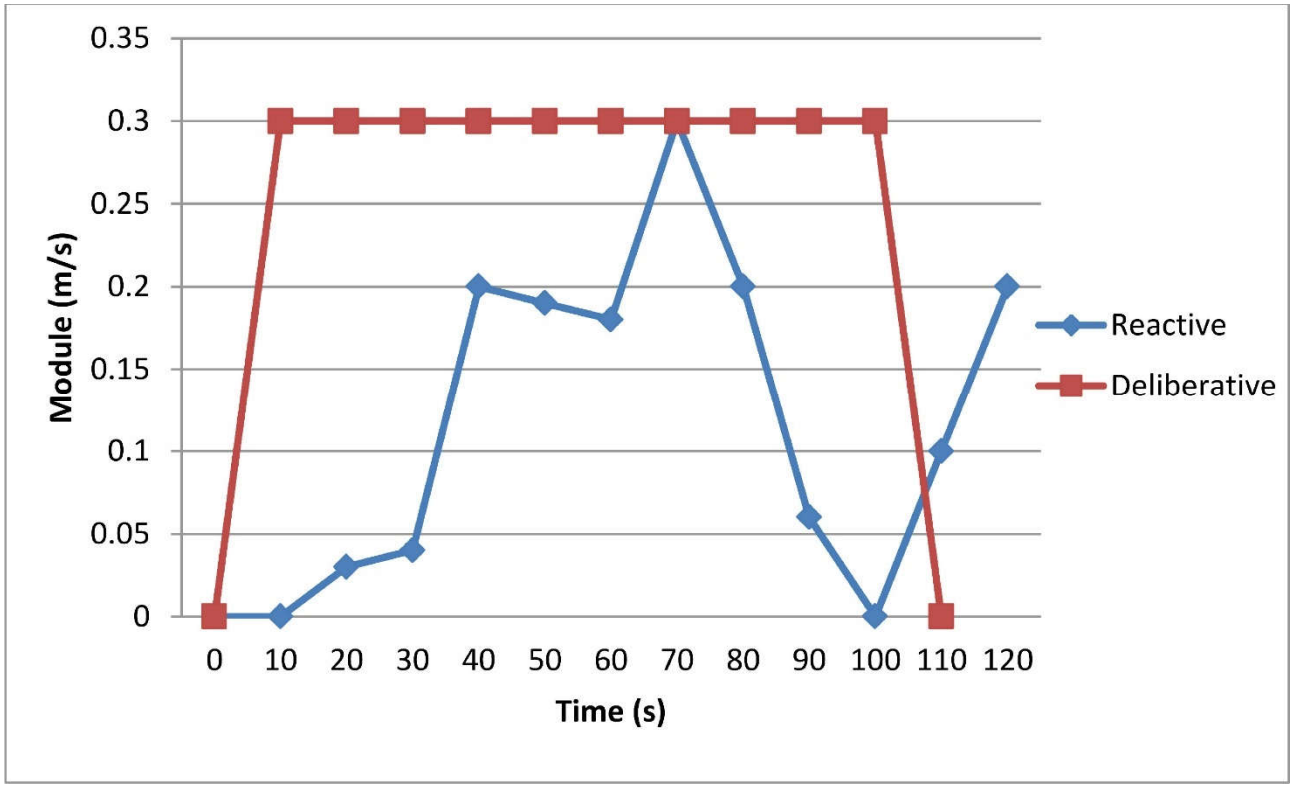

Рисунок 3.3 – Вектор швидкості в контексті аналізу модуля

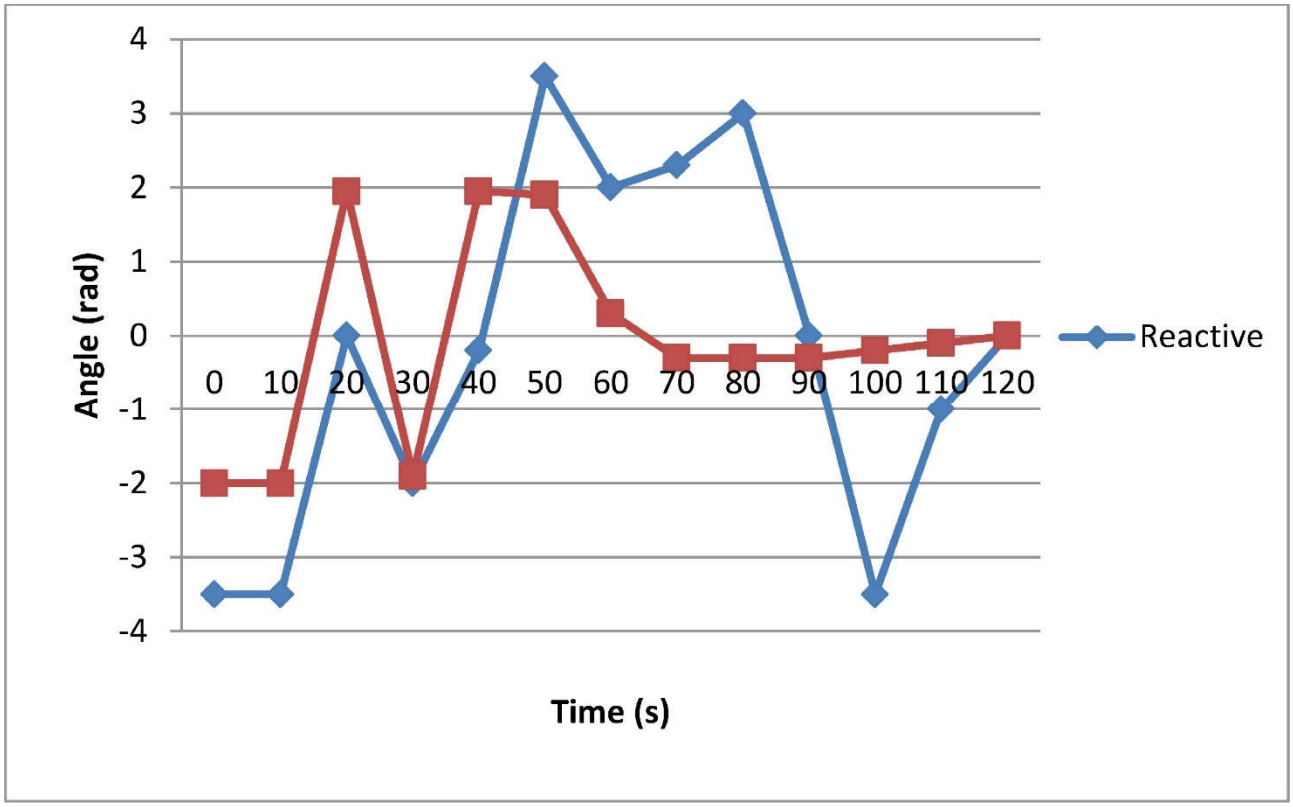

Рисунок 3.4 – Вектор швидкості в контексті аналізу кута

Результати досліджень підтверджують доцільність використання нейро-нечіткої системи для навігації мобільних роботів, особливо в умовах невизначеності або часткової втрати даних про середовище. Такий підхід забезпечує високу гнучкість, здатність до адаптації та зменшення ймовірності зіткнень.

Отже, у роботі було запропоновано інтегрований модуль навігації мобільного робота, який поєднує переваги реактивного та деліберативного підходів, водночас зменшуючи їхні основні недоліки, такі як ризик застрягання у локальному мінімумі та тривалий час планування. Для гармонійного поєднання обох стратегій використано нейро-нечітку систему, яка дозволяє узгоджувати керуючі сигнали, що формуються обома підходами, та визначати пріоритети в разі конфлікту між ними.

Реактивний рівень працює асинхронно, використовуючи дані з локальної карти та сенсорного оточення робота. Це забезпечує швидку реакцію на появу перешкод і зміну ситуації в безпосередній близькості. У той же час деліберативна

навігація реалізована на основі алгоритмів семплінгу з урахуванням концепції імовірнісної повноти, що дозволяє скоротити час планування навіть у складних і частково невідомих середовищах.

У процесі реалізації та тестування архітектури використовувалися популярні фреймворки та інструменти – такі як MoveIt!, ROS та симуляційне середовище V-REP. Завдяки цьому вдалося ефективно протестувати розроблену модель у середовищах з різним рівнем невизначеності.

Для створення практичної експериментальної бази використовувався мобільний робот Amigobot у поєднанні з глибинним сенсором Kinect – рішення, яке активно використовується в наукових лабораторіях і водночас є відносно недорогим для реалізації. Така комбінація дозволяє створити гнучку та адаптивну систему навігації з можливістю впровадження в реальні апаратні рішення.

Результати симуляцій підтвердили ефективність поєднання реактивного та деліберативного підходів у межах єдиної нейро-нечіткої моделі. Рух робота в різних сценаріях був стабільним і безпечним, із динамічним переходом між локальними реакціями та глобальним плануванням маршруту відповідно до ситуації в навколишньому середовищі.

Таким чином, запропоновані рішення можуть слугувати основою для побудови інтелектуальних систем автономного руху в мобільній робототехніці, особливо в умовах обмежених сенсорних даних або часткової невизначеності карти.

ВИСНОВКИ

1. У результаті виконання кваліфікаційної роботи було розроблено, реалізовано та експериментально перевірено програмний модуль управління рухом мобільного робота, який поєднує реактивну та деліберативну навігацію з використанням нейро-нечіткої логіки. Такий підхід дозволив об'єднати переваги обох стратегій управління – швидку реакцію на динамічні перешкоди та здатність планувати оптимальні маршрути в складних і частково невідомих середовищах.

2. У рамках роботи отримано наступні результати:

– проведено огляд сучасних методів навігації мобільних роботів, включно з традиційними алгоритмами та підходами на основі штучного інтелекту;

– проаналізовано відомі рішення, які застосовують нечітку логіку, штучні нейронні мережі та комбіновані системи;

– сформульовано постановку задачі та обґрунтовано вибір архітектури навігаційної системи;

– реалізовано архітектуру програмного модуля в середовищі ROS з використанням MoveIt!, симулятора V-REP, сенсора Kinect та алгоритмів потенційних полів;

– розроблено псевдокод і повнофункціональну реалізацію модулів реактивної, деліберативної навігації та нейро-нечіткого керування;

– проведено серію експериментів у симульованому середовищі з різними рівнями невизначеності карти;

– здійснено оцінку ефективності системи в порівнянні зі стандартними підходами навігації.

3. Результати експериментів засвідчили, що запропонований підхід демонструє вищу адаптивність та ефективність у складних сценаріях навігації. Особливо це проявляється у випадках, коли інформація про середовище є частково неточною або змінюється в реальному часі. Згідно з отриманими результатами, рівень успішного досягнення цілі у таких умовах зростав до 96%,

що свідчить про ефективну роботу комбінованої нейро-нечіткої системи.

4. Запропонований програмний модуль може бути використаний як основа для створення інтелектуальних систем автономного управління мобільними роботами. Завдяки модульній архітектурі та підтримці стандартів ROS, система є гнучкою, масштабованою та придатною для інтеграції як у симуляційні середовища, так і у фізичні апаратні рішення.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Wardana I.N.K. Application of laptop-based robot as restaurant waiter. KNS&I. 2015. Vol. Pp. 9–10.

2. Markkandan S., Sivasubramanian S., Mulerikkal J., Shaik N., Jackson B., Narayanan L. Massive MIMO codebook design using Gaussian mixture model based clustering. Intelligent Automation and Soft Computing. 2022. Vol. 32(1). Pp. 361–375.

3. Dewi T., Risma P., Oktarina Y., Nawawi M. Neural network simulation for obstacle avoidance and wall follower robot as a helping tool for teaching-learning process in classroom. Proceedings of the International Conference on Engineering and Applied Technology (ICEAT). 2017. Pp. 705–717.

4. Dezfoulian S.H., Wu D., Ahmad I.S. A generalized neural network approach to mobile robot navigation and obstacle avoidance. Intelligent Autonomous Systems. 2012. Vol. 12. Pp. 25–42.

5. Engedy I., Horvath G. Artificial neural network-based mobile robot navigation. Proceedings of the IEEE International Symposium on Intelligent Signal Processing. 2009. Pp. 241–246.

6. Mohanty P.K., Parhi D.R. A new intelligent motion planning for mobile robot navigation using multiple adaptive neuro-fuzzy inference system. Natural Science. 2014. Vol. 8(5). Pp. 2527–2535.

7. Savage J., Muñoz S., Matamoros M., Osorio R. Obstacle avoidance behaviors for mobile robots using genetic algorithms and recurrent neural networks. IFAC Proceedings Volumes. 2013. Vol. 46(24). Pp. 141–146.

8. Karur K., Sharma N., Dharmatti C., Siegel J. A survey of path planning algorithms for mobile robots. Vehicles. 2021. Vol. 3(3). Pp. 448–468.

9. Tang Y., Zakaria M., Younas M. Path planning trends for autonomous mobile robot navigation: A review. Sensors. 2025. Vol. 25. Article 1206.

10. Bouraine S., Fraichard T., Salhi H. Provably safe navigation for mobile robots with limited field-of-views in dynamic environments. Autonomous Robots. 2011. Vol. 32. Pp. 267–283.

11. Sánchez Ibáñez J.R., Perez-del-Pulgar C., Garcia A. Path planning for autonomous mobile robots: A review. Sensors. 2021. Vol. 21. Article 7898.

12. Li L., Li X., Shi Z., Chang G. Research on SLAM and path planning based on ROS robot. CDSR Proceedings. 2020. Article 109.

13. Milford M., Wyeth G. Hybrid robot control and SLAM for persistent navigation and mapping. Robotics and Autonomous Systems. 2010. Vol. 58(9). Pp. 1096–1104.

14. Koval V., Zahorodnia D., Adamiv O. An image segmentation method for obstacle detection in a mobile robot environment. Proceedings of the International Conference on Advanced Computer Information Technologies (ACIT). 2019. Pp. 475–478.

15. Adamiv O., Koval V., Kapura V., Dorosh V., Sapozhnyk G. Mobile robot navigation method for environment with dynamical obstacles. Proceedings of the 5th IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS). 2009. Pp. 515–518.

16. Adamiv O., Sachenko A., Kapura V. Gradient method for autonomous robot navigation. Proceedings of the International Conference TCSET. 2008. Pp. 640–642.

17. Larasati N., Dewi T., Oktarina Y. Object following design for a mobile robot using neural network. Computer Engineering and Applications Journal. 2017. Vol. 6(1). Pp. 5–14.

18. Ren L., Wang W., Du Z. A new fuzzy intelligent obstacle avoidance control strategy for wheeled mobile robot. Proceedings of the IEEE International Conference on Mechatronics and Automation (ICMA). 2012. Pp. 1732–1737.

19. Yousfi N., Rekik C., Jallouli M., et al. Optimized fuzzy controller for mobile robot navigation in a cluttered environment. Proceedings of the IEEE 7th International Multi-Conference on Systems, Signals and Devices. 2010. Pp. 1–7.

20. Qing-Yong B., Shun-ming L., Wei-yan S., et al. A fuzzy behavior–based architecture for mobile robot navigation in unknown environments. Proceedings of the IEEE International Conference on Artificial Intelligence and Computational Intelligence. 2009. Pp. 257–261.

21. Boubertakh H., Tadjine M., Glorennec P., et al. A simple goal seeking navigation method for a mobile robot using human sense, fuzzy logic and reinforcement learning. Journal of Automation and Control. 2008. Vol. 18(1). Pp. 23–27.

22. Muthu T., Gloude R.T., Swaminathan S., et al. Fuzzy logic controller for autonomous navigation. Proceedings of the IEEE International Conference on Communications and Signal Processing (ICCSP). 2012. Pp. 81–92.

23. Zou A.M., Hou Z.G., Fu S.Y. Neural networks for mobile robot navigation: A survey. Advances in Neural Networks. 2006. Pp. 1218–1226.

24. Xiao H., Liao L., Zhou F. Mobile robot path planning based on Q–ANN. Proceedings of the IEEE International Conference on Automation and Logistics. 2007. Pp. 2650–2654.

25. Rai N., Rai B. Neural network based closed loop speed control of DC motor using Arduino Uno. International Journal of Engineering Trends and Technology. 2013. Vol. 4(2). Pp. 137–140.

26. Iwendi C., Ponnan S., Munirathinam R., Srinivasan K., Chang C.Y. An efficient and unique TF/IDF algorithmic model-based data analysis for handling applications with big data streaming. Electronics. 2019. Vol. 8. Article 1331.

27. Yang S.X., Meng M. Neural network approaches to dynamic collision–free trajectory generation. IEEE Transactions on Systems, Man and Cybernetics: Applications and Reviews. 2001. Vol. 31(3). Pp. 302–318.

28. Enireddy C.V., Karthikeyan D., Babu V. One hot encoding and LSTM based deep learning models for protein secondary structure prediction. Soft Computing. 2022. Vol. 26(4). https://link.springer.com/article/10.1007/s00500-022-06783-9.

29. Motlagh O., Nakhaeinia D., Tang S.H., et al. Automatic navigation of mobile robots in unknown environments. Neural Computing and Applications. 2014. Vol. 24(7). Pp. 1569–1581.

30. Gavrilov A.V., Lee S. An architecture of hybrid neural network-based navigation system for mobile robot. Proceedings of the IEEE 7th International Conference on Intelligent Systems Design and Applications. 2007. Pp. 587–590.

31. Singh M.K., Parhi D.R. Path optimisation of a mobile robot using an artificial neural network controller. International Journal of Systems Science. 2011. Vol. 42(1). Pp. 107–120.

32. Al Mutib K., Mattar E. Neuro-fuzzy controlled autonomous mobile robotics system. Proceedings of the IEEE 13th International Conference on Modelling and Simulation. 2011. Pp. 1–7.

33. Leonid T.T., Jayaparvathy R. Classification of elephant sounds using parallel convolutional neural network. Intelligent Automation and Soft Computing. 2022. Vol. 32(3). Pp. 1415–1426.

34. Будаковський-Капанюк А.Р. Застосування нечіткої логіки для управління рухом мобільних роботів. Інтелектуальні інформаційні технології в прикладних дослідженнях : тези доп. студентської наук.-практ. конф. (м. Тернопіль, 27–29 травня 2025 р.). Тернопіль, 2025. С. 24-27.

35. Островерхов В. М., Біловус Л. І., Возьний К. З., Луцишин О. О., Монастирський Г. Л., Надвиничний С. А., Питель С. В., Шандрук С. К. Загальні методичні рекомендації з підготовки, оформлення, захисту та оцінювання кваліфікаційних робіт здобувачів вищої освіти першого (бакалаврського) і другого (магістерського) рівнів. Тернопіль: ЗУНУ, 2024. 83 с.

36. Комар М.П., Саченко А.О., Васильків Н.М., Гладій Г.М., Коваль В.С., Ліп’яніна-Гончаренко Х.В. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Штучний інтелект» спеціальності 122 «Комп’ютерні науки» за першим (бакалаврським) рівнем вищої освіти. Тернопіль: ЗУНУ, 2024. 57 с.

Додаток А

Програмний код

```
# --- Файл: reactive_navigation.py ---
import rospy
from sensor_msgs.msg import LaserScan
from geometry_msgs.msg import Twist
import numpy as np

class ReactiveNavigator:
    def __init__(self):
        self.cmd_pub = rospy.Publisher("/cmd_vel", Twist, queue_size=10)
        rospy.Subscriber("/scan", LaserScan, self.scan_callback)
        self.twist = Twist()

    def scan_callback(self, data):
        ranges = np.array(data.ranges)
        front = np.min(ranges[0:10].tolist() + ranges[-10:].tolist())
        left = np.mean(ranges[30:60])
        right = np.mean(ranges[300:330])

        force_x = 1.0
        force_y = (right - left)

        self.twist.linear.x = 0.2 if front > 0.5 else 0.0
        self.twist.angular.z = -force_y * 0.5

        self.cmd_pub.publish(self.twist)

if __name__ == '__main__':
    rospy.init_node('reactive_navigation')
    nav = ReactiveNavigator()
```

```
    rospy.spin()

# --- Файл: deliberative_navigation.py ---
import rospy
import moveit_commander
from geometry_msgs.msg import PoseStamped

class DeliberativePlanner:
    def __init__(self):
        moveit_commander.roscpp_initialize([])
        rospy.init_node('deliberative_navigation')
        self.group = moveit_commander.MoveGroupCommander("base")
        self.goal_pub = rospy.Publisher('/move_base_simple/goal', PoseStamped, queue_size=10)

    def send_goal(self, x, y):
        goal = PoseStamped()
        goal.header.frame_id = "map"
        goal.header.stamp = rospy.Time.now()
        goal.pose.position.x = x
        goal.pose.position.y = y
        goal.pose.orientation.w = 1.0
        self.goal_pub.publish(goal)

if __name__ == '__main__':
    planner = DeliberativePlanner()
    planner.send_goal(2.0, 1.0)

# --- Файл: decision_fuzzy_controller.py ---
import rospy
from std_msgs.msg import Float32
from geometry_msgs.msg import Twist
import skfuzzy as fuzz
import numpy as np
```

```
class FuzzyDecisionMaker:
    def __init__(self):
        rospy.init_node('fuzzy_decision_node')
        self.cmd_pub = rospy.Publisher("/cmd_vel", Twist, queue_size=10)
        rospy.Subscriber("/safety_level", Float32, self.safety_callback)
        rospy.Subscriber("/plan_validity", Float32, self.plan_callback)
        self.safety = 0.0
        self.validity = 0.0

    def safety_callback(self, msg):
        self.safety = msg.data
        self.make_decision()

    def plan_callback(self, msg):
        self.validity = msg.data
        self.make_decision()

    def make_decision(self):
        deliberative = fuzz.trimf(np.arange(0, 1.1, 0.1), [0.6, 0.8, 1.0])
        reactive = fuzz.trimf(np.arange(0, 1.1, 0.1), [0.0, 0.2, 0.4])
        decision_value = min(self.safety, self.validity)
        output = fuzz.interp_membership(np.arange(0, 1.1, 0.1), deliberative, decision_value)

        twist = Twist()
        if output > 0.6:
            rospy.loginfo("Deliberative mode")
            twist.linear.x = 0.3
        else:
            rospy.loginfo("Reactive mode")
            twist.linear.x = 0.1
            twist.angular.z = 0.5
```

```
        self.cmd_pub.publish(twist)


if __name__ == '__main__':
    controller = FuzzyDecisionMaker()
    rospy.spin()


# --- Файл: launch/navigation.launch ---
# <launch>
#     <node pkg="navigation_module" type="reactive_navigation.py" name="reactive_nav" output="screen"/>
#     <node pkg="navigation_module" type="deliberative_navigation.py" name="deliberative_nav" output="screen"/>
#     <node pkg="navigation_module" type="decision_fuzzy_controller.py" name="fuzzy_controller" output="screen"/>
# </launch>
```

Додаток Б

Копія публікації

Західноукраїнський національний університет

Факультет комп'ютерних інформаційних технологій

Кафедра інформаційно-обчислювальних систем і управління

ЗБІРНИК ТЕЗ ДОПОВІДЕЙ

Студентської науково-практичної конференції

ІНТЕЛЕКТУАЛЬНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ В ПРИКЛАДНИХ ДОСЛІДЖЕННЯХ

(IITAR-2025)

27-29 травня 2025 року

Тернопіль

2025

Збірник тез доповідей студентської науково-практичної конференції «Інтелектуальні інформаційні технології в прикладних дослідженнях» (IITAR-2025). (Тернопіль, 27-29 травня 2025 року). Тернопіль: ЗУНУ, 2025. 372 с.

До збірника увійшли тези доповідей учасників студентської науково-практичної конференції "Інтелектуальні інформаційні технології в прикладних дослідженнях" (IITAR – 2025), що відбувалась у рамках AI Week на базі кафедри інформаційно-обчислювальних систем і управління Західноукраїнського національного університету. Мета конференції — об'єднати науковців, освітян, представників IT-бізнесу та здобувачів освіти для обміну досвідом, презентації сучасних досліджень і впровадження інноваційних рішень у сфері комп'ютерних наук, штучного інтелекту та суміжних галузей.

За зміст наукових праць та достовірність наведених фактологічних і статистичних матеріалів відповідальність несуть автори публікацій та їхні наукові керівники. У збірнику зберігається стилістика та орфографія авторів матеріалів.

Склад організаційного комітету конференції

Керівництво оргкомітету

Мирослав Комар, д.т.н., професор, професор кафедри інформаційно-обчислювальних систем і управління

Христина Ліп'яніна-Гончаренко, д.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління

Надія Васильків, к.т.н., в.о. завідувача кафедри інформаційно-обчислювальних систем і управління

Члени програмного комітету

Василь Коваль, к.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління, заступник декана факультету комп'ютерних інформаційних технологій

Олександр Осолінський, к.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління

Павло Биковий, к.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління

Діана Загородня, к.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління

Ігор Майків, к.т.н., доцент кафедри інформаційно-обчислювальних систем і управління

Члени оргкомітету

Андрій Івасечко, викладач кафедри інформаційно-обчислювальних систем і управління

Дмитро Дюг, викладач кафедри інформаційно-обчислювальних систем і управління

Христина Юрків, студентка ФКІТ ЗУНУ, технік лабораторії з проблем інформаційних технологій

Мар'яна Соя, студентка ФКІТ ЗУНУ, лаборант кафедри інформаційно-обчислювальних систем і управління

Микола Телька, студент ФКІТ ЗУНУ, лаборант кафедри інформаційно-обчислювальних систем і управління, студентський декан факультету комп'ютерних інформаційних технологій

Зміст

Будаковський-Капанюк Артем
студент групи КНШІ-41
a.r.budakovskyi.kapaniuk@gmail.com
Західноукраїнський національний університет
Тернопіль, Україна

ЗАСТОСУВАННЯ НЕЧІТКОЇ ЛОГІКИ ДЛЯ УПРАВЛІННЯ РУХОМ МОБІЛЬНИХ РОБОТІВ

У системах автономної навігації все ширше застосовуються методи штучного інтелекту, серед яких особливе місце займають нечіткі логічні системи [1] та нейронні мережі [2]. Нечітка логіка дозволяє моделювати процес прийняття рішень у невизначених умовах, наближаючись до людського способу мислення, тоді як нейронні мережі забезпечують здатність до навчання на основі сенсорних даних.

Використання нечітких систем керування для мобільної навігації дозволяє об'єднати переваги реактивного та деліберативного управління. Реактивна навігація краще підходить для динамічних середовищ, де рішення приймаються на основі актуальних сенсорних даних, тоді як деліберативна навігація орієнтована на побудову оптимального шляху за наявності часткової або повної інформації про середовище [3]. Дослідження показують, що поєднання нейронних мереж та нечіткої логіки у рамках нейро-нечітких систем дозволяє істотно підвищити ефективність управління рухом мобільних роботів у складних умовах.

Проєктування архітектури програмного модуля для автономного управління рухом мобільного робота базується на принципах модульності, гнучкості та адаптивності. Основу архітектури становить багаторівнева система, яка інтегрує деліберативне та реактивне управління із нейро-нечітким контролером для забезпечення адаптації до змін у середовищі. Взаємодія між компонентами модуля реалізована у середовищі ROS (Robot Operating System), що дозволяє здійснювати паралельну обробку даних, обмін повідомленнями між вузлами та контроль за роботою кожного модуля в реальному часі.

Загалом архітектура складається з трьох логічних рівнів (рисунок 1): сенсорного рівня, рівня прийняття рішень та виконавчого рівня. Кожен рівень виконує визначені функції, а взаємозв'язки між ними забезпечують замкнутий цикл управління: від отримання даних про середовище до формування команд для виконавчих механізмів робота.

На сенсорному рівні встановлені пристрої для збору даних про навколишнє середовище – зокрема, глибинний сенсор Kinect, який генерує потоки інформації у форматі RGB-D (зображення та глибина). Ці дані надходять до вузла формування карти, де відбувається реконструкція 2D-площини з відображенням перешкод і вільних зон. Зібрана інформація також використовується для оновлення поточного

положення робота на карті (локалізація) та фіксації змін у середовищі.

Рисунок 1 – Архітектура модуля для автономного управління рухом мобільного робота

Наступним є рівень прийняття рішень, який містить декілька основних модулів:

1. Модуль деліберативної навігації, який будує глобальний маршрут від поточного положення до цілі, використовуючи алгоритми семплінгу. Найчастіше використовуються варіанти алгоритмів типу RRT (Rapidly-exploring Random Trees). Побудова маршруту базується на множині станів $S \subseteq \mathbb{R}^n$, між якими будується граф $G = (V, E)$, де V – множина вершин (допустимих положень), E – ребра, що з'єднують ці вершини.

2. Модуль реактивної навігації, що формує локальні траєкторії обходу перешкод, використовуючи метод потенційних полів. Реактивний рух ґрунтується на обчисленні результуючого векторного поля $\vec{F}(x, y)$, що включає притягуюче поле до цілі $\vec{F}_{attr}$ і відштовхуюче поле від перешкод $\vec{F}_{rep}$:

$$\vec{F}(x, y) = \vec{F}_{attr}(x, y) + \sum_{i=1}^{n} \vec{F}_{rep}^{(i)}(x, y) \quad (1)$$

Притягуюче поле зазвичай визначається як:

$$\vec{F}_{attr} = -k_{att} \cdot \nabla\left(\frac{1}{2} \cdot d^2(x, y)\right), \quad (2)$$

де $d(x, y)$ – відстань до цілі;

k_{att} – коефіцієнт притягання.

Відштовхуюче поле моделюється за допомогою гаусового ядра для кожної перешкоди:

$$V_{rep}(x,y) = A \cdot \exp\left(-\left(\frac{(x-x_0)^2}{2\sigma_x^2} + \frac{(y-y_0)^2}{2\sigma_y^2}\right)\right), \quad (3)$$

де A – максимальна інтенсивність поля;

σ_x, σ_y– параметри ширини по осях;

(x_0, y_0) – центр перешкоди.

3. Нейро-нечітка система прийняття рішень, яка здійснює узгодження між деліберативним і реактивним режимами на основі двох параметрів.

Припустимо, що маємо два вхідні параметри:

$p \in [0,1]$ – ступінь валідності маршруту;

$s \in [0,1]$ – рівень безпеки руху.

На виході формується коефіцієнт ухваленого рішення $f \in [0,1]$, за допомогою якого обчислюється результуюча швидкість:

$$V = f \cdot V_{delib} + (1-f) \cdot V_{react} \quad (4)$$

де V_{delib} – швидкість за глобальним планом;

V_{react} – швидкість за реактивною моделлю.

Кінцевим є виконавчий рівень, який містить модуль керування приводами. На основі обчислених швидкісних векторів формується керуючий сигнал $u(t)$, що подається на актуатори. Керування здійснюється за допомогою регулятора, який задається рівнянням:

$$u(t) = K_p \cdot e(t) + K_i \int_0^t e(\tau)\, d\tau \quad (5)$$

де $e(t)$ – поточна похибка між бажаним і фактичним положенням;

K_p, K_i – коефіцієнти регулювання.

Усі компоненти модуля функціонують в єдиному інформаційному середовищі, що дозволяє організувати асинхронну обробку даних, забезпечити паралельну роботу модулів і гнучко масштабувати систему при додаванні нових функцій або сенсорів.

Таким чином, архітектура програмного модуля управління забезпечує високу ступінь автономності мобільного робота, дозволяє ефективно поєднувати глобальне планування з локальною адаптацією та створює основу для реалізації поведінкових стратегій з використанням штучного інтелекту. Вона також забезпечує модульність, прозорість внутрішніх процесів і можливість тестування в симуляційному середовищі V-REP перед переходом до реального апаратного прототипу.

Список використаних джерел

1. Dewi T., Risma P., Oktarina Y., Nawawi M. Neural network simulation for obstacle avoidance and wall follower robot as a helping tool for teaching-learning process in classroom. *Proceedings of the International Conference on Engineering and Applied Technology (ICEAT)*. 2017. Pp. 705–717.

2. Mohanty P.K., Parhi D.R. A new intelligent motion planning for mobile robot navigation using multiple adaptive neuro-fuzzy inference system. *Natural Science*. 2014. Vol. 8(5). Pp. 2527–2535.
3. Savage J., Muñoz S., Matamoros M., Osorio R. Obstacle avoidance behaviors for mobile robots using genetic algorithms and recurrent neural networks. *IFAC Proceedings Volumes*. 2013. Vol. 46(24). Pp. 141–146.