Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризированих технологій
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

другий (магістерський)
(рівень вищої освіти)

РОЗРОБЛЕННЯ ПРОГРАМНОГО МОДУЛЯ ПОБУДОВИ ОПТИМАЛЬНОГО МАРШРУТУ МОБІЛЬНОГО РОБОТА У 2D-ПРОСТОРІ
(тема)

Виконав студент 2 курсу, групи КТРСм-22-2
Борисов Георгій Олександрович
(прізвище, ім'я, по батькові)

Спеціальності 151 Автоматизація та комп'ютерно-інтегровані технології
(код і повна назва напряму)

Тип програми Освітньо-професійна
Освітня програма Комп'ютеризовані та робототехнічні системи
(назва)

Керівник доц. каф. КІТАР Максимова С.С.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри КІТАР _____ Невлюдов І. Ш.
(підпис) (прізвище, ініціали)

2024

Я, як студент ХНУРЕ, розумію і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

20.01.2024  Борисов Г.О.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет АКТ
Кафедра КІТАР
Рівень вищої освіти другий (магістерський)
Спеціальність 151 Автоматизація та комп’ютерно-інтегровані технології
Тип програми Освітньо-професійна
Освітня програма Комп'ютеризовані та робототехнічні системи
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав.кафедри _____
(підпис)
«__»_____2024р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Борисову Георгію Олександровичу
(шифр і назва)

1. Тема роботи: Розроблення програмного модуля побудови оптимального маршруту мобільного робота у 2D-просторі
Затверджена наказом університету від №1288Ст від 03.11.2023
2. Термін подання студентом роботи до екзаменаційної комісії 24.01.2024р.
3. Вихідні дані до роботи: 3.1 Мова програмування Python; 3.2 Побудова шляху у двовимірному просторі; 3.3 Розмір карти для експериментальних досліджень 512×512 пікселів
4. Перелік питань, що потрібно опрацювати в роботі: 4.1 Вступ; 4.2 Аналіз сучасних методів та алгоритмів побудови маршруту; 4.3 Аналіз особливостей побудови маршруту у двомірному просторі; 4.4 Аналіз алгоритму обрізання Greedy Algorithm; 4.5 Математичний опис навколишнього середовища та мобільного робота; 4.6 Удосконалення методу побудови маршруту мобільного робота у двомірних координатах 4.7 Аналіз та вибір алгоритму побудови шляху; 4.8 Розробка алгоритму обрізання шляхів на базі Greedy Algorithm; 4.9 Розробка моделі згладжування шляху; 4.10 Вибір середовища розробки; 4.11 Розробка алгоритму обрізання шляхів; 4.12 Розробка алгоритму згладжування шляхів; 4.13 Розробка загального алгоритму роботи програми; 4.14 Реалізація коду; 4.15 Проведення досліджень та аналіз отриманих результатів; 4.16 Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій Графічний демонстраційний матеріал в форматі PowerPoint(*.ppt) формату А4 –12 сторінок.

6. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по-батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз методів пошуку та побудови маршруту для мобільного робота	1.09.2023-18.09.2023	виконано
2	Математичний опис навколишнього середовища та мобільного робота	19.09.2023-29.09.2023	виконано
3	Удосконалення методу побудови маршруту мобільного робота у двомірних координатах	30.09.2023-9.10.2023	виконано
4	Аналіз та вибір алгоритму побудови шляху	10.10.2023-15.10.2023	виконано
5	Розробка алгоритму обрізання шляхів на базі Greedy Algorithm	16.10.2023-24.10.2023	виконано
6	Розробка моделі згладжування шляху	25.10.2023-2.11.2023	виконано
7	Вибір середовища розробки	3.11.2023-4.11.2023	виконано
8	Розробка алгоритму обрізання шляхів	5.11.2023-7.11.2023	виконано
9	Розробка алгоритму згладжування шляхів	8.11.2023-11.11.2023	виконано
10	Розробка загального алгоритму роботи програми	12.11.2023-16.11.2023	виконано
11	Реалізація програмної частини	17.11.2023-28.11.2023	виконано
12	Проведення досліджень та аналіз отриманіх результатів	29.11.2023-18.12.2023	виконано

Дата видачі завдання 1 вересня 2023р.

Студент _____ Борисов Г.О.
(підпис) (прізвище, ініціали)

Керівник роботи Максимова С.С.
(підпис) (прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 81 с., 2 табл., 11 рис., 2 дод., 16 джерел.

АВТОНОМНА ТЕХНІКА, МОБІЛЬНІ РОБОТИ, ПРОГРАМНИЙ МОДУЛЬ, ОПТИМАЛЬНИЙ МАРШРУТ, НАВІГАЦІЯ, ЖАДІБНИЙ АЛГОРИТМ, РОЗРОБКА, СТРАТЕГІЯ РУХУ, АВТОМАТИЗАЦІЯ, ГЛОБАЛЬНА ОПТИМІЗАЦІЯ.

Мета дослідження – підвищення ефективності виробництва за рахунок розробки програмного забезпечення для навігації мобільного робота.

Об’єкт дослідження – процес побудови оптимального маршруту мобільного робота.

Предмет дослідження – алгоритм планування шляху та програма побудови маршруту.

В даній кваліфікаційній роботі проведено аналіз сучасних методів та алгоритмів побудови маршруту. Проаналізовано особливості побудови маршруту у двомірному просторі та алгоритм обрізання Greedy Algorithm. Наступним етапом було проведено математичний опис навколишнього середовища та мобільного робота. Удосконалено метод побудови маршруту мобільного робота у двомірних координатах. Далі було проведено аналіз та вибір алгоритму побудови шляху. Після проведеного аналізу було розроблено алгоритм обрізання шляхів на базі Greedy Algorithm та моделі згладжування шляху. Наступним етапом був вибір середовища розробки. Після було розроблено алгоритм обрізання шляхів, алгоритм згладжування шляхів та загальний алгоритм роботи програми. Реалізовано код програми та проведено експериментальні дослідження, проаналізовано отримані результати.

ABSTRACT

Explanatory note: 81 pages, 2 tables, 11 figures, 2 app, 16 sources.

AUTONOMOUS TECHNOLOGY, MOBILE ROBOTS, SOFTWARE MODULE, OPTIMAL ROUTE, NAVIGATION, GREEDY ALGORITHM, DEVELOPMENT, MOTION STRATEGY, AUTOMATION, GLOBAL OPTIMIZATION.

The purpose of the research is to improve production efficiency through the development of software for navigating mobile work.

The object of research is the process of building the optimal route of mobile work.

The subject of the research is the implementation of the path planning algorithm and the route construction program.

In this qualification work, an analysis of modern methods and algorithms of route construction was carried out. The features of route construction in two-dimensional space and the Greedy Algorithm pruning algorithm are analyzed. The next stage was a mathematical description of the environment and the mobile robot. The method of constructing a route of a mobile robot in two-dimensional coordinates has been improved. Next, the analysis and selection of the path construction algorithm was carried out. After the analysis, a path cutting algorithm was developed based on the Greedy Algorithm and the path smoothing model. The next stage was the selection of the development environment. After that, the path cutting algorithm, the path smoothing algorithm and the general algorithm of the program were developed. The program code was implemented, experimental studies were conducted, and the obtained results were analyzed.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

BRRT – bidirectional rapidly exploring random tree;

IDE – середовище розробки;

RT – rapidly exploring random tree.

ВСТУП

В умовах стрімкого розвитку автономної техніки та розширення сфер використання мобільних роботів, розробка програмного модуля для побудови оптимального маршруту у 2D-просторі стає ключовим етапом у вдосконаленні їхньої ефективності та пристосованості до різноманітних завдань.

Побудова оптимального маршруту є критичною задачею для ефективного функціонування мобільних роботів. Це стає актуальним в умовах росту автоматизації та впровадження роботизованих систем у різноманітних виробничих і комерційних середовищах.

Використання оптимальних маршрутів дозволяє ефективно використовувати ресурси мобільного робота, зменшує час переміщення та підвищує загальну продуктивність. У виробництві це може означати швидше переміщення виробничих роботів між робочими точками, у логістиці – оптимізацію доставки товарів, а в медицині – швидше доставлення медичних засобів чи допомоги.

З урахуванням швидкого розвитку технологій та впровадження штучного інтелекту, розробка програмного модуля для побудови оптимального маршруту отримує новий стимул. Сучасні методи та алгоритми можуть враховувати різноманітні умови, включаючи динамічні перешкоди та змінні параметри середовища.

Такий програмний модуль необхідний для створення гнучких та адаптивних систем навігації, які здатні пристосовуватися до змінних умов та ефективно виконувати завдання в реальному часі.

Мета дослідження – підвищення ефективності виробництва за рахунок розробки програмного забезпечення для навігації мобільної роботи.

Об’єкт дослідження – процес побудови оптимального маршруту мобільної роботи.

Предмет дослідження – реалізація алгоритму планування шляху та програми побудови маршруту.Для досягнення поставленої мети необхідно вирішити такі завдання:

- проаналізувати сучасні методи та алгоритми побудови маршруту;
- проаналізувати особливості побудови маршруту у двомірному просторі;
- проаналізувати алгоритм обрізання Greedy Algorithm;
- провести математичний опис навколишнього середовища та мобільного робота;
- провести удосконалення методу побудови маршруту мобільного робота у двомірних координатах;
- провести аналіз та вибір алгоритму побудови шляху;
- розробити алгоритм обрізання шляхів на базі Greedy Algorithm;
- розробити модель згладжування шляху;
- проаналізувати та вибрати середовище розробки;
- розробити алгоритму обрізання шляхів;
- розробити алгоритму згладжування шляхів;
- розробити загального алгоритму роботи програми;
- реалізувати програмний код;
- провести експериментальні дослідження та проаналізувати отримані результати.

Кваліфікаційна робота виконана згідно ДСТУ 3008 – 15 [1] та керуючись навчальним посібником з дипломного проекту [2] та методичними вказівками [3]. Результати кваліфікаційної роботи описані в статті, яку можна побачити в гугл академії [4].

1 АНАЛІЗ МЕТОДІВ ПОШУКУ ТА ПОБУДОВИ МАРШРУТУ ДЛЯ МОБІЛЬНОГО РОБОТА

1.1 Аналіз сучасних методів та алгоритмів побудови маршруту

У рамках аналізу сучасних методів та алгоритмів побудови маршруту в двовимірному просторі, важливо розглянути різноманіття підходів, що застосовуються для вирішення цієї задачі. Сучасні технології дозволяють розробляти алгоритми, які забезпечують оптимальну та ефективну навігацію мобільних роботів [5].

Алгоритм A* є одним із найпоширеніших та ефективних алгоритмів для пошуку найкоротшого шляху в графах, що використовується для побудови оптимального маршруту у двовимірному просторі. Цей алгоритм поєднує в собі особливості алгоритмів пошуку в ширину і пошуку за кращим першим (Best-First Search) та використовує евристику для пришвидшення процесу пошуку.

Основні кроки алгоритму A* виглядають наступним чином:

– ініціалізація: визначення точки старту та точки фінішу. Створення початкового вузла та ініціалізація його параметрів, таких як вартість (загальна вартість шляху від початку до поточного вузла), оцінка вартості до фінішу (евристична оцінка від поточного вузла до фінішу), та сумарна оцінка (сума вартості та евристичної оцінки);

– створення відкритого та закритого списків. Відкритий список містить вузли, які ще потрібно розглянути. Закритий список містить вузли, які вже були розглянуті;

– головний цикл. До тих пір, поки відкритий список не стане порожнім або не буде досягнуто фінішної точки: обрати вузол з найменшою сумарною оцінкою у відкритому списку, перемістити цей вузол з відкритого списку в

закритий, розглянути всі сусіди поточного вузла. Для кожного сусіда: якщо сусід вже є в закритому списку, проігнорувати його, якщо сусід не знаходиться в відкритому списку, додати його туди та встановити йому батьківський вузол (поточний вузол), якщо сусід вже є в відкритому списку, перевірити, чи новий шлях є коротшим, якщо так, оновити вартість та батьківський вузол;

– відновлення шляху: по закінченні алгоритму, якщо ми досягли фінішної точки, можна відновити оптимальний шлях, переходячи від фінішного вузла до початкового, використовуючи батьківські вузли.

Основною перевагою алгоритму A* є його здатність до ефективного знаходження оптимального маршруту завдяки використанню евристичної функції, яка сприяє пришвидшенню процесу пошуку. Алгоритм A* широко використовується у робототехніці, в іграх та в інших областях, де важлива оптимізація пошуку шляху [6].

Наступним є алгоритм Вейвфронт – це простий та ефективний метод для побудови маршруту у двовимірному просторі. В основі його роботи лежить ідея "хвиль", яка розповсюджується від точки старту до точки фінішу, маркуючи кожен вузол числом, яке представляє відстань від початку:

– простота та ефективність: алгоритм Вейвфронт відзначається своєю простотою та легкістю в імплементації, що робить його привабливим в багатьох випадках;

– безпека від змін: він може легко адаптуватися до змін у середовищі або динамічних умов, оскільки переоцінює вартості в залежності від нових обставин;

– можливість врахування обмежень: алгоритм може легко враховувати обмеження, такі як статичні чи динамічні перешкоди, шляхи з обмеженим доступом, інші обставини;

– працездатність в широких просторах: алгоритм Вейвфронт добре працює в широких відкритих просторах, де важливо уникнути статичних або динамічних перешкод.

Хоча алгоритм Вейвфронт може бути менш ефективним у вузьких та більш складних середовищах порівняно з деякими іншими методами, він залишається популярним та ефективним рішенням для багатьох задач побудови маршруту [7].

Алгоритм потенційних полів – це метод побудови маршруту для мобільних роботів, який базується на ідеї використання потенційних полів для керування рухом. У цьому методі кожному об'єкту у середовищі (включаючи перешкоди та цільові точки) присвоюється потенціал, який визначає взаємодію з мобільним роботом.

Основна ідея полягає в тому, що робот рухається від зон високого потенціалу (наприклад, перешкод або точок, які слід уникнути) до зон низького потенціалу (цільової точки). Потенціали можуть бути визначені як функції відстані від об'єктів, де велика відстань відповідає низькому потенціалу, і, навпаки, мала відстань – високому.

Взаємодія між роботом та об'єктами моделюється силовими полями. Робот відчуває сили, спрямовані від кожного об'єкта, і рухається в напрямку найменшого опору, враховуючи ці сили. Це дозволяє роботу ефективно обходити перешкоди та досягати цільових точок.

Однією з переваг цього методу є його простота та ефективність в реальному часі. Він дозволяє уникати перешкод, адаптуючись до змін в середовищі, і може використовуватися в різних задачах, включаючи навігацію мобільних роботів, в тому числі віддалених керованих апаратах.

Однак алгоритм потенційних полів також може мати обмеження, такі як можливість застрягання у локальних мінімумах, де робот може отримати неправильний вектор напрямку через створене потенціальне поле. Також, у

нього може бути складно управляти великим числом перешкод чи в сложних середовищах [8].

Метод генетичних алгоритмів є еволюційним підходом до розв'язання оптимізаційних завдань, включаючи побудову оптимального маршруту мобільного робота. Цей метод інспірується природним процесом еволюції та генетичної селекції.

Основна ідея полягає в застосуванні принципів еволюції для генерації та вдосконалення популяції можливих маршрутів. Початкова популяція складається з індивідів, кожен з яких представляє собою потенційний маршрут мобільного робота. Кожен індивід оцінюється за якоюсь функцією придатності, яка визначає, наскільки ефективним є його маршрут.

Генетичні алгоритми включають етапи генерації нової популяції на основі поточної та вибору найбільш придатних індивідів для переходу до наступного покоління. Операції, такі як схрещування (комбінування генетичного матеріалу двох індивідів) та мутація (зміна деяких генетичних характеристик), застосовуються для створення нових маршрутів.

Процес повторюється протягом кількох поколінь, під час яких покращені маршрути успадковуються, а менш ефективні заміщуються новими. Генетичні алгоритми мають здатність ефективно обходити просторі оптимізації, шукати рішення в глобальному контексті та адаптуватися до змінних умов [9].

У випадку побудови маршруту для мобільного робота, генетичні алгоритми можуть бути використані для оптимізації маршруту враховуючи різні фактори, такі як час, енергія або безпека. Цей метод є особливо ефективним у випадках, коли простір можливих маршрутів складний та великий, і коли необхідно швидко знаходити глобально оптимальне рішення.

Також, слід відзначити використання машинного навчання та нейронних мереж у сучасних методах побудови маршруту. Методи глибокого навчання можуть навчати модель на основі великої кількості

даних та дозволяють робити прогнози та оптимізувати маршрути в реальному часі.

Аналіз сучасних підходів до побудови маршруту свідчить про значний розмаїття методів, які враховують різні аспекти задачі. Отже, вибір конкретного методу повинен базуватися на вимогах конкретного завдання, характеристиках середовища та обмеженнях мобільного робота [10].

1.2 Особливості побудови маршруту у двомірному просторі

Побудова маршруту у двовимірному просторі включає в себе ряд особливостей та аспектів, що враховуються при розробці алгоритмів та методів навігації для мобільних роботів.

Геометрична модель простору в контексті побудови маршруту у двовимірному середовищі включає в себе визначення точок, ліній, поверхонь та їх взаємозв'язків з метою представлення структури простору та об'єктів в ньому. Ця модель грає важливу роль у розробці алгоритмів навігації для мобільних роботів, оскільки вона надає математичний фреймворк для вирішення задачі знаходження оптимального маршруту.

Координати точок визначають положення об'єктів та перешкод у просторі. Це може бути представлено системою координат, де кожна точка характеризується двома або трьома числами, що визначають її положення у просторі. Лінії та поверхні визначають геометричні області та об'єкти у просторі. Вони можуть представляти стіни, коридори, області забороненого доступу, або будь-які інші особливості, які робот повинен враховувати під час навігації. Геометрична модель також дозволяє визначати відстані, кути та інші геометричні параметри між об'єктами. Це важливо для розрахунків шляху та вибору оптимального напрямку руху.

Алгоритми побудови маршруту, які базуються на геометричній моделі простору, можуть використовувати графи для представлення взаємозв'язків

між точками та лініями, що значно полегшує обчислення та оптимізацію. Геометрична модель простору дозволяє алгоритмам навігації розуміти та взаємодіяти з оточуючим середовищем, визначати оптимальний маршрут і уникати перешкод. Вона є ключовою складовою для успішного функціонування систем навігації мобільних роботів у реальному світі.

Обхідність перешкод є однією з ключових особливостей при побудові маршруту для мобільних роботів у двовимірному просторі. Цей аспект визначає, наскільки ефективно робот може уникати або обходити різноманітні об'єкти та перешкоди, що можуть зустрічатися на його шляху. Обхідність перешкод передбачає не лише здатність виявляти перешкоди, але і розробку стратегій обходу для забезпечення безперешкодного руху. Алгоритми навігації повинні бути здатні ефективно обчислювати нові маршрути, якщо виникають нові перешкоди або якщо попередній маршрут стає недоступним через зміни у середовищі.

Динамічні перешкоди, такі як інші рухомі об'єкти чи люди, представляють виклик, оскільки робот повинен пристосовуватися до їхнього руху та уникати зіткнень. Це вимагає постійного відслідковування оточуючого простору та швидкого реагування на зміни.

При розробці алгоритмів обходження перешкод важливо враховувати різні типи перешкод, їх форму та розміри, а також враховувати умови обмеженого простору. У вузьких коридорах чи серед зібраної архітектури робот може стикнутися з ускладненнями обходження перешкод. Забезпечення ефективності та надійності обхідності перешкод визначає успіх в галузі автономної навігації мобільних роботів у реальних умовах. Адаптивні та гнучкі стратегії обходження є ключовими для досягнення безперешкодної та безпечної навігації в різноманітних середовищах.

Мінімізація функціоналу витрат у контексті побудови маршруту для мобільного робота є стратегічним підходом, спрямованим на оптимізацію різноманітних параметрів, таких як час переміщення, витрати енергії та інші

обчислювальні або фізичні фактори. Цей підхід передбачає врахування не лише самого шляху між точкою А та точкою Б, а й різноманітних умов та обставин, що можуть впливати на загальні витрати робота. Мінімізація функціоналу витрат включає в себе розгляд оптимальних стратегій руху, які дозволяють роботові зменшити енергоспоживання та скоротити час переміщення.

Врахування динамічних умов є важливим аспектом мінімізації витрат, оскільки вони можуть впливати на оптимальний вибір шляху та режим руху. Системи мають бути здатні реагувати на зміни у середовищі та адаптуватися до нових умов для збереження ресурсів. Крім того, урахування конкретних обмежень робота, таких як обмежені ресурси енергії чи обмежені можливості руху, є важливим елементом мінімізації функціоналу витрат. Розробка алгоритмів, які враховують ці обмеження, сприяє ефективному використанню роботом своїх ресурсів. Мінімізація функціоналу витрат ставить перед собою завдання знайти баланс між різними критеріями оптимізації, такими як час, енергія та ефективність руху. Цей підхід сприяє розробці адаптивних та ефективних стратегій навігації для мобільних роботів у реальних умовах.

Врахування динамічних умов у контексті побудови маршруту для мобільного робота – це аспект, що передбачає адаптацію системи навігації до змін у навколишньому середовищі та умовах руху. Цей підхід враховує рухомі об'єкти, зміни в структурі простору та інші фактори, що можуть виникнути під час роботи в реальному світі. Врахування динамічних умов передбачає постійне моніторинг та аналіз оточуючого середовища для виявлення змін. Системи навігації повинні бути здатні реагувати на нові перешкоди, рухомі об'єкти чи зміни у структурі простору, а також враховувати ці зміни при побудові нового маршруту.

Урахування динамічних умов також включає в себе адаптивні стратегії обходження та вибору оптимального напрямку руху. Робот повинен бути

здатний пристосовуватися до різних умов, враховуючи такі параметри, як швидкість руху об'єктів, їхній напрямок та взаємодію з роботом. Врахування динамічних умов також може включати передачу та отримання інформації між різними роботами або системами, яка дозволяє їм координувати свої рухи та уникати конфліктів. Це особливо важливо в сценаріях, де працюють декілька мобільних роботів або агентів. Врахування динамічних умов є важливим аспектом для забезпечення безперешкодної та безпечної навігації мобільних роботів в реальних умовах, де середовище та умови руху можуть змінюватися непередбачувано.

Моделювання та аналіз динамічних ситуацій у контексті побудови маршруту для мобільного робота включає в себе вирішення завдань, пов'язаних із змінами в середовищі та умовах руху, які можуть впливати на навігацію. Цей підхід орієнтований на створення систем, які в змозі адаптуватися до реальних сценаріїв та динамічних обставин. Моделювання динамічних ситуацій включає в себе використання математичних та комп'ютерних моделей для відображення рухомих об'єктів, їхнього поведінки та взаємодії з оточуючим середовищем. Це може включати в себе прогнозування траєкторій інших об'єктів, визначення швидкості та прискорення, а також врахування можливих змін у руховому середовищі.

Аналіз динамічних ситуацій передбачає оцінку можливих взаємодій між рухомим роботом та іншими об'єктами у просторі. Це може включати в себе розпізнавання та прогнозування можливих конфліктних ситуацій, де робот може зіткнутися з іншими об'єктами або перешкодами. Аналіз дозволяє визначити оптимальні рішення та стратегії обходження для забезпечення безперешкодної навігації. Ефективне моделювання та аналіз динамічних ситуацій дозволяє роботам пристосовуватися до непередбачуваних змін у навколишньому середовищі та забезпечує надійні та безпечні стратегії руху. Такий підхід стає ключовим у сценаріях, де роботи

взаємодіють з різноманітними динамічними елементами або в умовах, де середовище може швидко змінюватися.

1.3 Аналіз алгоритму обрізання Greedy Algorithm

Алгоритм обрізання "Greedy" (жадібний алгоритм) є простим та евристичним методом оптимізації, який зазвичай використовується для знаходження локально оптимального рішення проблем з множинами. Цей алгоритм дозволяє приймати на кожному кроці найкраще доступне рішення з точки зору поточного стану системи, надіючись, що це призведе до глобально оптимального результату.

Основна ідея "Greedy Algorithm" полягає в тому, щоб на кожному кроці обирати оптимальний елемент (зазвичай найменший або найбільший) з множини доступних елементів. Після обрання елемента, він додається до часткового рішення, і множина доступних елементів оновлюється, виключаючи той, який вже вибраний.

Процес триває до досягнення критерію завершення або вичерпання множини елементів. Зазвичай "Greedy Algorithm" швидко дає рішення, але не гарантує глобальну оптимальність.

Хоча "Greedy Algorithm" може бути ефективним в багатьох випадках, важливо враховувати, що він може привести до неправильного глобального рішення, оскільки вибір найкращого варіанту на кожному кроці не завжди приводить до найкращого можливого рішення у всьому просторі можливих рішень. Також, "Greedy Algorithm" часто не враховує контекст та взаємодії між елементами, що може призвести до невірних рішень в деяких ситуаціях.

Враховуючи ці обмеження, аналіз ефективності "Greedy Algorithm" повинен враховувати конкретні характеристики завдання та враховувати його придатність до конкретного контексту використання. Основі узагальнені характеристики даного алгоритму представлені в таблиці 1.1.

Таблиця 1.1 – узагальнені характеристики алгоритму Greedy Algorithm

Характеристика	Опис
Тип алгоритму	Жадібний алгоритм
Призначення	Локальна оптимізація, вирішення задач з множинами
Оптимізаційний критерій	Вибір найкращого елемента на кожному кроці
Глобальна оптимальність	Не гарантує глобально оптимального рішення
Часова складність	Зазвичай ефективний, залежить від конкретного випадку
Простір пам'яті	Зазвичай невеликий, залежить від кількості доступних елементів
Стійкість до великих даних	Залежить від конкретного випадку та оброблюваних даних
Сфери застосування	Задачі оптимізації, задачі вибору найкращого варіанту на кожному етапі

1.4 Постановка задач дослідження

В ході проведеного аналізу було виявлено, що тема даного дослідження є актуальною. Метою дослідження є підвищення ефективності виробництва за рахунок розробки програмного забезпечення для навігації мобільної роботи. Об’єктом дослідження є процес побудови оптимального маршруту мобільної роботи. Предметом дослідження є реалізація алгоритму планування шляху та програми побудови маршруту. Методами дослідження є алгоритмізація та системний аналіз. Для досягнення поставленої мети потрібно вирішити наступні завдання:

- провести математичний опис навколишнього середовища та мобільного робота;
- провести удосконалення методу побудови маршруту мобільного робота у двомірних координатах;
- провести аналіз та вибір алгоритму побудови шляху;
- розробити алгоритм обрізання шляхів на базі Greedy Algorithm;
- розробити модель згладжування шляху;
- проаналізувати та вибрати середовище розробки;

- розробити алгоритму обрізання шляхів;
- розробити алгоритму згладжування шляхів;
- розробити загального алгоритму роботи програми;
- реалізувати програмний код;
- провести експериментальні дослідження та проаналізувати отримані результати.

2 РОЗРОБКА МАТЕМАТИЧНОЇ МОДЕЛІ ОПИСУ МОБІЛЬНОГО РОБОТУ У ДВОМІРНОМУ ПРОСТОРІ

2.1 Математичний опис навколишнього середовища та мобільного робота

Нехай у межах даних досліджень, припустимо, що мобільний робот рухається відомою та обмеженою двовимірною площиною, яка дискретизована у вигляді сітки. Кожна точка сітки відповідає невеликій області реального простору. Якщо простір у цій галузі повністю прохідний, його стан вважається вільним, інакше воно вважається зайнятим. Конструкція колісного мобільного робота стандартна і складається з двох приводних коліс з лівого та правого боку, обґрунтуванням вибору даної конструкції є її доступність та широке розповсюдження, що в майбутньому спростить процес моделювання [4]. Загальний вигляд обраної конструкції мобільного робота наведено на рисунку 2.1.

Рисунок 2.1 – Mini Round Chassis 2WD DIY Smart Car

Для вирішення поставленої задачі в рамках даних досліджень необхідно розробити кінематичну модель обраного двоколісного мобільного робота 2WD у двомірній площині.

Кінематична модель мобільного робота – це математичний опис руху робота на основі його кінематичних параметрів та рівнянь. Ця модель описує, як змінюються положення та орієнтація робота у просторі залежно від керуючих входів (наприклад, швидкості коліс) без урахування динамічних ефектів, таких як сили та моменти.

Кінематична модель може бути лінійною або нелінійною залежно від складності руху та конструкції робота. Вона може описувати різні типи роботів, включаючи колісні роботи, роботи з ногами, роботи з маніпуляторами та інші.

Важливі елементи, які можуть включати кінематичну модель мобільного робота, включають:

– визначення поточного положення робота в тривимірному просторі (x, y, z) та його орієнтації (кути нахилу, тангажу та нишпорення);

– керуючі входи у вигляді швидкості або кутові швидкості коліс, двигунів або інших елементів, що впливають на рух робота;

– кінематичні рівняння, що пов'язують положення та орієнтацію робота з його керуючими входами. Ці рівняння можуть бути лінійними чи нелінійними, залежно від типу моделі;

– обмеження руху, що накладаються на кінематичну модель, можуть враховувати обмеження руху, такі як максимальні швидкості та кутові швидкості;

– траєкторії та планування оптимальних траєкторій та шляхів руху робота.

Кінематична модель не враховує динамічних ефектів, таких як інерція, сили тертя та зовнішні впливи. Вона надає більш простий та обчислювально ефективний спосіб моделювання руху робота для цілей управління та

планування шляху. У складніших випадках, коли необхідно враховувати динамічні ефекти, використовується динамічна модель [11].

Внаслідок цього, представимо обраного мобільного робота в площині координат XY та опишемо базові точки для розробки кінематичної моделі. На рисунку 2.2 представлена кінематична модель мобільного робота 2WD.

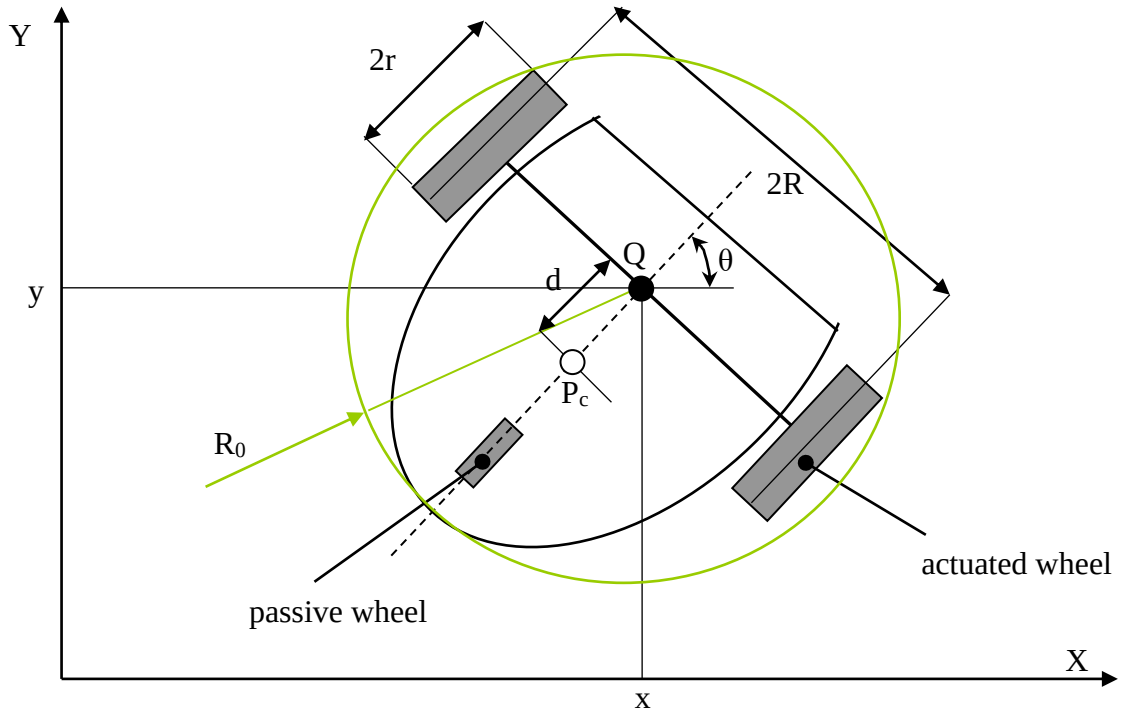

Рисунок 2.2 – Кінематична модель мобільного робота 2WD

Позначимо через:

P_c – центр мас мобільного робота, що є важливою частиною проектування та управління роботом. Центр мас є точку, в якій можна вважати сконцентрованою всю масу робота;

Q – середня точка двох колес, що приводяться в дію, облік середньої точки двох коліс є важливим аспектом при створенні кінематичної моделі мобільного робота, щоб забезпечити його стабільний та керований рух, а також для розробки ефективних алгоритмів управління та навігації;

d – відстань між P_c і Q, впливає те що, як зміна швидкостей коліс впливає рух центру мас робота. Це важливо для розробки алгоритмів керування рухом, які дозволяють роботу переміщуватись у потрібному напрямку та виконувати задані маневри;

R – половина ширини колеса, що приводиться в рух зліва направо. Врахування половини ширини колеса, що приводиться в рух зліва направо (або право наліво) у кінематичній моделі мобільного робота, є важливим аспектом при розробці цієї моделі. Це робиться для врахування особливостей руху робота з диференціальним приводом або іншими системами, де кожне колесо може керуватися незалежно;

r – радіус колеса. Врахування радіуса колеса в кінематичній моделі мобільного робота є важливим аспектом, це дає можливість для обліку розмірів та особливостей коліс робота, які значно впливають на його рух і маневреність;

R_o – радіус найменшого кола з центром Q, що охоплює всього робота. Цей радіус, також відомий як радіус обігу або радіус маневру, визначає, наскільки близько мобільний робот може під'їхати до перешкоди або повернути в обмеженому просторі.

Також визначимо три змінні, пов'язані з координатами робота:

θ – кут курсу мобільного робота, оскільки кут курсу є напрямом, у якому рухається робот щодо його початкової орієнтації. Цей кут є ключовою змінною при описі руху та керуванні роботом;

x та y – координати Q, необхідно для точного опису положення та руху робота у просторі. Ці змінні використовуються в кінематичній моделі робота і для вирішення задач навігації та управління.

В рамках даних досліджень для розробки кінематичної моделі мобільного робота 2WD, поставимо такі умови, що колеса не прослизають. Що дає можливість припустити, що кінематична модель мобільного робота 2WD (рис. 2.2) є – нелономний зв'язок, як показано у виразі (2.1).

Неголономний зв'язок – це обмеження на рух системи, при якому положення та швидкість частки або об'єкта пов'язані між собою. Неголономні зв'язки мають важливе значення в механіці та робототехніці, особливо при описі руху мобільних роботів та механічних систем, які мають обмеження на рух, такі як колеса та інші елементи, які можуть рухатися лише певним чином:

$$x\sin\theta - y\cos\theta = 0\,. \tag{2.1}$$

Виходячи з цього, кінематичну модель мобільного робота 2WD можна представити у вигляді (2.2-2.3):

$$\dot{q} = J(q)z = \frac{r}{2}\begin{bmatrix} \cos\theta & \cos\theta \\ \sin\theta & \sin\theta \\ R^{-1} & -R^{-1} \end{bmatrix}\begin{bmatrix} \upsilon_{wr} \\ \upsilon_{wf} \end{bmatrix}, \tag{2.2}$$

$$\tau = M\dot{z} + C(\dot{q})z + Dz + \tau_d\,, \tag{2.3}$$

де $q = [x, y, \theta]^T$ – вектор положення у двомірному просторі;

$z = [\upsilon_{wr}, \upsilon_{wf}]^T$ – вектор швидкості, де υ_{wr} – кутова швидкість правого та υ_{wf} лівого колеса відповідно.

Виходячи з цього, динаміку руху мобільного робота у двовимірному просторі можна представити у вигляді набору наступних моделей:

$$M = \begin{bmatrix} m_{11} & m_{22} \\ m_{12} & m_{21} \end{bmatrix}, \tag{2.4}$$

де M – матриця інерції, яка використовується для опису динаміки руху мобільного робота у деяких контекстах. Ця матриця інерції пов'язана з моментом інерції та масою мобільного робота;

m_{11} і m_{22} – представляють моменти інерції мобільного робота щодо його осей x та y відповідно;

m_{12} і m_{21} – представляють недіагональні елементи інерції матриці, які можуть вказувати на можливі зв'язки між двома осями (x, y) в контексті динаміки робота.

Внаслідок чого, динаміка руху мобільного робота в кутових координатах виглядатиме так:

$$C(\dot{q}) = \frac{r^2 m_c d}{2R} \begin{bmatrix} 0 & \dot{\theta} \\ -\dot{\theta} & 0 \end{bmatrix}, \tag{2.5}$$

де $C(\dot{q})$ – це матриця Коріолісових та відцентрових компонентів у рівняннях руху. Вона залежить від швидкостей $(\dot{q})$ мобільного робота. Ця матриця враховує взаємодію між кутовими швидкостями суглобів і може виникати при обертальних рухах робота;

r – радіус колеса;

c – це деяка константа, яка може залежати від фізичних параметрів робота та його динаміки;

d – ця відстань від центру мас робота до осі обертання;

$\dot{\theta}$ – це кутова швидкість, похідна кута θ по часу.

Інерції для опису динаміки руху мобільного робота можна представити у вигляді наступної матриці:

$$D = \begin{bmatrix} d_{11} & 0 \\ 0 & d_{22} \end{bmatrix}, \tag{2.6}$$

де D – матриця інерції, яка описує, як маса та розподіл маси мобільного робота впливають на його рух. Ця матриця зазвичай є квадратною матрицею розміром $N \cdot N$, де N – кількість ступенів свободи робота;

d_{11} и d_{22} – це діагональні елементи інерції матриці. Вони є моментами інерції робота навколо осей, які відповідають його ступеням свободи. Ці моменти інерції можуть залежати від конкретних характеристик робота та його геометрії. В даному випадку, в рамках досліджень з точки зору двомірних координат, матриця інерції має розмір, що може вказувати на те, що робот має два ступені свободи, пов'язані з його обертальними рухами навколо різних осей. Моменти інерції (d_{11} и d_{22}) для кожного з цих ступенів свободи можуть мати різні значення, що враховує відмінності у розподілі маси та геометрії робота щодо різних осей. Що дозволить провести аналізу його руху та визначення, які зусилля та моменти можуть виникати при його русі та маневруванні.

Для опису динаміки руху мобільного робота в контексті його обертального руху можна представити у вигляді наступної моделі:

$$m_{11} = \frac{r^2}{4R^2}(mR^2 + I) + I_w, \tag{2.7}$$

де m_{11} – це елемент матриці інерції, який відповідає першому ступені свободи робота в його обертальному русі. Він описує, як маса та розподіл маси впливають на обертальний рух робота;

r – це радіус колеса;

R – це відстань від центру маси мобільного робота до точки, навколо якої відбувається обертання. Ця відстань може бути пов'язана з геометрією робота та його моментом обертання;

m – маса мобільного робота;

I – момент інерції мобільного робота щодо точки обертання. Він характеризує розподіл маси робота та його здатність до обертальних рухів;

ω – кутова швидкість обертання робота навколо вибраної точки.

Опис обертання мобільного робота в іншій площині можна описати наступною кінематичною моделлю:

$$m_{12} = \frac{r^2}{4R^2}(mR^2 - I), \tag{2.8}$$

де m_{12} – це елемент матриці інерції, який відповідає другому ступеню свободи робота у його обертальному русі.

Всі інші елементи описані у моделі (2.7).

Для опису загальної маси мобільного робота (m), яка включає два компоненти: масу мобільної бази (m_c) та масу, пов'язану з обертанням коліс або інших рушійних частин ($2m_w$):

$$m = m_c + 2m_w, \tag{2.9}$$

де m – це загальна маса мобільного робота. Вона включає всі компоненти і елементи робота, включаючи його корпус, електроніку, датчики, акумулятори і т. д;

m_c – це маса мобільної бази, тобто маса всього робота без урахування маси рухомих частин, таких як колеса або гусениці;

$2m_w$ – це маса, пов'язана з обертанням рушійних частин робота, помножена на кутову швидкість (w), з якою вони обертаються. Це враховує внесок у загальну масу від цих рухомих частин, які також роблять свій внесок у динаміку руху робота.

Загальний момент інерції (I) мобільного робота, враховуючи різні компоненти та їх внесок у момент інерції:

$$I = m_c d^2 + 2m_w R^2 + I_c + 2I_m, \tag{2.10}$$

де I – це загальний момент інерції мобільного робота. Момент інерції – це фізична величина, яка описує, наскільки об'єкт чинить опір зміні своєї швидкості обертання. У цьому контексті це момент інерції всього робота щодо його осі обертання;

mc – це маса мобільної бази робота. Це маса всього робота без урахування частин, що рухаються, таких як колеса або гусениці;

d – це відстань від осі обертання (зазвичай центру робота) до маси мобільної бази. Ця відстань може бути важливою при розрахунку моменту інерції, оскільки вона впливає його величину;

$2mw$ – це момент інерції, пов'язаний з обертанням рухомих частин робота (наприклад, коліс) щодо їхньої осі обертання. Це враховує внесок у момент інерції від частин робота, що обертаються;

R – це радіус колеса або відстань від осі обертання до зовнішнього краю частин, що обертаються, наприклад, коліс;

I_c – це момент інерції інших компонентів робота, які не є частинами, що рухаються (наприклад, електроніка, акумулятори і т. д.). Це момент інерції цих компонентів щодо осі обертання;

$2I_m$ – це момент інерції, пов'язаний з масою частин робота, що рухаються (наприклад, маса коліс) щодо їх осі обертання.

Опишемо вектор моментів (τ), діють на мобільного робота. Вектор моментів зазвичай представляється як вектор із двома компонентами, які можуть бути записані у вигляді:

$$\tau = [\tau_r, \tau_f]^T, \tag{2.12}$$

де $\tau_г$ – це компонент моменту щодо поздовжньої осі робота. Вона описує момент, який впливає робота і обертає його навколо поздовжньої осі, що проходить через центр мас робота;

τ_f – це компонент моменту щодо поперечної осі робота. Вона визначає момент, який впливає на робота і обертає його навколо поперечної осі, що також проходить через центр мас робота.

Ці моменти можуть бути результатом зовнішніх сил або зусиль, що управляють, і вони можуть впливати на орієнтацію робота в просторі і на його здатність обертатися або маневрувати.

Вектор динамічних моментів (τ_d), які виникають під час руху мобільного робота. Цей вектор має два компоненти:

$$\tau_d = [\tau_{d1}, \tau_{d2}]^T, \tag{2.13}$$

де τ_{d1} – перший компонент вектор динамічних моментів. Вона описує момент, викликаний дією всіх зовнішніх сил, які можуть повертати робота навколо його поздовжньої осі. Це може бути пов'язане з нахилами, вітром чи іншими факторами, що впливають на орієнтацію робота;

τ_{d2} – друга компонент вектора динамічних моментів. Вона визначає момент, викликаний дією всіх зовнішніх сил, які можуть повертати робота навколо його поперечної осі. Це може бути пов'язано з бічними впливами чи іншими факторами, що впливають на орієнтацію робота.

Без урахування індуктивності двигуна, динаміка постійних струмів DC-моторів, що управляють провідними колесами, представляється в такому вигляді:

$$\tau_m = K_T i_a, \tag{2.14}$$

де τ_m – представляє собою момент (або крутний момент) двигунів мобільного робота;

K_T – позначає постійну моменту двигуна, яка характеризує зв'язок між струмом, що протікає через двигун, і створюваним ним моментом;

i_a – це струм, що надходить у двигуни мобільного робота.

Для опису зв'язку між керуючою напругою (u), струмом (i_a) та кутовою швидкістю ($\dot{\theta}_m$) двигунів мобільного робота. Вона враховує електричні характеристики двигунів (опір і зворотний зв'язок) і показує, як зміна напруги і струму впливає швидкість обертання двигунів і, отже, на рух мобільного робота.

$$u = R_a i_a + K_E \dot{\theta}_m, \tag{2.15}$$

де u – це вектор напруги (або сигналу, що управляє) для двигунів мобільного робота. Цей вектор включає значення напруги для кожного з двигунів;

R_a – це опір двигуна (або електрична характеристика), що впливає на величину струму у двигуні при заданій напрузі;

a – це коефіцієнт зворотного зв'язку між кутовою швидкістю двигуна та струмом у ньому;

i_a – це вектор струму у двигунах мобільного робота. Він включає значення струму для кожного з двигунів;

K_E – це коефіцієнт зворотного зв'язку між кутовою швидкістю двигуна і створюваною ним зворотною електромагнітною силою (back electromotive force, EMF);

$\dot{\theta}_m$ – це вектор кутової швидкості (похідний кута повороту за часом) для двигунів мобільного робота.

Варто логічно припустити, що напруга повинна бути обмежена так як $|u_r \le u_{\max}|$ та $|u_l \le u_{\max}|$. Параметри K_T, K_E и R – є константою крутного моменту двигуна, коефіцієнтом зворотної електрорушійної з опором.

Як можна бачити з рисунка 2.1, на макеті вибраного мобільного робота використовується стандартний DC Electric Motor 3V-6V Dual Shaft Geared, який містить редуктор, внаслідок чого необхідно врахувати як прискорення двигуна (а отже, рух мобільного робота) залежить від струму в двигуні, його кутової швидкості та параметрів, пов'язаних з його моментом сили. Тому передавальне число даного двигуна можна визначити з виразу:

$$n_i = \frac{\dot{\theta}_{mi}}{v_{wi}} = \frac{\tau_i}{\tau_{mi}}; i = r, l, \tag{2.16}$$

де n_i – це прискорення двигуна. Цей вираз описує, як прискорення двигуна залежить від кутової швидкості (θ_{mi}) та моменту сили (τ_i), що створюється двигуном. Тут сила струму (i) може бути (r) для правого двигуна та (l) для лівого двигуна;

τ_i – це момент сили, що створюється двигуном. Він залежить від струму (i) та зворотної електромагнітної сили (back electromotive force, EMF) двигуна;

τ_m – це момент сили, який потрібний для руху мобільного робота. Цей момент сили залежить від різних факторів, таких як маса робота, тертя, нахил поверхні і т.д;

v – це коефіцієнт зв'язку між струмом у двигуні (i) та моментом сили, який він створює (τ_i);

w – це коефіцієнт зв'язку між кутовою швидкістю двигуна (θ_{mi}) та моментом сили, який він створює (τ_i).

Варто пояснити чому у вираз (2.16) присутній такий тип запису, як “=”, в даному випадку мають на увазі рівність отриманих результатів, але за умов дотримання однакового прискорення двигунів n_i як правого, так і лівого.

Динамічну модель мобільного робота, яка визначає, як управляючі сигнали впливають на узагальнені швидкості робота у певній конфігурації. Якобіан є математичним зв'язок між керуючими сигналами і рухом робота в даній конфігурації:

$$\dot{q} = J(q)z\,, \tag{2.17}$$

де $J(q)$ – це якобіан (матриця Якобі, яка пов'язує вектор сигналів, що управляють z із узагальненими швидкостями $\dot{q}$), який залежить від поточної конфігурації мобільного робота;

$\dot{q}$ – є вектором узагальнених швидкостей мобільного робота. Ці швидкості можуть включати лінійні і кутові швидкості, які визначають рух робота;

z – це вектор керуючих впливів або керуючих сигналів, що подаються на мобільний робот. Ці сигнали можуть включати кутові швидкості коліс, моменти сили або інші параметри, що управляють рухом робота.

Синтезуючи моделі (2.2-2.17), можна отримати повну динамічну модель двоколісного мобільного робота у двовимірній системі координат:

$$\dot{z}'' = ''M^{-1}(R_a^{-1}NK_T u - (C(\dot{q}) + D + P_1)z - \tau_d)\,, \tag{2.18}$$

де $\dot{z}$ – є вектор прискорення мобільного робота. Ці прискорення можуть включати лінійні і кутові прискорення, які визначають рух і поворот робота відповідно;

M^{-1} – це обернена матриця інерції робота. Вона назад пов'язує вектор прискорень ($\dot{z}$) з вектором сигналів керування (u), які подаються на мобільний робот;

R^{-1} – це зворотна матриця кінематичної моделі робота, яка пов'язує узагальнені швидкості ($\dot{q}$) з вектором сигналів керування (u);

a – це вектор прискорень, спричинених дією зовнішніх сил, наприклад, гравітацією;

N – це вектор реакцій зв'язків робота з довкіллям;

K_T – це матриця перетворення з простору суглобових координат в простір випереджальних сигналів, що управляють (наприклад, моментів на колесах);

$C(\dot{q})$ – це матриця коріолісових і відцентрових сил, пов'язаних з кутовими швидкостями;

D – це матриця демпфування, що представляє сили тертя та інші втрати у системі;

P_1 – це матриця, що представляє пропорційні сигнали, що управляють, які можуть використовуватися для управління роботом в певних режимах;

τ_d – це вектор моментів, створюваних зовнішніми джерелами, наприклад, двигунами коліс.

Синтезований вираз являє собою повну динамічну модель двоколісного мобільного робота у двовимірній системі координат. Вона описує, як прискорення робота ($\dot{z}$) пов'язані з керуючими сигналами (u), зовнішніми силами (a) та реакціями від навколишнього середовища (N). У цій моделі враховуються як інерційні, і неінерційні (фрикційні, відцентрові та інші) ефекти, що впливають рух робота.

2.2 Удосконалення методу побудови маршруту мобільного робота у двомірних координатах

Грунтуючись на виведеному виразі, що являє собою повну динамічну модель двоколісного мобільного робота у двовимірній системі координат, у виразі (2.18), що дозволяє враховувати такі особливості при побудові маршруту переміщення мобільного робота:

– точність руху, запропонована динамічна модель дозволяє більш точно прогнозувати рух робота у відповідь певні команди руху чи управляючі сигнали. Це дозволяє врахувати фізичні обмеження робота, такі як динамічні обмеження на прискорення та максимальні швидкості. При плануванні шляху робота можна врахувати ці обмеження, щоб забезпечити безпечний та стабільний рух;

– динамічна модель дозволяє врахувати сили тертя та опору, які можуть впливати на рух робота. Це особливо важливо при плануванні шляху на нерівній поверхні або в умовах опору, таких як пісок або бруд. Облік цих факторів дозволяє більш реалістично спроектувати шлях та запобігти можливим проблемам, пов'язаним з тертям;

– уникнення колізій: запропонована модель дозволяє передбачати, як робот рухатиметься поблизу перешкод. Це дозволяє роботу уникати колізій з навколишніми об'єктами, враховуючи як геометричні параметри робота, а й його динамічні характеристики. Планування шляху з урахуванням цієї інформації дозволяє роботу уникати небезпечних ситуацій;

– оптимізація енергоспоживання за рахунок визначення динамічних характеристик робота, можна оптимізувати його рух з погляду енергоспоживання. Це важливо для мобільних роботів, що працюють на акумуляторах, тому що оптимізований рух може продовжити час автономної роботи.

Таким чином, запропонована повна динамічна модель дозволяє більш точно та ефективно розробити алгоритми побудови маршруту з урахуванням його фізичних можливостей та обмежень.

Нехай процес планування маршруту від початкової точки S до кінцевої точки E, в рамках даних досліджень назвемо плануванням шляху. Шлях, запланований у даному описі, позначимо через P. Припустимо, що P може містити кілька проміжних точок, внаслідок чого, позначимо їх через $P(k)$, де k-1-а точка P, за умови, що $k=0$ до n_p, а n_p – кількість точок у P.

Визначимо базові умови для планування шляху, які мають задовольняти наступним умовам (2.19-2.22).

$$P(0) = S \ , \tag{2.19}$$

де $P(0)$ – є планований шлях мобільного робота у початковий час (зазвичай позначається як початкове становище робота);

S – позначає початкове положення робота у двовимірній системі координат (зазвичай задається як точка з координатами x, y).

Таким чином, формула вказує, що початкове положення планованого шляху мобільного робота (у момент часу 0) дорівнює початковому положенню робота, заданому точкою $S(x, y)$ у двовимірній системі координат. Це означає, що у початковий час планований шлях робота збігається з його початковим становищем.

$$\begin{gathered} P(n_p - 1) = E \\ state(B(P_x(k), P_y(k))) = free, \forall k \in \aleph, k - n_p \ , \\ P(k+1) \in N(P), \forall k \in \mathrm{N}, k < n_p - 1 \end{gathered} \tag{2.20}$$

де $P(n_p - 1)$ – є планований шлях мобільного робота в момент часу;

E – це номер кроку планування шляху;

$P_x(k)$ та $P_y(k)$ – являють собою координати мобільного робота на етапі планування шляху k у двовимірній системі координат. Зазвичай k позначає поточний крок або час у процесі планування;

$B(P_x(k), P_y(k))$ – є функцією зворотного перетворення, яка переводить координати мобільного робота в комірку або вузол на сітці (зазвичай це використовується в контексті планування на сітці);

$state(B(P_x(k), P_y(K)))$ – позначає стан (статус) даного осередку або вузла на сітці після застосування функції зворотного перетворення. Це може включати в себе інформацію про те, чи є осередок вільним, зайнятим або якось інакше характеризується для планування руху;

$free$ – позначає функцію, яка визначає, чи є стан (статус) осередку або вузла на сітці вільним для проходження мобільним роботом;

$P(k+1)$ – є планована точка на шляху мобільного робота на наступному кроці $k+1$. Ця точка повинна належати безлічі $N(P)$, де $N(P)$ позначає набір доступних або дозволених точок, в яких мобільний робот може перебувати або переміщатися;

$\forall k \in N, k < n_p - 1$ – вказує на те, що ця умова має виконуватися для кожного кроку планування шляху 0 до $np-1$. Тут np є кількість кроків у плануванні шляху.

Таким чином у (2.20) задаються такі умови:

– що на n_p 1-му кроці планування шляху мобільного робота планований шлях досягає кінцевої точки E. Це означає, що на цьому етапі планування шлях робота завершується і досягає своєї кінцевої мети в точці E;

– статус певного осередку або вузла на сітці після зворотного перетворення його координат $(P_x(k), P_y(k))$ є вільним для проходження мобільним роботом, що відповідає значенню функції $free$;

– запланована точка на шляху мобільного робота на наступному кроці $k+1$ повинна знаходитися в межах доступних або дозволених точок $N(P)$ для кожного кроку планування шляху від 0 до $np-1$. Ця умова забезпечує коректність планування шляху, враховуючи доступні місця для переміщення робота на кожному кроці.

При застосуванні запланованого шляху до навігації двоколісних мобільних роботів слід зважати на властивість безперервності орієнтації. Через нелономне обмеження в (2.1) роботу необхідно слідувати процесу

«стоп-поворот-йти» щоразу, коли він стикається з переривчастою точкою напрямку, що потребує багато часу та енергії через велику зміну швидкості. Тому ми очікуємо, що запланований шлях матиме властивість спрямованої безперервності, щоб задовольняти нерівність:

$$\left|P_\theta(k+1)-P_\theta(k)\right|<\theta_t, \forall k\in N, k<n_p-1\ , \tag{2.21}$$

де $P_\theta(k)$ – являє собою кут повороту (або напрямок) на k-м етапі в плануванні шляху;

k – індекс етапу у плануванні шляху;

$P_\theta(k+1)$ – є кутом повороту на наступному етапі $(k+1)$-м етапі) у плануванні шляху;

n_p – загальна кількість етапів у плануванні шляху;

θ_t – деяка межа кута повороту, який визначає допустиму різницю у кутах між послідовними етапами.

Умова $\left|P_\theta(k+1)-P_\theta(k)\right|<\theta_t$ означає, що різниця в кутах повороту між послідовними етапами k-м та $(k+1)$-м) має бути менше заданої межі (θ_t). Ця умова гарантує, що запланований шлях має властивість спрямованої безперервності, тобто зміна напрямку на шляху відбувається поступово та обмежено заданою межею.

Якщо нам потрібно отримати додаткову динамічну інформацію про шлях, а саме лінійну швидкість $P_v(k)$ та кутову швидкість $P_w(k)$, змінну k в (2.20) слід розглядати як дискретний час. Необхідно задовольнити додаткові обмеження (2.22), де Δk – часовий інтервал, а max – максимальне лінійне прискорення, а αmax – максимальне кутове прискорення:

$$\begin{gathered}\left|P_\upsilon(k+1)-P_\upsilon(k)\right|\le\alpha_{\max}\Delta k\ ;\\ \left|P_w(k+1)-P_w(k)\right|\le\alpha_{\max}\Delta k, \forall k\in N, k<n_p-1,\end{gathered} \tag{2.22}$$

де $\alpha_{\max}$ – це максимальні значення прискорення та кутового прискорення відповідно;

Δk – крок часу між етапами планування шляху.

Ці обмеження забезпечують, що зміни швидкості та кутової швидкості на шляху обмежені значеннями $\alpha_{\max}$ відповідно. Це важливо для забезпечення безпеки руху та плавності зміни траєкторії мобільного робота. Умова дійсна для всіх етапів планування шляху ($k < n_p - 1$).

2.3 Висновки до 2 розділу

В ході виконання 2 розділу кваліфікаційної роботи магістра, було проведено математичний опис навколишнього середовища та мобільного робота. Після, грунтуючись на виведеному виразі, що являє собою повну динамічну модель двоколісного мобільного робота у двовимірній системі координат, це дозволило враховувати особливості побудови маршруту переміщення мобільного робота. Далі було удосконалено метод побудови маршруту мобільного робота у двомірних координатах.

3 РОЗРОБКА АЛГОРИТМА ПЛАНУВАННЯ ШЛЯХУ У ДВОВИМІРНІЙ СИСТЕМІ КООРДИНАТ

3.1 Аналіз та вибір алгоритму побудови шляху

RRT (Rapidly Exploring Random Tree) – це алгоритм планування шляху, який широко використовується для задач планування руху мобільних роботів та інших систем. Його ключова ідея полягає в тому, щоб швидко досліджувати простір станів, створюючи випадкове дерево, яке може бути використане для знаходження шляху від початкового до цільового стану [12].

Принцип роботи алгоритму RRT:

– ініціалізація: починаємо з побудови дерева, що складається лише з початкового стану;

– розширення дерева: на кожній ітерації генеруємо випадкову точку в просторі станів. Потім знаходимо найближчу точку в дереві до згенерованої випадкової точки;

– рух у напрямку випадкової точки: створюємо нову точку, рухаючись із найближчої точки у напрямку згенерованої випадкової точки на певну відстань;

– перевірка на зіткнення: перевіряємо, чи не перетинається нова точка з перешкодами. Якщо ні, додаємо нову точку до дерева і з'єднуємо її з найближчою точкою;

– перевірка досягнення мети: перевіряємо, досягли ми цільового стану. Якщо так, то шлях знайдено і можна завершити алгоритм;

– повторення: повторюємо кроки 2-5 доти, доки знайдено шлях до мети чи доки вичерпаються спроби.

Недоліки алгоритму RRT:

випадковість: використання випадкових точок може призвести до неоптимальних шляхів і залежить від розподілу випадкових точок;

– неоптимальність: знайдений шлях не завжди є оптимальним, особливо якщо алгоритм працює з обмеженим числом ітерацій;

– схильність до локальних мінімумів: алгоритм може застрягти у локальних мінімумах чи складних конфігураціях простору станів;

– не враховує динаміку: RRT не враховує динамічних обмежень робота, що може призвести до планування шляхів, які робот не може виконати в реальному часі;

– чутливість до початкових умов: результати можуть залежати від початкового розташування дерева у просторі станів.

BRRT (Bidirectional Rapidly Exploring Random Tree) – це модифікація алгоритму RRT, який намагається зменшити довжину знайденого шляху та збільшити ефективність планування шляху за рахунок одночасного розширення дерева від початкової та кінцевої точок. Ось як працює BRRT:

– ініціалізація: почніть із побудови двох дерев – одного від початкової точки та одного від кінцевої точки;

– розширення дерев: на кожній ітерації алгоритму генеруються випадкові точки в просторі станів і шукаються найближчі точки в кожному з дерев;

– рух у напрямі випадкових точок: кожному дереву створюються нові точки, рухаючись від найближчих точок у бік згенерованих випадкових точок;

– перевірка на зіткнення: перевіряються зіткнення нових точок в обох деревах. Якщо нові точки безпечні, вони додаються до відповідного дерева та з'єднуються з найближчими точками;

– перевірка на з'єднання дерев: перевіряється, чи можуть бути нові точки з обох дерев з'єднані. Якщо це відбувається, шлях знайдено та алгоритм завершує роботу;

– повторення: повторюються кроки 2-5 доти, доки знайдено шлях до мети чи доки досягнуто максимальну кількість ітерацій [13].

Переваги BRRT включають більш ефективне використання інформації про початкову і кінцеву точки, що може призвести до більш оптимальних шляхів. Однак у BRRT також є недоліки:

– складність реалізації: алгоритм BRRT більш складний у реалізації порівняно із звичайним RRT через необхідність керування двома деревами;

– чутливість до початкових умов: як і RRT, BRRT також чутливий до початкових умов, що може вплинути на результати планування шляху;

– схильність до застрягання: BRRT також схильний до ризику застрявання в локальних мінімумах, особливо в складних середовищах з великою кількістю перешкод;

– незважаючи на недоліки, BRRT залишається ефективним методом для планування шляху, особливо в завданнях, де потрібне знаходження оптимального шляху у складних та динамічних навколишніх умовах.

Побудуємо таблицю 3.1, порівняння застосування алгоритмів RRT та BRRT для вирішення задачі побудови шляху мобільного робота у двовимірній системі координат.

Таблиця 3.1 – Порівняння алгоритмів RRT та BRRT для вирішення задачі побудови шляху мобільного робота у двовимірній системі координат

Критерії порівняння	RRT	BRRT
1	2	3
Принцип роботи	Одне дерево, розширення у випадковому напрямку та перевірка на зіткнення	Два дерева, розширення у випадковому напрямку та перевірка на зіткнення з обох сторін
Складність реалізації	Відносно простий у реалізації	Більш складний у реалізації через управління двома деревами
Чутливість до початкових умов	Чутливий, результати можуть залежати від початкових умов	Також чутливий до початкових умов
Ефективність у складних середовищах	Може застрягати у локальних мінімумах	Також схильний до ризику застрявання в локальних мінімумах, але може знайти більш оптимальні шляхи

Продовження таблиці 3.1

1	2	3
Застосовність	Широко використовується у випадках, де необхідно швидко знайти шлях у просторах великих розмірів	Добре підходить для завдань, де початкова та кінцева точки відомі заздалегідь і необхідно знаходити оптимальні шляхи
Складність довкілля	Менш ефективний у складних середовищах із безліччю перешкод	Може краще справлятися у складних середовищах, завдяки двом деревам та можливості знаходити більш оптимальні маршрути.
Швидкість пошуку шляху	Може бути швидше у простих середовищах	Зазвичай повільніший через керування двома деревами
Примітки	Часто використовується у випадках, де точка призначення невідома	Корисний, коли точка призначення відома і потрібно знайти найоптимальніший шлях

Розглянемо класичний алгоритм RRT, який традиційно є методом випадкової вибірки з неевристичними ітераціями. Загальний вид псевдокоду алгоритму RRT представлений нижче:

```
1:T.Initialize(S)
2:for i = 0 to Parameters.MaxSearchingNumber
3:    nr = RandomSampling(Map)
4:    nc = FindNearest(T, nr)
5:    nn = Extend(nc, nr)
6:    if CollisionCheck (Map, nc, nn) continue
7:    nn.parent = nc
8:    T.Add(nn)
9:    if !GoalTest(Map, nn, E) continue
10:  return SelectPath (T, E)
11:end
12: return Error.PathPlanningFailed.OutOfSearchingNumber
```

Цей код є основною структурою алгоритму RRT (Rapidly-Exploring Random Tree) для пошуку шляху в просторі станів мобільного робота. Давайте розберемо кожен крок алгоритму:

– ініціалізація: початок роботи алгоритму. Створюється порожнє дерево T, яке буде шлях робота в просторі станів;

– цикл пошуку шляху: алгоритм виконує ітерації в межах максимальної кількості спроб (MaxSearchingNumber) або до тих пір, поки не буде знайдено шлях до цільової точки;

– випадкова вибірка (Random Sampling): генерується випадкова точка nr в просторі станів карти;

– пошук найближчої точки (FindNearest): з існуючого дерева T знаходиться найближча до nr точка nc;

– розширення (Extend): намагаємося розширити дерево від nc у напрямку nr, щоб отримати нову точку nn;

– перевірка на зіткнення (Collision Check): перевіряється, чи не перетинається шлях між nc і nn із перешкодами на карті. Якщо так, то йдемо до наступної ітерації;

– присвоєння батька (Setting Parent): якщо nn пройшла перевірку зіткнення, їй присвоюється батьківська точка nc;

– додавання крапки до дерева (Adding Node): нова точка nn додається до дерева T;

– перевірка досягнення мети (Goal Test): перевіряється, чи є nn цільової точкою. Якщо так, то шлях знайдено, і алгоритм завершує роботу;

– вибір шляху (Selecting Path): якщо шлях не було знайдено в межах максимальної кількості спроб, вибирається шлях від початкової точки до знайденої nn;

– повернення знайденого шляху: знайдений шлях повертається як результат роботи алгоритму;

– помилка: переповнення числа спроб (Error Handling): Якщо максимальна кількість спроб вичерпана, повертається повідомлення про помилку, що вказує на невдачу в пошуку шляху.

Якщо робота може вільно переміщатися в навколишньому середовищі, стратегія двостороннього пошуку може зменшити діапазон пошуку. Двонаправлений RRT (BRRT), покращений алгоритм RRT, здійснює пошук від початкової та кінцевої точки. Псевдокод алгоритму BRRT наведено нижче:

```
1:Ts.Initialize(S)
2:Te.Initialize(E)
3:T = {Ts, Te}
4:N = {S, E}
5:for i = 0 to Parameters.MaxSearchingNumber
6:    nr = RandomSampling(Map)
7:    for j = 0 to 1
8:        nc = FindNearest(T(j), nr)
9:        nn = Extend(nc, nr)
10:       if CollisionCheck(Map, nc, nn) continue
11:       nn.parent = nc
12:       T(j).Add(nn)
13:       if !GoalTest(Map, nn, N(j)) continue
14:       return SelectPath (T(j), T(1 − j), nn, S, E)
15:   end
16:end
17:return Error.PathPlanningFailed.OutOfSearchingNumber
```

Отже, є два дерева пошуку T_s та T_e. Псевдокод двонаправленої RRT задається наступним чином. У стані ініціалізації тільки S в T_s і тільки E E в

T_e. T – масив осередків, включаючи T_s та T_e. n_r – точка випадкової вибірки на карті. Рядки з 7 по 15 є подвійним циклом для двостороннього пошуку. n_c – найближча до n_r точка в поточному пошуку дерева. n_n – точка, що простягається від n_c на випадкову відстань. CollisionCheck (Map, n_c, n_n) призначений для перевірки наявності перешкод між n_c та n_n, які можуть спричинити зіткнення. GoalTest(Map, n_n, $N(j)$) – перевірити, чи немає перешкод між n і ціль $N(j)$. SelectPath ($T(j)$, $T(1-j)$, n_n, S, E) призначений для побудови шляху від S до E. Ця функція створює перший шлях Path 1 на основі $T(1-j)$ починаючи з n_n, створює другий шлях Path 2 на основі $T(1-j)$, починаючи з n_n, змінює місцями Path 1 як Path 3, об'єднує Path 3 і Path 2 як Path і, нарешті, повертає Path або його перевернуте бачення Reverse (Path) в залежності від того, чи є перша точка Path S або E. Якщо я досягнув максимальної кількості пошуку, BRRT повертає помилку, яка вказує, що кількість пошуку досягнуто.

Як можна бачити з результатів порівняння двох алгоритмів RRT та BRRT, вони створюють безліч непотрібних та трудомістких поворотів. Тому необхідний механізм покращення шляху. Для покращення шляху використовуються три механізми: обрізання шляху, згладжування шляху та трапецієподібний профіль швидкості.

3.2 Розробка алгоритму обрізання шляхів на базі Greedy Algorithm

Жадібний метод (Greedy Algorithm) – це алгоритмічний підхід до прийняття рішень, при якому на кожному етапі вибирається локально оптимальне рішення, сподіваючись, що це призведе до глобально оптимального результату. Жадібні алгоритми працюють відповідно до локальної жадібної стратегії вибору, яка означає вибір найкращого в

поточному стані варіанта без урахування наслідків для майбутніх кроків або глобального оптимуму [14].

З точки вирішення поставлення завдань у рамках даних досліджень Greedy Algorithm дозволить проводити планування шляху, при якому робот приймає рішення про наступний крок на основі поточного стану та намагається знайти найбільш оптимальний шлях до цільової точки, виключаючи більш довгі або менш оптимальні шляхи. У загальному вигляді процес планування шляху буде виглядати наступним чином:

– ініціалізація: почніть з початкової точки та встановіть її як поточну позицію робота;

– цикл планування;

– обчисліть доступні напрямки або сусідні точки з поточної позиції;

– оцініть кожен доступний напрямок відповідно до будь-якої функції вартості (наприклад, відстань до цільової точки);

– виберіть напрямок з найменшою вартістю;

– перейдіть у вибраний напрямок і встановіть його як нову поточну позицію робота;

– повторюйте цей процес до досягнення цільової точки.

Цей метод приймає локальні оптимальні рішення кожному кроці, і робот рухається до мети, вибираючи найменший шлях кожному етапі. Однак він не гарантує перебування глобально оптимального шляху, оскільки робот може застрягти у локальному мінімумі чи максимумі. Тому псевдокод алгоритму обрізки шляху матиме такий вигляд:

```
1:PrunedPath.Initialize(Path(0))
2:i = 0
3:while i < length(Path) − 1
4:      for j = length(Path) −1 to i + 1 by −1
5:              if CollisionCheck(Map, Path(i), Path(j)) continue
6:              PrunedPath.Add(Path(j))
```

```
7:              i = j
8:              break
9:      end
10:end
11: return Pruned
```

Представлений псевдокод алгоритму дозволяє видалити надлишкові точки зі шляху мобільного робота, які впливають на обхід перешкод і, таким чином, спрощують шлях мобільного робота, зберігаючи його безпеку.

3.3 Розробка моделі згладжування шляху

Мета згладжування шляху – знайти дугу для заміни шляху навколо точок шляху, щоб забезпечити безперервність орієнтації. На рисунку 3.1 показаний приклад згладжування колії на п'ятиточковій колії.

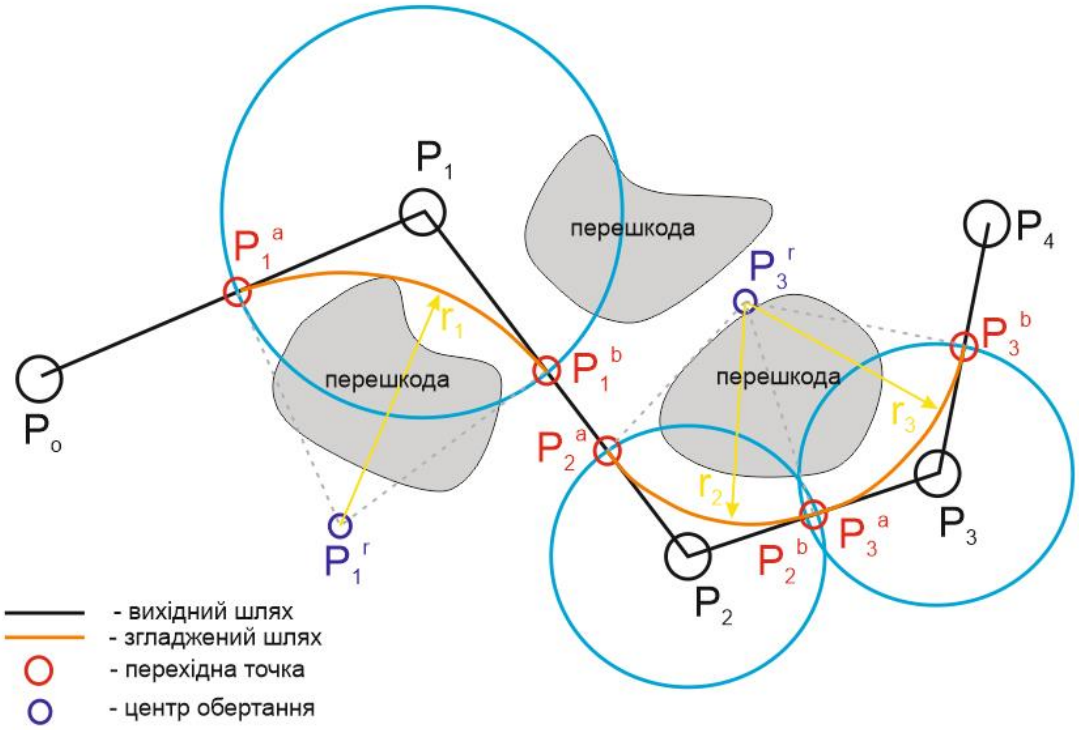

Рисунок 3.1 – Приклад згладжування шляху на 5-точковому шляху

Як можна бачити з рисунку 3.1, є три дуги для заміни частини вихідного шляху. Для кожної точки шляху, крім першої та останньої, її радіус обертання, центр обертання та дві точки дотику представлені як r_i, P_i^r, P_i^a та P_i^b відповідно. Співвідношення між цими точками можна описати у вигляді:

$$\begin{bmatrix} x_i^a \\ y_i^a \end{bmatrix} = \begin{bmatrix} x_i - r_i \sin(|\Delta\theta_i| / 2)\cos(\theta_i) \\ x_i - r_i \sin(|\Delta\theta_i| / 2)\sin(\theta_i) \end{bmatrix}; \tag{3.1}$$

$$\begin{bmatrix} x_i^b \\ y_i^b \end{bmatrix} = \begin{bmatrix} x_i - r_i \sin(|\Delta\theta_i| / 2)\cos(\theta_{i+1}) \\ x_i - r_i \sin(|\Delta\theta_i| / 2)\sin(\theta_{i+1}) \end{bmatrix}, \tag{3.2}$$

де x_i^a, y_i^a – початкові координати точки в локальній системі координат робота;

x_i^b, y_i^b – кінцеві координати після переміщення на P_i^b;

$\Delta\theta_i$ – кут повороту робота в радіанах;

θ_i – початковий кут напряму робота в локальній системі координат робота (щодо осі (x));

r_i – радіус повороту робота (відстань від точки повороту робота до центру робота).

Початковий кут напрямку робота в локальній системі координат (θ_i) знаходиться за таким виразом:

$$\theta_i = \tan^{-1}\left(\frac{y_i - y_{y-1}}{x_i - x_{i-1}}\right) + 2m_i\pi,\ 0 \le \theta_i < 2\pi, \tag{3.3}$$

де θ_i – кут між двома точками в радіанах;

(x_i, y_i) – координати поточної точки;

$(x_{i-1},\ y_{i-1})$ – координати попередньої точки.

Кут повороту робота в радіанах $\Delta\theta_i$ можна одержати з наступного виразу (3.4). Вона представляє різницю в кутах між поточним та попереднім сегментами шляху, враховуючи можливі множники n_i, які представляють кількість обертів або витків шляху. Це важлива частина алгоритмів, таких як алгоритм Дубінса, які знаходять оптимальний шлях між двома точками з урахуванням обмежень руху:

$$\Delta\theta_i = \theta_i - \theta_{i-1} + 2n_i\pi, -\pi \le \theta_i < \pi, \tag{3.4}$$

де $\Delta\theta_i$ – є зміна кута між двома сусідніми сегментами шляху;

θ_i – кут напряму поточного сегмента шляху;

θ_{i-1} – кут напряму попереднього сегмента шляху;

n_i – множник, який становить кількість обертів чи витків шляху.

3.4 Висновки до 3 розділу

В ході написання третього розділу даної кваліфікаційної роботи, спочатку було проведено аналіз та вибір алгоритму побудови шляху. Після проведеного аналізу, з результатів порівняння двох алгоритмів RRT та BRRT, можна бачити, що вони створюють безліч непотрібних та трудомістких поворотів. Тому для покращення шляху використовуються три механізми: обрізання шляху, згладжування шляху та трапецієподібний профіль швидкості. Далі було проведено розробку алгоритму обрізання шляхів на базі Greedy Algorithm та розробку моделі згладжування шляху.

4 РОЗРОБКА ПРОГРАМИ ПОБУДОВИ МАРШРУТУ

4.1 Вибір середовища розробки

Для розробки на мові програмування Python існує багато середовищ розробки (IDE) та текстових редакторів, з яких кожен має свої переваги і функціонал. PyCharm – це інтегроване середовище розробки для мови програмування Python, розроблене компанією JetBrains. Це потужне інструментарій, спеціально спроектований для полегшення роботи розробників Python у всіх аспектах процесу програмування [15].

PyCharm надає широкий спектр інструментів та функціоналів, що полегшують розробку на Python. Його можливості включають автоматичне завершення коду, вбудовану систему контролю версій, інтеграцію з віртуальними оточеннями, інструменти аналізу коду, візуальний редактор баз даних та інше.

Дане середовище розробки активно оновлюється та підтримується великою спільнотою та розробниками компанії JetBrains. Це означає, що ви можете очікувати регулярні оновлення, виправлення помилок та новий функціонал. PyCharm має інтеграцію з багатьма іншими інструментами та сервісами, такими як Docker, Flask, Django та інші. Також він підтримує розширення (плагіни), що дозволяє вам налаштовувати його під свої потреби.

Середовище розробки PyCharm підтримує інструменти для роботи з науковими бібліотеками, такими як NumPy, SciPy, Matplotlib, і іншими, що робить його привабливим для наукових досліджень та аналізу даних. Вбудовані інструменти аналізу коду, автоматична підсвічування синтаксису, можливість використання шаблонів коду та інші функції сприяють збільшенню продуктивності розробника [16].

Хоча PyCharm має платні версії, він також пропонує безкоштовну Community Edition з базовим функціоналом, що може бути важливим для початківців або для тих, хто не потребує всіх функцій Professional Edition.

4.2 Розробка алгоритму обрізання шляхів

Алгоритм обрізання шляхів – це метод оптимізації маршрутів у мобільних роботів чи інших подібних задачах. Його основна мета полягає в усуненні зайвих або непотрібних частин шляху, щоб покращити його ефективність та пристосованість до умов середовища. На рисунку 4.1 представлено розроблений алгоритм обрізання шляхів.

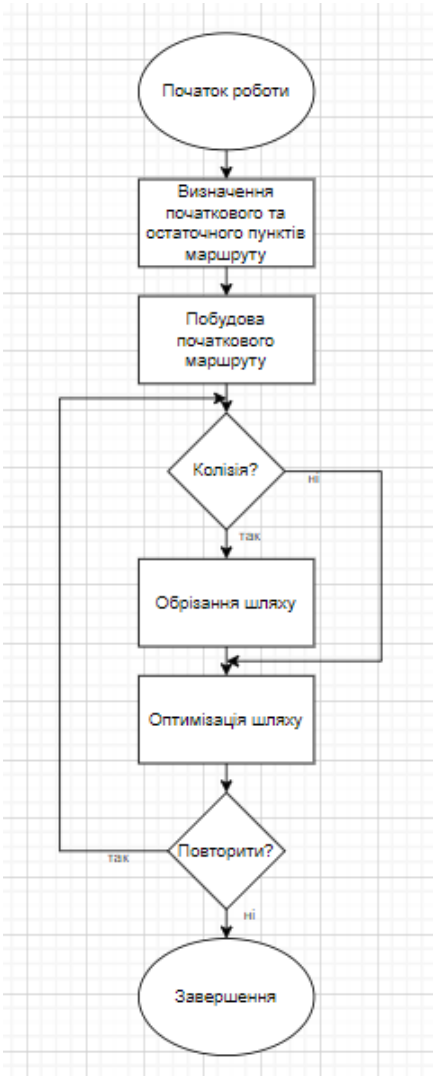

Рисунок 4.1 – Алгоритм обрізання шляхів

Алгоритм обрізання шляхів є етапом в оптимізації маршрутів для мобільних роботів та інших подібних систем. Його завданням є виявлення та усунення частин шляху, які можуть призвести до непрохідності через перешкоди чи інші обмеження в середовищі. Давайте розглянемо загальний алгоритм більш детально.

Після початку роботи, розпочинаємо з визначення початкового та кінцевого пунктів маршруту. Це може бути задано координатами чи іншими параметрами. Далі застосовуємо алгоритм для побудови початкового маршруту від початкового до кінцевого пункту. Перевіряємо кожний вузол маршруту на наявність колізій з перешкодами чи обмеженнями в середовищі, а якщо виявлена колізія, виконуємо обрізання маршруту у цьому місці, можливо, знаходячи нові точки, які обходять область колізій. Після, використовуємо алгоритми оптимізації для скорочення кількості вузлів шляху, зберігаючи при цьому загальний напрямок маршруту. Повторюємо процес перевірки колізій, обрізання та оптимізації до тих пір, поки не отримаємо оптимальний маршрут. Завершуємо процес, коли отримали оптимальний маршрут, який уникав перешкод та враховував інші обмеження. Даний алгоритм може бути реалізований різними способами в залежності від конкретних вимог та умов завдання.

4.3 Розробка алгоритму згладжування шляхів

Алгоритм згладжування шляхів використовується для покращення якості маршрутів, зменшення кількості вузлів та уникнення непотрібних поворотів, зроблених алгоритмами пошуку шляху. Основна ідея полягає в тому, щоб застосувати фільтрацію або оптимізацію до вихідного шляху, щоб згладити його форму.

На рисунку 4.2 представлено алгоритм згладжування шляхів.

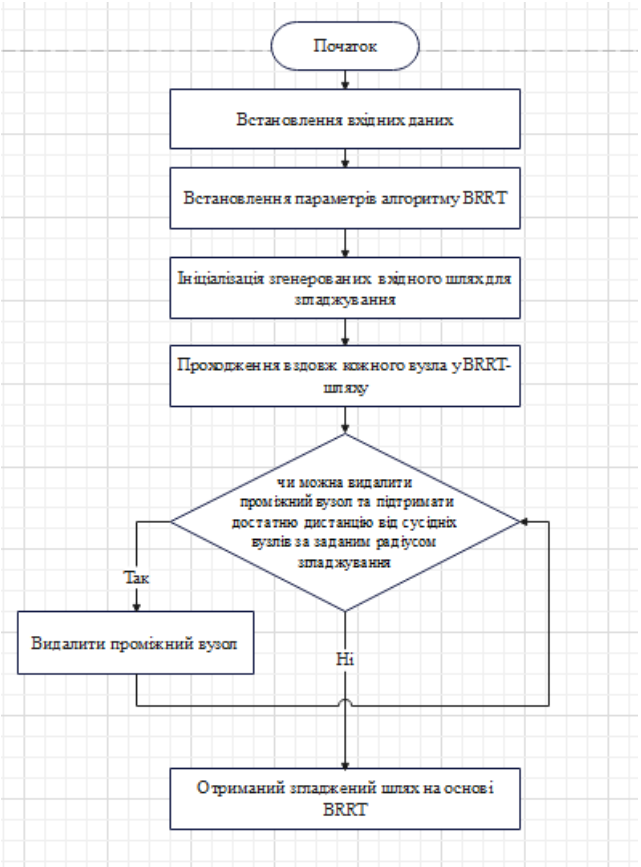

Рисунок 4.2 – Алгоритм згладжування шляхів

Розроблений алгоритм згладжування шляху на основі BRRT допоможе покращити якість шляху, видаляючи гострі кути та зменшуючи вибігання, що може поліпшити виконання руху для роботів або інших автономних систем, але важливо зберегти придатність шляху для реального виконання.

Даний алгоритм отримує початковий шлях, який був згенерований алгоритмом BRRT, далі встановлюються параметри алгоритму, зокрема радіус згладжування та кількість ітерацій алгоритму. Після цього ініціалізується згенерований BRRT-шлях як вхідний шлях для згладжування і для кожної ітерації виконуються наступні кроки: спочатку відбувається прохдження вздовж кожного вузла у BRRT-шляху, за винятком першого та останнього вузола, потім відбувається перевірка, чи можна видалити проміжний вузол та підтримати достатню дистанцію від сусідніх вузлів за заданим радіусом згладжування, якщо можна то видаляється проміжний

вузол і це повторюється певну кількість разів до тих пір, поки шлях не залишиться незмінним. І в кінці отримуємо згладжений шлях.

4.4 Розробка загального алгоритму роботи програми

Загальний алгоритм роботи програми використовує випадкову стратегію генерації точок для ефективного розведення простору станів та побудови маршруту від початкової до кінцевої точки, уникючи перешкод. Створений алгоритм спочатку ініцалізує середовище: задає розміри середовища, у розумінні ширини та висоти, встановлює початкову та кінцеву точку шляху, створює список перешкод та задає параметри алгоритму, такі як: кількість ітерацій та розмір кроку.

Потім використовує функції перевірки колізій, для забезпечення того, щоб нова точка, яка генерується під час кожної ітерації алгоритму, не знаходилася в області, яка зайнята перешкодами. Функція перевірки колізій гарантує, що лише безпечні точки додаються до дерева.

Далі йде функція пободови дерева, де спочатку створюється початкове дерево, що містить лише початкову точку, потім проводиться задана кількість ітерацій для побудови маршруту. На кожній ітерації генерується випадкова точка у серидовищі, після цього алгоритм знаходить найближчу точку в поточному дереві до випадкової точки.

Наступним кроком йде візуалізація середовища, де відображається початкова та кінцева точки, а також перешкоди у середовищі, далі візуалізується з'єднання між точками дерева. В кінці виводиться побудований маршрут.

На рисунку 4.3 представлено загальний алгоритм роботи програми.

Рисунок 4.3 – Загальний алгоритм роботи програми

4.5 Реализація коду

```
import numpy as np
import matplotlib.pyplot as plt

# Параметри середовища
width = 10
```

```
height = 10

# Початкова та кінцева точки
start_point = (1, 1)
goal_point = (8, 8)

# Генерація перешкод у середовищі
obstacles = [(3, 3), (4, 4), (5, 5)]

# Параметри алгоритму BRRT
num_iterations = 500
step_size = 1.0

# Функція для перевірки колізій із перешкодами
def is_collision_free(point):
    for obstacle in obstacles:
        if np.linalg.norm(np.array(point) - np.array(obstacle)) < 1.0:
            return False
    return True

# Функція для побудови маршруту за алгоритмом BRRT
def build_rrt(start, goal, num_iterations, step_size):
    tree = [start]

    for _ in range(num_iterations):
        random_point = np.random.rand(2) * np.array([width, height])
        nearest_point_index = np.argmin([np.linalg.norm(np.array(random_point) - np.array(point)) for point in tree])
```

```
            nearest_point = tree[nearest_point_index]

            new_point = nearest_point + step_size * (random_point - nearest_point)
            new_point = tuple(new_point)

            if is_collision_free(new_point):
                tree.append(new_point)

            # Візуалізація
            plt.plot([nearest_point[0], new_point[0]], [nearest_point[1], new_point[1]], color='blue', alpha=0.5)

        return tree

    # Візуалізація початкової та кінцевої точок, перешкод
    plt.scatter(*start_point, color='green', marker='o', label='Start')
    plt.scatter(*goal_point, color='red', marker='o', label='Goal')
    for obstacle in obstacles:
        plt.scatter(*obstacle, color='black', marker='x', label='Obstacle')

    # Побудова маршруту
    rrt_tree = build_rrt(start_point, goal_point, num_iterations, step_size)

    # Отображение графики
    plt.title('BRRT Path Planning')
    plt.legend()
    plt.grid(True)
    plt.show()
```

Результат роботи програми без оптимізації представлено на рисунку 4.4.

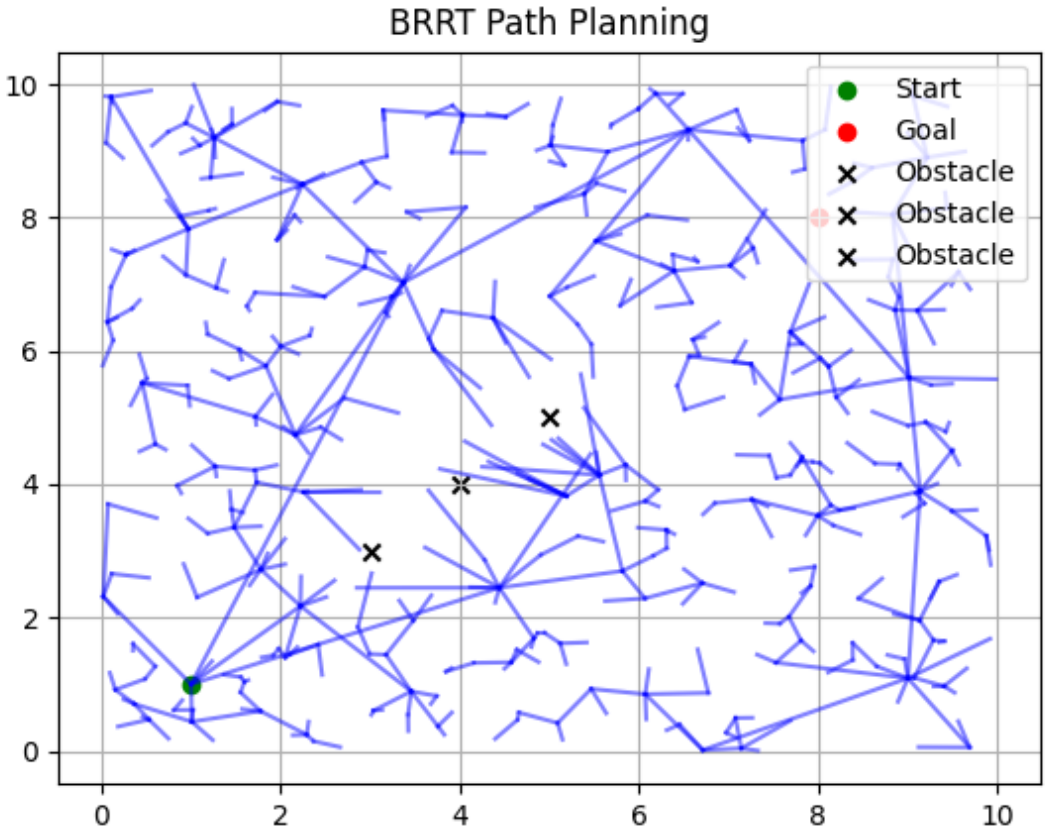

Рисунок 4.4 – Результат роботи програми без оптимізації

4.6 Проведення досліджень та аналіз отриманих результатів

Для проведення досліджень щодо побудови маршруту мобільного робота у двомірному просторі у розробленій програмі. Візьмемо карту розміром 512×512 пікселів, початок координат визначається у верхньому лівому кутку карти. Початкова точка відзначена кружком і розташована у точці (15, -30). Точка воріт відзначена ромбом і розташована в точці (270-212). Приклад карти на початок моделювання наведено рисунку 4.5.

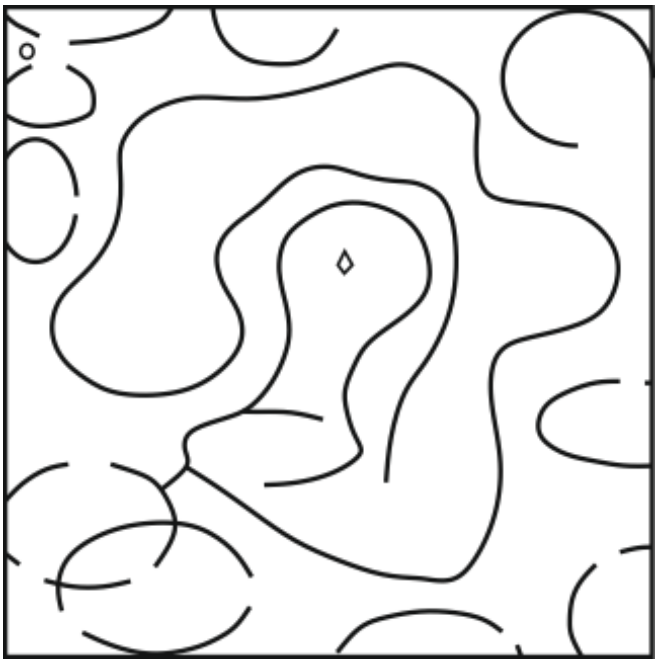

Рисунок 4.5 – Приклад двомірної карти до початку моделювання

Проведемо моделювання побудови маршруту – досягнення мобільного робота від початкової точки до точки виходу з використанням програмної реалізацій алгоритму BRRT. Отриманий результат моделювання з побудови маршруту представлено рисунку 4.6.

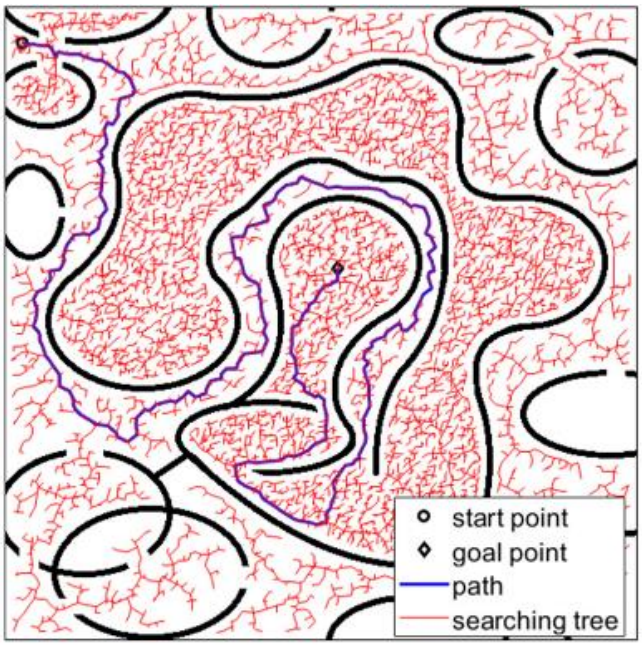

Рисунок 4.6 – Результат моделювання щодо побудови маршруту на базі алгоритму BRRT

Як можна побачити з отриманого результату рисунку 4.2, маршрут, запланований алгоритмом BRRT показаний синьою лінією, а дерево пошуку – червона лінія. Очевидно, що шлях включає безліч непотрібних поворотів.

Проведемо повторне дослідження побудови маршруту за однакових початкових умов, але при цьому застосуємо алгоритм, що згладжує, з підпункту 3.3. Результат моделювання з використанням алгоритму, що згладжує, наведено на рисунку 4.7.

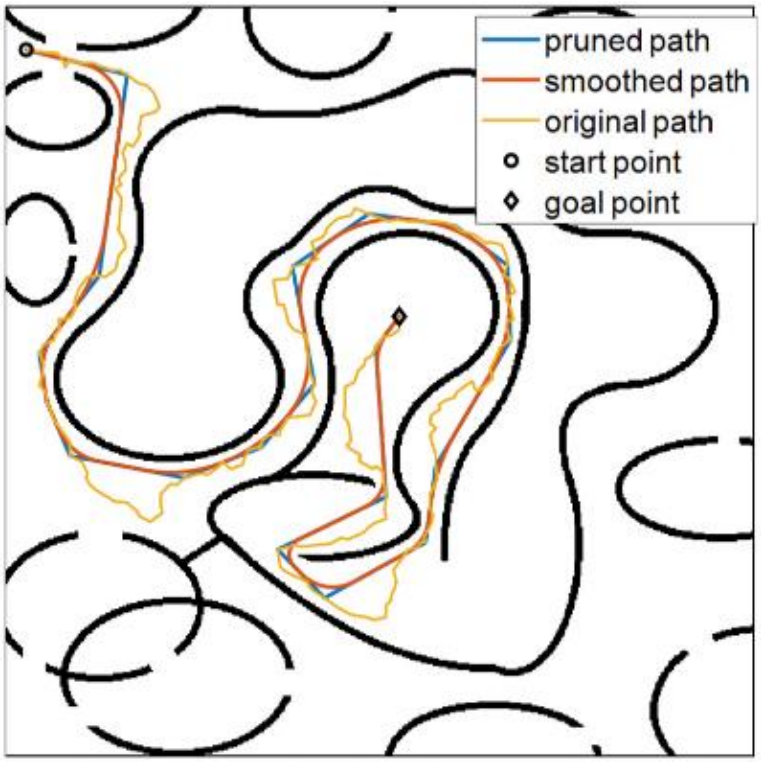

Рисунок 4.7 – Моделювання побудови маршруту рахунок застосування методу обрізання та згладжування алгоритму BRRT

Мінімальна та максимальна відстань висування BRRT встановлюється в діапазоні від 3 до 10. Лінійна та кутова швидкості згладженого шляху показані на рисунку 4.8.

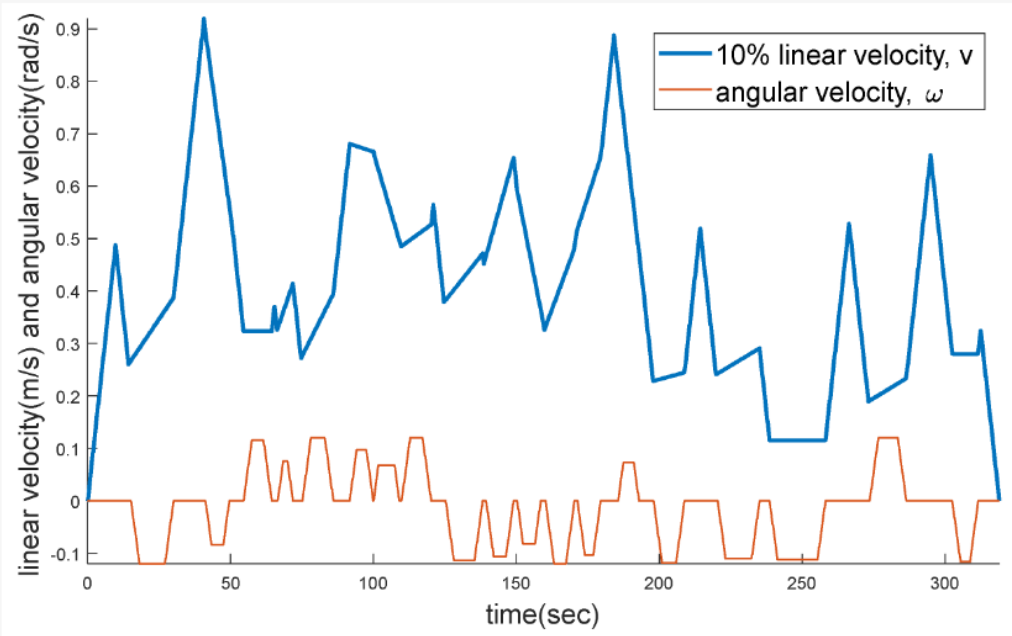

Рисунок 4.8 – Графік лінійної та кутової швидкості згладженої траєкторії

4.7 Охорона праці

Для проведення розрахунків освітлення лабораторії, важливо враховувати кілька ключових параметрів, таких як потужність світлових джерел, висота розташування світильників, коефіцієнт відбиття поверхні, та відстань між світильниками.

Визначимо вихідні параметри:

– потужність світлових джерел P = 4000 лм;

– площа лабораторії A = 100 м²;

– висота підвісу світильників H = 2,5 м;

– коефіцієнт відбиття стін та стелі ρ = 0,8;

– коефіцієнт витрати світлового потоку CU = 0,7.

Розрахуємо світловий поток $Flux$ за формулою:

$$P \ / \ A \ = \ 4000 \ / \ 100 \ = \ 40 \ лм / м^2. \qquad (4.1)$$

Витрата світлового потоку на висоті H :

$$Flux_H = Flux \cdot (H^2) / (H^2 + D^2), \quad (4.2)$$

де D – відстань між світильниками.

Загальна витрата світлового потоку:

$$Total\_Flux = Flux\_H \cdot CU \cdot \rho. \quad (4.3)$$

При умові, що відстань між світильниками (D) дорівнює 4 м, то світловий потік на висоті H:

$$Flux_H = 40 \cdot (2,5^2) / (2,5^2 + 4^2) \approx 12,17 \text{ лм/м}^2.$$

Потім обчислимо загальну витрату світлового потоку:

$$Total\_Flux = 12,17 \cdot 0,7 \cdot 0,8 \approx 6,79 \text{ лм/м}^2.$$

Отже, при заданих параметрах, у даній лабораторії буде приблизно 6,79 люменів світлового потоку на квадратний метр [17].

4.8 Висновки до 4 розділу

В ході написання четвертого розділу даної кваліфікаційної роботи, першим етапом для розробки на мові програмування Python було обрано середовище розробки PyCharm – потужний інструментарій, спеціально спроектований для полегшення роботи розробників Python у всіх аспектах процесу програмування. Наступним етапом було розроблено алгоритм обрізання шляхів, основною метою якого є усунення зайвих або непотрібних частин шляху, щоб покращити його ефективність та пристосованість до умов

середовища. Далі було розроблено алгоритм згладжування шляхів, який використовується для покращення якості маршрутів, зменшення кількості вузлів та уникнення непотрібних поворотів, зроблених алгоритмами пошуку шляху. Після було розроблено загальний алгоритм роботи програми та реалізовано код програми. Заключним етапом було проведення експериментальних досліджень та аналіз отриманих результатів.

ВИСНОВКИ

В ході написання даної кваліфікаційної роботи було проведено аналіз сучасних методів та алгоритмів побудови маршруту. Проаналізовано особливості побудови маршруту у двомірному просторі та алгоритм обрізання Greedy Algorithm. Наступним етапом було проведено математичний опис навколишнього середовища та мобільного робота. Удосконалено метод побудови маршруту мобільного робота у двомірних координатах. Далі було проведено аналіз та вибір алгоритму побудови шляху. Після проведеного аналізу було розроблено алгоритм обрізання шляхів на базі Greedy Algorithm та моделі згладжування шляху. Наступним етапом був вибір середовища розробки. Після було розроблено алгоритм обрізання шляхів, алгоритм згладжування шляхів та загальний алгоритм роботи програми. Реалізовано код програми та проведено експериментальні дослідження, проаналізовано отримані результати.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. ДСТУ 3008-15. Документація. Звіти у сфері науки та техніки. структура та правила оформлення. – Введ. 2015-06-22. – К. Держстандарт України, 2017 – 29 с.

2. Невлюдов, І.Ш. Дипломне проектування для студентів усіх форм навчання спеціальностей 151 «Автоматизація та комп'ютерно-інтегровані технології» [Текст]: навч. посіб. / І.Ш. Невлюдов, А.О. Андрусевич, О.В. Токарєва, Г.В. Пономарьова. – Київ-58, пр. Космонавта Комарова, 1, 2016. – 320с.

3. Методичні вказівки з підготовки та захисту кваліфікаційної роботи здобувачами другого (магістерського) рівня вищої освіти спеціальності 151 Автоматизація та комп'ютерно-інтегровані технології, освітньо-професійних програм: «Автоматизоване управління технологічними процесами»; «Комп'ютерно-інтегровані технологічні процеси і виробництва»; «Комп'ютеризовані та робототехнічні системи» / Упоряд.: І. Ш. Невлюдов Р. В. Артюх В. В. Безкоровайний Н. П. Демська В. В. Євсєєв О. І. Филипенко О. М. Цимбал. Харків: ХНУРЕ, 2021. 55 с.

4. Максимова С. Розробка програми для пошуку й побудови оптимального маршруту мобільного робота у двовимірному просторі / С. Максимова, Г. Борисов // Виробництво & Мехатронні Системи 2023 : тези доповідей VII-ої Міжнар. конф., 19-20 жовтня 2023 р. – Харків : ХНУРЕ, 2023. – С. 52–53.

5. Невлюдов І. Ш., Андрусевич А. О., Євсєєв В. В., Новоселов С. П., Демська Н. П. Проектування мобільних маніпуляційних роботів: Монографія. – Х. :, 2022. – 427 с.

6. Korkmaz, M., & Durdu, A. (2018, February). Comparison of optimal path planning algorithms. In *2018 14th International Conference on Advanced Trends*

in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET) (pp. 255-258). IEEE.

7. Zhang, Z., Schwartz, S., Wagner, L., & Miller, W. (2000). A greedy algorithm for aligning DNA sequences. *Journal of Computational biology*, *7*(1-2), 203-214.

8. Zhang, Y., & Gong, D. (2021, May). S-BRRT: A Spline-based Bidirectional RRT with Strategies under Nonholonomic Constraint. In 2021 33rd Chinese Control and Decision Conference (CCDC) (pp. 1753-1758). IEEE.

9. Viktoriia Bortnikova, Vladyslav Yevsieiev, Iryna Botsman, Igor Nevliudov, Kostiantyn Kolesnyk, Nazariy Jaworski. Queries classification using machine learning for implementation in intelligent manufacturing // Chapter 6 in Monograph «Methods and tools in CAD – selected issues». – Białystok (Poland): Publishing House of Bialystok University of Technology. – 2021. – PP. 63-74.

10. Моделі та методи кіберфізичних виробничих систем в концепції Industry 4.0: монографія / І. Ш. Невлюдов, В. В. Євсєєв, А. О. Андрусевич, С. С. Максимова; – Oktan Print – Prague. 2023. – 321 с.

11. Невлюдов І.Ш. Автоматизована система керування технологічними процесами в SCADA системі TRACE MODE 6: Навчальний посібник / І.Ш. Невлюдов, А.О. Андрусевич, В.В. Євсєєв, С.С. Максимова, М.Г. Стародубцев, В.В.Невлюдова. Кривий Ріг: Криворізький коледж НАУ, 2018. 320 с.

12. Бернацький, А. П. (2023). Метод планування шляху автономного наземного робота з використанням модифікації динамічного двонаправленого rrt-алгоритму. Системи і технології зв’язку, інформатизації та кібербезпеки.

13. Wang, J., Hirota, K., Wu, X., Dai, Y., & Jia, Z. (2021). Hybrid bidirectional rapidly exploring random tree path planning algorithm with reinforcement learning. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, *25*(1), 121-129.

14. Shafique, K., & Shah, M. (2005). A noniterative greedy algorithm for multiframe point correspondence. *IEEE transactions on pattern analysis and machine intelligence*, *27*(1), 51-65.

15. PyCharm IDE для професійної розробки на Python // PyCharm, 2023. URL: https://www.jetbrains.com/ru-ru/pycharm/ (дата звернення : 18.11.2023).

16. Ми створюємо наше програмне забезпечення, щоб ви могли насолоджуватися створенням свого // Jet brains, 2023. URL: https://www.jetbrains.com/ (дата звернення : 18.11.2023).

17. Охорона праці в офісі. Вимоги до робочого місця офісного працівника // Сайт GCC. URL: https://gc.ua/uk/oxorona-praci-v-ofisi-vimogi-do-robochogo-miscya-ofisnogo-pracivnika/ (дата звернення: 27.12.2023).