Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

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

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Розроблення системи автоматизації для визначення оптимального маршруту переміщення маніпуляційного робота на виробництві в умовах невизначеності
(тема)

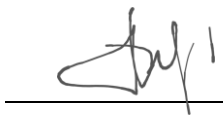
Виконав:
Здобувач 4 року навчання,
групи АКТАКІТ-21-3
Данило ГЕРАСИМЕНКО
(власне ім'я прізвище)
Спеціальності 151 Автоматизація та комп`ютерно-інтегровані технології
(код і повна назва спеціальності)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма Автоматизація та комп`ютерно-інтегровані технології
(повна назва освітньої програми)
Керівник доцент Рауф АЛЛАХВЕРАНОВ
(посада, власне ім'я прізвище)

Допускається до захисту

Завідувач кафедри КІТАР _____ Ігор НЕВЛЮДОВ
(підпис) (власне ім'я прізвище)

2025 р.

Я, Герасименко Данило Євгенович, як здобувач вищої освіти ХНУРЕ, розумію та підтримую політику закладу з академічної доброчесності. Я не надавав і неодержував недозволену допомогу під час підготовки кваліфікаційної роботи. Я не використовував штучний інтелект для підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

«12» червня 2025 р.  Данило ГЕРАСИМЕНКО

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет Автоматики і комп'ютеризованих технологій
Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки
Рівень вищої освіти перший (бакалаврський)
Спеціальність 151 Автоматизація та комп`ютерно-інтегровані технології
Тип програми освітньо-професійна
Освітня програма Автоматизація та комп`ютерно-інтегровані технології
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри КІТАР _____
(підпис)
« 28 » квітня 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Герасименка Данилу Євгеновичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення системи автоматизації для визначення робота оптимального маршруту переміщення маніпуляційного на виробництві в умовах невизначеності
затверджена наказом по університету від " 19 " травня 2025р. № 390 Ст.
2. Термін подання здобувачем роботи " 25 " червня 2025р.
3. Вихідні дані до роботи 3.1 Мобільний робот TurtleBot 3 Burger;
3.2 Замкнуте невизначене середовище;
3.3 Хвильовий алгоритм;
3.4 Мова програмування – JavaScript;
3.5 Операційна система – Microsoft Windows 10;
3.5 Оформлення текстової документації – ДСТУ 3008-2015.

4. Перелік питань, що потрібно опрацювати в роботі 4.1 Вступ;
4.2 Аналіз технічного завдання та предметної області;
4.3 Розрахунок визначення оптимального маршруту переміщення маніпуляційного мобільного робота;
4.4 Розроблення програмного забезпечення оптимального маршруту переміщення маніпуляційного мобільного робота;
4.5 Охорона праці;
4.6 Висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Демонстраційний матеріал представлений у форматі презентації PowerPoint (*.ppt) – 18 с. формату А4

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

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 технічного завдання та предметної області	28.04 – 04.05.25	виконано
2	Розрахунок визначення оптимального маршруту переміщення маніпуляційного мобільного робота	05.05 – 14.05.25	виконано
3	Розроблення програмного забезпечення оптимального маршруту переміщення маніпуляційного мобільного робота	15.05 – 28.05.25	виконано
4	Охорона праці	29.05 – 11.06.25	виконано
5	Оформлення пояснювальної записки	12.06 – 15.06.25	виконано
6	Подання роботи на перевірку Інтернет-системою StrikePlagiarism	16.06 – 18.06.25	виконано
7	Подання роботи на рецензію	19.06 – 21.06.25	виконано
8	Подання роботи на підпис зав. кафедри	22.06 – 24.06.25	виконано
9	Подання кваліфікаційної роботи в ЕК	25.06.25	виконано

Дата видачі завдання 28.04.2025р.

Здобувач _____ Данило ГЕРАСИМЕНКО
(підпис)

Керівник роботи _____ доцент Рауф АЛЛАХВЕРАНОВ
(підпис) (посада, власне ім'я прізвище)

РЕФЕРАТ

Пояснювальна записка: 65 с., 1 табл., 16 рис., 2 дод., 20 джерел.

МОБІЛЬНИЙ РОБОТ, МАНІПУЛЯТОР, СИСТЕМА КЕРУВАННЯ, НЕВИЗНАЧЕНЕ СЕРЕДОВИЩЕ, ОПТИМАЛЬНИЙ МАРШРУТ, ХВИЛЬОВИЙ АЛГОРИТМ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ.

Мета роботи – створення програмно-апаратного комплексу для демонстрації функціонування системи визначення оптимального маршруту маніпуляційного робота в умовах імітованої невизначеності.

Об'єкт розробки – переміщення маніпуляційного мобільного робота в невизначеному середовищі.

Предмет розробки – процеси коригування траєкторій руху мобільних роботів у динамічному виробничому просторі.

В кваліфікаційній роботі було проаналізовано види маніпуляційних мобільних роботів та методи вибору траєкторії оптимального маршруту. Для подальших досліджень за темою роботи було обрано мобільний робот TurtleBot 3 Burger. Для імітації реальних умов було створено детерміноване середовище. Програмно реалізовано хвильовий алгоритм, за яким розраховується і будується траєкторія оптимального маршруту робота. Розроблено програмне забезпечення обчислення траєкторії оптимального маршруту маніпуляційного мобільного робота в умовах невизначеного середовища на виробництві.

Отримані результати роботи можна віднести до Цілі сталого розвитку 9 «Промисловість, інновації та інфраструктура», зокрема до пункту 9.4 «Розвиток високотехнологічного машинобудування».

ABSTRACT

Explanatory note: 65 pp., 1 tab., 16 figs., 2 appendices, 20 sources.

MOBILE ROBOT, MANIPULATOR, CONTROL SYSTEM, UNCERTAIN ENVIRONMENT, OPTIMAL ROUTE, WAVE ALGORITHM, SOFTWARE.

Purpose – to develop software for determining the optimal route of movement of a manipulative mobile robot in an uncertain environment of a production facility

Object of development – movement of a manipulative mobile robot in an uncertain environment.

Subject of development – software for calculating the optimal route for moving a manipulative mobile robot in an uncertain environment.

The qualification work analyzed the types of manipulative mobile robots and methods for choosing the optimal route trajectory. The TurtleBot 3 Burger mobile robot was chosen for further research on the topic. To simulate real conditions, a deterministic environment was created. A wave algorithm was programmatically implemented, which calculates and builds the trajectory of the optimal route of the robot. Software for calculating the trajectory of the optimal route of a manipulative mobile robot in an uncertain environment at work has been developed.

The results of the work can be attributed to Sustainable Development Goal 9 “Industry, Innovation and Infrastructure”, in particular to paragraph 9.4 “Development of high-tech engineering”.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БД – база даних;

ПЗ – програмне забезпечення;

ПК – персональний комп’ютер;

РТС – робототехнічна система;

СК – система керування.

ВСТУП

Одним із важливих напрямків у галузі машинобудування визначимо напрямок зі створення автоматизованих систем для допоміжного виробництва. Особливого значення в галузі автоматизованих транспортних систем набуває розроблення наукових основ проєктування мобільних транспортних роботів і роботизованих транспортних комплексів різного призначення, а також їхня відповідність технологічним і конструктивним вимогам.

Однією з переваг автоматизації транспортних засобів визначимо як скорочення часу очікування вантажів на робочих місцях, а також скорочення витрат енергії шляхом ефективного планування маршрутів оптимального маршруту. Таким чином, у системі керування мобільним транспортним роботом одним із ключових засобів є алгоритм прокладання траси, мета якого полягає у формуванні маршруту переміщення робота зовнішнім середовищем.

Так, мова йде про те, що мобільному роботу, котрий долає перешкоди, необхідно взяти вантаж і перемістити його в зазначене місце за мінімальний пройдений шлях або за мінімальний час проходження траєкторії. Втім мобільний робот піддається впливу перешкод, як-от діям зовнішніх сил, які відхиляють його від заданої траєкторії на кшталт непрохідних ділянок. Наступна складність пов'язана з тим, що моделюється рух не ідеального, а інерційного робота. Таким чином, під час оптимального маршруту нелінійним маршрутом буде простежуватись відхилення реальної траєкторії від заданої.

З метою успішної навігації в просторі система мобільного робота повинна вміти будувати та керувати параметрами оптимального маршруту (задавати кут повороту коліс і швидкість їхнього обертання), правильно

інтерпретувати інформацію про навколишнє середовище, що надходить від сенсорів, і весь час відстежувати власні координати.

Об'єкт розробки – переміщення маніпуляційного мобільного робота в невизначеному середовищі.

Предмет розробки – програмне забезпечення обчислення оптимального маршруту переміщення маніпуляційного мобільного робота в умовах невизначеного середовища.

Мета роботи – розроблення програмного забезпечення для визначення оптимального маршруту переміщення маніпуляційного мобільного робота в невизначеному середовищі виробничого приміщення.

Для досягнення поставленої мети необхідно виконати такі завдання:

− вивчити алгоритми оптимального маршруту пермищення мобільних роботів;

− вибрати мобільного робота з маніпулятором;

− побудувати детерміновану карту;

− проаналізувати хвильовий алгоритм;

− розробити програму визначення оптимального маршруту переміщення маніпуляційного мобільного робота в невизначеному середовище;

− опрацювати питання, що пов’язані з охороною праці;

– оформити пояснювальну записку згідно з ДСТУ 3008–15 [1] та послуговуючись з рекомендаціями з підготовки і оформлення кваліфікаційної роботи здобувачами першого (бакалаврського) рівня вищої освіти [2-3], отримані результати роботи можна віднести до Цілі сталого розвитку 9 «Промисловість, інновації та інфраструктура», зокрема до пункту 9.4 «Розвиток високотехнологічного машинобудування».

1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ ТА ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз видів маніпуляційних мобільних роботів

З’явилися нові методи щодо керування та навігації мобільними роботами. У зв’язку з цим діяльність оператора, що керує роботом, полягає у спостереженні власне за роботом та формулювання різних завдань. Водночас додається зворотний зв'язок, зокрема мовленнєвий. З огляду на, це процес керування стає схожим на діалог між людиною та роботом, що супроводжується графічними та мовленнєвими повідомленнями. Найважливішу роль набуває система навігації, тому що робот повинен самостійно оцінювати навколишнє середовище, а також планувати свій шлях, зокрема у разі присутності інших рухомих об'єктів у робочій зоні. Із появою автоматичних розв’язань таких задач істотно полегшилися завдання людини, що керує роботом, але для цього потрібно розроблення «інтелектуальної» системи керування роботом. До списку таких завдань можна віднести і формування задачі щодо автоматичного повернення робота у разі втрачання зв'язку з оператором, розв’язання якої підвищує надійність робототехнічної системи [4].

Крім того, змінюється і характер діяльності людини, котра керує роботами. Так, вона перестає керувати безпосередньо рухами робота, а починає вказувати лише на завдання, що повинен робот розв’язати. Усе це власне змінює і характер системи керування, оскільки повинні враховуватися можливості сприйняття оператора й особливості прийнятих ним рішень. Для того, щоб розв’язати такі завдання, доцільно послуговуватися нечіткою логікою як на етапі сприйняття інформації, так і на етапі планування дій разом із етапом приймання оперативних рішень. За допомогою використання просторово-часових відносин і лінгвістичних змінних діалог між людиною, що керує роботом, і власне роботом реалізується ще більше. Це дає

можливість говорити про те, що робототехнічні системи можуть бути системами кооперативного керування.

На практиці багато завдань, на кшталт звичайного моніторингу місцевості або радіаційної та хімічної розвідки, розв'язують шляхом застосування автономних мобільних роботів. До того ж деякі роботи призначені для боротьби з пожежами та стихійними лихами, отже, вони можуть брати участь у групових завданнях.

Застосування маніпуляційних мобільних роботів стало нагальним і для промислової галузі. Якщо послуговуватись цими роботами спільно з іншими автоматизованими засобами, то це сприятиме економічному зростанню підприємства. Наразі маніпуляційні мобільні роботи є одним із важливих компонентів автоматизованого виробництва. Такі роботи активно функціонують у гнучкому автоматизованому виробництві, тобто збільшують продуктивність праці, але забезпечують незмінний рівень якості [4].

На промислових виробництвах роботів можуть залучати до різних видів робіт, як правило, не виходячи за межі типових проєктів. Так, для робота вибирають конкретні завдання, з огляду на його технічні характеристики. Крім того, визначають кількість роботів для виконання певних завдань, вирішуються питання щодо інфраструктури живлення (силові підводки, подача охолоджувальної рідини – в разі застосування рідинного охолодження елементів устаткування), а також їхньої інтеграції до виробничого процесу (забезпечення заготовками, напівфабрикатами і повернення готового продукту до автоматичної лінії для виконання наступної технологічної операції).

1.1.1 Маніпуляційний мобільний робот iRobot Warrior

Велика кількість роботів розроблена для військових цілей. Так, iRobot Warrior (рисунок 1.1) є роботизованою платформою, розробленою компанією iRobot, з метою переміщення потенційно небезпечних предметів (як-от снарядів) без загрози для людини [5].

Рисунок 1.1 – Маніпуляційний мобільний робот iRobot Warrior

Крім того, цього робота можна залучати для розчищення шляху, гасіння пожежі чи розвідки. Водночас за допомогою роботизованої «руки» можна відтягувати поранених солдат із поля бою, вхопившись за їхній одяг. Усе це можливо, оскільки робот устаковано відеокамерами та датчики. Вага платформи без маніпулятора становить 165,6 кг, а з маніпулятором – 226,8 кг. Максимальна відстань, на яку маніпулятор може витягнутися, становитиме до 192,2 см, тримати вантаж у висячому положенні – до 31,6 кг. Однак у закритому положенні маніпулятор може утримувати вантаж до 136,1 кг. До того ж робот спроможний долати перешкоди заввишки до 47 см і рухатися під кутом нахилу до 45º. Заряд батареї дозволяє працювати від 4 до 10 годин. Оператор може ним керувати на відстані до 800 м [5].

Warrior значно відрізняється від інших військових роботів, які можуть виконувати лише ті дії, що задаються оператором. У ньому присутня функція, що у разі втрати зв'язку задає шлях до місця, на якому може відновитися сеанс зв'язку. Warrior передає оператору, котрий ним керує, аудіо- та відеоінформацію через радіоканал [5].

1.1.2 Маніпуляційний мобільний робот KUKA KMR iiwa

Маніпуляційні мобільні роботи у виробничому процесі здатні реалізовувати ключові та допоміжні технологічні операції. До ключових технологічних операцій належать операції безпосереднього виконання формоутворення, зміни лінійних розмірів заготовки тощо. Тоді як до допоміжних технологічних операцій належать транспортні операції, зокрема операції із завантаження та вивантаження технологічного обладнання. Як приклад одного із таких роботів наведемо KUKA KMR iiwa (рисунок 1.2) [6].

Рисунок 1.2 – Маніпуляційний мобільний робот KUKA KMR iiwa

Маніпуляційний робот KMR iiwa розроблений для співпраці з людиною і є мобільним. В одній системі він поєднує сильні сторони чутливого робота легкої конструкції LBR iiwa і мобільної автономної платформи. KMR iiwa не прив'язаний до місця і характеризується своєю гнучкістю – ідеальна умова для виконання різних вимог [6].

До його щоденних обов'язків належить розподіл гвинтів, кілець ущільнювачів, гайок і інших дрібних деталей. Так, наприклад, замовлення, тобто бокси, виставляються на центральний складський стелаж, а KMR iiwa через однакові проміжки часу перевіряє окремі полиці і забирає доставлені коробки з деталями.

Власне реалізується даний процес за таким принципом: робот бере коробку і підносить її до сканера QR-коду, котрий заздалегідь вбудовано в платформу, – так він розпізнає, куди необхідно її доставити. Потім автономна платформа пересувається, тобто перевозить коробку через виробничий цех і автоматично подає на робоче місце [6].

Зауважимо, що розподіл дрібних деталей шляхом залучення роботизованих систем вже набув широкого застосування, на відміну від галузі виробництва й оброблення великих деталей, де виснажливі операції в незручному положенні (як-от монтаж над головою) створюють певну незручність для робітників, тут автономні помічники тільки почали набирати популярність [6].

Мобільні роботи рухаються вздовж виробничих ліній, тому легко забезпечують швидку зміну окремих видів продукції. Крім того, безпечні помічники устатковані лазерними датчиками, завдяки яким розпізнають людей та предмети, що з’являються на шляху. Кожен датчик забезпечує кут зору 220°, щоб система змогла успішно подолати як стаціонарні, так і рухомі перешкоди. В майбутньому планується вдосконалити роботів, щоб вони могли перевозити вантажі з місця на місце [6].

Швидкість реакції реалізується завдяки датчикам шарнірних моментів. Так, KUKA KBR iiwa миттєво розпізнає контакт, зменшуючи одночасно зусилля і швидкість. Шляхом регулювання положення та гнучкості робот маніпулює чутливими деталями без защемлення і зрізів.

Робот із високоефективним регулюванням. KBR iiwa швидко розпізнає контури, регулюючи зусилля. Він фіксує правильне положення монтажу і водночас швидко і точно встановлює деталі з точністю моментів за осями ±2 % від максимального моменту. KBR iiwa миттєво знаходить дрібні філігранні деталі, без сторонньої допомоги.

Самостійність. Система керування KUKA Sunrise Cabinet роботом KBR iiwa спрощує швидке введення в експлуатацію навіть у межах складних

завдань. Роботу можна делегувати незручні та монотонні завдання, які він виконає надійно і без сторонньої допомоги.

Щоб мобільні роботи могли більш тісно взаємодіяти між собою і реалізовувати на виробництві більше операцій, потрібно базуватися не лише на розробленні стандартів. Швидкість розповсюдження мобільних роботів на виробництві залежить від таких чинників: засобів розпізнавання середовища, оброблення даних, систем безпеки, а також передусім від визнання і прийняття людьми таких роботів.

1.1.3 Маніпуляційний мобільний робот на платформі OTTO 1500

OTTO Motors спільно з Yaskawa Motoman розробили робота для транспортування важких вантажів, який обладнали автономною платформою з вантажопідйомністю до 1500 кг. Напрямки застосування майже безмежні. Складається робот із мобільної платформи OTTO 1500, промислового маніпулятора Motoman MH12, роботизованого захвату Robotiq 2-Finger 85 Gripper (рисунок 1.3) [7].

Рисунок 1.3 – Маніпуляційний мобільний робот
на платформі OTTO 1500

Усі ці механічні пристрої були з'єднані воєдино, в результаті чого розроблений робот містить сучасні технології. Завдяки безконтактним датчикам і системам планування шляху, OTTO 1500 може рухатися виробничим майданчиком або складом без ризику нанесення травм працівникам. Маніпулятор реалізовано 6-осьовим роботом Motoman MH12 із максимальною досяжністю 1,4 м і 12 кг корисного навантаження [7].

Роботизований захват Robotiq 2-Finger 85 Gripper дозволяє роботу виконувати різні завдання. Так, він може механічно пристосуватися до роботи з будь-якими об'єктами. Характеризується він такими технічними параметрами:

– габаритні розміри – 1190×1810×400 мм;

– корисне навантаження – 1500 кг;

– максимальна швидкість – 2,0 м/с;

– номінальний безперервний час автономної роботи – 6 год;

– просвіт – 16 мм;

– спроможність підніматися по нахилу до 5 %;

– радіус повороту дорівнює 0;

– точність позиціонування – +/- 25 мм.

Серед завдань для розробників актуальним було створення дуже надійних систем навігації та картографування, котрі переважно покладаються на лазерні датчики. Крім того, для навігації та картографії застосовані технології одометра та інерційних даних. До того ж у подібних системах багато компаній використовують камери. На сьогодні відеосистеми пройшли довгий шлях розвитку і відіграватимуть важливу роль в майбутньому теж. Однак наразі для розв’язання навігаційних завдань OTTO покладається здебільшого на лазерні датчики [7].

1.1.4 Маніпуляційний мобільний робот TurtleBot 3 Burger

Як приклад мобільного робота було обрано TurtleBot 3 Burger (розроблений Interbotix Labs та Open Source Robotics Foundation), зовнішній вигляд якого продемонстровано на рисунку 1.4 [8].

Рисунок 1.4 – Маніпуляційний мобільний робот TurtleBot 3 Burger

Його відмінна особливість характеризується маніпулятором Pincher MK3 – роботизованою рукою з 5 ступенями свободи. Це вперше, коли робот TurtleBot пропонує вбудовану підтримку роботизованого маніпулятора. Його перевага полягає в круглій формі, що дозволяє роботу в обмеженому просторі розгортатися на місці, не зачіпаючи корпусом стіни [9].

Робот спрямований на виконання рухомих дій і функцій керування в процесі маніпуляційних робіт. Іншими словами, це автоматичний пристрій, що складається з маніпулятора і перепрограмованого пристрою керування, що формує керувальний вплив, тобто задають необхідні рухи виконавчим органам маніпулятора. До того ж, він, як і робот KUKA KMR iiwa, призначений для розподілу гвинтів, кілець-ущільнювачів, гайок та інших дрібних деталей.

Програмному забезпеченню (ПЗ) і СК робота властива модульна структура, у зв’язку з цим допускається модернізація та розширення в частині доробок, забезпечення завадостійкості, тестування підвищення надійності, самодіагностики, а також виконання додаткових функцій і поліпшення інших тактико-технічних характеристик [9].

За допомогою автономної СК реалізується керування роботом, тобто персональним комп'ютером. СК має зв'язок з підсистемами датчиків, зв'язку та керування.

Платформа TurtleBot завжди реалізувала своє призначення через доведення винахідникам і розробникам у більш доступному форматі таких складних технологій на кшталт автономної навігації та роботизованої маніпуляції. TurtleBot 3 Burger володіє багатьма можливостями, зокрема застосовує перероблене шасі, має вбудовану підтримку роботизованих маніпуляторів та обчислювальний модуль IntelJoule 570x [9].

TurtleBot 3 Burger як стандартна опція властивий роботизований маніпулятор Pincher MK3 5 DOF. Це перший робот TurtleBot зі стандартною підтримкою маніпулятора, де MK3 дозволяє TurtleBot 3 Burger перекладати предмети й інструменти в природному середовищі. Контролер Arbotix забезпечує інтерфейс для Pincher MK3. Це дозволяє керувати маніпулятором шляхом високорівневих команд [9].

TurtleBot 3 Burger має в якості стандартної опції роботизований маніпулятор Pincher MK3 5 DOF. Це перший робот TurtleBot зі стандартною підтримкою маніпулятора. MK3 дозволяє TurtleBot 3 Burger перекладати предмети і інструменти в природному середовищі. Контролер Arbotix забезпечує інтерфейс для Pincher MK3, що дозволяє управляти маніпулятором за допомогою високорівневих команд [9].

Також, в TurtleBot 3 Burger застосовується модуль обчислень IntelJoule 570x, який дозволяє легко інтегрувати до TurtleBot 2i дві камери RealSense 3D. Так, камера Intel ZR300 RealSense застосовується для навігації і

картографії, водночас камера ближнього радіусу дії SR300 RealSense допомагає керувати маніпулятором MK3 [9].

Постійно зростаючим вимогам розробників робототехніки сьогодні вже не відповідають наявні на ринку готові плати для їхніх потреб. Так, розробники нових роботизованих систем очікують більш продуктивні та енергоефективні рішення для вдосконалення роботів, які спроможні розв’язувати складні завдання, як-от: одночасну просторову локалізацію та картографування (3DSLAM), розпізнавання об'єктів і стеження, ідентифікація осіб і голосу [9].

Поєднання Joule 570x і RealSense також реалізують апаратне прискорення для вбудованої графіки IntelIris. Попередні версії TurtleBot застосовували для обчислень нетбуки чи ноутбуки [9].

Основні можливості:

– програмне забезпечення TurtleBot 3 Burger ROS/демо-можливості;

– автономна навігація завдяки точковій хмарі;

– ідентифікація зони карти і шляхових точок;

– маніпуляція і сортування об'єктів роботизованою рукою Pincher MK3 RoboArm;

– запобігання перешкод і планування шляху;

– дистанційне управління;

– автономне підзаряджання з док-станції;

– контролер Arbotix-M.

Апаратні засоби:

– CPU – Intel Joule 570X;

– платаGumstixNodana;

– 4 GB RAM;

– 16 GB eMMC;

– 802.11AC WiFi / Bluetooth 4.0;

– ОС Ubuntu 16.04 / ROS Kinetic;

– сенсори: SR300 RealSense 3D камера, ZR300 RealSense 3D камера;

– акселерометр, гіроскоп, компас;

– сенсори країв і перешкод.

Технічні характеристики:

– батареї – 3S LiPo 4500mAh;

– максимальна поступальна швидкість – 70 см/с;

– максимальна швидкість обертання – 180 град/с (гіроскопічна характеристика > 110 град/с буде погіршуватися);

– корисне навантаження – 2 кг (без руки), 1 кг (з ручним приводом);

– подолання порогів – 12 мм;

– очікуваний час роботи – від 4 до 6 год. (час роботи залежить від навантаження);

– час заряджання – від 2 до 3 год.

СК мобільним роботом має розв’язувати такі завдання:

− оброблення сенсорних даних (даних від інтерфейсу з оператором) з метою збирання інформації про роботу і зовнішнє середовище навколо нього;

− планування заходів щодо окреслення цільового завдання і планування послідовності підзадач, які необхідні для виконання цього завдання;

− формування таких програмних траєкторій оптимального маршруту МР, які б призводили до виконання роботом локальної підзадачі (наприклад, прибуття до цільової точки в середовищі з перешкодами);

– формування таких вхідних впливів на виконавчі механізми робота, що призводили б до максимально точної та швидкої реалізації ними програмної траєкторії оптимального маршруту.

1.2 Аналіз методів вибору оптимального маршруту переміщення

Сучасна робототехніка набула розповсюдження як відповідь на запити комплексної автоматизації. Так, в результаті поєднання маніпуляторів, керованих людиною, із системами ЧПУ верстатів та іншого технологічного

обладнання виникли автоматичні машини принципово нового типу. Мова йде про роботів із програмним керуванням, тобто роботів першого покоління.

Успіхи функціонування перших роботів посприяли швидкому зростанню потреб у них, а, отже, і вимог до їхніх можливостей. Активного розвитку набули роботи з комбінованим керуванням, в яких програмне керування доповнюється керуванням від людини-оператора. Так, виникли роботи проміжного 1,5-го покоління з супервізорним, а в подальшому й інтерактивним керуванням.

Водночас із першими кроками втілення теорії адаптивного керування почали з’явилися і перші адаптивні роботи. Це роботи вже другого покоління, котрі були устатковані сенсорами.

Зважаючи на темпи розвитку систем адаптивного керування, поступово стали застосовувати в них методи штучного інтелекту. Коли ці технології посіли визначальне місце в алгоритмічному забезпеченні систем керування, було сформовано нове, третє покоління роботів – інтелектуальні.

Серед найпоширеніших алгоритмів вибору траєкторії оптимального маршруту мобільного робота можна виокремити такі:

- алгоритм пошуку A*;
- алгоритм Дейкстри;
- хвильовий алгоритм;
- маршрутний алгоритм;
- навігаційна сітка;
- ієрархічні алгоритми;
- обхід перешкод;
- розділяй і володарюй;
- алгоритм повороту Креша.

Пошук A* є алгоритмом пошуку за першим найкращим збігом на графі, який знаходить маршрут із найкоротшим шляхом від початкової вершини до кінцевої. Алгоритм A* набуває повного значення, що завжди

знаходить рішення, якщо таке існує. Узагальненням для нього стає двонаправлений евристичний алгоритм пошуку [10].

У зазначеному алгоритмі порядок обходу вершин доцільно визначати евристичною функцією «відстань+ оцінювання». Така функція є поєднанням двох інших: функції оцінювання досягнення даної вершини з початкової клітинки, і функції евристичної оцінки відстані від розглянутої вершини до кінцевої.

До цільової вершини не є потрібним, щоб функція переоцінювала відстань. Наприклад, для задавання маршрутизації може бути відстань до цілі по прямій лінії, оскільки фізично це найменша можлива відстань між двома точками [10].

Пошук A* покроково опрацьовує всі шляхи, що ведуть від початкової вершини до кінцевої, поки не знайде мінімальний. За аналогією до всіх поінформованих алгоритмів пошуку, спочатку він переглядає ті маршрути, які «здаються» ведуть до мети. Цей алгоритм обчислює значення шляху від початкової вершини, а не від попередньої. Під час вибору вершини він враховує, до того ж, увесь пройдений до неї шлях [10].

На початку роботи простежуються вузли, суміжні з початковим, і вибирається з них той, який має мінімальне значення вартості, після чого цей вузол розкривається. На кожному етапі алгоритм опрацьовує безліч шляхів із початкової точки до всіх ще не розкритих вершин графа – безліччю приватних рішень, що розміщується в черзі з пріоритетом. Пріоритет шляху можна визначити як суму значення досягнення з початкової вершини до кінцевої. Алгоритм продовжує працювати доти, поки значення оцінювання цільової вершини не виявиться меншим за будь-яке значення в черзі, або поки все дерево не буде переглянуто. Серед безлічі рішень вибирається рішення з найменшим оцінюванням [10].

Чим меншою буде евристика функції кінцевої вершини, тим більшим стане пріоритет. Отже, для реалізації черги можна застосовувати сортувальні дерева. Припустимо, що зі свого боку алгоритм A* обходить мінімальну

кількість вершин, оскільки він працює з «оптимістичним» значенням шляху через вершину. Так, про оптимістичність йдеться в тому разі, що, якщо він піде через цю вершину, то в алгоритму «є шанс», що реальне значення результату дорівнюватиме цьому значенню, проте ніяк не менше. Втім, з огляду на те, що A* є поінформованим алгоритмом, то зазначена рівність може бути цілком можливою. Коли A* завершує пошук, то алгоритм, згідно з визначенням, знаходить шлях, справжнє значення якого менше за оцінювання значення будь-якого шляху через будь-який відкритий вузол. Однак через те, що ці позначення є оптимістичними, то відповідні вузли можна без сумнівів відкинути. Перефразуємо, A* ніколи не втратить можливості мінімізувати довжину шляху, отже, є допустимим [10].

Алгоритм Дейкстри знаходить всі найкоротші шляхи з від початкової точки до всіх інших. Послуговуючись усією необхідною інформацією можна дізнатися як швидше дістатися з місця старту до фінішу. Проте у цього методу є й недолік – не може обробляти графи з ребрами, які мають негативне значення.

Покрокова робота алгоритму полягає в тому, що на кожному кроці він «відвідує» одну вершину і намагається вибрати таку, щоб значення зменшувалося. Робота алгоритму завершується, коли всі вершини опрацьовані. В іншому разі, з вершин, які ще не розглянуті, вибирається та, що має мінімальну позначку. Вивчаються всілякі маршрути, в яких вершини є передостаннім пунктом. Так, вершини, до яких ведуть ребра з невідвіданих вершин, визначимо як сусіди цієї вершини. Для кожного сусіда вершини, крім позначених відвіданих, розглядається нова довжину шляху, що дорівнює сумі значень поточного позначення і довжини ребра, що з'єднує вершини з цим сусідом. Якщо отримане значення довжини менше за значення, що позначає сусіда, то отримане значення позначки заміняється значенням довжини. Опрацювавши всіх сусідів, вершину позначають як відвідану, а крок повторюється до завершення алгоритму [11].

Зі свого боку хвильовий алгоритм шукає найкоротший шлях на планарному графі. Він належить до алгоритмів, які ґрунтуються на методах пошуку в ширину. Так, алгоритм дискретизує робоче поле, що набуває вигляду обмеженої замкнутої лінії фігури. Зазначена фігура не обов'язково повинна бути прямокутної форми, розбитої на прямокутні комірки. Усі комірки розподіляються на стартову і фінішну, а також комірки прохідні та непрохідні [11].

Алгоритм призначений для пошуку найкоротшого шляху від стартової комірки до фінішної, у разі, якщо це можливо, проте у разі відсутності шляху, з’являється повідомлення про непрохідність. Власне робота алгоритму складається з трьох етапів: ініціалізації, поширення хвилі та відновлення шляху.

Під час ініціалізації будується зображення безлічі комірок оброблюваного поля, кожній з яких приписуються атрибути прохідності чи непрохідності, до того ж запам'ятовуються стартова і фінішна комірки.

Далі від стартової комірки прокладається шлях до сусідньої, водночас перевіряється, чи вона прохідна, і чи не належить раніше поміченому на шляху осередку. Сусідні комірки прийнято класифікувати подвійно, як-от: околиці Мура і околиці фон Неймана. Відмінність між ними полягає в тому, що в околиці фон Неймана сусідніми вважаються тільки 4 комірки по вертикалі і горизонталі, а в околиці Мура – всі 8 комірок, зокрема і діагональні. У разі виконання умов прохідності і неналежності її до комірок, раніше позначених на шляху, атрибут комірки записується як число, що дорівнює кількості кроків від стартової комірки. Таким чином, від стартової комірки на першому кроці буде 1. Кожна клітинка, що позначається числом кроків від стартової комірки, стає стартовою і від неї спрямовуються чергові кроки до сусідніх комірок. Зрозуміло, що за таких дій буде знайдено шлях від початкової комірки до кінцевої, до того ж виконати черговий крок з будь-якої створеної на шляху комірки буде неможливо [11].

Відновлення найкоротшого шляху реалізується в зворотному напрямку. Під час вибору комірки, від фінішної до стартової, на кожному кроці вибирається комірка, що містить атрибут відстані від стартової на одиницю менше за поточну комірку. Вірогідно, що таким чином знаходиться найкоротший шлях між парою заданих комірок. Однак може бути кілька мінімальних числових довжин шляху, за аналогією до пошуку шляху як в околицях Мура, так і фон Неймана. Вибір остаточного шляху в застосунках обґрунтовано іншими міркуваннями, що знаходяться поза межами цього алгоритму.

Маршрутний алгоритм отримав свою назву, оскільки одночасно втілює як формування фронту, так і прокладання траси. Джерело хвилі на кожному кроці визначається як кінцевий елемент ділянки траси, що було прокладено на попередніх кроках.

Так, у маршрутному алгоритмі розглядаються восьмиелементні околиці вихідного елемента. Від кожного елемента оточення оцінюється відстань до кінцевого елемента. Таким чином, прораховується вісім значень відстаней, з яких вибирається мінімальне. Елемент, для якого відстань виявилася мінімальною, позначають як елемент траси. Процес повторюватиметься стільки, доки відстань не буде дорівнювати нулю, тобто поки кінцевого елемента не буде досягнуто. Втім обхід заборонених елементів виконується з огляду на інтуїцію розробника.

Навігаційна сітка застосовується під час пошуку шляху за повністю відомою місцевістю заздалегідь. Головна відмінність даного методу полягає у відсутності чіткої дискретизації цього простору пошуку, тоді як у попередніх методах простір замінюють вузли графів або комірки масивів. Практично це сприяє побудові більш реалістичних траєкторій оптимального маршруту.

Крім того, простір, який представлено навігаційною сіткою, може містити додаткову інформацію, що впливає на поведінку виконавця. Особливо корисно це під час конструювання штучного інтелекту,

призначення якого не обмежується пошуком шляху. Так, керівник штучним інтелектом здебільшого для оптимізації своєї діяльності потребує даних про тактичні характеристики простору. Зі свого боку, штучний інтелект, який автоматизує роботу, як правило послуговується навігаційною сіткою для структурування інформації про приміщення [11].

Зауважимо, що пошук точного найкоротшого шляху серед великих масштабів займає багато часу через велику кількість ребр, за якими потрібно цей пошук виконувати. З метою поліпшення швидкодії можна моделювати природну ієрархію, на кшталт у транспортній системі, де вибір автомагістралі краще за дороги місцевого значення. Після побудови ієрархічної мережі доцільно використати модифікований алгоритм Дейкстри для двох напрямків задля обчислення маршруту між вихідною та кінцевою точками. Загалом цей механізм розрахунку в ієрархічній мережі, де ребрам присвоюються вагові коефіцієнти на основі часу в дорозі, є цілком актуальним. Він імітує звичайні поїздки мережею автошляхів.

Загальна задача в цьому випадку окреслюється шляхом забезпечення присутності в мережі ієрархій більш високого порядку. Для розв’язання даного завдання виконується одночасний пошук з вихідного і кінцевого розташування, а також точок стикування або в'їзду на дороги верхнього рівня, після чого – пошук по дорогах верхнього рівня, поки не зустрінуться сегменти з джерела та призначення. Оскільки пошук обмежується ієрархією верхнього рівня, то він опрацьовує меншу кількість ребр, що сприяє підвищенню швидкодії. Зауважимо, що мова йде про евристичний алгоритм і його мета полягає у високій швидкодії та продуктивному рішенні, проте він не гарантує, що найкоротший шлях буде знайдено. Задля успішної роботи евристичного алгоритму слід підключити ієрархію вищого рівня, у зв’язку з тим, що він не переходить на рівень нижче у разі досягнення тупикового кінця.

Розділяй і володарюй – це важлива парадигма розроблення алгоритмів, мета якої полягає в рекурсивному розбитті розв'язуваної задачі на дві або

більше підзадач того ж типу, проте меншого розміру, а також комбінуванні їхніх розв’язань для отримання відповіді до вихідної задачі. Розбиття виконуватимуться до тих пір, поки всі підзадачі не стануть елементарними.

Алгоритм повороту Креша не відразу прораховує весь шлях, а тільки на один крок. Для цього необхідно між початком і кінцем оптимального маршруту провести пряму лінію, визначивши правила, за якими власне відбуватиметься рух. Зважаючи, на сказане вище, це буде не дуже продуктивний спосіб. Так, робот поводитиметься дивно, вибираючи неправильний маршрут, до того ж іноді його рухи будуть зациклюватися. Власне опис методу пошуку шляху наведено нижче:

– до кінцевої точки будується пряма лінія. Координати секції, в якій робот перебував (на один крок) завжди запам'ятовуються;

– у разі появи перешкоди робот застосовує правило правої руки. Він переміщається праворуч до тих пір, поки не знайде вільний прохід. А потім рухається далі;

– перші два пункти повторюватимуться доти, доки кінцевого пункту оптимального маршруту не буде досягнуто;

– якщо робот виявився в попередній точці, то доцільно змінити правило правої руки на правило лівої руки, а потім повторити всю процедуру знову.

Крім того, може бути реалізована трохи ускладнена версія алгоритму, коли під час виявлення перешкоди виконавець рухається в один бік уздовж краю перешкоди, доки рух до кінця шляху по прямій не стане еквівалентним оптимального маршруту від перешкоди.

Однак суттєвим мінусом алгоритму повороту Креша є те, що його залучення на лабіринтоподібній місцевості практично ніколи не дозволяє знайти будь-який шлях.

Зважаючи на вимоги технічного завдання для вибору траєкторії обраного мобільного робота, доцільно застосувати хвильовий алгоритм з

огляду на детермінованість середовища, статичність системи та відносну простоту для реалізації.

Для написання програми було вибрано такі мови програмування, як Canvas (HTML), CSS, JavaScript, що дозволяє активувати програму з будь-якого браузера.

2 РОЗРАХУНОК ВИЗНАЧЕННЯ ОПТИМАЛЬНОГО МАРШРУТУ ПЕРЕМІЩЕННЯ МАНІПУЛЯЦІЙНОГО МОБІЛЬНОГО РОБОТА

2.1 Розроблення схеми керування роботом

Обраний маніпуляційний мобільний робот TurtleBot 3 Burger має круглу форму (див. рисунок 1.4). Зважаючи на це, під час керування ним немає необхідності додавати припуски на занос, оскільки конструкція робота дозволяє обертатися на 360° майже на одному місці. Зауважимо, що робот устаткований роботизованою рукою з п’ятьма ступенями свободи. Власне керування мобільним роботом реалізується через СК із модульною структурою.

Створенню СК мобільним роботом передує побудова системи за ієрархічним багаторівневим принципом. Так, у разі підвищення ієрархічного рангу підсистеми підвищуватиметься і ступінь інтелектуальності (рисунок 2.1) [11].

Система поведінки вважається верхньою ланкою ієрархії, далі слідує система оптимального маршруту, а нижчою ланкою даної ієрархії буде система, що керує виконавчими механізмами. До того ж, окрім усіх перерахованих підсистем, до структури входить обчислювальна система, котра повинна володіти деякими інтелектуальними можливостями, та керування мобільним роботом.

Для того, щоб завдання, поставлене перед роботом, можна було виконати, система поведінки формує його доцільне функціонування. Для системи визначення оптимального маршруту на виході система формує цілевказання: необхідний стан приводів робота, команди керування режимами роботи обчислювальної системи, цільову точку шляху.

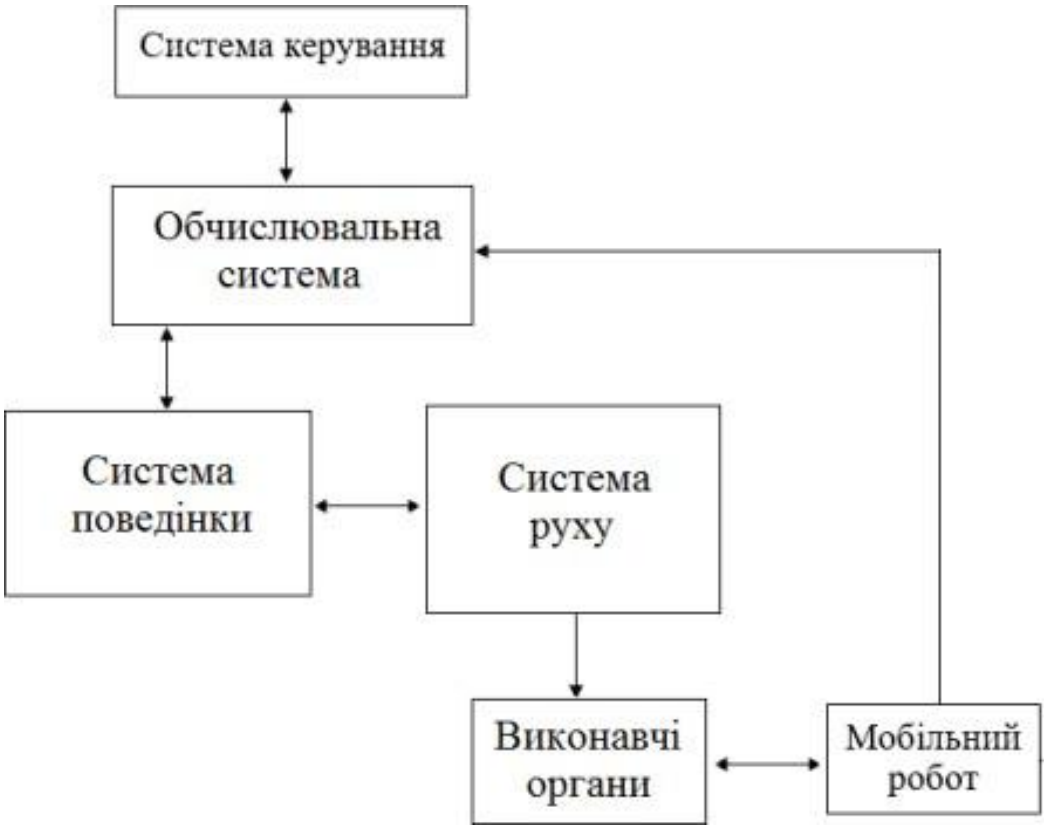


Рисунок 2.1 – Структурна схема системи керування маніпуляційним мобільним роботом

Планування програмних траєкторій оптимального маршруту призначена для того, щоб спрямувати робота до вказаного цільового стану в середовищі з перешкодами, з урахуванням його динамічних характеристик. Власне система поведінки формується цільовим станом, а командне значення швидкостей лінійного руху і повороту мобільного робота – на виході системи [12].

Завдання для виконавчих органів мобільного робота розв’язує система виконавчих органів, тобто вона повинна реалізувати інтерфейс із апаратною частиною його (електричні та механічні пристрої, необхідні для функціонування мобільного робота).

Для зручності використання в СК мобільного робота, обчислювальна система збирає, опрацьовує і перетворює сенсорну інформацію на сигнали.

Підсистеми приймають ті чи інші дані на основі параметрів, які перетворюються з відеосигналу.

Маніпуляційний мобільний робот TurtleBot 3 володіє обчислювальним модулем IntelJoule 570x. Разом із потужною обчислювальною системою Joule 570x дозволяє роботу легко інтегрувати дві камери IntelRealSense 3D. Так, камеру Intel ZR300 RealSense застосовують для того, щоб орієнтуватися на місцевості, водночас камера SR300 RealSense призначена для керування маніпулятором MK3 [9].

2.2 Розроблення детермінованого середовища

З огляду на вимоги технічного завдання та тематику кваліфікаційної роботи, середовище, в якому рухається мобільний робот, повинно бути невизначеним. Таким чином, було обрано замкнене середовище. Іншими словами, за замовчуванням вважається, що мобільний робот уже перебуває в середовищі та не полишатиме його за звичайних умов [13].

Як детерміноване середовище було створено карту розміром 650×650 пікселів, яку наочно продемонстровано на рисунку 2.2.

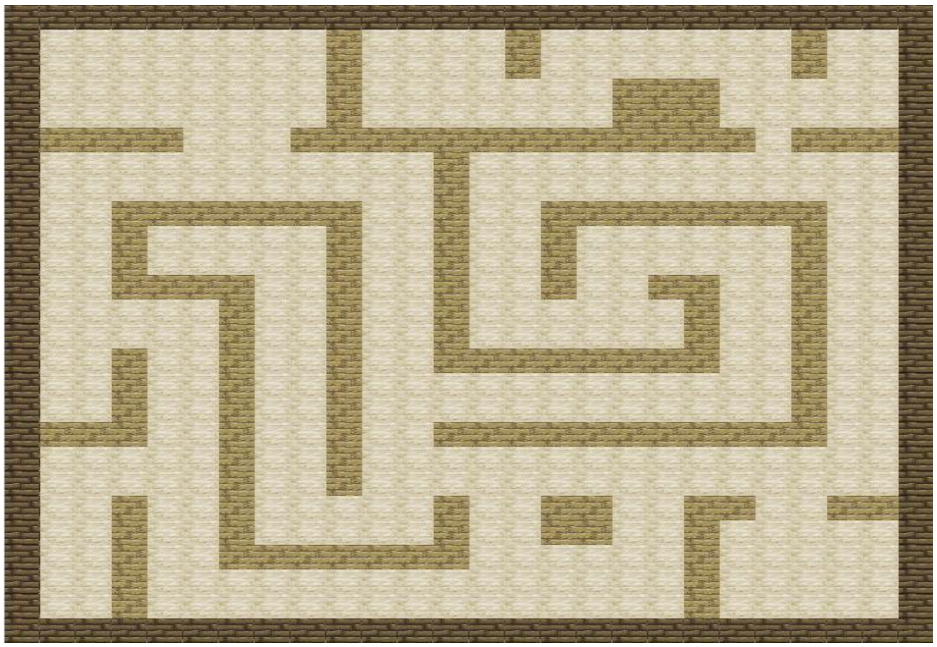


Рисунок 2.2 – Вид замкнутого невизначеного середовища

Інформація про стіни або про кожний елемент, який відповідає певній ділянці стіни, опрацьовується роботом задля прокладання маршруту. Координати комірки середовища визначаються індексами деякої комірки. Точку огляду було обрано зверху, тобто над картою, що б без перешкод можна було спостерігати за рухом робота. Одна ділянка поля (25×25 пікселів) становить 1×1 м. Середовище – це замкнута система, де зовнішні темні стіни не мають виходу до зовнішнього середовища. Товщина цих стін становить 1 м. Координати верхнього лівого кута поля |350; 8|, верхнього правого – |1000;8|, нижнього лівого – |350; 658| і нижнього правого – |1000; 658|.

Зважаючи на те, що на виробництві розташовані столи, роботизовані стаціонарні ділянки та різне обладнання, то на карті зображені тупики та скрізні проходи. Крім того, є три окремі кімнати. Для простоти і наочності товщина всіх внутрішніх стін обрана однаковою, тому, як і зовнішні стіни, становить 1 м.

Оскільки робот може пересуватися тільки рівною поверхнею, то розглядати висоту стін буде зайвим. Місця, де розміщено обладнання та інші предмети, позначені на карті як стіни, тобто нездолані перешкоди.

Для наочності визначатимемо не координати пікселів, а блоки стін. Внутрішня стіна розділяє кімнату від цеху, що виходить із зовнішньої лівої стіни, має протяжність у 4 м (від |1; 5| до |4; 5| блоку). Далі стіна переривається і залишається прохід до кімнати шириною в 3 м. Потім розташовується ще стіна довжиною в 13 м (від |8; 7| до |20; 5|), утім вона розділяє дві кімнати, розташовані поруч із основним цехом, і має виступ у другій кімнаті 3×2 м (|17; 3| до |20; 3| на |17; 3| до |17; 4| відповідно). Далі знаходиться прохід до другої кімнати шириною в 1 метр. Потім простягається стіна завдовжки в 3 м (з |22; 5| до |24; 5|), поки не врізається у зовнішню стінку. Стінка, що розділяє дві сусідні кімнати, виходить з верхньої зовнішньої стіни |9; 1| до внутрішньої |9; 4|. У другій кімнаті є дві стіни, що виходять з верхньої зовнішньої стіни, по 2 м (|14; 1| до |14; 2| і |22; 1| до |22; 2|).

Стіна виходить з довгої стіни має вигляд спіралі і неглибокий тупик. Перший відрізок спрямований донизу (з |12; 6| до |12; 14|), другий – праворуч (з |13; 14| до |19; 14|), третій – вгору (з |19; 15| |19; 11 |) і четвертий повертає ліворуч (|18; 11|). Наступна стіна бере початок із зовнішнього боку спіралі, що створює наскрізний простір. Перша ділянка спрямована догори (з |15; 11| до |15;8|), друга – праворуч (з |16; 8| до |22; 8|), а третя – донизу (з |22; 9| до |22; 17|) і четверта – ліворуч (з |21, 17| до |12, 17|).

Довга стіна утворює глибокий тупик, а також складається з семи частин. Перша її частина йде вгору (з |9; 19| до |9; 8|), друга – ліворуч (з |8; 8| до |3; 8|), а третя – донизу (з |3; 9| до |3; 11|), четверта – праворуч (з |4; 11| до |6; 11|), п'ята знову донизу (з |6; 12| до |6; 22|), шоста – праворуч (з |7; 22| до |12; 22|) і сьома – вгору (з |12; 21| до |12; 20|).

Стіна, що утворює невеликий тупик, бере свій початок з лівої зовнішньої стіни (з |1; 19| до |3; 17|) і повертає догори (з |3; 16| до |3; 14|). У нижньому лівому кутку стіна, що виходить із нижньої зовнішньої стіни (з |3; 24| до |3; 20|) також утворює тупик. Посеред поля розташована стіна розміром 2×2 м (|15; 20| до |16; 20| на |15; 20| до |15; 21|). Третя кімната утворена з декількох стін. Перша виходить з нижньої зовнішньої стіни (з|19; 24| до |19; 20|), повертає праворуч (|20; 20|) і переривається. Інша стіна бере свій початок із правої зовнішньої стіни (з |24; 20| до |23; 20|) на 2 м, залишаючи вхід до кімнати шириною 2 м.

2.3 Розроблення алгоритму оптимального маршруту переміщення робота

Для побудови траєкторії оптимального маршруту переміщення маніпуляційного мобільного робота було використано хвильовий алгоритм. Це один із найшвидших та ефективних методів для об'єктів, які стоять, оскільки можна один раз знайти шлях до мети і починати до неї рух. Отже, для переміщення об'єктів з одного місця на інше такий спосіб найкраще

підходить підприємству. Проте, якщо ж об'єкт буде рухатися, то необхідно кожен ігровий такт прораховувати шлях до нього.

Для побудови робочої зони карти маніпуляційного мобільного робота, зовнішнє середовище дискредитується, набуваючи вигляду обмеженої замкненою лінією фігуру. Така фігура не обов'язково має бути прямокутною, розбитою на прямокутні осередки, в окремому випадку – квадратні. Всі комірки поля розбиваються на підмножини:

– вільні, тобто під час пошуку шляху їх можна проходити;

– перешкоди – шлях через цю комірку заборонений;

– стартову і фінішну комірки, куди необхідно перевезти вантаж.

Так, призначення стартової і фінішної комірок є умовними, тобто достатньо вказати пару комірок, між якими необхідно знайти найкоротший шлях. Зважаючи на все це, доцільно побудувати хвильовий алгоритм (рисунок 2.3).

Наприклад, у початковий момент часу карта, де перебуває маніпуляційний мобільний робот, буде відома, а він власне знаходиться на ділянці, що позначена синім кольором, яка вважається початком координат карти, котра створюється. Ділянки з номерами зеленого кольору вважаються потенційно прохідними. Для перевірки їхньої прохідності мобільний робот повинен послідовно сканувати дані ділянок. Будується образ комірок поля, що опрацьовується, і кожній комірці приписуються значення, що відображають її прохідність. Повинна запам'ятовуватися як стартова, так і фінішна комірки. Для наступного кроку, від стартової комірки, перевіряється її прохідність, щоб вона не належала раніше позначеній комірці [13, 14].

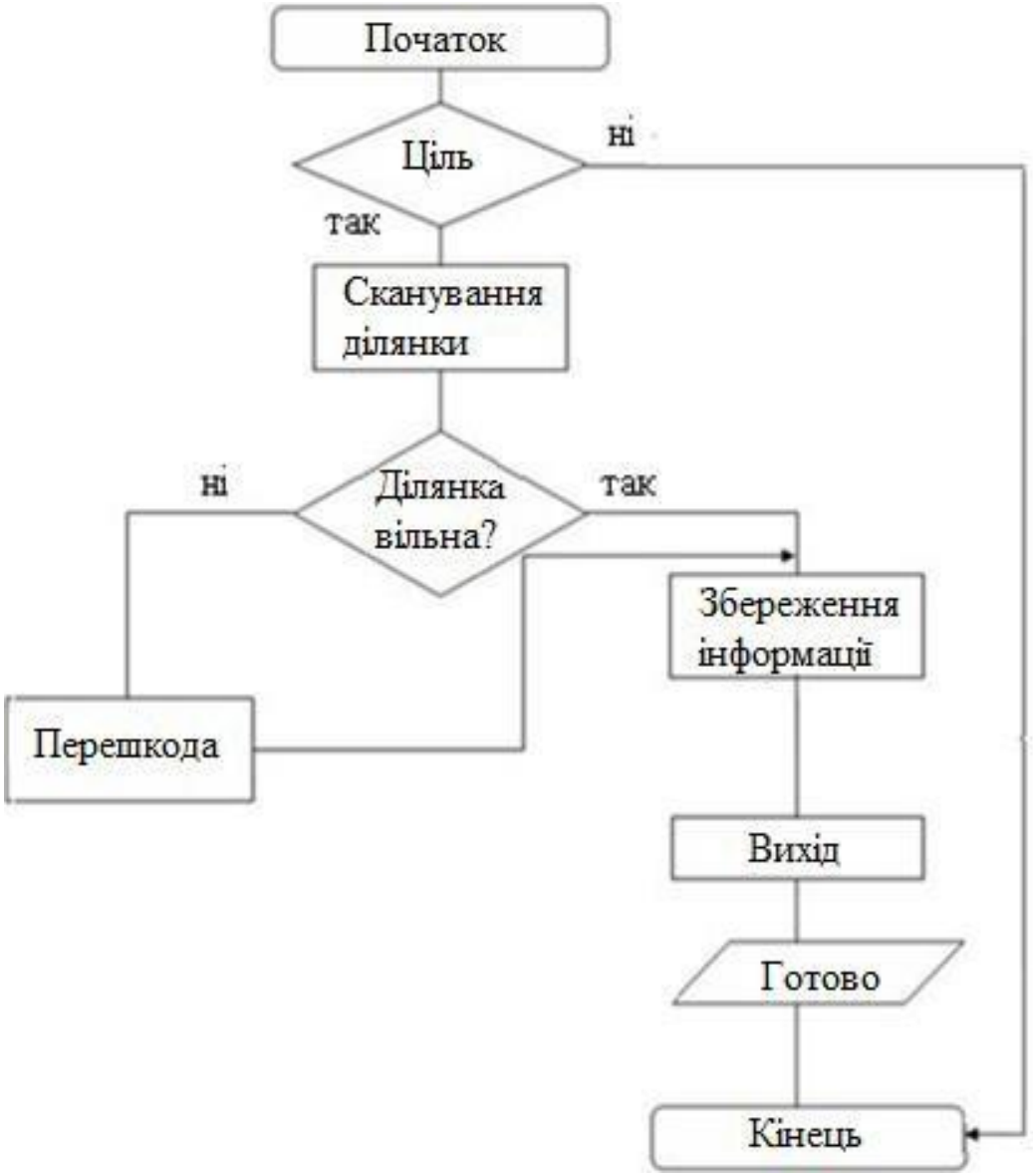


Рисунок 2.3 – Хвильовий алгоритм

Під час перевірки комірки на прохідність, а також її неприналежність до комірок, які раніше вже набули позначення на шляху, до атрибуту комірки записується число, що дорівнює кількості кроків від стартової комірки, проте від стартової комірки на першому кроці це буде 1. Кожна клітинка, що позначена числом кроків від стартової комірки, стає стартовою і з неї розпочинається черговий крок до сусідньої комірки. Зрозуміло, що у разі таких дій рано чи пізно буде знайдено шлях від старту до фінішу (рисунок 2.4), або черговий крок з будь-якої створеної протягом шляху комірки буде неможливий.

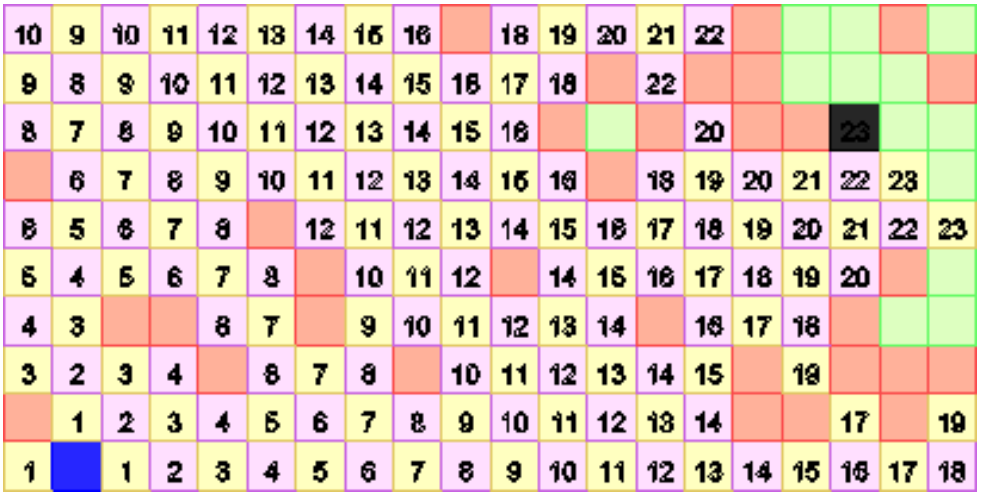


Рисунок 2.4 – Визначення кількості кроків до мети

Якщо відомо зміщення робота (Δx, Δy) щодо центру поточної ділянки, то обчислити індекси даної ділянки можна за такими формулами:

$$x_{ind} = \frac{x_i + \Delta x}{\Delta S}, \tag{2.1}$$

$$y_{ind} = \frac{y_i + \Delta y}{\Delta S}. \tag{2.2}$$

Коли всі комірки між стартом і фінішом позначені цифрою, то за допомогою хвильового алгоритму будується траєкторія найкоротшого шляху (рисунок 2.5).

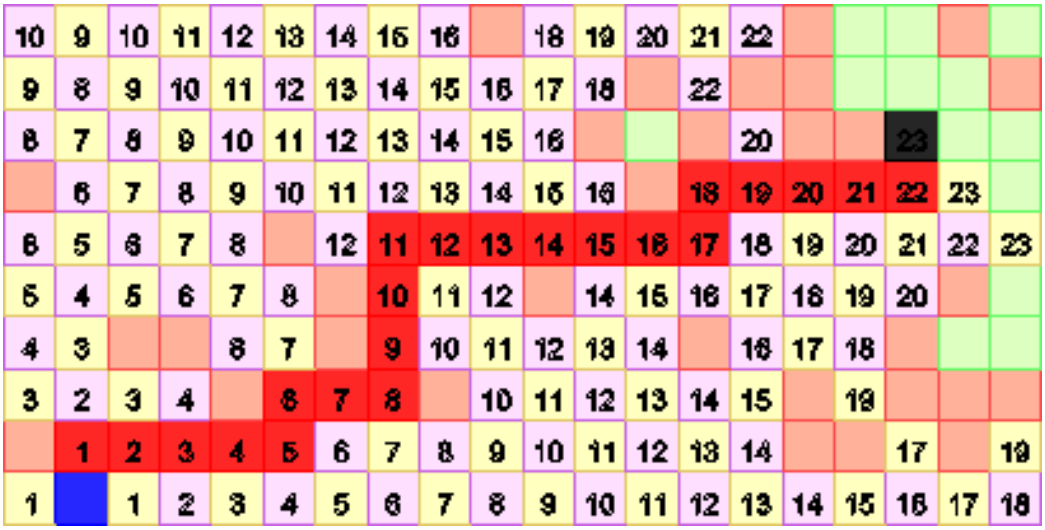


Рисунок 2.5 – Складання траєкторії оптимального маршруту

Якщо дві ділянки мають однаковий пріоритет, то перевага надається тій, що розташована по вертикалі. Це аргументується тим, що побудова траєкторії розпочинається від кінцевої комірки.

Далі за за траєкторією, котра була побудована найкоротшим шляхом, робот починає йти через центр кожної ділянки. До того ж величина переміщення дорівнює максимальному з габаритних розмірів ділянок, на які дискредитується середовище. У подальшому на карту наносяться координати центру кожної ділянки, що обчислюються як:

$$x_i = x_{i-1} \pm \Delta S \cdot \cos\alpha, \tag{2.3}$$

$$y_i = y_{i-1} \pm \Delta S \cdot \sin\alpha, \tag{2.4}$$

де x_{i-1}, y_{i-1} – абсциса й ордината центру попередньої ділянки;

ΔS – величина переміщення;

α – курсовий кут робота.

Так, у формулі (2.3) переважає знак «+», якщо і ділянка розташована правіше $x_i > i$-1, а знак «-», якщо лівіше. У формулі (2.4) знак«+» переважає, якщо і ділянка розташована вище за i-1, а знак «-», якщо нижче. Крім того, поряд із координатами центру кожної ділянки ставляться у відповідність індекси зсуву за координатними осями x, y щодо початкової ділянки. Це робиться для того, щоб індекси ділянок були відповідними щодо упорядкованої збереженої карти середовища в пам'яті мобільного робота. Якщо ж якась ділянка буде зайнятою, то роботу необхідно визначити, чи присутня на ній перешкода i з якою метою додавати інформацію про дану ділянку на карту. Зазначений процес триватиме доти, поки маніпуляційний мобільний робот не досягне кінцевої ділянки [14].

2.4 Розроблення схеми обходу перешкод

Якщо необхідність обходу перешкоди роботом є нагальною, то можна послуговуватись таким алгоритмом керування для ділянок, які віддаленні від робота більш ніж на *n* осередків матриці А (рисунок 2.6). Так, для кожного з m напрямків оптимального маршруту маніпуляційного мобільного робота, що можливі в даний момент, складається значимість Рі даного напрямку, що прораховується як:

$$P_i = \sum g_k P_{ki}, \tag{2.5}$$

де P_{ki} – оцінка *i*-го напряму блоком оцінки ($Б_1 – Б_5$);

g_k – значення оцінки.

За силу, що спрямовує черговий крок, вибирається напрямок m, який набув найбільшої значимості P_m (рисунок 2.6).

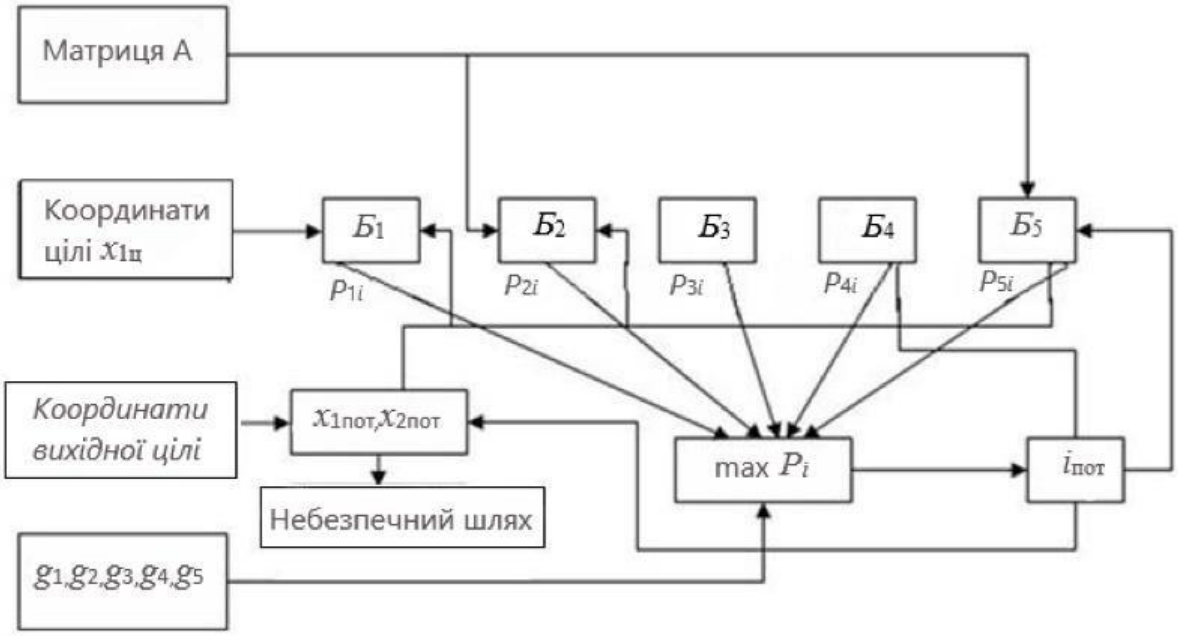


Рисунок 2.6 – Структурна схема обходу небезпечних ділянок траси

– $Б_1$ відповідає за напрямок до цілі з поточної складової на кроці i:

$$P_{1i} = \frac{(x_{1ц}-x_{1пот})\cos\pi/4(i-1)+(x_{2ц}-x_{2пот})\sin\pi/4(i-1)}{\sqrt{(x_{1ц}-x_{1пот})^2+(x_{2ц}-x_{2пот})^2}}. \quad (2.6)$$

– $Б_2$ відповідає за сусідні зайняті клітини поточного стану, зокрема P_{2i}=1, якщо комірка ($x_{1пот}$ + Δx_{1i}, $x_{2пот}$ + Δx_{2i}) зайнята та P_{2i}=0 – в іншому випадку.

– $Б_3$ відповідає за випадкові значення P_{3i}=z, де випадкові числа вибираються з рівноцінною ймовірністю на відрізку |0, 1|.

– $Б_4$ відповідає за інерцію робота, значення P_{4i}=1, для напрямку i_0, за яким було зроблено попередній крок, P_{4i} =1/2 – для напрямків, які відрізняються від i_0 на π/4, та P_{4i}=0 для інших напрямків;

– $Б_5$ відповідає за імітацію “ближню передбачливість” та дає сигнал про найпростіші тупикові ситуації за один крок. Значення P_{5i}=1, якщо після кроку в i-м напрямку мобільного роботу доведеться повертатися на кут, більший ніж π/4, та P_{5i}=0, якщо після кроку в i-м напрямку можливо продовжити рух, не повертаючись більше ніж на π/4.

Обрано п’ять параметрів для значень:

– g_1 відповідає за важливість дотримання напряму до цілі, у разі g_1 <0 робот віддаляється від мети;

– g_2 відповідає за ставлення мобільного робота до перешкод: у разі g_2 <0 мобільний робот уникає перешкод; у разі g_2>0 мобільний робот прагне йти через комірки, які містять перешкоди;

– g_3 відповідає за ступінь хаотичності в рухах мобільного робота і може змінюватися в процесі пошуку шляху в складних ситуаціях. Проте ймовірне і «усвідомлення» тупику, коли останні n кроків не привели до скорочення відстані до цілі:

$$r_{ха} = \sqrt{(x_{1ц} - x_{1пот})^2 + (x_{2ц} - x_{2пот})^2},$$

при цьому значимість g_3 поступово збільшується в діапазоні |0, 1|, поки пошук керування не стає випадковим. Таким чином, за допомогою рівняння дифузії описується потрапляння на будь-яку незайняту клітку, і у разі, коли мобільний робот опиняється в глухому куті, він обов'язково знайде вихід;

– g_4 відповідає за плавні повороти і дотримання оптимального маршруту по прямій при $g_4>0$, мобільний робот кидається з боку в бік і схильний до різких поворотів при $g_4 <0$;

– g_5 відповідає за обережність мобільного робота, при $g_5<0$ мобільний робот застерігається від перпендикулярного наближення до стіни і прямування в кут.

Отже, для безпечного досягнення цілі, небезпечні ділянки на полі стають не прохідними, тобто позначаються як стіна. Зважаючи на це, траєкторія оптимального маршруту переміщення маніпуляційного мобільного робота планується в обхід таких небезпечних ділянок.

3 РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ОПТИМАЛЬНОГО МАРШРУТУ ПЕРЕМІЩЕННЯ МАНІПУЛЯЦІЙНОГО МОБІЛЬНОГО РОБОТА

3.1 Вибір мови програмування

JavaScript – це мова програмування, що працює за інтерактивним принципом організації. Мета її розроблення полягала в тому розроблялася, щоб web-сторінки робити «живими». Цій мові властива підтримка функціонального, об'єктно-орієнтованого й імперативного стилів. Щоб отримати програмний доступ до об’єктів програми застосовується JavaScript. До того ж, в браузерах здебільшого використовується як мова сценаріїв. JavaScript властиві такі архітектурні риси, як-от: слабка і динамічна типізація, прототипне програмування, автоматичне керування пам’яттю, функції як об’єкти першого класу [15].

Під час створення JavaScript планувалося зробити її схожою на Java, водночас легкою для застосування людьми, котрі не є в ній фахівцями. До того ж під час її розроблення впливало дуже багато мов. Утім більшість мов програмування, що є актуальними для web-розроблення, належать компаніям або організаціям, однак не JavaScript.

JavaScript – це об'єктно-орієнтована мова в порівнянні з іншими клас-орієнтованими. Вона базується на прототипуванні, водночас містить низку властивостей функціональних мов – каринг, об'єкти як списки, функції як об'єкти першого класу, замикання, що надає мові додаткової гнучкості, анонімні функції.

Незважаючи на схожість синтаксису між C і JavaScript, остання характеризується суттєвими відмінностями:

– автоматичне прибирання сміття;

– об'єкти з можливістю інтроспекції;

– анонімні функції;

– функції як об'єкти першого класу;

– автоматичне приведення типів;

Проте у даній мові відсутні такі корисні речі, зокрема:

– стандартна бібліотека: немає інтерфейсу програмування застосунків щодо роботи з файловою системою, керування потоками введення/виведення, базових типів для бінарних даних;

– стандартні інтерфейси до web-серверів і БД;

– система керування пакетами, котра відстежувала б залежності і автоматично встановлювала їх.

Утім синтаксис JavaScript у багатьох випадках і можливостях подібний до синтаксису C і Java, хоча, семантично мова наближена до Self, Smalltalk або Ліспі.

Структурно JavaScript можна представити у вигляді трьох чітко розмежованих одна від одної об'єднаних частин:

– ядро (ECMAScript);

– об'єкт на модель браузера (BrowserObjectModelабоBOM);

– об'єкт на модель документа (DocumentObjectModelабоDOM).

Сутність JavaScript, яка узгоджується з визначенням DOM, може бути окремою від об’єктної моделі документа. Об'єктна модель браузера й об'єктна модель документа є недопустимими у разі застосування JavaScript у середовищах, відмінних від браузера.

В ECMAScript не визначаються методи введення/виведення інформації, проте вона не є браузерною мовою. Для неї більш притаманне таке поняття, як основа для побудови скриптових мов. Специфікація ECMAScript описує об'єкти, регулярні вирази, типи даних, ключові і зарезервовані слова, інструкції, оператори. Разом із тим не обмежує користувачів похідних мов у розширенні їх новими складовими.

Об'єктна модель браузера – це зона між об'єктною моделлю документа й ядром. Ключове її призначення міститься в керуванні вікнами браузера і

забезпеченні взаємодії між ними. Кожне з цих вікон подається як об'єкт Window або центральний об'єкт DOM.

Окрім керування вікнами, в межах цієї моделі, як правило забезпечується підтримка браузерами таких властивостей:

– керування кадрами;

– підтримка затримки у реалізації коду і зациклення із затримкою;

– системні діалоги;

– керування адресою відкритої сторінки;

– керування інформацією про браузер;

– керування інформацією про параметри монітора;

– обмежене керування історією перегляду сторінок;

Підтримка роботи з HTTPcookie.

Інтерфейс програмування застосунків для HTML і XML-документів – це об’єктна модель документа. Документ можна сформувати як дерево об'єктів, яке має властивості, згідно з DOM, що дозволяють проводити з ним різні маніпуляції [15]:

– генерація і додавання вузлів;

– отримання вузлів;

– зміна вузлів;

– зміна зв'язків між вузлами;

– видалення вузлів.

Щоб додати JavaScript-код на сторінку, доцільно застосовувати теги <script></ script>, хоча вони і є рекомендованими, проте їх не обов'язково розташовувати всередині контейнера <head>. Так, в одному документі може бути безліч контейнерів <script>. Атрибут «type = 'text / javascript'» зазначати не обов’язково, дане значення використовується за замовченням [15].

HTML – це мова програмування, на базі якої реалізується розмітка документів. Більшість вебсторінок написано на HTML. Текстові документи із HTML-розміткою, що мають розширення .html або .htm, обробляються браузерами, котрі, зі свого боку, відображають документ у його

форматованому вигляді. Завдяки ним користувачі можуть послуговуватись зручним інтерфейсом для запиту вебсторінок та їхнього перегляду, а також надсилати введені користувачем дані на сервер. Власне за допомогою цієї мови можна визначити, як текст, гіперпосилання або мультимедіа будуть відображатися в браузері [16].

Для роботи з Canvas необхідні базові знання HTML і JavaScript, однак у цілому вона є дуже простою. Так, Canvas є елементом HTML5, тому за допомогою скриптів створює растрове двомірне зображення. Зліва зверху розташовано початок відліку блоку. Від нього розпочинає будуватися кожен блок-елемент. Розмір простору координат не завжди відображає розмір фактичної площі [15, 17, 18].

Створення статей, графіків, а також ігрового поля в браузерних іграх – це є головною метою Canvas. До того ж можливе застосування для побудови графів і створення колажів або анімації.

Утім Canvas має деякий недолік, що може ускладнити роботам завдання щодо розпізнавання капчі. Під час функціонування Canvas із сервера завантажується набір точок (або алгоритм промальовування), а не зображення, за якими браузер промальовує (капчу).

Крім того, можна додавати відео, зображення та текст, застосовувати кольорову заливку, градієнт і обведення контурів, а також створювати фігури за допомогою вказівок контрольних точок, утім змінюється ширина ліній, стиль з'єднань ліній і пензлик малювання ліній [15].

Як особливості Canvas визначимо:

– зміну висоти і ширини полотна;

– розташування початку відліку (точка 0,0) у лівому верхньому куті;

– відсутність 3D-контексту, проте наявність окремих нестандартизованих розробок;

– зазначення кольору тексту і розміру шрифту як у CSS.

Недоліки Canvas полягають у:

– надмірному навантаженні процесора й оперативної пам'яті;

– обмеження збирача сміття не дозволяють очистити пам'ять;

– самостійному обробленні дій з об'єктами;

– поганій продуктивність при високому розширенні;

– незручності створенні кожного елемента окремо;

– можливості створення на сторінках спеціальних «маячків», що йменуються CanvasFingerprinting, для відстеження користувачів у мережі.

Переваги Canvas:

– на противагу SVG зручніше мати справу з великою кількістю елементів;

– присутнє апаратне прискорення;

– можливість маніпулювання кожним пікселем;

– можливість застосування фільтрів оброблення зображень;

– наявність багатьох бібліотек.

CSS – мова, що формує зовнішній вигляд HTML-сторінки. Здебільшого нею послуговуються для оптимізації і структурування коду. На її основі можна задавати колір, розташовувати окремі блоки, шрифту та інших аспектів. Головна мета CSS полягає у розмежовуванні опису логічної структури вебсторінки з описом зовнішнього вигляду цієї вебсторінки. Це посприяло збільшенню доступності документа, а також надало більшої гнучкості та можливості керувати його поданням. Застосовуючи CSS, можна демонструвати один документ різними методами виведення або стилями, зокрема шляхом друкованого подання, екранного уявлення або читання голосом [16].

Таблицям стилів притаманні правила власне CSS. Дані таблиці можуть розташовуватися як власне у вебдокументі, зовнішній вигляд якого вони описують, так і в окремих файлах, що містять формат CSS. Загалом, формат CSS – це звичайний текстовий файл. У файлі не знаходиться нічого, крім переліку правил CSS і коментарів до них. Отже, таблиці можуть бути підключені або впроваджені до описуваного ними вебдокумента за допомогою чотирьох різних способів [16]:

– у разі розташування таблиці стилів в окремому файлі, її можна підключити до вебдокумента за допомогою тега <link>, який знаходиться в цьому документі між тегами <head> і </ head>. (Тег <link> матиме атрибут href, що є позначенням адреси цієї таблиці стилів). Усі правила цієї таблиці чинні протягом усього документа;

– у разі розташування таблиці стилів в окремому файлі, її можна підключити до вебдокумента за допомогою директиви @import, що знаходиться в цьому документі між тегами <style></ style, відразу після тега <style>, що також вказує на адресу цієї таблиці стилів. Усі правила цієї таблиці чинні протягом усього документа;

– у разі опису таблиці стилів власне у документі, вона може знаходитись у ньому між тегами <style> і </ style>. Усі правила цієї таблиці також чинні протягом усього документа;

– у разі опису таблиці стилів власне у документі, вона може міститися в тілі якогось окремого тега цього документа. Всі правила цієї таблиці діють тільки на складові цього тега.

У перших двох випадках мається на увазі, що з документом використовуються зовнішні таблиці стилів, а в інших двох – внутрішні таблиці стилів.

Передусім HTML-документи створюються на підставі ієрархії елементів, яку можна наочно представити в деревовидній формі. Так, елементи HTML один для одного можуть бути батьківськими, сестринськими, дочірніми, елементами-пращурами, елементами-нащадками [16].

Елемент може стати батьком іншого елемента, якщо в ієрархічній структурі документа він розташований відразу, безпосередньо над цим елементом. Елемент може стати пращуром іншого елемента, якщо в ієрархічній структурі документа він розташований десь вище цього елемента.

Наприклад, до структури документа належать два абзаци *p*, які містять шрифт із напівжирним шрифтом *b*. Тоді елементи *b* вважатимуться дочірніми

елементами своїх батьківських елементів *p*, а також нащадками свого пращура body. Зі свого боку, для елементів *p* елемент body буде лише батьком. Водночас ці два елементи *p* уважатимуться сестринськими елементами, оскільки походять з одного із батьків – body [17].

Так, за допомогою селекторів у CSS можуть задаватися не лише поодинокі елементи, але й такі, що є нащадками, дочірніми або сестринськими елементами інших елементів.

3.2 Розроблення программного забезпечення оптимального маршруту переміщення робота

Для розроблення системи доцільно створити детерміноване середовище, вибрати мобільного робота з маніпулятором, утілити хвильовий алгоритм траєкторії оптимального маршруту робота разом із динамікою робота [19].

Спершу необхідно з’ясувати розмірність однієї комірки та їхню кількість на карті (рисунок 3.1).

```
// Настройки карти
var block_size = 25;// Розмір в пікселях одного блоку
varmap_size = 26;// Кількість блоків (NxN = квадрат)
var game = 'set_robot';
```

Рисунок 3.1 – Задаються параметри карти

Заповнюємо карту вільними комірками, тобто якими робот пересуватиметься (рисунок 3.2).

```
// Задаем пустоту на карті
for (var i = 0; i != map_size; i++) {
for (var j = 0; j != map_size; j++) {
map[i][j] = 0;
   }
}
```

Рисунок 3.2 – Створюються вільні комірки

Потім будуємо зовнішні стіни карти, тобто границі, за які робот виходити не зможе (рисунок 3.3).

```
// Задаємо границі карти
for (i = 0; i != map_size; i++) {
   map[i][0] = 2;
   map[i][map_size - 1] = 2;
   map[0][i] = 2;
   map[map_size - 1][i] = 2;
}
```

Рисунок 3.3 – Створюються зовнішні стіни

Разом із тим створюємо внутрішні стіни за кожною координатою комірки, тому стислий приклад наведено на рисунку 3.4.

```
// Задаємо стіни карти
map[1][9] = 1;
map[1][14] = 1;
map[1][22] = 1;
map[2][9] = 1;
map[2][14] = 1;
map[2][22] = 1;
map[3][9] = 1;
map[3][17] = 1;
map[3][18] = 1;
map[3][19] = 1;
```

Рисунок 3.4 – Створюються внутрішні стіни

Для опису функціонування системи було побудовано алгоритм програми, який продемонстровано на рисунку 3.5.

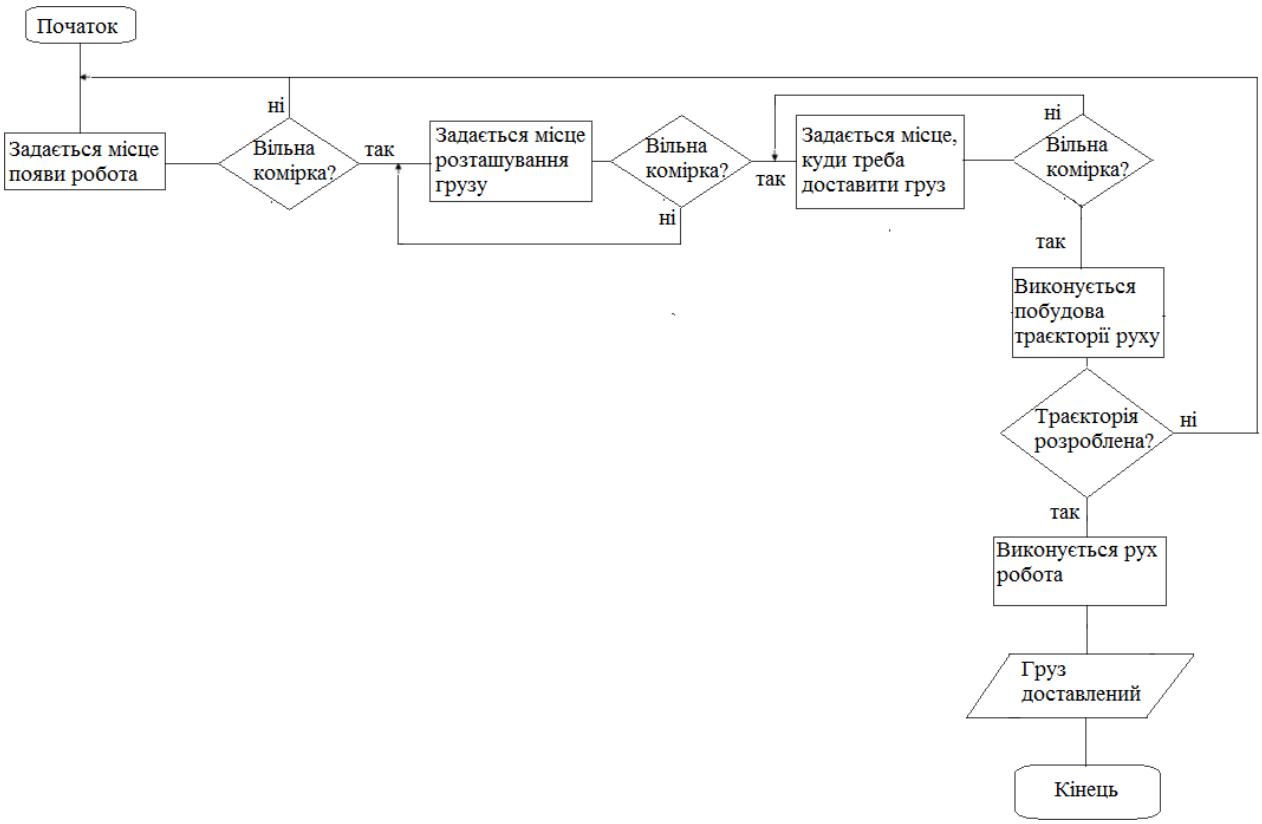


Рисунок 3.5 – Алгоритм функціонування програми

Код, який наведено у тексті нижче, задає місце появи мобільного робота та перевіряє, щоб зазначена комірка була вільною. Якщо місце буде вказано на стінку, то з’явиться вікно, в якому повідомлятиметься, що розмістити робота у цьому місці неможливо.

```
// Оброблення натискання на вікно динамічних об'єктів
function game_click(x, y) {
//Визначаємо сценарій оброблення
 switch (game) {
case'set_robot':
//Натискання на місце, де буде робот
var click_possition = map_position_zero(x, y, map, map_size, block_size);
```

```
//Отримуємо індекси для масиву карти згідно з координатами натискання
if(false==click_possition){
alert ('Тут неможливо розташувати робота, спробуйте ще раз');
}else {
game = 'set_cargo'; map[click_possition[0]][click_possition[1]]=100; robot_x = click_possition[0];
robot_y = click_possition [1];
 game_render(document.getElementById('game'),map,map_size, block_size);
// Відображаємо робота в області натискання setTimeout("alert('Оберить місто на карті для развантаження')",100);
}
break;
```

За аналогією обирається місце, де з’являється вантаж, який необхідно транспортувати. Якщо місце буде спрямовуватись на стінку, то з’явиться вікно, в якому повідомлятиметься, що розмістити вантаж у цьому місці неможливо. Текст програми наведено нижче.

```
case'set_cargo':
//Натискання  на місце, де буде вантаж
var click_possition = map_position_zero(x, y, map, map_size, block_size);
//Отримуємо індекси для масиву карти згідно з координатами натискання
if(false==click_possition){
alert('Тут неможливо розташувати робота, спробуйте ще раз');
}
else{
```

```
game = 'set_point'; map[click_possition[0]][click_possition[1]]=101;
cargo_x = click_possition[0];
cargo_y = click_possition[1];
game_render(document.getElementById('game'),map,map_size,
block_size);
// Відображаємо вантаж в області натискання
setTimeout ("alert (Оберіть місце на карті для розвантаження)",100);
}
break;
```

Як і у варіанті з роботом та вантажем, обирається місце, куди робот повинен перевезти вантаж. Якщо місце буде вказано на стінку, то з’явиться вікно, в якому повідомлятиметься, що розмістити вантаж у цьому місці неможливо. Текст програми наведено нижче.

```
case'set_point':
// Натискання на місце, де буде фінішна точка
var click_possition = map_position_zero(x, y, map, map_size, block_size);
//Отримуємо індекси для масиву карти згідно з координатами
натискання
if(false==click_possition){
alert ('Тут неможливо розташувати розвантаження, спопробуйте ще
раз');
}else {
game = 'set_start'; map[click_possition[0]][click_possition[1]]=102; point_x
= click_possition[0];
point_y = click_possition[1];
game_render(document.getElementById('game'),map,map_size,
block_size);
// Відображаємо точку
```

```
setTimeout ("alert ('Робот прораховує маршрут')",100);
```

Слідом будується оптимальний маршрут переміщення робота, котрий обчислюється за допомогою хвильового алгоритму. Нижче наведено частину коду, де описується крок розповсюдження хвилі.

```
//Запускаємо нескінченний цикл for (; ;) {
for (var i = 0; i != map_size; i++) { for(varj=0;j!= map_size; j++){
// Задаємо значення біля старту +1 (подібно рекурсії), у разі досягненні фінішу, в зворотному порядку переглядаємо найменші значення і записуємо маршрут цих індексів масиву карти
if(map_arr[i][j]==Ni){
//i + 1
if(map_arr[i+1][j]!==undefined){ if (map_arr[i + 1][j] == 253) {
x.splice(0,x.length); y.splice(0,y.length); xx = i + 1;
yy=j;
for(varii=0;ii!=Nk;ii++){ map_right = 999;
if(map_arr[xx+1][yy]!==undefined){ map_right = map_arr[xx + 1][yy];
}
map_left= 999;
if(map_arr[xx-1][yy]!==undefined){ map_left = map_arr[xx - 1][yy];
}
```

У тексті програми, що наведено нижче, прописано обчислення оптимального маршрут.

```
varpoint_map=arrayClone(map);
var cargo_way = way(100, 101, map, map_size);
// Обчислюємо короткий маршрут до вантажу
var point_way = way(101, 102, point_map, map_size);
```

// Обчислюємо короткий маршрут від вантажу до фінішу

Далі будуємо траєкторію всього маршруту, з’єднуючи шлях від місця появи робота до вантажу, а потім шлях до місця, куди вантаж повинен бути транспортований (рисунок 3.6). Текст програми наведено нижче.

```
//Збираємо маршрут до вантажу і до фінішу в загальний маршрут
way_size = 2 + way[0].length + point_way[0].length;
way_full[0][0]=robot_x; way_full[0][1]=robot_y;
for(vari=0;i<way[0].length;i++){ way_full[i + 1][0] = way[0][i];
way_full[i+1][1]=way[1][i];
}
way_full[way[0].length + 1][0] = cargo_x; way_full[way[0].length + 1][1] =
cargo_y; for(vari=0;i<point_way[0].length;i++){
way_full[i+way[0].length+2][0]=point_way[0][i];
way_full[i+way[0].length+2][1]= point_way[1][i];
}
```

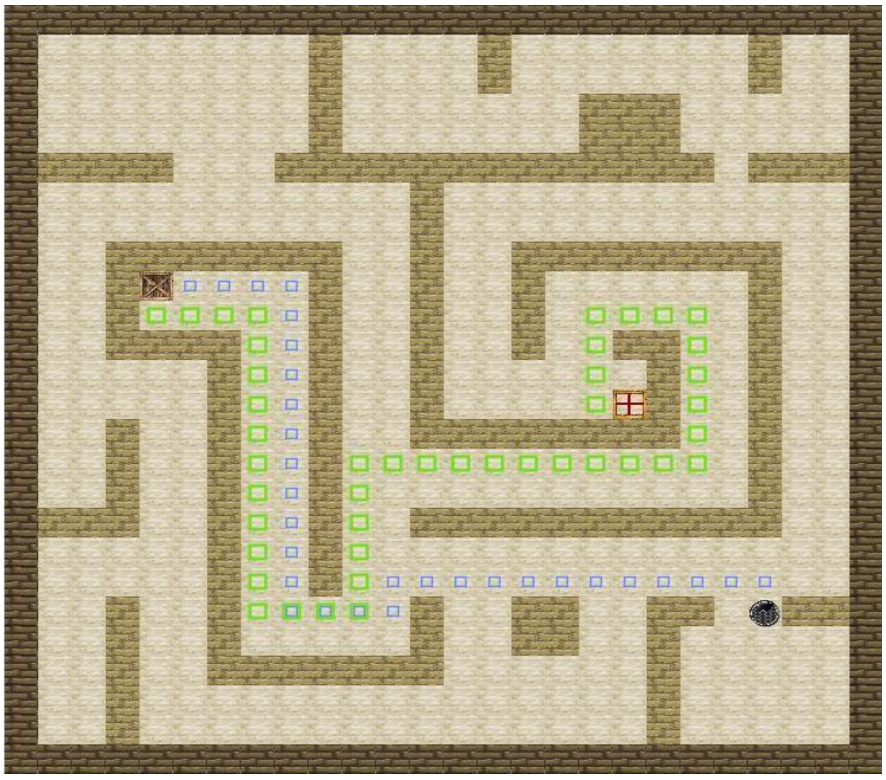


Рисунок 3.6 – Траєкторія всього шляху мобільного робота

Потім необхідно зробити так, щоб робот рухався побудованим маршрутом, що наведено у тексті нижче.

```
// Переміщення робота functionway_go(){
varmap_canvas=document.getElementById('game'); if (gi < way_size) {
index++;
```

Нижче у тексті наведено як відображається переміщення робота вгору, вниз і по сторонах.

```
//Відображаємо переміщення робота
//Up
if(way_full[gi][0]>way_full[gi+1][0]){
way_render(map_canvas, 'dirt', way_full[gi + 1][0] * block_size,
way_full[gi][1] * block_size, block_size, block_size);// Clear block
way_render(map_canvas, 'robot', way_full[gi][0] * block_size -index + 1,
way_full[gi][1] * block_size + 1, block_size - 2, block_size - 2);// Clear block
}
```

Нижче у тексті наведено завершення оптимального маршруту переміщення робота, коли він досягає кінцевої точки.

```
if(index>block_size){
//Переходимо до наступного кроку маршруту index = 0;
gi++;
}
setTimeout(way_go,33);
// Повторно викликаємо поточну функцію із затримкою 0.033 сек
}
else{
```

```
//Робот прибув на місце
way_render(map_canvas,'dirt', point_x * block_size, point_y
* block_size, block_size, block_size);
way_render(map_canvas,'cargo',point_x*block_size,point_y* block_size,
block_size, block_size);
aler t('Робот доставив вантаж');
}
}
```

Лістинг програми наведено у стислому вигляді, проте повний варіант продемонстровано у Додатку А.

4 ОХОРОНА ПРАЦІ

4.1 Аналіз умов праці на робочому

На робочому місці оператора ПК згідно виникають небезпечні та шкідливі фактори: підвищений рівень шуму, несприятливі мікрокліматичні умови, недостатній рівень освітленості, шкідливі речовини, підвищений рівень електромагнітних випромінювань радіочастот, висока напруга електричної мережі, статична електрика та інші. Робота з ПК супроводжується також підвищеним ступенем напруженості трудового процесу. При систематичному впливі виробничих факторів, які не відповідають нормативним показникам, зростає рівень професійно зумовленої захворюваності працюючих та можуть виникнути професійні захворювання органів зору, руху, нервової системи. Таким чином, вивчення умов праці на робочому місці оператора ПК є необхідною умовою запобігання негативних наслідків впливу небезпечних та шкідливих факторів.

Організація робочого місця. Приміщення, в якому знаходиться робоче місце оператора ПК, загальною площею 48 м^2, і висотою стелі 3,5 м. У приміщенні знаходиться 6 робочих місць з ПК. Кожне робоче місце обладнане робочім столом, стільцем та персональним комп'ютером, що складається з монітора, системного блоку, клавіатури та миші.

4.2 Промислова безпека на робочому

Живлення ПК здійснюється від трифазної чьотирьох електричної мережі змінного струму з глухо-заземленою нейтраллю і напругою 220 В, частотою 50 Гц. Згідно НПАОП 40.1-1.21-98 приміщення можна віднести до

категорії без підвищеної небезпеки, так як в приміщенні відсутні чинники, які викликають підвищену або особливу небезпеку.

Для створення безпечних умов праці необхідно провести ряд організаційних і технічних заходів. Згідно НПАОП 40.1-1.32-01 для запобігання ураження людини електричним струмом в приміщенні застосовується система занулення.

4.3 Виробнича санітарія у приміщенні

Робота оператора ПК за енерговитратами відноситься до категорії легких робіт. В таблиці 4.1 наведені оптимальні параметри мікроклімату в приміщеннях, де виконуються роботи операторського типу [20].

Таблиця 4.1 – Параметри мікроклімату для приміщень з ПК

Період року	Параметр мікроклімату	Величина
Холодний	Температура повітря в приміщенні; відносна вологість; швидкість руху повітря	22 – 24 °С; 40 – 60 %; до 0,1 м/с
Теплий	Температура повітря в приміщенні; відносна вологість; швидкість руху повітря	23 – 25 °С; 40 – 60 %; 0,1 – 0,2 м/с

Виміряні за допомогою приладів температура та вологість у лабораторії відповідають вказаним у таблиці для теплого періоду року. Слід зазначити, що для нормалізації параметрів мікроклімату слід використовувати у приміщеннях кондиціювання повітря, або забезпечити подачу свіжого повітря системами вентиляції.

Лабораторія, де виконується розробка конструкції модуля, має наступні характеристики:

– площа приміщення 48 м^2 (8×6 м);

– висота – 3,5 м;

– кількість робочих місць – 6 шт.;

– обладнання – стіл з ПК і периферією – 6 шт.

Приміщення, відповідно до ДНАОП 0.00-1.31-99, має забезпечувати 6 м^2 площі та 20 м^3 обсягу на одне окреме робоче місце з ПК [20]. Площа приміщення 48 м^2 та об'єм 168 м^3, на кожне робоче місце приходится 8 м^2 площі і об'єм 28 м^3, тобто вимога виконана.

Приміщення з ПК повинні мати природне і штучне освітлення відповідно до ДБН В.25-28-2006 «Природне і штучне освітлення». Природне світло повинно проникати через бічні світлові прорізи, зорієнтовані, як правило, на північ або північний схід, і забезпечувати коефіцієнт природної освітленості (КПО) не нижче 1,5 %.

Рівень загального штучного освітлення приміщення можна перевірити за допомогою методу питомої потужності, викладеної в [20].

Розрахункова формула методу:

$$W = \frac{W_{\Sigma}}{S}, \tag{4.1}$$

де W – питома потужність, Вт/м^2;

S – площа приміщення, м^2;

W_{Σ} – загальна потужність освітлювальної установки Вт, яка розраховується за формулою:

$$W_{\sum} = W_{св} \cdot n_{св}, \tag{4.2}$$

де $W_{св}$ – потужність одного світильника, Вт;

$n_{св}$ – кількість світильників в приміщенні.

$$W_{\sum} = 100 \cdot 4 = 400 \text{ Вт}, \quad (4.3)$$

$$W = \frac{400}{48} = 8{,}33 \text{ Вт/м}^2. \quad (4.4)$$

Питомої потужності 8,33 Вт/м^2 по таблиці Б.3 із [20] відповідає освітленість в 250 лк при мінімальній допустимої освітленості 300 лк.

Отже, для створення сприятливих зорових умов в лабораторії необхідно збільшити кількість світильників або замінити лампи в світильниках на більш потужні.

4.4 Пожежна безпека виробничого приміщення

Пожежна безпека – стан об'єкта, при якім виключається можливість пожежі, а у випадку його виникнення запобігає вплив на людей небезпечних факторів пожежі й забезпечується захист матеріальних цінностей.

Пожежна безпека забезпечується системою запобігання пожежі й системою пожежного захисту. У всіх службових приміщеннях обов'язково повинен бути «План евакуації людей при пожежі», що регламентує дії персоналу у випадку виникнення вогнища загоряння, що й указує місця розташування пожежної техніки.

Горючими компонентами у виробничому приміщенні є: перегородки, двері, підлоги, ізоляція кабелів і ін.

Протипожежний захист – це комплекс організаційних і технічних заходів, спрямованих на забезпечення безпеки людей, на запобігання пожежі, обмеження його поширення, а також на створення умов для успішного гасіння пожежі.

Джерелами запалювання у виробничому приміщені можуть бути електронні схеми від ПК, прилади, застосовувані для технічного обслуговування, пристрою електроживлення, кондиціювання повітря, де в

результаті різних порушень утворюються перегріті елементи, електричні іскри й дуги, здатні викликати загоряння горючих матеріалів.

У сучасних ПК дуже висока щільність розміщення елементів електронних схем. У безпосередній близькості друг від друга розташовуються сполучні проведення, кабелі. При протіканні по них електричного струму виділяється значна кількість теплоти. При цьому можливо оплавлення ізоляції. Для відводу надлишкової теплоти від ПК служать системи вентиляції й кондиціювання повітря. При постійній дії ці системи являють собою додаткову пожежну небезпеку.

Енергопостачання виробничого приміщення здійснюється за допомогою трансформаторної станції та за допомогою двигун-генераторних агрегатів. На трансформаторних підстанціях особливу небезпеку представляють трансформатори які мають масляне охолодження. У зв'язку із цим перевагу слід віддавати сухим трансформаторам.

ВИСНОВКИ

Наразі проведено величезну кількість досліджень, які пов'язані із розробленням алгоритмів керування, що шляхом залучення мобільних роботів забезпечують розв’язання таких нетривіальних операцій: уточнення карти, планування траєкторій, обхід перешкод, що виникають під час руху, проникнення до важкодоступних зон тощо.

Подальші дослідження нових типів мобільних роботів стимулюються численними застосунками в різноманітних галузях людської діяльності (автоматизації керування рухом транспортних засобів, боротьбі з тероризмом і розмінуванні підозрілих предметів, роботі в умовах сильної задимленості під час пожежогасіння, інспектуванні територій, забруднених хімічними речовинами, самостійному патрулюванні певних територій тощо).

З метою ефективного функціонування інтелектуальні мобільні роботи устатковані системою сприйняття зовнішнього середовища, засобами аналізу ситуацій та прийняття рішень. До того ж здійснюють планування руху, зокрема і побудову траси.

Під час роботи над кваліфікаційною роботою проаналізовані методи алгоритму оптимального маршруту мобільних роботів, обрано маніпуляційний мобільний робот TurtleBot 3 Burger, складено детерміновану карту, розроблено та проаналізовано хвильовий алгоритм, розроблено програму оптимального маршруту переміщення маніпуляційного мобільного робота.

Розроблено програмне забезпечення обчислення оптимального маршруту переміщення маніпуляційного мобільного робота в умовах замкнутого невизначеного середовища виробничого приміщення.

У розділі «Охорона праці» розраховано штучне освітлення в дослідницькій лабораторії, де виконувалась кваліфікаційна робота.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008:2015 Інформація та документація «Звіти у сфері науки і техніки». Структура та правила офомрлювання. / В. Земцева; Ю. Поліщук, канд. фіз.-мат. наук; Р. Санченко, канд. техн. наук; Л. Шрамко; А. Ямчук (науковий керівник) ДП «УкрНДНЦ» від 22 червня 2015р. № 61 з 2017- 07-01.

2. Методичні вказівки з підготовки кваліфікаційної роботи для здобувачів першого (бакалаврського) рівня вищої освіти денної і заочної форми навчання спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» освітньої програми «Автоматизація та комп'ютерно-інтегровані технології» / Упоряд.: І. Ш. Невлюдов, О. І. Филипенко, О. В. Токарєва, С. П. Новоселов, О. В Сичова. – Харків: ХНУРЕ, 2023. – 64 с.

3. Методичні вказівки з підготовки кваліфікаційної роботи бакалавра для студентів усіх форм навчання спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» освітньої програми «Автоматизація та комп'ютерно-інтегровані технології» / Упоряд.: І. Ш. Невлюдов, А. О. Андрусевич, О. В. Токарєва, С. П. Новоселов, О. В Сичова. – Харків: ХНУРЕ, 2022. – 55 с.

4. Технічні засоби автоматизації: Підручник / І. Ш. Невлюдов, А. О. Андрусевич, О. І. Филипенко, Н. П. Демська, С. П. Новоселов. – Кривий Ріг : Криворізький коледж НАУ, 2019. – 366 с.

5. Робот iRobot Warrior [Електронний ресурс]. – Режим доступу: https://wikipedia.org/wiki/IRobot_Warrior/ (Дата звернення 12.04.2025). – Загл. з екрану.

6. Робот KUKA KMR iiwa [Електронний ресурс]. – Режим доступу: https://www.kuka.com/kmr-iiwa/ (Дата звернення 13.04.2025). – Загл. з екрану

7. Робот OTTO 1500 [Електронний ресурс]. – Режим доступу: http://www.plm.pw/2016/09/otto-1500-motoman-mh12f-kuka-kmr-iiwa.html/ (Дата звернення 14.04.2025). – Загл. з екрану.

8. Робот TurtleBot 3 Burger [Електронний ресурс]. – Режим доступу: https://robotics.ua/news/prototypes/6112-turtlebot_2i_novyj_mobilnyj_robot/ (Дата звернення 15.04.2025). – Загл. з екрану.

9. Handbook of Industrial Robotics. 2nd ed / Ed. by S. Y. Nof. – New York: John Wiley & Sons, 2019. – 1378 p. – ISBN 978-0-471-17783-8. – P. 3-5.

10. Хазіна С. А. Комп’ютерне моделювання фізичного процесу у різних програмних середовищах / М-во освіти і науки України, Нац. пед. ун-т імені М. П. Драгоманова. – К. : НАУ, 2018. – С. 93-97.

11. Нариньяни А. С Програмування в обмеженнях і недовизначених моделях // Інформаційні технології №7, 2018. К., Видавництво "Машинобудування". – С. 13-22.

12. Хвильовий алгоритм [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Хвильовий_алгоритм (Дата звернення 01.05.2025). – Загл. з екрану.

13. Опис хвильового алгоритму [Електронний ресурс]. – Режим доступу: https://habr.com/post/264189/ (Дата звернення 02.05.2025). – Загл. з екрану.

14. Robin Nixon. Building Dynamic Websites with PHP, MySQL, JavaScript, CSS and HTML5, 6th Edition / Publisher(s): O'Reilly Media, Inc., 2021. – 828 p.

15. Jurgen Wolf. HTML and CSS: The Comprehensive Guide / Publisher(s): Rheinwerk Computing, 2023. – 814 p.

16. Ben Frain. Responsive Web Design with HTML5 and CSS. Develop future-proof responsive websites using the latest HTML5 and CSS techniques, 3rd Edition / Publisher(s): Wiley, 2020. – 408 p.

17. Mikael Olsson. Java 17 Quick Syntax Reference. A Pocket Guide to the Java SE Language, APIs, and Library. 3rd Ed. / Publisher(s): Apress, 2022. – 132 p.

18. Douglas Crockford. How JavaScript Works / Publisher(s): Douglas Crockford, 2021. – 280 c.

19. John Paxton, Adam D. Scott & Shelley Powers. JavaScript Cookbook: Programming the Web. 3rd Ed. / Publisher(s): O'Reilly Media, Inc., 2021. – 650 p.

20. Комплекс навчально-методичного забезпечення навчальної дисципліни «Організація керування умовами праці» підготовки освітнього рівня бакалавр усіх спеціальностей та усіх напрямів університету [Електронний ресурс] / ХНУРЕ; розроб.: Т.Є. Стиценко, Г.В. Пронюк, Н.М. Сердюк. – Харків, 2017. – 108 с.