

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ
СІКОРСЬКОГО»**

Навчально-науковий інститут атомної та теплової енергетики
Кафедра інженерії програмного забезпечення в енергетиці

ДО ЗАХИСТУ ДОПУЩЕНО

В.о. завідувача кафедри
Олександр КОВАЛЬ

«__» _____ 2026 р.

Дипломна робота

**на здобуття ступеня бакалавра
за освітньо-професійною програмою «Інженерія програмного забезпечення
інтелектуальних кібер-фізичних систем в енергетиці»
спеціальності 121 Інженерія програмного забезпечення**

**на тему: «Побудова карти перешкод для робота-маніпулятора на основі
комп'ютерного зору з візуалізацією в RViz»**

Виконала:

студентка IV курсу, групи ТВ-321

Кушнір Катерина Вікторівна

(прізвище, ім'я, по батькові)

(підпис)

Керівник:

доцент, к.т.н. Варава Іван Андрійович

(посада, науковий ступінь, вчене звання, прізвище та ініціали)

(підпис)

Рецензент:

доцент кафедри кібербезпеки та комп'ютерної інженерії
Київського національного університету будівництва і архітектури,
д.т.н., доцент Гуменний Дмитро Олександрович

(посада, науковий ступінь, вчене звання, прізвище та ініціали)

(підпис)

Засвідчую, що у цій дипломній роботі
немає запозичень із праць інших
авторів без відповідних посилань.

Студент _____

(підпис)

Київ – 2026

**Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»**

Навчально-науковий інститут атомної та теплової енергетики

Кафедра інженерії програмного забезпечення в енергетиці

Рівень вищої освіти перший (бакалаврський)

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення
інтелектуальних кібер-фізичних систем в енергетиці»

ЗАТВЕРДЖУЮ

В.о. завідувача кафедри

_____ Олександр КОВАЛЬ

«___» _____ 2026р.

ЗАВДАННЯ

на дипломну роботу студенту

_____ Кушнір Катерині Вікторівні

(прізвище, ім'я, по батькові)

1. Тема роботи: «Побудова карти перешкод для робота-маніпулятора на основі
комп'ютерного зору з візуалізацією в RViz»

керівник роботи доцент, к.т.н. Варава Іван Андрійович

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «28» травня 2026р. № 1993-с від
28.05.2026

2. Строк подання студентом роботи «10» червня 2026р.

3. Вихідні дані до роботи: операційна система Ubuntu 20.04, мова програмування
Python, платформа ROS Noetic, бібліотека OpenCV, середовище візуалізації RViz,
бібліотеки NumPy та cv_bridge, RGB-камера для отримання відеоданих

4. Зміст (дипломної роботи) пояснювальної записки (перелік завдань, які потрібно
розробити): Створити модель виявлення перешкод на основі комп'ютерного зору,
оцінити відстань до об'єктів у робочій зоні маніпулятора, побудувати карту
перешкод у ROS2, реалізувати візуалізацію карти у RViz, провести тестування і
налагодження програмного продукту для подальшого використання.

5. Перелік ілюстративного матеріалу: актуальність теми, постановка задачі,
діаграма компонентів, проектування системи, оцінка відстані від об'єктів _____

6. Дата видачі завдання «30» жовтня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Строки виконання етапів роботи	Примітка
1	Отримання завдання	31.10.2025	
2	Дослідження предметної області	01.11.2025 – 01.12.2025	
3	Дослідження існуючих рішень	02.12.2025 – 31.12.2025	
4	Постановка вимог до проектування системи	01.01.2026 – 31.01.2026	
5	Розробка програмного продукту	01.02.2026 – 03.05.2026	
6	Тестування	04.05.2026 – 10.05.2026	
7	Захист програмного продукту	11.05.2026 – 15.05.2026	
8	Оформлення дипломної роботи	18.05.2026 – 31.05.2026	
9	Передзахист	01.06.2026 – 05.06.2026	
10	Захист	15.06.2026 – 19.06.2026	

Студент

(підпис)

Катерина КУШНІР

(ім'я, прізвище)

Керівник роботи

(підпис)

Іван БАРАВА

(ім'я, прізвище)

РЕФЕРАТ

Структура та обсяг дипломної роботи: робота містить 111 сторінок, 18 рисунків, 2 додатки та 35 використаних джерел.

Метою роботи є розробка програмної системи побудови карти перешкод для робота-маніпулятора на основі технологій комп'ютерного зору з можливістю візуалізації результатів у середовищі RViz. Для досягнення поставленої мети необхідно реалізувати функціональний комплекс, який включає: отримання відеоданих із камери, обробку зображень, виявлення перешкод, побудову карти навколишнього середовища, передачу даних між ROS-вузлами та візуалізацію результатів роботи системи у режимі реального часу.

Практичне значення одержаних результатів полягає у створенні програмного застосунку, який дозволяє автоматизувати процес аналізу навколишнього середовища та побудови карти перешкод для робототехнічних систем. Розроблена система може бути використана у роботах-маніпуляторах, системах автоматизованого виробництва, мобільній робототехніці та інших сферах, де необхідне своєчасне виявлення об'єктів і уникнення зіткнень.

Ключові слова: робот-маніпулятор, комп'ютерний зір, ROS, RViz, OpenCV, Python, карта перешкод, occupancy grid map, робототехнічна система, виявлення перешкод, обробка зображень, навігація.

ABSTRACT

Structure and scope of the thesis: the work contains 111 pages, 18 figures, 2 appendices and 35 references.

The purpose of the work is to develop a software system for obstacle map construction for a robotic manipulator based on computer vision technologies with visualization in the RViz environment. To achieve this goal, it is necessary to implement a functional complex that includes: capturing video data from a camera, image processing, obstacle detection, environment map construction, data transmission between ROS nodes, and real-time visualization of the system operation results.

The practical value of the obtained results lies in the development of a software application that automates the process of environment analysis and obstacle map construction for robotic systems. The developed system can be used in robotic manipulators, automated manufacturing systems, mobile robotics, and other areas where timely object detection and collision avoidance are required.

Keywords: robotic manipulator, computer vision, ROS, RViz, OpenCV, Python, obstacle map, occupancy grid map, robotic system, obstacle detection, image processing, navigation.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ	8
ВСТУП	11
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА СУЧАСНИХ МЕТОДІВ ПОБУДОВИ КАРТ ПЕРЕШКОД	13
1.1 Особливості роботів-маніпуляторів у сучасних робототехнічних системах	13
1.2 Основні задачі навіації та уникнення перешкод	15
1.3 Аналіз методів комп'ютерного зору для виявлення об'єктів	17
1.4 Порівняння з аналогами	20
1.5 Аналіз сучасних програмних платформ для робототехнічних систем	22
1.6 Постановка задачі дипломної роботи	25
Висновки до розділу 1	27
2 АНАЛІЗ ТЕХНОЛОГІЙ ТА ІНСТРУМЕНТАЛЬНИХ ЗАСОБІВ РОЗРОБКИ	29
2.1 Операційна система Ubuntu для робототехнічних систем	29
2.2 Платформа Robot Operating System (ROS)	31
2.3 Середовище візуалізації RViz	34
2.4 Бібліотека OpenCV для задач комп'ютерного зору	37
2.5 Мова програмування Python	40
2.6 Методи цифрової обробки зображень	42
2.7 Алгоритми виявлення та класифікації перешкод	45
Висновки до розділу 2	48
3 ПРОЕКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ ПОБУДОВИ КАРТИ ПЕРЕШКОД	50
3.1 Архітектура програмної системи	50
3.2 Проектування функціональних модулів системи	53
3.3 Організація взаємодії між компонентами ROS	56
3.4 Розробка алгоритму отримання відеоданих	58
3.5 Розробка алгоритму виявлення перешкод	60
3.6 Розробка алгоритму побудови карти перешкод	63
3.7 Реалізація візуалізації карти в RViz	65
3.8 Інтеграція системи з роботом-маніпулятором	67

Висновки до розділу 3	70
4 РОЗРОБКА ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАСТОСУНКУ	71
4.1 Реалізація програмного забезпечення	71
4.2 Налаштування середовища ROS та RViz	74
4.3 Тестування системи виявлення перешкод	77
4.4 Аналіз точності побудови карти	79
4.5 Оцінка продуктивності системи	81
4.6 Аналіз результатів роботи програмного застосунку	84
Висновки до розділу 4	86
5 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА ПІД ЧАС РОБОТИ З РОБОТОТЕХНІЧНИМИ СИСТЕМАМИ	87
5.1 Аналіз небезпечних факторів під час роботи з роботами-маніпуляторами	87
5.2 Вимоги електробезпеки при роботі з обладнанням	89
5.3 Організація безпечного робочого місця	91
Висновки до розділу 5	93
ВИСНОВКИ	95
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	97
ДОДАТОК А Лістинг програми	100
ДОДАТОК Б Презентація	104

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

Canny Edge Detection – алгоритм виділення контурів об’єктів на цифровому зображенні.

Contour Detection (Виявлення контурів) – метод визначення меж об’єктів на зображенні.

Gaussian Blur – алгоритм згладжування зображення для зменшення шумів та покращення якості обробки.

FPS (Frames Per Second) – кількість кадрів, які система обробляє за одну секунду.

Image Segmentation (Сегментація зображення) – процес поділу зображення на окремі області для спрощення аналізу об’єктів.

LiDAR (Light Detection and Ranging) – сенсор, який визначає відстань до об’єктів за допомогою лазерного випромінювання.

Marker – графічний об’єкт у RViz, який використовується для відображення точок, ліній, фігур та інших елементів сцени.

MarkerArray – набір маркерів для одночасного відображення декількох об’єктів у RViz.

Message (Повідомлення) – структура даних, що використовується для передачі інформації між вузлами ROS.

Node (Вузол) – окремий програмний модуль у ROS, який виконує визначену функцію та взаємодіє з іншими вузлами через повідомлення.

Occupancy Grid Map – карта зайнятості; спосіб представлення навколишнього середовища у вигляді набору комірок, які містять інформацію про наявність або відсутність перешкод.

OpenCV (Open Source Computer Vision Library) – бібліотека з відкритим кодом для реалізації алгоритмів комп’ютерного зору та цифрової обробки зображень.

Point Cloud (Хмара точок) – набір просторових координат точок, що описують форму об'єктів та навколишнього середовища.

Publisher – вузол ROS, який публікує повідомлення у визначену тему.

PUMA 560 (Programmable Universal Machine for Assembly) – промисловий шестивісний робот-маніпулятор, який використовується для моделювання та досліджень у робототехніці.

Python – високорівнева мова програмування, що використовується для розробки програмного забезпечення робототехнічних систем.

Real-Time Processing (Обробка в режимі реального часу) – режим роботи системи, за якого дані обробляються без суттєвих затримок.

RGB-камера (Red, Green, Blue Camera) – камера, яка формує кольорове зображення на основі трьох базових кольорових каналів.

RGB-D камера (Red, Green, Blue and Depth Camera) – камера, що додатково отримує інформацію про глибину сцени.

ROS (Robot Operating System) – програмна платформа для розробки робототехнічних систем, яка забезпечує взаємодію між окремими компонентами робота.

ROS2 (Robot Operating System 2) – сучасна версія платформи ROS, орієнтована на підвищену продуктивність, безпеку та роботу в режимі реального часу.

RViz (ROS Visualization Tool) – середовище візуалізації, що використовується для відображення моделі робота, даних сенсорів, карт навколишнього середовища та результатів роботи алгоритмів.

SLAM (Simultaneous Localization and Mapping) – технологія одночасного визначення положення робота та побудови карти навколишнього середовища.

Subscriber – вузол ROS, який отримує повідомлення із визначеної теми.

TF (Transform Frames) – механізм ROS для визначення та відстеження взаємного розташування систем координат.

Thresholding (Порогова обробка) – метод сегментації зображення шляхом розділення пікселів за значенням яскравості.

Topic (Тема) – канал передачі повідомлень між вузлами ROS за принципом видавець-підписник.

Ubuntu – операційна система на базі Linux, яка використовується як основна платформа для роботи з ROS та робототехнічними застосунками.

URDF (Unified Robot Description Format) – формат опису моделі робота для використання у ROS.

Комп'ютерний зір (англ. Computer Vision) – галузь штучного інтелекту, що займається отриманням, обробкою та аналізом інформації із цифрових зображень і відеопотоків.

Робот-маніпулятор – автоматизована механічна система, призначена для виконання операцій переміщення та маніпулювання об'єктами у робочому просторі.

ВСТУП

У сучасних умовах стрімкого розвитку робототехніки та автоматизації виробничих процесів усе більшого значення набуває використання інтелектуальних робототехнічних систем, здатних працювати в умовах змінного навколишнього середовища. Роботи-маніпулятори широко застосовуються в промисловості, логістиці, медицині, енергетиці та інших галузях, де необхідна висока точність, швидкість виконання операцій та мінімізація впливу людського фактора.

Однією з найважливіших задач сучасної робототехніки є забезпечення безпечної взаємодії робота з навколишнім середовищем. Для виконання поставлених завдань робот-маніпулятор повинен мати можливість аналізувати простір навколо себе, визначати наявність перешкод та своєчасно реагувати на зміни робочого середовища. Помилки під час визначення положення об'єктів можуть призвести до зіткнень, пошкодження обладнання або порушення технологічного процесу.

Для вирішення цієї задачі все частіше використовуються методи комп'ютерного зору. Завдяки цифровим камерам та алгоритмам обробки зображень робот може отримувати інформацію про навколишні об'єкти, визначати їх розташування та відображати результати у зручному для користувача вигляді. Використання комп'ютерного зору дозволяє знизити вартість системи порівняно зі спеціалізованими сенсорами та забезпечити достатню точність для багатьох практичних задач.

Сучасні робототехнічні платформи, зокрема ROS2, надають широкі можливості для розробки модульних систем керування роботами та обробки даних від сенсорів. Для візуалізації результатів роботи широко застосовується середовище RViz, яке дозволяє відображати моделі роботів, карти навколишнього середовища та виявленні перешкоди.

Актуальність даної роботи полягає у необхідності створення програмної системи, здатної виявляти перешкоди за допомогою комп'ютерного зору та будувати карту робочого простору робота-маніпулятора. Реалізація такої системи дозволяє підвищити безпечність роботи роботизованих комплексів та створює основу для подальшого розвитку інтелектуальних систем керування.

Метою дипломної роботи є розробка програмної системи побудови карти перешкод для робота-маніпулятора PUMA 560 на основі комп'ютерного зору з візуалізацією результатів у середовищі RViz.

Об'єктом дослідження є процес виявлення та відображення перешкод у робочому просторі робота-маніпулятора. Предметом дослідження є методи комп'ютерного зору, засоби ROS2 та алгоритми побудови карти перешкод для робототехнічних систем.

Практичне значення роботи полягає у створенні програмного продукту, який може використовуватися для візуалізації перешкод та аналізу робочого простору роботів-маніпуляторів у режимі реального часу.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ПОБУДОВИ КАРТ ПЕРЕШКОД ДЛЯ РОБОТІВ-МАНІПУЛЯТОРІВ

1.1 Особливості роботів-маніпуляторів у сучасних робототехнічних системах

У сучасних умовах стрімкого розвитку автоматизації та інтелектуальних технологій роботи-маніпулятори набувають широкого застосування в багатьох сферах діяльності людини. Вони активно використовуються у промисловості, логістиці, медицині, енергетиці, аграрному секторі та автоматизованих виробничих комплексах. Використання роботизованих систем дозволяє підвищити точність виконання операцій, зменшити вплив людського фактора, оптимізувати виробничі процеси та забезпечити безпечну роботу в небезпечних або важкодоступних умовах [1].

Робот-маніпулятор являє собою автоматизовану механічну систему, здатну виконувати переміщення об'єктів або робочих інструментів у визначеному просторі відповідно до заданої програми. Основними складовими робота-маніпулятора є механічна конструкція, система приводів, контролери керування, сенсори та програмне забезпечення. Завдяки поєднанню апаратної та програмної частин сучасні маніпулятори можуть виконувати складні технологічні операції з високою точністю та швидкістю [3].

Однією з ключових особливостей сучасних роботів-маніпуляторів є здатність працювати в динамічному середовищі. На відміну від класичних автоматизованих механізмів, які функціонують у жорстко визначених умовах, сучасні робототехнічні системи повинні адаптуватися до змін навколишнього середовища, аналізувати перешкоди та приймати рішення у режимі реального часу [10].

Для безпечного функціонування робот-маніпулятор повинен мати можливість виявляти перешкоди та будувати карту навколишнього середовища.

Наявність системи аналізу простору дозволяє уникати зіткнень із предметами та забезпечувати коректне переміщення виконавчих механізмів. Помилки в роботі системи навігації можуть призвести до аварійних ситуацій, пошкодження обладнання або порушення виробничого процесу [15].

Одним із найефективніших підходів до аналізу навколишнього середовища є використання технологій комп'ютерного зору. Комп'ютерний зір забезпечує отримання інформації про форму, положення та переміщення об'єктів за допомогою камер і алгоритмів обробки зображень. На відміну від звичайних датчиків відстані, системи комп'ютерного зору здатні отримувати значно більший обсяг інформації про навколишній простір, що дозволяє формувати більш точні карти перешкод [5].

У сучасних робототехнічних системах широко використовуються RGB-D камери, стереокамери та LiDAR-системи. Використання depth-сенсорів дозволяє отримувати інформацію про глибину сцени та формувати тривимірне представлення навколишнього середовища [2].

Для розробки робототехнічних систем широко використовується платформа Robot Operating System (ROS). Дана система являє собою набір програмних інструментів і бібліотек, які забезпечують взаємодію між окремими компонентами робота. Використання ROS дозволяє організувати передачу даних між камерами, алгоритмами обробки зображень, системами керування та модулями побудови карти середовища [9].

Важливим компонентом ROS є середовище RViz, яке використовується для візуалізації роботи робототехнічних систем. За допомогою RViz можна відображати модель робота, траєкторії руху, дані сенсорів та карту навколишнього середовища у двовимірному або тривимірному просторі [8].

На рисунку 1.1 представлено загальну структуру робототехнічної системи з використанням комп'ютерного зору.

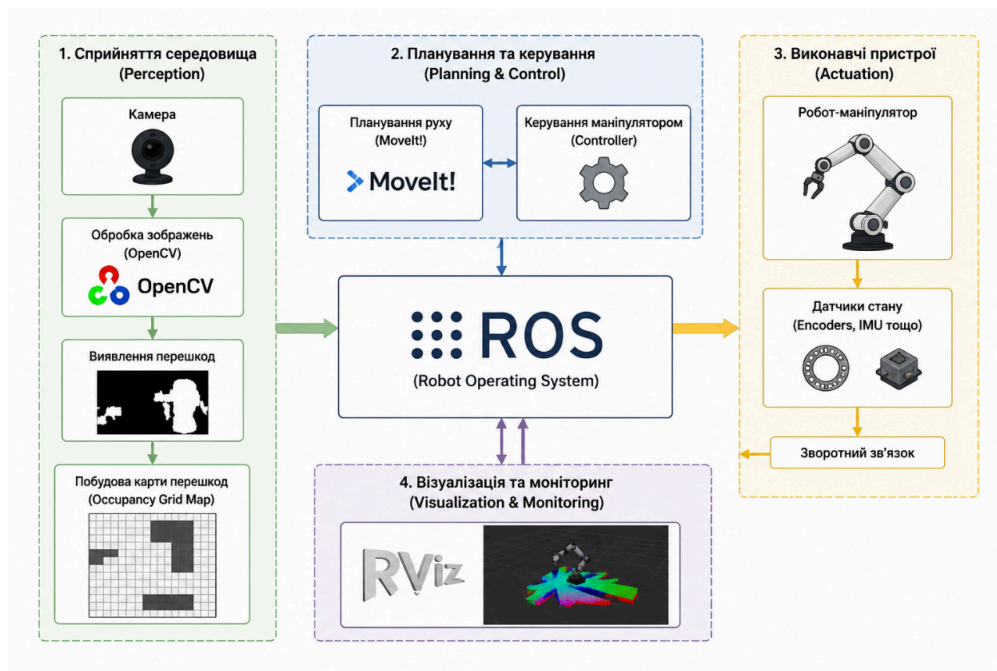


Рисунок 1.1 – Структура робототехнічної системи

Таким чином, сучасні роботи-маніпулятори є складними інтелектуальними системами, ефективність роботи яких значною мірою залежить від здатності аналізувати навколишнє середовище та своєчасно виявляти перешкоди. Використання технологій комп'ютерного зору, платформ ROS та засобів візуалізації RViz дозволяє створювати адаптивні та безпечні робототехнічні системи, здатні працювати в умовах реального середовища.

1.2 Основні задачі навігації та уникнення перешкод

Однією з найважливіших задач у робототехнічних системах є забезпечення безпечної та ефективної навігації робота у навколишньому середовищі. Для роботів-маніпуляторів навігація полягає не лише у переміщенні самого робота, а й у правильному визначенні траєкторії руху виконавчого механізму з урахуванням наявних перешкод та обмежень робочого простору [15].

Під час роботи роботизованої системи можуть виникати різноманітні перешкоди, які ускладнюють виконання поставлених задач. До таких перешкод

належать статичні об'єкти, наприклад стіни, обладнання або інструменти, а також динамічні об'єкти, які змінюють своє положення у просторі. Особливу складність створюють рухомі об'єкти, оскільки система повинна не лише визначити їхнє поточне положення, а й прогнозувати можливий напрямок руху [10].

Основною задачею системи навігації є побудова безпечної траєкторії руху робота від початкової точки до кінцевої позиції. Для цього необхідно забезпечити:

- отримання інформації про навколишнє середовище;
- виявлення перешкод;
- аналіз просторового розташування об'єктів;
- побудову карти середовища;
- розрахунок безпечної траєкторії руху;
- уникнення зіткнень у реальному часі.

Одним із ключових етапів навігації є побудова карти перешкод. Карта являє собою цифрове представлення навколишнього середовища, яке містить інформацію про розташування об'єктів та вільний простір для переміщення робота [25].

Для виявлення перешкод у робототехніці застосовуються різноманітні сенсори. Найпоширенішими є камери, ультразвукові датчики, інфрачервоні сенсори та LiDAR-системи. Проте останніми роками найбільшої популярності набули саме системи комп'ютерного зору, оскільки вони дозволяють отримувати детальну інформацію про навколишнє середовище за допомогою звичайних камер [5].

Система уникнення перешкод повинна працювати в режимі реального часу. Це означає, що робот має постійно аналізувати нові дані із сенсорів та швидко реагувати на зміни навколишнього середовища. У випадку появи нової перешкоди система повинна оперативно перебудувати маршрут або змінити траєкторію руху маніпулятора [27].

Таким чином, задачі навігації та уникнення перешкод є одними з найважливіших складових сучасних робототехнічних систем. Ефективність роботи робота-маніпулятора значною мірою залежить від точності аналізу навколишнього середовища, швидкості обробки даних та якості алгоритмів побудови карти перешкод.

1.3 Аналіз методів комп'ютерного зору для виявлення об'єктів

Комп'ютерний зір є одним із найважливіших напрямів сучасної робототехніки та штучного інтелекту. Основною задачею комп'ютерного зору є отримання, аналіз та інтерпретація інформації із цифрових зображень або відеопотоку. У робототехнічних системах технології комп'ютерного зору застосовуються для навігації, виявлення об'єктів, аналізу навколишнього середовища та побудови карт перешкод [5].

Для роботів-маніпуляторів системи комп'ютерного зору мають особливе значення, оскільки саме вони дозволяють роботу взаємодіяти з навколишнім середовищем. Використання камер та алгоритмів обробки зображень забезпечує можливість визначати форму, розмір, положення та рух об'єктів у просторі. Отримана інформація використовується для побудови траєкторії руху та уникнення зіткнень із перешкодами [12].

Одним із найпоширеніших підходів у комп'ютерному зорі є цифрова обробка зображень. До основних методів обробки належать:

- фільтрація шумів;
- перетворення кольорового простору;
- виділення контурів;
- сегментація зображення;
- визначення ключових точок;

- розпізнавання об'єктів.

Першим етапом аналізу зображення зазвичай є попередня обробка. Вона необхідна для покращення якості зображення та зменшення впливу шумів. Для цього використовуються різноманітні фільтри, наприклад Gaussian Blur або Median Blur. Після цього зображення може бути переведене у відтінки сірого, що дозволяє спростити подальшу обробку та зменшити обсяг обчислень [14].

Одним із найбільш популярних методів виявлення об'єктів є виділення контурів. Контури дозволяють визначити межі об'єктів та оцінити їхню форму. Для цього часто використовується алгоритм Canny Edge Detection, який забезпечує ефективне визначення границь об'єктів на зображенні [4].

Після виявлення контурів система може аналізувати розміри та положення перешкод. Для задач робототехніки важливим є не лише визначення форми об'єкта, а й оцінка його координат у просторі.

Для задач розпізнавання об'єктів також використовуються методи сегментації зображення. Сегментація полягає у поділі зображення на окремі області відповідно до кольору, яскравості або текстури. Це дозволяє виділяти об'єкти із загального фону та аналізувати їх окремо [12].

У сучасних системах комп'ютерного зору все частіше застосовуються методи машинного навчання та нейронні мережі. Особливу популярність отримали згорткові нейронні мережі (CNN), які демонструють високу ефективність у задачах класифікації та розпізнавання об'єктів [21].

Одним із найбільш відомих сучасних алгоритмів є YOLO (You Only Look Once). Даний алгоритм дозволяє виконувати виявлення об'єктів у режимі реального часу та забезпечує високу швидкість роботи. YOLO широко використовується у робототехнічних системах завдяки можливості одночасно визначати декілька об'єктів на одному кадрі [13].

Крім YOLO, у системах комп'ютерного зору використовуються також алгоритми SSD (Single Shot Detector), Faster R-CNN та Mask R-CNN. Кожен із них має свої переваги та недоліки залежно від вимог до точності та швидкодії системи.

Для реалізації алгоритмів комп'ютерного зору часто використовується бібліотека OpenCV. Вона містить великий набір готових інструментів для обробки зображень, роботи з відео, виділення контурів, сегментації та машинного навчання [7].

Важливою задачею комп'ютерного зору є визначення відстані до об'єктів. Для цього можуть використовуватися стереокамери або RGB-D камери, які дозволяють отримувати інформацію про глибину сцени. На основі цих даних система будує тривимірну модель навколишнього середовища та визначає розташування перешкод відносно робота [29].

На рисунку 1.3 представлено приклад роботи системи комп'ютерного зору для виявлення перешкод.

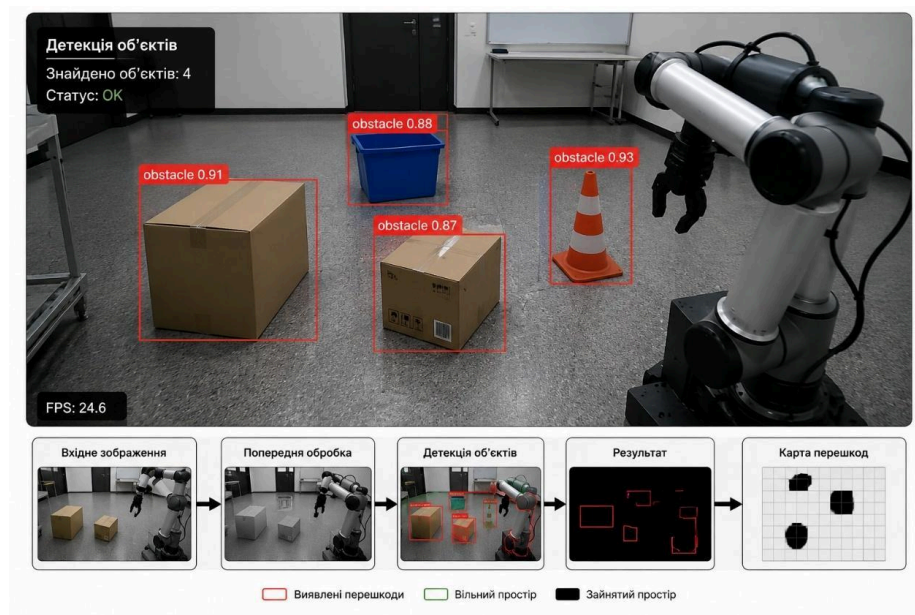


Рисунок 1.3 – Виявлення об'єктів за допомогою комп'ютерного зору

Таким чином, методи комп'ютерного зору є важливою складовою сучасних робототехнічних систем. Вони дозволяють роботам-маніпуляторам аналізувати навколишнє середовище, виявляти перешкоди та забезпечувати безпечне переміщення у робочому просторі. Використання сучасних алгоритмів обробки зображень та машинного навчання дозволяє значно підвищити ефективність роботи систем навігації та побудови карт перешкод.

1.4 Порівняння з аналогами

Однією з основних задач робототехнічних систем є побудова карти навколишнього середовища для забезпечення безпечної навігації та уникнення перешкод. Карта середовища являє собою цифрову модель простору, яка містить інформацію про розташування об'єктів, перешкод та вільних зон для переміщення робота [10].

Побудова карти дозволяє робототехнічній системі орієнтуватися у просторі, визначати власне положення та планувати траєкторію руху. Без використання карти навігація робота була б значно менш ефективною та могла б призводити до зіткнень із перешкодами [15].

У сучасній робототехніці використовуються різні типи карт навколишнього середовища. Найбільш поширеними є:

- двовимірні карти;
- тривимірні карти;
- карти глибини;
- occupancy grid map;
- point cloud map.

Двовимірні карти є найбільш простими у реалізації та широко застосовуються у мобільній робототехніці. Такі карти відображають положення перешкод на площині та дозволяють визначати вільний простір для переміщення робота. Основною перевагою двовимірних карт є невеликі вимоги до обчислювальних ресурсів [26].

Тривимірні карти забезпечують значно точніше представлення навколишнього середовища. Вони дозволяють враховувати висоту об'єктів, форму перешкод та складні просторові структури. Побудова 3D-карт особливо важлива для роботів-маніпуляторів, які працюють у складному середовищі та взаємодіють із великою кількістю об'єктів [25].

Одним із найпоширеніших методів побудови карти є occupancy grid map. Даний метод передбачає поділ простору на окремі комірки, кожна з яких містить інформацію про наявність або відсутність перешкоди. Якщо в певній області виявлено об'єкт, відповідна комірка позначається як зайнята. У випадку відсутності перешкод комірка вважається вільною [10].

Для побудови карт навколишнього середовища широко використовуються алгоритми SLAM (Simultaneous Localization and Mapping). Основною задачею SLAM є одночасне визначення положення робота та створення карти простору [26].

Алгоритми SLAM активно застосовуються у системах автономної навігації, мобільних роботах та роботах-маніпуляторах. Для роботи алгоритмів можуть використовуватися камери, LiDAR, depth-сенсори або комбінація декількох типів датчиків [27].

Одним із сучасних підходів є використання point cloud map — карти хмари точок. У даному випадку навколишнє середовище представляється у вигляді великої кількості точок у тривимірному просторі. Кожна точка містить координати та інформацію про поверхню об'єкта. Хмари точок широко використовуються у робототехніці та системах комп'ютерного зору [25].

Для побудови карти за допомогою комп'ютерного зору робот отримує дані з камери у вигляді відеопотоку або окремих зображень. Після обробки зображення система визначає положення об'єктів та формує модель навколишнього середовища. Отримані дані можуть передаватися до ROS та відображатися у середовищі RViz [8].

Візуалізація карти перешкод у RViz дозволяє аналізувати роботу алгоритмів у режимі реального часу. Користувач може бачити положення робота, виявлені перешкоди та траєкторію руху. Це значно спрощує тестування та налагодження робототехнічної системи [8].

Таким чином, методи побудови карт навколишнього середовища є важливою складовою сучасної робототехніки. Використання алгоритмів SLAM, occupancy

grid map та технологій комп'ютерного зору дозволяє роботам-маніпуляторам ефективно орієнтуватися у просторі та безпечно взаємодіяти з навколишнім середовищем.

1.5 Аналіз сучасних програмних платформ для робототехнічних систем

Розробка сучасних робототехнічних систем потребує використання спеціалізованих програмних платформ, які забезпечують взаємодію між окремими компонентами робота, обробку даних сенсорів, побудову карт навколишнього середовища та керування виконавчими механізмами. Використання готових платформ дозволяє значно спростити процес розробки програмного забезпечення та забезпечити сумісність між різними модулями системи [9].

Однією з найпоширеніших платформ у сфері робототехніки є Robot Operating System (ROS). Дана система являє собою набір програмних бібліотек та інструментів, які використовуються для створення робототехнічних застосунків. Основною перевагою ROS є модульна архітектура, що дозволяє розділяти систему на окремі незалежні компоненти [6].

У ROS кожен функціональний модуль реалізується у вигляді окремого вузла (node). Вузли можуть обмінюватися інформацією через спеціальні канали передачі повідомлень — topics. Такий підхід забезпечує гнучкість системи та спрощує інтеграцію нових програмних компонентів [9].

Для робототехнічних систем ROS забезпечує:

- передачу даних між сенсорами;
- роботу із камерами;
- підтримку навігації;
- побудову карт середовища;
- керування роботами;

- інтеграцію із системами комп'ютерного зору.

Однією з важливих переваг ROS є підтримка великої кількості готових пакетів та бібліотек. Це дозволяє використовувати вже реалізовані алгоритми та скорочує час розробки програмного забезпечення [6].

Для задач візуалізації у ROS використовується середовище RViz. Даний інструмент дозволяє відображати:

- модель робота;
- карту навколишнього середовища;
- траєкторії руху;
- координати об'єктів;
- хмари точок;
- дані сенсорів.

RViz широко використовується під час розробки та тестування робототехнічних систем, оскільки дозволяє аналізувати роботу алгоритмів у режимі реального часу [8].

Ще одним важливим компонентом робототехнічних платформ є MoveIt. Даний програмний пакет використовується для планування траєкторії руху роботів-маніпуляторів та забезпечення уникнення зіткнень. MoveIt інтегрується з ROS та дозволяє реалізувати кінематичні розрахунки, аналіз робочого простору та автоматичне планування руху [18].

Для моделювання робототехнічних систем часто використовується середовище Gazebo. Воно дозволяє створювати віртуальні моделі роботів та тестувати алгоритми навігації без використання реального обладнання. Gazebo підтримує фізичне моделювання, роботу з сенсорами та інтеграцію з ROS [30].

У сфері комп'ютерного зору найбільш поширеною платформою є бібліотека OpenCV. Вона забезпечує обробку зображень, аналіз відеоданих, виявлення об'єктів та реалізацію алгоритмів машинного навчання [7].

Для задач штучного інтелекту та розпізнавання об'єктів використовуються також TensorFlow та PyTorch. Дані фреймворки дозволяють створювати та навчати нейронні мережі для задач комп'ютерного зору та автоматичного аналізу зображень [22].

На рисунку 1.5 представлено взаємодію основних програмних компонентів робототехнічної системи.

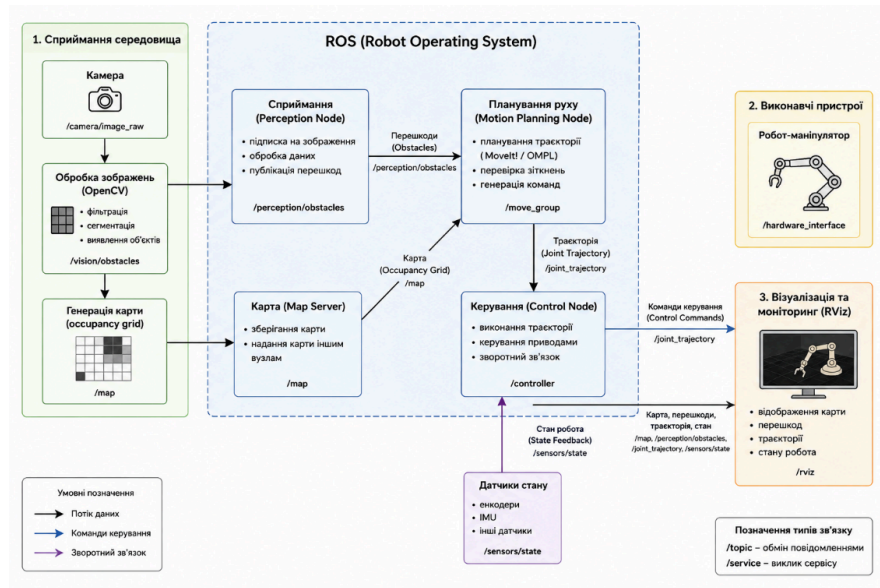


Рисунок 1.5 – Програмні компоненти робототехнічної системи

Під час вибору програмної платформи для реалізації системи побудови карти перешкод важливими критеріями є:

- стабільність роботи;
- підтримка обробки даних у режимі реального часу;
- можливість інтеграції із сенсорами;
- наявність документації;
- підтримка спільноти розробників.

У результаті проведеного аналізу було встановлено, що найбільш ефективним рішенням для реалізації системи побудови карти перешкод є використання платформи ROS у поєднанні з бібліотекою OpenCV та середовищем

RViz. Даний підхід забезпечує ефективну взаємодію між компонентами системи та дозволяє реалізувати обробку відеоданих і побудову карти навколишнього середовища у режимі реального часу.

1.6 Постановка задачі дипломної роботи

У результаті проведеного аналізу предметної області було встановлено, що сучасні робототехнічні системи потребують ефективних методів аналізу навколишнього середовища та побудови карти перешкод для забезпечення безпечної навігації роботів-маніпуляторів.

Незважаючи на значний розвиток технологій робототехніки та комп'ютерного зору, задачі виявлення об'єктів та побудови карти середовища залишаються актуальними. Особливо важливими є питання забезпечення роботи системи у режимі реального часу, підвищення точності визначення координат перешкод та ефективної візуалізації результатів роботи алгоритмів [10].

Основною метою дипломної роботи є розробка програмної системи побудови карти перешкод для робота-маніпулятора на основі технологій комп'ютерного зору з можливістю візуалізації результатів у середовищі RViz.

Для досягнення поставленої мети необхідно виконати такі задачі:

- провести аналіз предметної області робототехнічних систем;
- дослідити сучасні методи побудови карт навколишнього середовища;
- проаналізувати алгоритми комп'ютерного зору для виявлення перешкод;
- виконати аналіз програмних платформ ROS та RViz;
- розробити архітектуру програмної системи;
- реалізувати алгоритм отримання та обробки відеоданих;
- реалізувати алгоритм виявлення перешкод;

- реалізувати побудову карти навколишнього середовища;
- забезпечити візуалізацію результатів у RViz;
- провести тестування та оцінку ефективності роботи системи.

Об'єктом дослідження є процес побудови карти навколишнього середовища для робототехнічних систем.

Предметом дослідження є методи комп'ютерного зору, алгоритми виявлення перешкод та програмні засоби побудови карти для робота-маніпулятора.

Під час виконання роботи планується використання:

- операційної системи Ubuntu;
- платформи ROS;
- середовища RViz;
- бібліотеки OpenCV;
- мови програмування Python.

Очікуваним результатом роботи є створення програмного застосунку, який забезпечуватиме:

- отримання відеоданих із камери;
- виявлення перешкод;
- побудову карти навколишнього середовища;
- передачу даних у ROS;
- візуалізацію результатів у RViz.

На рисунку 1.6 представлено загальну схему роботи системи побудови карти перешкод.



Рисунок 1.6 – Загальна схема роботи системи

Практичне значення роботи полягає у можливості використання розробленої системи у задачах навігації робототехнічних комплексів, автоматизації виробничих процесів та системах уникнення зіткнень для роботів-маніпуляторів.

Таким чином, реалізація системи побудови карти перешкод на основі технологій комп'ютерного зору є актуальною задачею сучасної робототехніки та має важливе практичне значення для розвитку інтелектуальних робототехнічних систем.

Висновки до розділу 1

У рамках У даному розділі було проведено аналіз предметної області побудови карт перешкод для роботів-маніпуляторів та досліджено сучасні підходи до реалізації систем навігації у робототехнічних комплексах.

Було розглянуто особливості сучасних роботів-маніпуляторів та визначено важливість використання систем комп'ютерного зору для аналізу навколишнього середовища та уникнення зіткнень. Проведений аналіз показав, що ефективність роботи робототехнічних систем значною мірою залежить від точності виявлення перешкод та швидкості обробки даних [1].

У процесі дослідження було проаналізовано сучасні методи комп'ютерного зору, алгоритми виявлення об'єктів та методи побудови карт навколишнього середовища. Особливу увагу приділено алгоритмам SLAM, occupancy grid map та технологіям цифрової обробки зображень [10].

Також було проведено аналіз сучасних програмних платформ для робототехнічних систем. Встановлено, що використання ROS, OpenCV та RViz забезпечує ефективну взаємодію між програмними компонентами системи та дозволяє реалізувати побудову карти перешкод у режимі реального часу [6].

У результаті проведеного аналізу було сформульовано мету та основні задачі дипломної роботи, визначено об'єкт і предмет дослідження, а також обґрунтовано вибір програмних засобів для реалізації системи побудови карти перешкод для робота-маніпулятора.

2 АНАЛІЗ ТЕХНОЛОГІЙ ТА ІНСТРУМЕНТАЛЬНИХ ЗАСОБІВ РОЗРОБКИ

2.1 Операційна система Ubuntu для робототехнічних систем

Для реалізації сучасних робототехнічних систем важливе значення має вибір операційної системи, яка забезпечує стабільну роботу програмного забезпечення, підтримку апаратних компонентів та можливість інтеграції із робототехнічними платформами. У сфері робототехніки найбільш поширеним рішенням є операційна система Ubuntu, яка базується на Linux та активно використовується для роботи з платформою ROS [17].

Ubuntu є операційною системою з відкритим вихідним кодом, що забезпечує широкі можливості для розробки та тестування програмного забезпечення. Вона підтримує багатозадачність, роботу із мережевими службами, драйверами сенсорів та програмними бібліотеками для робототехніки [6].

Однією з основних переваг Ubuntu є висока стабільність роботи. Для робототехнічних систем це особливо важливо, оскільки помилки або збої операційної системи можуть призвести до некоректної роботи алгоритмів навігації або втрати даних із сенсорів.

Операційна система Ubuntu має велику кількість програмних пакетів та бібліотек, які використовуються у сфері робототехніки. Зокрема, вона забезпечує підтримку:

- ROS;
- OpenCV;
- Gazebo;
- RViz;

- MoveIt;
- TensorFlow;
- PyTorch.

Важливою перевагою Ubuntu є підтримка командного рядка та автоматизації процесів. Це дозволяє швидко запускати окремі програмні модулі, налаштовувати середовище розробки та виконувати тестування системи.

Для розробки робототехнічних застосунків Ubuntu також забезпечує ефективну роботу з багатопотоковими процесами. Це дозволяє одночасно виконувати:

- отримання відеоданих;
- обробку зображень;
- передачу даних у ROS;
- побудову карти;
- візуалізацію результатів.

Ще однією важливою особливістю Ubuntu є сумісність із платформою ROS. Більшість версій ROS офіційно підтримуються саме для Ubuntu, що спрощує встановлення та налаштування робототехнічного програмного забезпечення [6].

На рисунку 2.1 представлено загальну структуру взаємодії Ubuntu із компонентами робототехнічної системи.



Рисунок 2.1 – Використання Ubuntu у робототехнічній системі

У процесі виконання дипломної роботи використовується операційна система Ubuntu 20.04 LTS, оскільки дана версія забезпечує стабільну роботу ROS Noetic та підтримує необхідні бібліотеки для реалізації систем комп'ютерного зору.

Таким чином, Ubuntu є ефективною платформою для розробки робототехнічних систем завдяки стабільності роботи, підтримці сучасних програмних бібліотек та сумісності із платформою ROS.

2.2 Платформа Robot Operating System (ROS)

Robot Operating System (ROS) є однією з найпоширеніших програмних платформ у сфері робототехніки. Вона являє собою набір програмних бібліотек та інструментів, які використовуються для створення програмного забезпечення робототехнічних систем [9].

Основною перевагою ROS є модульна архітектура, яка дозволяє розділяти складну систему на окремі незалежні компоненти. Кожен функціональний

елемент реалізується у вигляді окремого вузла (node), який може взаємодіяти з іншими вузлами через спеціальні канали передачі повідомлень — topics [6].

ROS забезпечує:

- передачу даних між компонентами системи;
- роботу із сенсорами;
- обробку відеоданих;
- побудову карт навколишнього середовища;
- керування роботами;
- інтеграцію із системами візуалізації.

Взаємодія між вузлами здійснюється за допомогою механізму publisher/subscriber. Один вузол може передавати дані, а інший — отримувати їх та виконувати подальшу обробку. Даний підхід дозволяє забезпечити незалежність компонентів системи та спрощує масштабування програмного забезпечення [9].

У межах робототехнічної системи ROS використовується для:

- передачі відеопотоку із камери;
- передачі координат перешкод;
- оновлення карти навколишнього середовища;
- взаємодії із RViz;
- інтеграції із роботом-маніпулятором.

Основними компонентами ROS є:

- nodes;
- topics;
- messages;

- services;
- rosbag;
- launch-файли.

Launch-файли використовуються для автоматичного запуску декількох вузлів одночасно. Це значно спрощує процес запуску робототехнічної системи та дозволяє швидко виконувати тестування програмного забезпечення.

Важливою перевагою ROS є велика кількість готових пакетів. Наприклад, для задач побудови карти середовища можуть використовуватися готові пакети SLAM, а для керування маніпуляторами — MoveIt [18].

Для тестування робототехнічних систем ROS інтегрується із середовищем Gazebo. Це дозволяє моделювати роботу робота та перевіряти алгоритми без використання реального обладнання [30].

На рисунку 2.2 представлено структуру взаємодії вузлів у ROS.

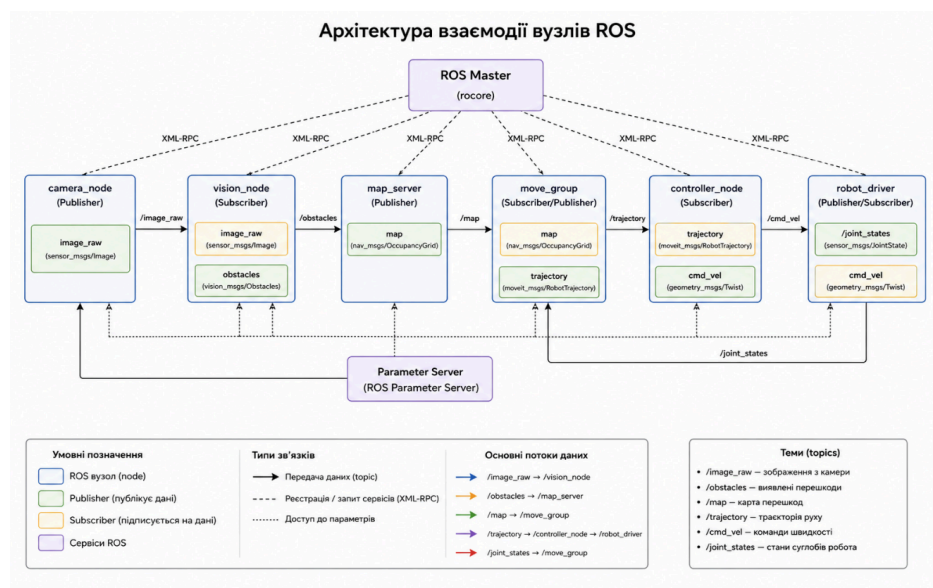


Рисунок 2.2 – Архітектура взаємодії вузлів ROS

Однією з важливих особливостей ROS є підтримка роботи у режимі реального часу. Це особливо важливо для задач комп'ютерного зору та навігації, де система повинна швидко реагувати на зміни навколишнього середовища.

Під час виконання дипломної роботи використовується ROS Noetic, який забезпечує стабільну роботу із Ubuntu 20.04 та підтримує необхідні інструменти для реалізації системи побудови карти перешкод.

Таким чином, ROS є ефективною платформою для створення робототехнічних застосунків завдяки модульній архітектурі, підтримці великої кількості програмних пакетів та можливості інтеграції з різними апаратними та програмними компонентами.

2.3 Середовище візуалізації RViz

Для аналізу роботи робототехнічних систем та відображення результатів функціонування алгоритмів широко використовується середовище візуалізації RViz. Даний інструмент входить до складу платформи ROS та призначений для графічного представлення даних робототехнічної системи у двовимірному та тривимірному просторі [8].

RViz є одним із основних інструментів під час розробки робототехнічних застосунків, оскільки дозволяє в режимі реального часу аналізувати роботу сенсорів, алгоритмів навігації та систем побудови карт навколишнього середовища.

Основними можливостями RViz є:

- відображення моделі робота;
- візуалізація карти навколишнього середовища;
- відображення координат перешкод;
- перегляд траєкторії руху;
- відображення систем координат TF;
- візуалізація даних камер та сенсорів;

- робота з хмарами точок.

У робототехнічних системах RViz використовується для тестування алгоритмів та контролю правильності передачі даних між ROS-вузлами. За допомогою RViz користувач може аналізувати роботу системи без необхідності використання додаткових програмних засобів [6].

Для відображення карти навколишнього середовища використовується компонент Map. Він дозволяє переглядати occupancy grid map та аналізувати положення перешкод у робочому просторі.

Компонент RobotModel застосовується для відображення тривимірної моделі робота-маніпулятора. Це дозволяє контролювати положення окремих ланок робота та перевіряти правильність роботи алгоритмів планування руху [18].

Для відображення окремих об'єктів використовуються Marker та MarkerArray. Дані компоненти дозволяють позначати:

- координати перешкод;
- точки маршруту;
- напрямки руху;
- окремі елементи карти.

Особливе значення для робототехнічних систем має компонент TF, який використовується для роботи із системами координат. TF дозволяє визначати положення різних елементів системи відносно один одного та забезпечує синхронізацію координат між сенсорами, картою та моделлю робота [6].

У процесі побудови карти навколишнього середовища RViz може відображати:

- occupancy grid map;
- point cloud map;
- дані depth-камер;

- результати SLAM-алгоритмів.

На рисунку 2.3 представлено приклад візуалізації робототехнічної системи у RViz.

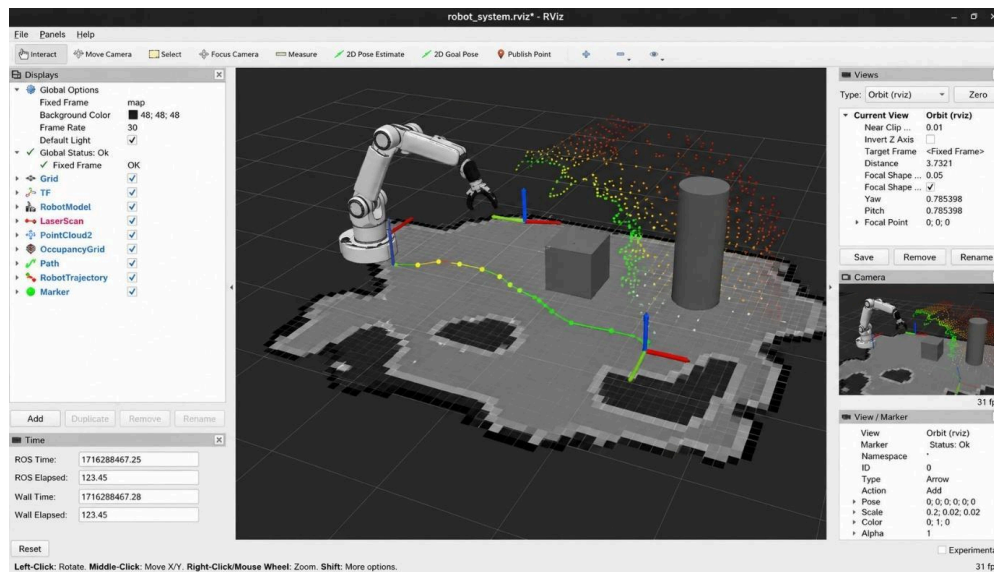


Рисунок 2.3 – Візуалізація робототехнічної системи у RViz

Однією з головних переваг RViz є підтримка тривимірної графіки. Користувач може змінювати кут огляду, масштабувати сцену та переглядати карту середовища з різних ракурсів. Це значно спрощує процес налагодження та тестування робототехнічних систем.

RViz також підтримує роботу у режимі реального часу. У процесі роботи системи карта та положення об'єктів автоматично оновлюються відповідно до нових даних із сенсорів.

У межах дипломної роботи RViz використовується для:

- відображення карти перешкод;
- візуалізації положення об'єктів;
- аналізу роботи алгоритмів комп'ютерного зору;
- тестування взаємодії між ROS-вузлами.

Таким чином, RViz є важливим компонентом робототехнічних систем та забезпечує ефективну візуалізацію результатів роботи алгоритмів навігації, побудови карт та аналізу навколишнього середовища.

2.4 Бібліотека OpenCV для задач комп'ютерного зору

Для реалізації систем комп'ютерного зору широко використовується бібліотека OpenCV (Open Source Computer Vision Library). Дана бібліотека є однією з найпопулярніших платформ для обробки зображень та відеоданих у робототехніці, системах автоматизації та штучному інтелекті [7].

OpenCV являє собою набір програмних інструментів та алгоритмів, які забезпечують:

- обробку цифрових зображень;
- роботу з відеопотоком;
- виділення контурів;
- сегментацію об'єктів;
- аналіз руху;
- розпізнавання об'єктів;
- реалізацію алгоритмів машинного навчання.

Бібліотека підтримує мови програмування Python, C++ та Java, однак у сфері робототехніки найбільш популярним є використання Python через простоту реалізації алгоритмів та інтеграцію з ROS [16].

Однією з головних переваг OpenCV є висока швидкість обробки даних. Бібліотека оптимізована для роботи у режимі реального часу та дозволяє обробляти відеопотік із мінімальними затримками.

Для задач побудови карти перешкод OpenCV використовується для:

- отримання відеоданих із камери;
- попередньої обробки зображення;
- визначення контурів об'єктів;
- аналізу форми перешкод;
- визначення координат об'єктів.

Першим етапом роботи системи є отримання кадру із камери. Для цього використовується клас VideoCapture, який дозволяє підключатися до пристроїв захоплення відео та отримувати зображення у режимі реального часу [4].

Після отримання кадру виконується попередня обробка зображення. На даному етапі можуть застосовуватися:

- Gaussian Blur;
- Median Blur;
- Bilateral Filter;
- перетворення у відтінки сірого.

Для визначення меж об'єктів використовується алгоритм Canny Edge Detection. Даний метод дозволяє ефективно виділяти контури перешкод та формувати карту меж об'єктів [14].

Після визначення контурів система виконує пошук окремих об'єктів за допомогою функції findContours. На основі отриманих контурів визначаються:

- площа об'єкта;
- координати;
- форма;
- центр перешкоди.

OpenCV також підтримує інтеграцію з алгоритмами машинного навчання та нейронними мережами. Це дозволяє реалізовувати сучасні системи автоматичного розпізнавання об'єктів на основі технологій штучного інтелекту [22].

На рисунку 2.4 представлено приклад обробки зображення за допомогою OpenCV.

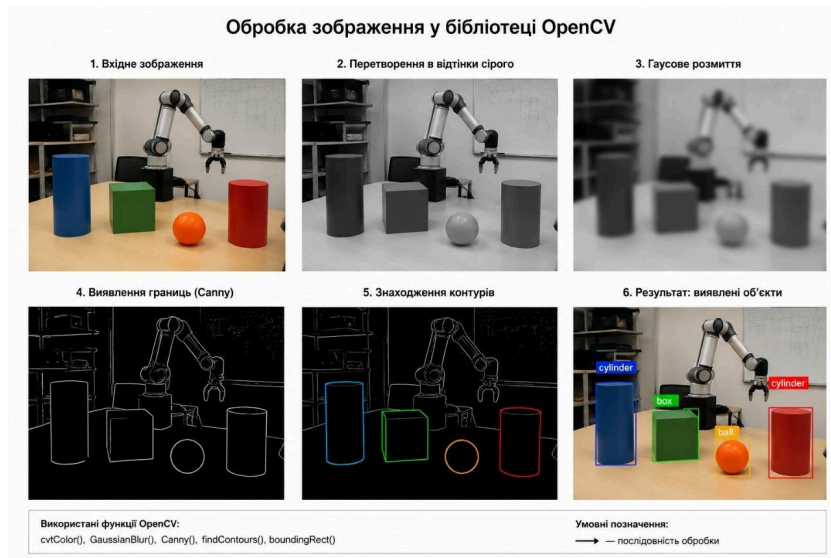


Рисунок 2.4 – Обробка зображення у бібліотеці OpenCV

Важливою перевагою OpenCV є наявність великої кількості документації та готових прикладів програмного коду. Це значно спрощує процес розробки систем комп'ютерного зору та дозволяє швидко інтегрувати бібліотеку у робототехнічні проекти.

У межах дипломної роботи OpenCV використовується як основний інструмент для реалізації алгоритмів обробки зображень та виявлення перешкод.

Таким чином, бібліотека OpenCV є ефективним засобом для реалізації систем комп'ютерного зору та забезпечує необхідний функціонал для побудови карти перешкод у робототехнічних системах.

2.5 Мова програмування Python

Для реалізації програмної системи побудови карти перешкод у межах дипломної роботи використовується мова програмування Python. Дана мова є однією з найпоширеніших у сфері робототехніки, комп'ютерного зору, аналізу даних та штучного інтелекту завдяки простоті використання, великій кількості бібліотек та високій швидкості розробки програмного забезпечення [16].

Python належить до високорівневих інтерпретованих мов програмування та підтримує об'єктно-орієнтований підхід до розробки застосунків. Основною перевагою Python є простий та зрозумілий синтаксис, що дозволяє швидко реалізовувати складні алгоритми та скорочує час розробки програмного забезпечення [21].

У сфері робототехніки Python активно використовується разом із:

- ROS;
- OpenCV;
- TensorFlow;
- PyTorch;
- NumPy;
- SciPy.

Використання Python у робототехнічних системах дозволяє реалізовувати:

- обробку відеоданих;
- аналіз зображень;
- роботу із сенсорами;
- побудову карт середовища;
- алгоритми машинного навчання;

- системи автоматичного керування.

Однією з важливих переваг Python є підтримка великої кількості бібліотек для математичних обчислень та обробки даних. Для задач комп'ютерного зору широко використовується бібліотека OpenCV, яка має повну підтримку Python та дозволяє реалізовувати алгоритми аналізу зображень із мінімальною кількістю програмного коду [7].

У робототехнічних системах Python часто використовується для створення ROS-вузлів. ROS має спеціальну бібліотеку rospy, яка забезпечує взаємодію програмного коду із ROS topics, messages та іншими компонентами системи [6].

Ще однією перевагою Python є підтримка багатопотокової та асинхронної обробки даних. Це дозволяє одночасно виконувати:

- отримання відеопотоку;
- аналіз зображень;
- побудову карти;
- передачу даних між вузлами системи.

Для реалізації алгоритмів машинного навчання Python інтегрується із TensorFlow та PyTorch. Дані фреймворки дозволяють створювати нейронні мережі для задач розпізнавання об'єктів та автоматичного аналізу навколишнього середовища [22].

Важливою перевагою Python є велика кількість навчальних матеріалів та документації. Це дозволяє швидко знаходити готові рішення та спрощує процес розробки програмного забезпечення.

У межах дипломної роботи Python використовується для:

- реалізації ROS-вузлів;
- роботи з OpenCV;
- обробки відеоданих;

- побудови карти перешкод;
- передачі даних у RViz.

Таким чином, мова програмування Python є ефективним інструментом для створення робототехнічних застосунків та забезпечує необхідний функціонал для реалізації систем комп'ютерного зору і побудови карти навколишнього середовища.

2.6 Методи цифрової обробки зображень

Цифрова обробка зображень є важливою складовою систем комп'ютерного зору та використовується для аналізу відеоданих, виявлення об'єктів та побудови карти навколишнього середовища. Основною задачею цифрової обробки є отримання корисної інформації із зображення та підготовка даних для подальшого аналізу [14].

У робототехнічних системах обробка зображень використовується для:

- виявлення перешкод;
- аналізу форми об'єктів;
- визначення координат;
- сегментації сцени;
- розпізнавання об'єктів;
- побудови карти середовища.

Першим етапом обробки зображення є попередня підготовка кадру. У процесі отримання відеоданих із камери зображення може містити шуми, спотворення або нерівномірне освітлення, що ускладнює подальший аналіз.

Для покращення якості зображення використовуються різноманітні методи фільтрації. Найбільш поширеними є:

- Gaussian Blur;
- Median Blur;
- Bilateral Filter.

Gaussian Blur використовується для згладжування зображення та зменшення шумів. Даний метод дозволяє покращити якість визначення контурів та зменшити кількість помилкових об'єктів [4].

Median Blur застосовується для видалення імпульсних шумів та забезпечує збереження меж об'єктів. Bilateral Filter використовується для згладжування зображення зі збереженням контурів.

Після фільтрації виконується перетворення кольорового зображення у відтінки сірого. Це дозволяє значно зменшити кількість обчислень та спростити подальшу обробку даних [14].

Одним із найважливіших етапів є виділення контурів об'єктів. Для цього широко використовується алгоритм Canny Edge Detection. Даний метод аналізує різницю яскравості між сусідніми пікселями та дозволяє визначати межі об'єктів на зображенні [4].

Після визначення контурів система може виконувати пошук окремих об'єктів та аналізувати їх характеристики. Для цього використовуються:

- contour detection;
- object segmentation;
- thresholding.

Сегментація зображення дозволяє поділити сцену на окремі області та виділити об'єкти із загального фону. Це особливо важливо для задач навігації роботів та побудови карти середовища [12].

У сучасних системах комп'ютерного зору також використовуються методи машинного навчання. Нейронні мережі дозволяють автоматично визначати об'єкти на зображенні та забезпечують високу точність роботи навіть у складних умовах [22].

Для робототехнічних систем важливе значення має швидкість обробки даних. Алгоритми повинні працювати у режимі реального часу та забезпечувати своєчасне оновлення карти навколишнього середовища.

На рисунку 2.6 представлено основні етапи цифрової обробки зображення.



Рисунок 2.6 – Етапи цифрової обробки зображення

У межах дипломної роботи методи цифрової обробки зображень використовуються для:

- аналізу відеопотоку;
- визначення контурів перешкод;
- побудови карти навколишнього середовища;
- передачі даних у систему візуалізації RViz.

Таким чином, цифрова обробка зображень є важливою складовою систем комп'ютерного зору та забезпечує можливість ефективного виявлення перешкод і побудови карти навколишнього середовища для роботів-маніпуляторів.

2.7 Алгоритми виявлення та класифікації перешкод

Однією з ключових задач робототехнічних систем є своєчасне виявлення та класифікація перешкод у навколишньому середовищі. Від точності роботи алгоритмів виявлення залежить безпечність переміщення робота-маніпулятора, ефективність побудови траєкторії руху та стабільність функціонування всієї системи [15].

Алгоритми виявлення перешкод використовуються для аналізу даних, отриманих із камер та сенсорів, а також для визначення положення об'єктів у просторі. Після виявлення перешкод система повинна класифікувати їх та передати отриману інформацію до модуля побудови карти навколишнього середовища.

У сучасних робототехнічних системах використовуються різні підходи до виявлення перешкод. Найбільш поширеними є:

- методи цифрової обробки зображень;
- алгоритми аналізу контурів;
- сегментація зображення;
- методи машинного навчання;
- нейронні мережі.

Класичні методи комп'ютерного зору базуються на аналізі зображення та визначенні характерних ознак об'єктів. Для цього використовуються алгоритми виділення контурів, визначення меж об'єктів та аналізу кольорових характеристик [14].

Одним із найбільш поширених підходів є *contour detection*. Після попередньої обробки система визначає контури об'єктів та формує набір

координат, які описують форму перешкоди. На основі отриманих даних можна визначити:

- площу об'єкта;
- положення перешкоди;
- форму;
- приблизну відстань до камери.

Для підвищення точності виявлення використовується сегментація зображення. Даний метод дозволяє поділити зображення на окремі області відповідно до певних характеристик, наприклад кольору або яскравості. Це дозволяє більш точно виділяти об'єкти із загального фону [12].

У складних умовах освітлення або при великій кількості об'єктів класичні методи можуть працювати недостатньо ефективно. У таких випадках використовуються алгоритми машинного навчання та нейронні мережі.

Сучасні нейронні мережі дозволяють автоматично визначати об'єкти на зображенні та виконувати їх класифікацію. Для задач робототехніки особливо популярними є алгоритми:

- YOLO;
- SSD;
- Faster R-CNN;
- Mask R-CNN.

Алгоритм YOLO (You Only Look Once) забезпечує високу швидкість роботи та дозволяє виконувати виявлення об'єктів у режимі реального часу. Основною перевагою даного алгоритму є можливість одночасного аналізу всього зображення та визначення декількох об'єктів на одному кадрі [13].

Алгоритм SSD (Single Shot Detector) також використовується для швидкого виявлення об'єктів. Він забезпечує баланс між швидкістю та точністю роботи системи.

Faster R-CNN демонструє високу точність класифікації об'єктів, однак потребує значних обчислювальних ресурсів та працює повільніше порівняно з YOLO [21].

Для задач побудови карти перешкод важливим є не лише виявлення об'єкта, а й визначення його координат у просторі. Для цього можуть використовуватися RGB-D камери або depth-сенсори, які дозволяють отримувати інформацію про глибину сцени [29].

Отримані координати об'єктів передаються до системи побудови карти, після чого інформація відображається у середовищі RViz. Це дозволяє візуально контролювати роботу алгоритмів та аналізувати точність визначення перешкод [8].

На рисунку 2.7 представлено приклад роботи алгоритму виявлення перешкод.

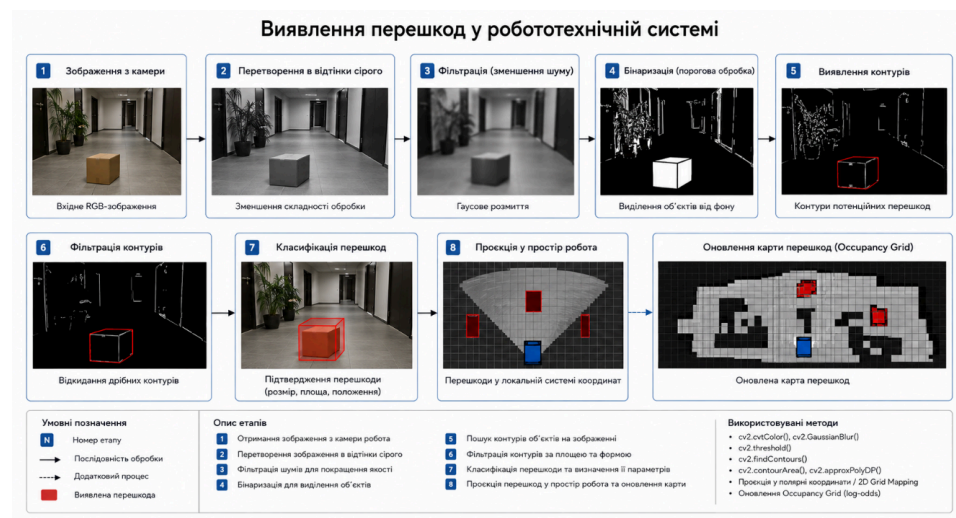


Рисунок 2.7 – Виявлення перешкод у робототехнічній системі

У межах дипломної роботи планується використання алгоритмів цифрової обробки зображень бібліотеки OpenCV для виявлення перешкод на основі аналізу контурів та форми об'єктів. Даний підхід дозволяє забезпечити достатню швидкість роботи системи та зменшити навантаження на обчислювальні ресурси.

Таким чином, алгоритми виявлення та класифікації перешкод є важливою складовою систем навігації роботів-маніпуляторів. Використання сучасних методів комп'ютерного зору та машинного навчання дозволяє забезпечити високу точність визначення об'єктів та безпечне функціонування робототехнічної системи.

Висновки до розділу 2

У даному розділі було проведено аналіз основних технологій та інструментальних засобів, які використовуються для розробки системи побудови карти перешкод для робота-маніпулятора.

Було розглянуто операційну систему Ubuntu, яка забезпечує стабільну роботу робототехнічних платформ та підтримує необхідні програмні компоненти для реалізації систем комп'ютерного зору [17]. Проведений аналіз показав, що Ubuntu є ефективним середовищем для роботи з платформою ROS та бібліотеками обробки зображень.

У процесі дослідження було проаналізовано можливості платформи Robot Operating System (ROS), яка використовується для організації взаємодії між окремими програмними модулями робототехнічної системи. ROS забезпечує передачу даних між сенсорами, алгоритмами обробки інформації та системами візуалізації [9].

Також було розглянуто середовище RViz, яке використовується для візуалізації карти навколишнього середовища, траєкторії руху робота та положення перешкод у тривимірному просторі [8].

Під час аналізу бібліотеки OpenCV було визначено її основні можливості для реалізації задач комп'ютерного зору та обробки відеоданих. OpenCV забезпечує високу швидкість роботи та містить великий набір алгоритмів для виявлення об'єктів, аналізу контурів та побудови карти перешкод [7].

Було також досліджено методи цифрової обробки зображень та алгоритми виявлення перешкод. Проведений аналіз показав, що використання методів аналізу контурів та сучасних алгоритмів комп'ютерного зору дозволяє ефективно визначати об'єкти у робочому просторі робота-маніпулятора [14].

У результаті проведеного аналізу було визначено основні технології, які будуть використані під час розробки програмної системи побудови карти перешкод. Використання Ubuntu, ROS, RViz, Python та OpenCV дозволить створити ефективну робототехнічну систему з можливістю виявлення перешкод та візуалізації результатів роботи у режимі реального часу.

3 ПРОЕКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ ПОБУДОВИ КАРТИ ПЕРЕШКОД

3.1 Архітектура програмної системи

Під час розробки робототехнічних застосунків важливе значення має правильне проектування архітектури програмної системи. Від структури програмного забезпечення залежить стабільність роботи системи, швидкість обробки даних, можливість масштабування та інтеграції з іншими компонентами робототехнічного комплексу [9].

У межах дипломної роботи було спроектовано програмну систему побудови карти перешкод для робота-маніпулятора на основі технологій комп'ютерного зору та платформи ROS. Основною задачею системи є отримання відеоданих із камери, виявлення перешкод, побудова карти навколишнього середовища та візуалізація результатів у RViz.

Під час проектування системи особлива увага приділялася:

- модульності архітектури;
- незалежності компонентів;
- швидкодії системи;
- можливості роботи у режимі реального часу;
- простоті масштабування.

Архітектура розробленої системи складається з декількох функціональних модулів, кожен із яких відповідає за виконання окремих задач. Основними компонентами системи є:

- модуль отримання відеоданих;

- модуль обробки зображень;
- модуль виявлення перешкод;
- модуль побудови карти;
- модуль передачі даних у ROS;
- модуль візуалізації у RViz.

Модуль отримання відеоданих відповідає за роботу з камерою та формування відеопотоку у режимі реального часу. Для реалізації даного модуля використовуються засоби бібліотеки OpenCV, які дозволяють отримувати окремі кадри та передавати їх до системи обробки [7].

Модуль обробки зображень виконує попередню підготовку кадрів. На даному етапі здійснюється:

- фільтрація шумів;
- згладжування зображення;
- перетворення кольорового простору;
- підготовка даних для аналізу контурів.

Після попередньої обробки дані передаються до модуля виявлення перешкод. Даний модуль виконує аналіз зображення та визначає координати об'єктів у робочому просторі. Для виявлення перешкод використовуються алгоритми contour detection та Canny Edge Detection [14].

Модуль побудови карти формує цифрове представлення навколишнього середовища на основі отриманих координат об'єктів. Для представлення робочого простору використовується occupancy grid map, яка дозволяє позначати зайняті та вільні області середовища [10].

Передача даних між окремими модулями реалізується за допомогою ROS topics. Такий підхід дозволяє забезпечити незалежність компонентів системи та спростує процес масштабування програмного забезпечення [6].

Для відображення карти перешкод використовується RViz. Система візуалізації дозволяє у режимі реального часу переглядати:

- карту навколишнього середовища;
- положення перешкод;
- модель робота;
- траєкторію руху;
- системи координат TF.

На рисунку 3.1 представлено загальну архітектуру програмної системи побудови карти перешкод.

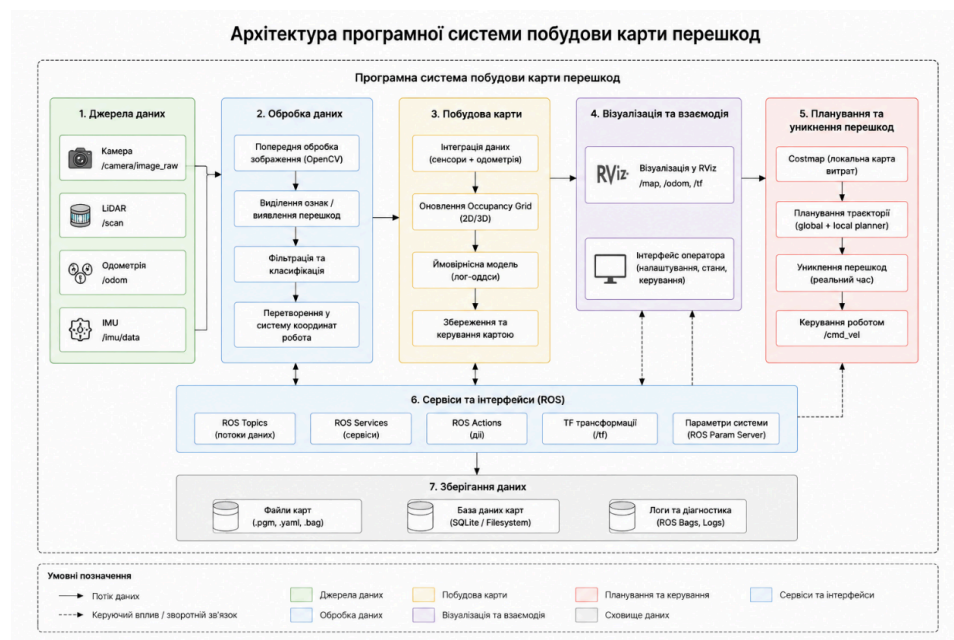


Рисунок 3.1 – Архітектура програмної системи

Під час проектування системи також враховувалася можливість подальшого розширення функціональності. Зокрема, у майбутньому система може бути доповнена:

- depth-камерами;
- LiDAR-сенсорами;
- алгоритмами SLAM;
- системами автоматичного планування траєкторії.

Таким чином, розроблена архітектура забезпечує ефективну взаємодію між компонентами системи та створює основу для реалізації програмного застосунку побудови карти перешкод для робота-маніпулятора.

3.2 Проектування функціональних модулів системи

Для забезпечення стабільної роботи програмного застосунку було виконано проектування функціональних модулів системи. Розподіл програмного забезпечення на окремі компоненти дозволяє спростити структуру коду, забезпечити незалежність модулів та підвищити ефективність роботи системи [9].

Кожен модуль виконує окрему функцію та взаємодіє з іншими компонентами через ROS topics. Такий підхід відповідає модульній архітектурі ROS та дозволяє реалізувати обробку даних у режимі реального часу [6].

Модуль отримання відеоданих відповідає за підключення до камери та передачу відеопотоку до інших компонентів системи.

Для реалізації даного модуля використовується бібліотека OpenCV та клас VideoCapture, який забезпечує:

- ініціалізацію камери;
- отримання окремих кадрів;
- перевірку коректності зчитування;
- передачу даних до ROS.

Отримані кадри публікуються у ROS topic:

/camera/image_raw

Після цього інші модулі системи можуть отримувати доступ до відеоданих та виконувати подальшу обробку. Модуль обробки зображень виконує попередній аналіз кадрів та підготовку даних для виявлення перешкод.

На даному етапі реалізуються:

- перетворення зображення у відтінки сірого;
- Gaussian Blur;
- фільтрація шумів;
- підготовка до виділення контурів.

Попередня обробка дозволяє покращити якість визначення меж об'єктів та зменшити кількість помилкових перешкод [14]. Модуль виявлення перешкод виконує аналіз контурів та визначення координат об'єктів у просторі.

Для реалізації даного модуля використовуються:

- Canny Edge Detection;
- contour detection;
- thresholding;
- object segmentation.

Після виявлення контурів система визначає:

- координати об'єкта;
- площу;
- форму;
- центр перешкоди.

Отримані дані передаються до topic:

/obstacles

Модуль побудови карти формує occupancy grid map на основі координат виявлених об'єктів [10]. Карта середовища представляється у вигляді набору комірок, кожна з яких містить інформацію про:

- наявність перешкоди;
- вільний простір;
- невідомі області.

У процесі роботи карта постійно оновлюється відповідно до нових даних із камери. Для візуалізації результатів роботи системи використовується RViz. Модуль забезпечує:

- відображення карти;
- візуалізацію перешкод;
- відображення систем координат;
- перегляд моделі робота.

На рисунку 3.2 представлено структуру взаємодії функціональних модулів системи.



Рисунок 3.2 – Взаємодія функціональних модулів системи

Під час проєктування модулів особлива увага приділялася мінімізації затримок передачі даних та оптимізації швидкодії системи. Для цього окремі компоненти працюють незалежно один від одного та можуть виконувати обробку даних паралельно.

Таким чином, розроблена структура функціональних модулів забезпечує стабільну роботу системи побудови карти перешкод та дозволяє ефективно реалізувати алгоритми комп'ютерного зору у робототехнічному застосунку.

3.3 Організація взаємодії між компонентами ROS

Для забезпечення стабільної роботи робототехнічної системи необхідно організувати ефективну взаємодію між окремими програмними компонентами. У розробленому застосунку для цього використовується платформа ROS, яка забезпечує передачу даних між модулями системи у режимі реального часу [6].

Основою архітектури ROS є вузли (nodes), які виконують окремі функції та можуть взаємодіяти між собою через канали передачі повідомлень — topics. У межах розробленої системи кожен функціональний модуль реалізований у вигляді окремого ROS-вузла [9]. Основними вузлами системи є:

- вузол отримання відеоданих;
- вузол обробки зображень;
- вузол виявлення перешкод;
- вузол побудови карти;
- вузол візуалізації.

Вузол отримання відеоданих відповідає за підключення до камери та передачу відеопотоку іншим компонентам системи. Отримані кадри публікуються у topic:

`/camera/image_raw`

Вузол обробки зображень підписується на даний topic та виконує попередню підготовку кадрів. Після обробки дані передаються до модуля виявлення перешкод.

Для взаємодії між вузлами використовується механізм publisher/subscriber. Один вузол виконує публікацію повідомлень, а інший — отримує їх та виконує необхідну обробку. Такий підхід забезпечує незалежність компонентів системи та дозволяє спростити масштабування програмного забезпечення [6]. Для передачі координат виявлених об'єктів використовується topic:

`/obstacles`

Після отримання координат система побудови карти формує occupancy grid map та передає результати до RViz. Для передачі карти використовується topic:

`/map`

Важливою перевагою ROS є підтримка асинхронної обробки даних. Кожен вузол може працювати незалежно від інших компонентів системи, що дозволяє забезпечити стабільну роботу застосунку навіть при великому навантаженні [9].

Для запуску системи використовується launch-файл, який автоматично запускає всі необхідні вузли та виконує налаштування параметрів середовища.

Приклад запуску системи:

```
roslaunch obstacle_map system.launch
```

Launch-файл дозволяє:

- запускати декілька вузлів одночасно;
- задавати параметри роботи системи;
- виконувати автоматичне налаштування середовища.

У процесі роботи системи важливе значення має використання TF-трансформацій. TF використовується для визначення положення різних елементів системи відносно один одного та забезпечує синхронізацію координат між:

- камерою;
- картою;
- моделлю робота;
- перешкодами.

Під час проєктування системи особлива увага приділялася мінімізації затримок передачі даних. Це особливо важливо для робототехнічних систем, які працюють у режимі реального часу та повинні швидко реагувати на зміни навколишнього середовища.

Таким чином, використання ROS забезпечує ефективну організацію взаємодії між програмними компонентами системи та дозволяє реалізувати стабільну роботу алгоритмів побудови карти перешкод.

3.4 Розробка алгоритму отримання відеоданих

Одним із найважливіших етапів роботи системи є отримання відеоданих із камери. Саме відеопотік є основним джерелом інформації про навколишнє середовище та використовується для подальшого виявлення перешкод і побудови карти [7].

Для реалізації отримання відеоданих використовується бібліотека OpenCV. Вона дозволяє працювати з USB-камерами та іншими пристроями захоплення зображення у режимі реального часу [4].

Під час запуску системи виконується ініціалізація камери за допомогою класу VideoCapture. Після успішного підключення система починає отримувати окремі кадри відеопотоку.

Алгоритм отримання відеоданих складається з таких етапів:

1. Ініціалізація камери.
2. Отримання кадру.

3. Перевірка коректності зчитування.
4. Масштабування зображення.
5. Передача кадру до модуля обробки.

На першому етапі система встановлює з'єднання із камерою та перевіряє її доступність. У випадку відсутності підключення система формує повідомлення про помилку та припиняє роботу.

Після успішної ініціалізації камера починає передавати кадри у режимі реального часу. Для підвищення швидкодії системи виконується масштабування зображення до фіксованого розміру.

Під час роботи застосунку система постійно перевіряє коректність отримання кадрів. Якщо зчитування кадру завершилося помилкою, виконується повторна спроба підключення до камери. Для передачі відеопотоку між ROS-вузлами використовується пакет `cv_bridge`, який забезпечує конвертацію зображень OpenCV у формат ROS messages [6].

Отримані кадри публікуються у топіс:

```
/camera/image_raw
```

Це дозволяє іншим модулям системи отримувати доступ до відеоданих та виконувати їх подальшу обробку. Нижче наведено приклад фрагмента програмного коду для отримання відеопотоку за допомогою OpenCV.

```
import cv2
camera = cv2.VideoCapture(0)
while True:
    ret, frame = camera.read()
    if not ret:
        break
    cv2.imshow("Camera", frame)
    if cv2.waitKey(1) == 27:
        break
camera.release()
```

`cv2.destroyAllWindows()`

У процесі роботи система отримує послідовність кадрів та передає їх до модуля обробки зображень для подальшого аналізу. Під час розробки алгоритму особлива увага приділяється забезпеченню стабільності роботи системи та мінімізації затримок обробки даних. Для цього використовувалися оптимізовані функції OpenCV та багатопотокова обробка відеопотоку.

Таким чином, розроблений алгоритм отримання відеоданих забезпечує стабільну передачу зображень у режимі реального часу та створює основу для подальшої роботи системи комп'ютерного зору.

3.5 Розробка алгоритму виявлення перешкод

Однією з основних задач розробленої системи є виявлення перешкод у робочому просторі робота-маніпулятора. Для забезпечення безпечної навігації система повинна своєчасно визначати об'єкти на зображенні, аналізувати їх координати та передавати отримані дані до модуля побудови карти навколишнього середовища [15].

У межах дипломної роботи для виявлення перешкод використовуються методи цифрової обробки зображень бібліотеки OpenCV. Даний підхід дозволяє забезпечити достатню швидкість роботи системи та реалізувати обробку відеоданих у режимі реального часу [7]. Алгоритм виявлення перешкод складається з таких етапів:

1. Отримання кадру із камери.
2. Попередня обробка зображення.
3. Виділення меж об'єктів.
4. Пошук контурів.
5. Визначення координат перешкод.
6. Передача результатів до ROS.

Після отримання кадру система виконує його попередню обробку. На даному етапі зображення перетворюється у відтінки сірого для зменшення кількості обчислень та спрощення подальшого аналізу [14].

Для зменшення впливу шумів використовується Gaussian Blur. Даний метод забезпечує згладжування зображення та покращує якість визначення контурів об'єктів [4]. Приклад попередньої обробки зображення наведено нижче.

```
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
blur = cv2.GaussianBlur(gray, (5, 5), 0)
```

Після попередньої обробки виконується визначення меж об'єктів за допомогою алгоритму Canny Edge Detection. Даний алгоритм аналізує зміну яскравості між сусідніми пікселями та дозволяє ефективно виділяти контури перешкод [14].

```
edges = cv2.Canny(blur, 50, 150)
```

Наступним етапом є пошук контурів об'єктів. Для цього використовується функція findContours бібліотеки OpenCV.

```
contours, _ = cv2.findContours(
    edges,
    cv2.RETR_EXTERNAL,
    cv2.CHAIN_APPROX_SIMPLE
)
```

Після визначення контурів система аналізує їх характеристики та виконує фільтрацію дрібних об'єктів, які можуть бути результатом шумів або помилок обробки зображення. Для кожного виявленого об'єкта визначаються:

- координати центру;
- площа контуру;
- ширина та висота об'єкта;
- межі перешкоди.

Для визначення координат використовується функція `boundingRect`.

`x, y, w, h = cv2.boundingRect(contour)`

Отримані координати використовуються для побудови карти навколишнього середовища та передачі інформації до ROS topics.

У процесі роботи системи координати об'єктів постійно оновлюються відповідно до нових даних із камери. Це дозволяє роботу-маніпулятору реагувати на зміни навколишнього середовища у режимі реального часу. Під час розробки алгоритму особлива увага приділялася:

- швидкодії системи;
- точності визначення контурів;
- зменшенню кількості помилкових спрацьовувань;
- стабільності роботи в умовах зміни освітлення.

Було встановлено, що використання методів попередньої обробки зображення дозволяє значно покращити якість визначення перешкод та зменшити вплив шумів на роботу системи. Для подальшого розвитку системи можливе використання алгоритмів машинного навчання та нейронних мереж, які дозволять підвищити точність класифікації об'єктів та реалізувати автоматичне розпізнавання типів перешкод [22].

Таким чином, розроблений алгоритм виявлення перешкод забезпечує стабільну роботу системи комп'ютерного зору та дозволяє ефективно визначати об'єкти у робочому просторі робота-маніпулятора.

3.6 Розробка алгоритму побудови карти перешкод

Після визначення координат об'єктів система повинна сформувати карту навколишнього середовища, яка буде використовуватися для навігації робота-маніпулятора та уникнення зіткнень. Побудова карти є одним із

найважливіших етапів роботи робототехнічної системи, оскільки саме карта містить інформацію про розташування перешкод та вільний простір для переміщення робота [10].

У межах дипломної роботи для представлення навколишнього середовища використовується occupancy grid map. Даний підхід широко застосовується у робототехніці та передбачає поділ простору на окремі комірки, кожна з яких містить інформацію про стан відповідної області середовища [26]. Кожна комірка карти може мати один із таких станів:

- вільна область;
- зайнята область;
- невідома область.

Якщо у певній частині простору виявлено перешкоду, відповідна комірка позначається як зайнята. У випадку відсутності об'єктів комірка вважається вільною. Алгоритм побудови карти складається з таких етапів:

1. Отримання координат перешкод.
2. Перетворення координат у систему карти.
3. Оновлення occupancy grid map.
4. Передача карти до ROS.
5. Візуалізація результатів у RViz.

Після отримання координат об'єктів система виконує їх масштабування відповідно до розмірів карти. Це необхідно для правильного відображення перешкод у робочому просторі. Для зберігання карти використовується двовимірний масив, елементи якого відповідають окремим коміркам середовища.

Приклад створення карти наведено нижче.

```
import numpy as np
```

```
map_grid = np.zeros((500, 500), dtype=np.int8)
```

Після визначення координат об'єкта відповідні комірки карти оновлюються.

```
map_grid[y:y+h, x:x+w] = 100
```

Значення:

- 0 — вільний простір;
- 100 — наявність перешкоди.

Після оновлення карти система передає дані до ROS topic:

```
/map
```

Для передачі карти використовується тип повідомлення `nav_msgs/OccupancyGrid`, який є стандартом ROS для представлення occupancy map [6].

У процесі роботи системи карта постійно оновлюється відповідно до нових даних із камери. Це дозволяє забезпечити адаптацію роботи до змін навколишнього середовища та своєчасне визначення нових перешкод.

Однією з важливих переваг occupancy grid map є простота реалізації та висока швидкість оновлення даних. Даний підхід дозволяє ефективно використовувати карту для задач навігації роботів та уникнення зіткнень [10]. Для підвищення точності побудови карти у майбутньому система може бути доповнена:

- depth-камерами;
- LiDAR-сенсорами;
- алгоритмами SLAM;
- тривимірними картами типу OctoMap [25].

Таким чином, розроблений алгоритм побудови карти перешкод забезпечує формування цифрового представлення навколишнього середовища та створює основу для реалізації безпечної навігації робота-маніпулятора.

3.7 Реалізація візуалізації карти в RViz

Для наочного відображення результатів роботи системи використовується середовище візуалізації RViz, яке входить до складу платформи ROS. Використання RViz дозволяє у режимі реального часу переглядати карту перешкод, положення робота-маніпулятора та результати роботи алгоритмів комп'ютерного зору [8].

Основною задачею системи візуалізації є забезпечення зручного відображення даних, отриманих у процесі роботи програмного застосунку. Це дозволяє аналізувати правильність визначення перешкод, контролювати побудову карти навколишнього середовища та оцінювати ефективність роботи алгоритмів.

Для передачі даних до RViz використовується механізм ROS topics. У процесі роботи система передає:

- карту навколишнього середовища;
- координати перешкод;
- положення робота;
- траєкторію руху;
- відеодані із камери.

Основним елементом візуалізації є occupancy grid map, яка відображає зайняті та вільні області робочого простору. Карта формується у процесі роботи алгоритму побудови перешкод та постійно оновлюється відповідно до нових даних із камери [10]. Для налаштування RViz було використано такі компоненти:

- Map;
- Grid;
- RobotModel;
- Marker;
- Image;
- TF.

Компонент Map використовується для відображення карти навколишнього середовища. Він отримує дані із topic:

/map

Компонент Marker застосовується для відображення окремих перешкод та допоміжних елементів системи. За допомогою маркерів можна позначати координати об'єктів, напрямок руху або інші важливі дані.

Для відображення моделі робота використовується компонент RobotModel. Це дозволяє візуально контролювати положення робота-маніпулятора у просторі та аналізувати взаємодію з навколишнім середовищем [18]. Компонент TF забезпечує відображення систем координат та використовується для синхронізації положення різних елементів системи [6].

Під час роботи системи RViz автоматично оновлює інформацію про навколишнє середовище. Це дозволяє у режимі реального часу спостерігати зміну положення перешкод та аналізувати роботу алгоритмів.

Однією з переваг використання RViz є можливість тривимірної візуалізації. Користувач може змінювати положення камери, масштабувати сцену та переглядати карту із різних ракурсів. Це значно спрощує процес тестування та налагодження системи.

Використання RViz також дозволяє швидко виявляти помилки у роботі алгоритмів побудови карти та перевіряти правильність передачі даних між ROS-вузлами. У межах дипломної роботи RViz використовується для:

- аналізу карти перешкод;
- контролю координат об'єктів;
- тестування алгоритмів навігації;
- перевірки роботи ROS topics;
- візуального аналізу роботи системи.

Таким чином, реалізована система візуалізації забезпечує наочне відображення результатів роботи алгоритмів комп'ютерного зору та дозволяє ефективно аналізувати процес побудови карти перешкод для робота-маніпулятора.

3.8 Інтеграція системи з роботом-маніпулятором

Одним із найважливіших етапів розробки програмного забезпечення є інтеграція системи побудови карти перешкод із роботом-маніпулятором. Даний процес забезпечує взаємодію між програмними компонентами та апаратною частиною робототехнічної системи [3].

Основною задачею інтеграції є передача інформації про навколишнє середовище до системи керування роботом. Отримані дані використовуються для аналізу робочого простору та уникнення зіткнень під час руху маніпулятора. У межах розробленої системи робот-маніпулятор взаємодіє з такими компонентами:

- система комп'ютерного зору;
- модуль побудови карти;
- система навігації;
- середовище візуалізації RViz;
- ROS-вузли керування.

Під час роботи система отримує відеодані із камери, виконує аналіз навколишнього середовища та визначає координати перешкод. Після цього інформація передається до ROS topics та використовується для оновлення карти робочого простору.

Для взаємодії між системою побудови карти та роботом використовується механізм TF-трансформацій. TF дозволяє визначати положення окремих елементів системи відносно різних систем координат та забезпечує коректну синхронізацію даних [6]. Завдяки використанню TF робот-маніпулятор може визначати:

- положення перешкод;
- власне положення у просторі;
- координати цілі;
- напрямок руху.

У процесі інтеграції було реалізовано передачу координат перешкод до системи керування рухом маніпулятора. Це дозволяє уникати небезпечних зон та коригувати траєкторію руху у випадку появи нових об'єктів [18].

Для тестування системи використовувалася віртуальна модель робота у середовищі RViz. Це дозволило перевірити коректність передачі даних та оцінити стабільність роботи алгоритмів без використання реального обладнання. На рисунку 3.8 представлено приклад інтеграції системи побудови карти із роботом-маніпулятором.

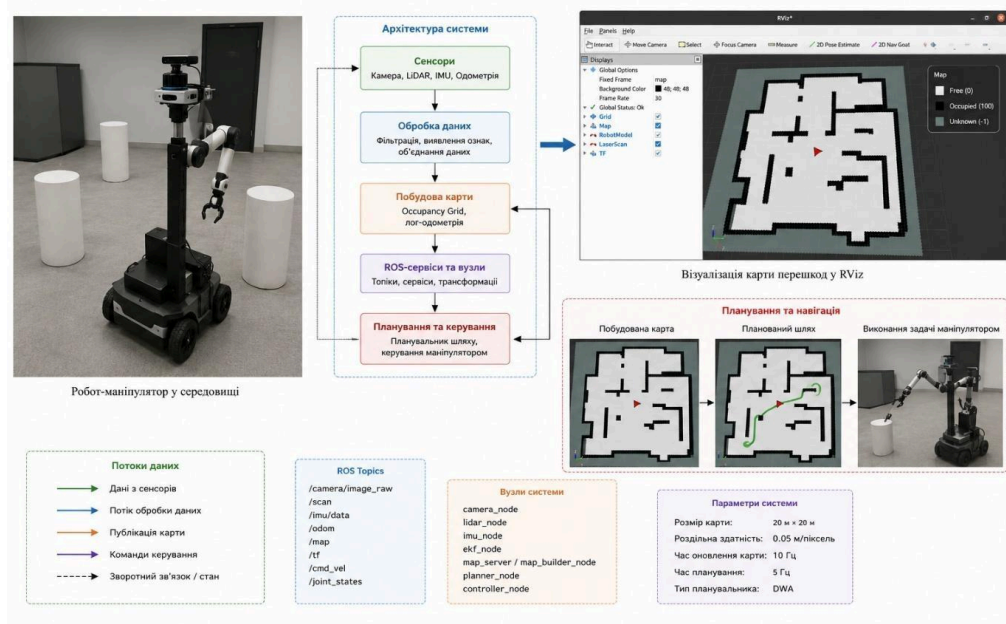


Рисунок 3.8 – Інтеграція системи з роботом-маніпулятором

Однією з важливих переваг розробленої системи є можливість подальшого розширення функціональності. У майбутньому система може бути доповнена:

- LiDAR-сенсорами;
- depth-камерами;
- алгоритмами машинного навчання;
- системами автоматичного планування траєкторії;
- тривимірними картами OctoMap [25].

Під час інтеграції особлива увага приділялася стабільності передачі даних та швидкості реакції системи на зміну навколишнього середовища. Це дозволяє забезпечити безпечну роботу робота-маніпулятора навіть у динамічному середовищі.

Таким чином, інтеграція системи побудови карти перешкод із роботом-маніпулятором забезпечує ефективну взаємодію між програмними та апаратними компонентами робототехнічної системи та створює основу для реалізації безпечної навігації у робочому просторі.

Висновки до розділу 3

У даному розділі було виконано проектування програмної системи побудови карти перешкод для робота-маніпулятора на основі технологій комп'ютерного зору.

Було розроблено архітектуру програмного застосунку та визначено основні функціональні модулі системи. Проведене проектування дозволило реалізувати модульну структуру програмного забезпечення та забезпечити незалежність окремих компонентів [9].

У процесі роботи було реалізовано алгоритм отримання відеоданих із камери, а також алгоритм виявлення перешкод на основі методів цифрової обробки зображень бібліотеки OpenCV. Використання алгоритмів аналізу контурів дозволило забезпечити стабільне визначення об'єктів у режимі реального часу [7].

Було реалізовано систему побудови карти навколишнього середовища у вигляді оцифрованої grid map. Розроблений алгоритм забезпечує формування цифрового представлення робочого простору та автоматичне оновлення карти відповідно до нових даних із камери [10].

Також було реалізовано візуалізацію карти перешкод у середовищі RViz. Використання даного інструменту дозволяє у режимі реального часу аналізувати роботу алгоритмів та контролювати положення перешкод у просторі [8].

У результаті інтеграції системи із роботом-маніпулятором забезпечено передачу даних між програмними компонентами та створено основу для реалізації безпечної навігації у робочому середовищі.

Отримані результати підтверджують можливість використання розробленої системи для задач побудови карти перешкод та візуалізації навколишнього середовища у робототехнічних системах.

4 РОЗРОБКА ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАСТОСУНКУ

4.1 Реалізація програмного забезпечення

Після завершення етапу проектування було виконано реалізацію програмної системи побудови карти перешкод для робота-маніпулятора. Розробка програмного забезпечення здійснювалася із використанням мови програмування Python, бібліотеки OpenCV та платформи ROS [6].

Основною метою реалізації програмного застосунку є забезпечення:

- отримання відеоданих із камери;
- обробки зображень;
- виявлення перешкод;
- побудови карти навколишнього середовища;
- передачі даних між ROS-вузлами;
- візуалізації результатів у RViz.

Під час розробки програмного забезпечення особлива увага приділялася:

- модульності структури коду;
- швидкодії системи;
- стабільності роботи;
- можливості роботи у режимі реального часу;
- простоті масштабування.

Програмний застосунок реалізований у вигляді окремих ROS-вузлів, кожен із яких відповідає за виконання певної функції. Такий підхід забезпечує

незалежність програмних модулів та дозволяє ефективно організувати обробку даних [9].

Для роботи із камерою використовувався клас VideoCapture бібліотеки OpenCV. Даний клас забезпечує отримання відеопотоку та передачу кадрів до модуля обробки зображення [7]. Нижче наведено приклад ініціалізації камери.

```
import cv2
camera = cv2.VideoCapture(0)
while True:
    ret, frame = camera.read()
    if not ret:
        break
    cv2.imshow("Camera", frame)
    if cv2.waitKey(1) == 27:
        break
camera.release()
cv2.destroyAllWindows()
```

Після отримання кадру система виконує його попередню обробку. На даному етапі реалізуються:

- перетворення у відтінки сірого;
- Gaussian Blur;
- фільтрація шумів;
- виділення контурів.

Для визначення меж об'єктів використовується алгоритм Canny Edge Detection [14].

```
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
blur = cv2.GaussianBlur(gray, (5, 5), 0)
edges = cv2.Canny(blur, 50, 150)
```

Після визначення контурів система виконує пошук окремих об'єктів за допомогою функції `findContours`.

```
contours, _ = cv2.findContours(  
    edges,  
    cv2.RETR_EXTERNAL,  
    cv2.CHAIN_APPROX_SIMPLE  
)
```

Для передачі даних між ROS-вузлами використовуються topics. Координати перешкод передаються до topic:

`/obstacles`

Карта навколишнього середовища публікується у topic:

`/map`

Для автоматичного запуску всіх компонентів системи використовується launch-файл ROS. Він дозволяє одночасно запускати:

- камеру;
- модуль обробки зображень;
- систему побудови карти;
- RViz.

Приклад запуску системи наведено нижче.

```
roslaunch obstacle_map system.launch
```

У процесі реалізації програмного забезпечення також використовувалися:

- `cv_bridge`;
- `sensor_msgs`;
- `nav_msgs`;
- `rospy`;

- NumPy.

На рисунку 4.1 представлено структуру програмного застосунку.

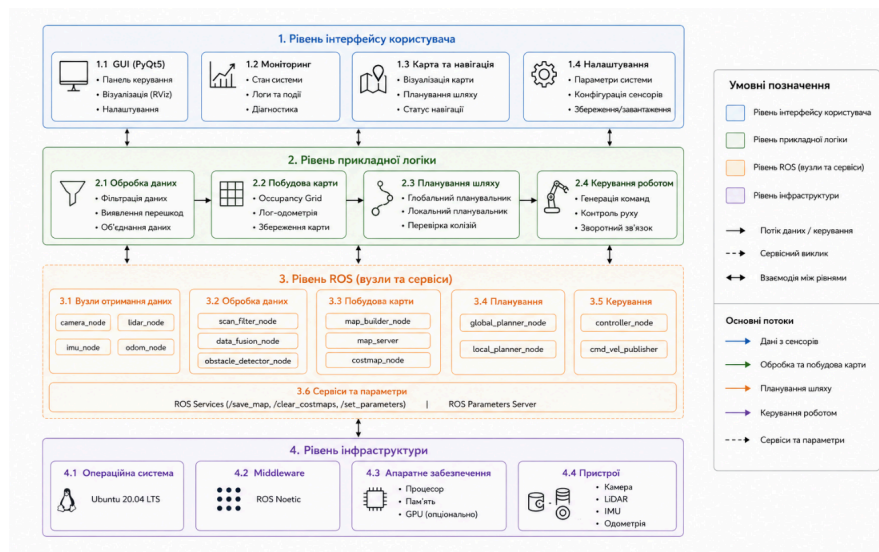


Рисунок 4.1 – Структура програмного забезпечення системи

Під час розробки проводилося поетапне тестування кожного модуля окремо, що дозволило своєчасно виявляти помилки та покращувати стабільність роботи системи.

Таким чином, реалізоване програмне забезпечення забезпечує отримання відеоданих, виявлення перешкод, побудову карти навколишнього середовища та передачу результатів до системи візуалізації RViz.

4.2 Налаштування середовища ROS та RViz

Для роботи розробленої системи було виконано налаштування операційної системи Ubuntu, платформи ROS та середовища візуалізації RViz [17]. У межах дипломної роботи використовується:

- Ubuntu 20.04 LTS;
- ROS Noetic;

- RViz;
- Python 3;
- OpenCV.

Першим етапом було встановлення операційної системи Ubuntu та необхідних пакетів ROS. Використання Ubuntu забезпечує стабільну роботу робототехнічного програмного забезпечення та підтримку необхідних бібліотек [6].

Після встановлення ROS було створено робочий простір catkin workspace, який використовується для розміщення ROS-пакетів та програмного коду. Для створення робочого простору використовувалися такі команди:

```
mkdir -p ~/catkin_ws/src  
cd ~/catkin_ws  
catkin_make
```

Після створення робочого простору виконувалося налаштування середовища ROS.

```
source devel/setup.bash
```

Для роботи із зображеннями та відеоданими було встановлено додаткові пакети:

- cv_bridge;
- image_transport;
- sensor_msgs.

Для реалізації побудови карти використовувався пакет nav_msgs, який забезпечує підтримку типу повідомлень OccupancyGrid [6].

Під час налаштування RViz було створено конфігурацію для відображення:

- карти навколишнього середовища;
- моделі робота;

- координат перешкод;
- систем координат TF;
- відеопотоку із камери.

Для цього використовувалися компоненти:

- Map;
- RobotModel;
- Marker;
- TF;
- Grid;
- Image.

Для автоматичного запуску системи використовувався launch-файл ROS, який забезпечує запуск усіх необхідних вузлів та налаштування параметрів роботи системи.

```
roslaunch obstacle_map system.launch
```

Після запуску системи RViz автоматично підключається до ROS topics та відображає:

- карту перешкод;
- модель робота;
- координати об'єктів;
- результати роботи алгоритмів комп'ютерного зору.

Особлива увага під час налаштування приділялася:

- синхронізації ROS topics;
- стабільності передачі даних;

- коректності TF-трансформацій;
- швидкодії системи.

Було встановлено, що використання ROS та RViz забезпечує ефективну взаємодію між програмними компонентами та дозволяє реалізувати візуалізацію карти перешкод у режимі реального часу [8].

Таким чином, проведене налаштування середовища забезпечило стабільну роботу системи побудови карти перешкод та створило необхідні умови для тестування розробленого програмного застосунку.

4.3 Тестування системи виявлення перешкод

Після реалізації програмного забезпечення було проведено тестування системи виявлення перешкод. Основною метою тестування була перевірка коректності роботи алгоритмів комп'ютерного зору, стабільності передачі даних між ROS-вузлами та можливості функціонування системи у режимі реального часу [5]. Тестування проводилося у декілька етапів:

1. Перевірка отримання відеопотоку.
2. Тестування алгоритмів обробки зображення.
3. Перевірка виявлення контурів об'єктів.
4. Тестування передачі даних у ROS.
5. Аналіз відображення карти у RViz.

На першому етапі перевірялася стабільність отримання відеоданих із камери. Система успішно отримувала відеопотік у режимі реального часу без критичних затримок або втрати кадрів. Після цього проводилося тестування алгоритмів попередньої обробки зображення. Було перевірено роботу:

- перетворення у відтінки сірого;

- Gaussian Blur;
- фільтрації шумів;
- алгоритму Canny Edge Detection.

У результаті тестування було встановлено, що використання Gaussian Blur дозволяє значно зменшити кількість шумів та покращити якість визначення контурів [14].

Наступним етапом було тестування алгоритму виявлення перешкод. У процесі роботи система аналізувала кадри відеопотоку та визначала координати об'єктів у робочому просторі.

Для перевірки роботи алгоритму використовувалися:

- статичні об'єкти;
- рухомі об'єкти;
- об'єкти різної форми та розміру.

Під час тестування система успішно визначала більшість перешкод та коректно передавала їх координати до ROS topics.

Особлива увага приділялася перевірці роботи системи у різних умовах освітлення. Було встановлено, що при недостатньому освітленні точність визначення контурів дещо зменшується. Однак використання попередньої обробки зображення дозволило частково компенсувати даний недолік.

Також проводилася перевірка передачі даних між ROS-вузлами. У процесі тестування система стабільно передавала координати перешкод та карту навколишнього середовища без помилок синхронізації [6].

Для оцінки продуктивності системи було виміряно середню швидкість обробки кадрів. У результаті тестування система забезпечувала обробку приблизно 20–25 кадрів за секунду, що є достатнім показником для роботи у режимі реального часу.

Під час тестування також оцінювалася:

- стабільність роботи ROS topics;
- коректність оновлення карти;
- швидкість передачі даних;
- навантаження на процесор.

Було встановлено, що використання модульної архітектури ROS дозволяє ефективно розподіляти навантаження між окремими компонентами системи та забезпечує стабільну роботу застосунку [9].

Таким чином, проведене тестування підтвердило працездатність алгоритмів комп'ютерного зору та можливість використання розробленої системи для задач побудови карти перешкод у робототехнічних системах.

4.4 Аналіз точності побудови карти

Одним із важливих критеріїв ефективності роботи системи є точність побудови карти навколишнього середовища. Для оцінки якості роботи алгоритмів було проведено аналіз точності визначення координат перешкод та коректності відображення об'єктів на карті [10].

У процесі тестування система формувала карту робочого простору на основі даних, отриманих із камери. Після виявлення об'єктів координати перешкод передавалися до модуля побудови карти та відображалися у RViz.

Для перевірки точності використовувалися контрольні об'єкти, положення яких було відоме заздалегідь. Отримані системою координати порівнювалися з реальним положенням об'єктів.

Під час аналізу оцінювалися:

- точність визначення координат;
- стабільність оновлення карти;

- швидкість реакції системи;
- коректність відображення перешкод.

Результати тестування показали, що система забезпечує достатню точність побудови карти для задач навігації робота-маніпулятора. Найкращі результати спостерігалися при стабільному освітленні та відсутності великої кількості дрібних об'єктів у кадрі. У процесі роботи було встановлено, що найбільший вплив на точність мають:

- якість відеопотоку;
- рівень освітлення;
- наявність шумів;
- роздільна здатність камери;
- швидкість руху об'єктів.

Для зменшення похибок використовувалися методи згладжування зображення та фільтрації контурів малого розміру [14].

Під час тестування також перевірялася стабільність оновлення карти у режимі реального часу. Система успішно реагувала на появу нових об'єктів та автоматично оновлювала карту відповідно до змін навколишнього середовища.

Було встановлено, що використання *occipancy grid map* дозволяє ефективно представляти інформацію про робочий простір та забезпечує достатню швидкість оновлення карти [26].

Під час тестування затримка оновлення карти залишалася незначною, що дозволяло системі працювати у режимі реального часу та своєчасно реагувати на появу нових перешкод.

Отримані результати підтвердили можливість використання розробленої системи для побудови карти перешкод у задачах навігації роботів-маніпуляторів.

Таким чином, проведений аналіз показав, що розроблена система забезпечує достатню точність побудови карти навколишнього середовища та може використовуватися для задач уникнення перешкод у робототехнічних системах.

4.5 Оцінка продуктивності системи

Для забезпечення ефективної роботи робототехнічної системи важливим фактором є продуктивність програмного забезпечення. Система побудови карти перешкод повинна працювати у режимі реального часу, швидко обробляти відеодані та своєчасно реагувати на зміни навколишнього середовища [15].

У процесі тестування було проведено оцінку продуктивності розробленого програмного застосунку. Основною метою дослідження було визначення швидкості обробки кадрів, стабільності роботи системи та рівня навантаження на обчислювальні ресурси. Під час оцінювання аналізувалися такі показники:

- швидкість обробки кадрів;
- затримка передачі даних;
- використання оперативної пам'яті;
- навантаження на процесор;
- стабільність роботи системи.

Тестування проводилося на персональному комп'ютері із такими характеристиками:

- процесор Intel Core i5;
- 16 ГБ оперативної пам'яті;
- операційна система Ubuntu 20.04;
- ROS Noetic.

У процесі роботи система виконувала:

- отримання відеопотоку;
- обробку зображення;
- визначення контурів;
- побудову карти;
- передачу даних у RViz.

Середня швидкість обробки відеопотоку становила приблизно 20–25 кадрів за секунду. Даний показник забезпечує комфортну роботу системи у режимі реального часу та дозволяє оперативно реагувати на зміни навколишнього середовища.

Під час тестування було встановлено, що найбільше навантаження на процесор виникає на етапі:

- обробки зображення;
- визначення контурів;
- оновлення карти.

Для підвищення продуктивності було використано:

- масштабування зображення;
- оптимізацію кількості обчислень;
- фільтрацію дрібних контурів;
- модульну архітектуру ROS [9].

Використання ROS дозволило розподілити навантаження між окремими вузлами системи та забезпечити незалежну роботу компонентів.

Також було проведено аналіз використання оперативної пам'яті. У процесі роботи система стабільно функціонувала без критичного збільшення споживання пам'яті або втрати продуктивності.

На рисунку 4.5 представлено графік зміни швидкості обробки кадрів під час роботи системи.



Рисунок 4.5 – Швидкість обробки відеопотоку

У процесі тривалого тестування система демонструвала стабільну роботу без критичних помилок або зависань. Передача даних між ROS-вузлами відбувалася коректно, а RViz своєчасно оновлював карту навколишнього середовища [8].

Проведений аналіз показав, що розроблений програмний застосунок має достатній рівень продуктивності для використання у робототехнічних системах та забезпечує стабільну роботу у режимі реального часу.

Таким чином, результати тестування підтверджують ефективність реалізованих алгоритмів та можливість використання системи для задач побудови карти перешкод і навігації роботів-маніпуляторів.

4.6 Аналіз результатів роботи програмного застосунку

Після завершення тестування було проведено загальний аналіз результатів роботи розробленої системи побудови карти перешкод для робота-маніпулятора. У процесі роботи програмний застосунок успішно виконував:

- отримання відеоданих із камери;
- обробку зображення;
- виявлення перешкод;
- побудову карти навколишнього середовища;
- передачу даних до ROS;
- візуалізацію результатів у RViz.

Під час тестування система стабільно працювала у режимі реального часу та своєчасно оновлювала інформацію про навколишнє середовище. Алгоритми комп'ютерного зору забезпечували коректне визначення більшості об'єктів у робочому просторі [5].

Проведений аналіз показав, що використання OpenCV є ефективним рішенням для реалізації задач виявлення перешкод. Методи аналізу контурів дозволили забезпечити достатню точність роботи системи при відносно невеликому навантаженні на обчислювальні ресурси [7].

Використання ROS забезпечило стабільну взаємодію між програмними компонентами та дозволило організувати передачу даних між вузлами системи у режимі реального часу [6].

Однією з важливих переваг розробленого застосунку є можливість візуалізації карти перешкод у RViz. Це значно спрощує процес аналізу роботи алгоритмів та дозволяє наочно контролювати положення об'єктів у просторі [8].

Під час тестування були також виявлені певні обмеження системи. Основними недоліками є:

- залежність точності роботи від умов освітлення;
- зниження якості визначення об'єктів при великій кількості шумів;
- обмеження використання звичайної RGB-камери для оцінки глибини сцени.

Для підвищення точності роботи системи у майбутньому доцільно реалізувати:

- використання depth-камер;
- інтеграцію LiDAR-сенсорів;
- застосування алгоритмів машинного навчання;
- автоматичне планування траєкторії руху;
- використання алгоритмів SLAM [26].

У результаті проведених досліджень було підтверджено можливість використання розробленої системи для задач навігації роботів-маніпуляторів та аналізу навколишнього середовища.

Проведений аналіз показав, що система забезпечує достатню точність побудови карти, стабільну роботу у режимі реального часу та ефективну взаємодію між програмними компонентами.

Таким чином, розроблений програмний застосунок забезпечує побудову карти перешкод на основі технологій комп'ютерного зору та може бути використаний як основа для подальшого розвитку інтелектуальних робототехнічних систем.

Висновки до розділу 4

У даному розділі було проведено реалізацію та тестування програмної системи побудови карти перешкод для робота-маніпулятора.

Було виконано налаштування середовища ROS та RViz, реалізовано основні програмні модулі системи та проведено тестування алгоритмів комп'ютерного зору [6].

У процесі тестування підтверджено працездатність системи виявлення перешкод та побудови карти навколишнього середовища. Розроблений застосунок забезпечує стабільне отримання відеоданих, визначення координат об'єктів та передачу результатів до системи візуалізації.

Проведений аналіз точності показав, що система забезпечує достатню якість побудови карти для задач навігації роботів-маніпуляторів [10]. Також було встановлено, що продуктивність програмного забезпечення дозволяє працювати у режимі реального часу.

У результаті проведених досліджень було визначено переваги та обмеження розробленої системи, а також запропоновано можливі напрямки подальшого вдосконалення програмного застосунку.

Отримані результати підтверджують ефективність використання технологій комп'ютерного зору, платформи ROS та середовища RViz для реалізації систем побудови карти перешкод у робототехнічних комплексах.

5 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА ПІД ЧАС РОБОТИ З РОБОТОТЕХНІЧНИМИ СИСТЕМАМИ

5.1 Аналіз небезпечних факторів під час роботи з роботами-маніпуляторами

Під час розробки та експлуатації робототехнічних систем важливе значення має забезпечення безпечних умов праці. Роботи-маніпулятори поєднують механічні, електронні та програмні компоненти, тому під час роботи з такими системами можуть виникати різноманітні небезпечні фактори, які здатні негативно впливати на здоров'я людини та безпечність виконання робіт [33].

Одним із основних небезпечних факторів є рухомі механічні елементи робота-маніпулятора. Під час роботи робот може виконувати швидкі переміщення окремих ланок, що створює ризик травмування оператора у випадку контакту з рухомими частинами.

До найбільш небезпечних механічних елементів належать:

- обертальні механізми;
- рухомі ланки маніпулятора;
- приводи;
- захоплювачі об'єктів.

Для зменшення ризику травмування необхідно обмежити доступ до робочої зони робота під час його функціонування. На виробництві для цього використовуються спеціальні захисні огороження, датчики безпеки та системи аварійного вимкнення.

Ще одним важливим фактором безпеки є електричний струм. Робототехнічні системи працюють від електромережі та містять значну кількість електронних компонентів. Пошкодження ізоляції кабелів або неправильне

підключення обладнання може призвести до ураження електричним струмом [33]. Для забезпечення електробезпеки необхідно:

- використовувати справне обладнання;
- контролювати стан кабелів;
- виконувати заземлення системи;
- дотримуватися правил експлуатації електрообладнання.

Під час роботи із комп'ютерною технікою значний вплив має навантаження на органи зору та опорно-руховий апарат. Тривала робота за комп'ютером може призводити до:

- втоми очей;
- погіршення постави;
- болю у спині;
- перенапруження м'язів.

Для зменшення негативного впливу необхідно правильно організувати робоче місце та дотримуватися режиму праці та відпочинку.

У процесі роботи робототехнічних систем також можуть виникати програмні помилки або збої у роботі сенсорів. Некоректна робота програмного забезпечення може призвести до неправильного визначення координат об'єктів або небезпечного руху маніпулятора. Для зменшення ризику аварійних ситуацій необхідно:

- виконувати тестування програмного забезпечення;
- використовувати системи аварійного вимкнення;
- контролювати стабільність роботи сенсорів;
- перевіряти коректність передачі даних між ROS-вузлами.

На рисунку 5.1 представлено основні небезпечні фактори під час роботи з роботами-маніпуляторами.



Рисунок 5.1 – Небезпечні фактори робототехнічної системи

Таким чином, під час роботи з роботами-маніпуляторами необхідно враховувати механічні, електричні та програмні фактори небезпеки та забезпечувати дотримання правил охорони праці.

5.2 Вимоги електробезпеки при роботі з обладнанням

Під час роботи з комп'ютерною технікою, робототехнічними системами та електронним обладнанням особливу увагу необхідно приділяти дотриманню вимог електробезпеки [33].

Електричний струм є одним із найнебезпечніших факторів виробничого середовища, оскільки може призвести до травмування людини, пошкодження обладнання або виникнення пожежі.

Для забезпечення безпечної роботи необхідно використовувати лише справне обладнання, яке відповідає технічним вимогам та має належний рівень ізоляції. Перед початком роботи необхідно перевірити:

- справність кабелів живлення;
- відсутність пошкоджень ізоляції;
- наявність заземлення;
- коректність підключення обладнання.

Забороняється:

- працювати із пошкодженими кабелями;
- торкатися електрообладнання мокрими руками;
- виконувати ремонт обладнання під напругою;
- використовувати несправні розетки або подовжувачі.

У випадку виникнення несправностей обладнання необхідно негайно припинити роботу та відключити систему від електромережі. Для захисту обладнання та користувача рекомендується використовувати:

- автоматичні вимикачі;
- стабілізатори напруги;
- джерела безперебійного живлення.

Особливу увагу необхідно приділяти організації кабельного підключення. Кабелі не повинні знаходитися у місцях можливого механічного пошкодження або перешкоджати переміщенню працівника.

Під час роботи з роботами-маніпуляторами важливим є наявність кнопки аварійного вимкнення. У випадку небезпечної ситуації це дозволяє швидко

припинити роботу системи та уникнути травмування або пошкодження обладнання.

Також необхідно дотримуватися вимог нормативних документів та правил експлуатації електрообладнання, визначених законодавством України [34]. На рисунку 5.2 представлено приклад організації електробезпеки робототехнічної системи.



Рисунок 5.2 – Засоби електробезпеки робототехнічної системи

Таким чином, дотримання вимог електробезпеки є необхідною умовою безпечної експлуатації робототехнічних систем та комп'ютерного обладнання.

5.3 Організація безпечного робочого місця

Організація робочого місця є важливим фактором забезпечення безпечних та комфортних умов праці під час роботи з комп'ютерними системами та робототехнічним обладнанням.

Робоче місце користувача повинно відповідати санітарно-гігієнічним та ергономічним вимогам. Правильна організація робочого простору дозволяє зменшити втому, покращити продуктивність праці та знизити ризик виникнення професійних захворювань [33].

Робочий стіл повинен мати достатню площу для розміщення:

- комп'ютера;
- монітора;
- клавіатури;
- миші;
- додаткового обладнання.

Монітор рекомендується розташовувати на відстані 50–70 см від очей користувача. Верхня межа екрана повинна знаходитися приблизно на рівні очей або трохи нижче.

Для забезпечення комфортної роботи необхідно використовувати зручне крісло з можливістю регулювання висоти та положення спинки.

Особливу увагу необхідно приділяти освітленню робочого місця. Освітлення повинно бути достатнім, але не створювати відблисків на поверхні монітора. Рекомендується використовувати комбіноване освітлення:

- природне;
- штучне.

Під час тривалої роботи за комп'ютером необхідно робити регулярні перерви. Це дозволяє зменшити навантаження на органи зору та опорно-руховий апарат. Рекомендується:

- робити короткі перерви кожні 45–60 хвилин;
- виконувати вправи для очей;
- змінювати положення тіла під час роботи.

Під час роботи із робототехнічними системами необхідно забезпечити вільний доступ до аварійного вимкнення обладнання та не допускати сторонніх осіб до робочої зони робота.

На рисунку 5.3 представлено приклад організації безпечного робочого місця оператора робототехнічної системи.



Рисунок 5.3 – Організація робочого місця оператора

Таким чином, правильна організація робочого місця є важливою умовою забезпечення безпечної та ефективної роботи з робототехнічними системами та комп'ютерним обладнанням.

Висновки до розділу 5

У даному розділі було проведено аналіз основних небезпечних факторів, які можуть виникати під час роботи з роботами-маніпуляторами та комп'ютерним обладнанням.

Було розглянуто механічні, електричні та організаційні фактори небезпеки, а також визначено основні заходи щодо забезпечення безпечних умов праці [33].

Особливу увагу приділено вимогам електробезпеки під час роботи із робототехнічними системами та правилами експлуатації електронного обладнання.

Також було проаналізовано вимоги до організації робочого місця користувача та визначено рекомендації щодо зменшення негативного впливу тривалої роботи за комп'ютером.

Проведений аналіз показав, що дотримання вимог охорони праці та правил безпеки є необхідною умовою ефективної та безпечної роботи робототехнічних систем.

ВИСНОВКИ

У даній дипломній роботі було розроблено програмну систему побудови карти перешкод для робота-маніпулятора на основі технологій комп'ютерного зору з можливістю візуалізації результатів у середовищі RViz.

У процесі виконання роботи було проведено аналіз предметної області робототехнічних систем та досліджено сучасні методи навігації, побудови карт навколишнього середовища і виявлення перешкод. Було встановлено, що використання технологій комп'ютерного зору є ефективним підходом для аналізу робочого простору та забезпечення безпечного переміщення роботів-маніпуляторів [5].

Під час виконання роботи було проведено аналіз існуючих програмних рішень та інструментальних засобів, які використовуються у сфері робототехніки. У результаті аналізу було обрано платформу ROS, бібліотеку OpenCV, середовище RViz та мову програмування Python як основні технології для реалізації програмного застосунку [6].

У межах роботи було спроектовано архітектуру програмної системи та реалізовано основні функціональні модулі:

- модуль отримання відеоданих;
- модуль обробки зображень;
- модуль виявлення перешкод;
- модуль побудови карти;
- модуль візуалізації результатів у RViz.

Для виявлення перешкод було використано алгоритми цифрової обробки зображень бібліотеки OpenCV, зокрема методи аналізу контурів та виділення меж об'єктів. Реалізований алгоритм забезпечує визначення координат перешкод у режимі реального часу та передачу отриманих даних до системи побудови карти

[14]. У процесі роботи було реалізовано формування occupancy grid map, яка використовується для представлення навколишнього середовища та відображення зайнятих і вільних областей робочого простору [10].

Для наочного відображення результатів роботи системи було використано середовище RViz. Реалізована система візуалізації дозволяє у режимі реального часу переглядати карту перешкод, положення об'єктів та результати роботи алгоритмів комп'ютерного зору [8].

Під час тестування програмного застосунку було підтверджено працездатність системи та стабільність роботи алгоритмів. Проведений аналіз показав, що система забезпечує достатню точність побудови карти та швидкість обробки даних для використання у робототехнічних системах.

У результаті виконання дипломної роботи було досягнуто поставленої мети та реалізовано систему побудови карти перешкод для робота-маніпулятора на основі комп'ютерного зору з візуалізацією у RViz.

Практичне значення отриманих результатів полягає у можливості використання розробленої системи для задач навігації робототехнічних комплексів, аналізу навколишнього середовища та уникнення зіткнень у процесі роботи роботів-маніпуляторів. Перспективами подальшого розвитку системи є:

- використання depth-камер та LiDAR-сенсорів;
- інтеграція алгоритмів машинного навчання;
- реалізація автоматичного планування траєкторії руху;
- побудова тривимірних карт навколишнього середовища;
- підвищення точності визначення координат об'єктів;
- використання алгоритмів SLAM та OctoMap [25].

Отримані результати підтверджують ефективність використання технологій комп'ютерного зору, платформи ROS та середовища RViz для реалізації сучасних робототехнічних систем.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Siegwart R. Introduction to Autonomous Mobile Robots / R. Siegwart, I. Nourbakhsh, D. Scaramuzza. – Cambridge : MIT Press, 2011. – 457 p.
2. Corke P. Robotics, Vision and Control / P. Corke. – Springer, 2017. – 693 p.
3. Siciliano B. Robotics: Modelling, Planning and Control / B. Siciliano, L. Sciavicco, L. Villani. – Springer, 2010. – 632 p.
4. Bradski G. Learning OpenCV: Computer Vision with the OpenCV Library / G. Bradski, A. Kaehler. – Sebastopol : O'Reilly Media, 2008. – 555 p.
5. Szeliski R. Computer Vision: Algorithms and Applications / R. Szeliski. – Springer, 2022. – 979 p.
6. Robot Operating System (ROS) Documentation [Electronic resource]. – Access mode: <https://www.ros.org/>
7. OpenCV Documentation [Electronic resource]. – Access mode: <https://opencv.org/>
8. RViz User Guide [Electronic resource]. – Access mode: <https://wiki.ros.org/rviz>
9. Quigley M. ROS: an open-source Robot Operating System / M. Quigley // ICRA Workshop on Open Source Software. – 2009. – P. 1–6.
10. Thrun S. Probabilistic Robotics / S. Thrun, W. Burgard, D. Fox. – Cambridge : MIT Press, 2005. – 647 p.
11. Hartley R. Multiple View Geometry in Computer Vision / R. Hartley, A. Zisserman. – Cambridge University Press, 2004. – 655 p.
12. Forsyth D. Computer Vision: A Modern Approach / D. Forsyth, J. Ponce. – Pearson, 2015. – 792 p.
13. Redmon J. You Only Look Once: Unified, Real-Time Object Detection / J. Redmon // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. – 2016. – P. 779–788.

14. Gonzalez R. Digital Image Processing / R. Gonzalez, R. Woods. – Pearson Education, 2018. – 976 p.
15. Lavallo S. Planning Algorithms / S. Lavallo. – Cambridge University Press, 2006. – 826 p.
16. Python Software Foundation [Electronic resource]. – Access mode: <https://www.python.org/>
17. Ubuntu Documentation [Electronic resource]. – Access mode: <https://ubuntu.com/>
18. MoveIt Documentation [Electronic resource]. – Access mode: <https://moveit.ros.org/>
19. Open Source Robotics Foundation [Electronic resource]. – Access mode: <https://www.openrobotics.org/>
20. Murphy R. Introduction to AI Robotics / R. Murphy. – Cambridge : MIT Press, 2019. – 648 p.
21. Bishop C. Pattern Recognition and Machine Learning / C. Bishop. – Springer, 2006. – 738 p.
22. Goodfellow I. Deep Learning / I. Goodfellow, Y. Bengio, A. Courville. – MIT Press, 2016. – 775 p.
23. Klette R. Concise Computer Vision / R. Klette. – Springer, 2014. – 429 p.
24. O’Kane J. A Gentle Introduction to ROS / J. O’Kane. – Chapman and Hall/CRC, 2023. – 282 p.
25. Hornung A. OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees / A. Hornung // Autonomous Robots. – 2013. – Vol. 34. – P. 189–206.
26. Durrant-Whyte H. Simultaneous Localization and Mapping: Part I / H. Durrant-Whyte, T. Bailey // IEEE Robotics & Automation Magazine. – 2006. – Vol. 13. – P. 99–110.

27. Bailey T. Simultaneous Localization and Mapping (SLAM): Part II / T. Bailey, H. Durrant-Whyte // IEEE Robotics & Automation Magazine. – 2006. – Vol. 13. – P. 108–117.
28. NVIDIA Developer Documentation [Electronic resource]. – Access mode: <https://developer.nvidia.com/>
29. Intel RealSense Documentation [Electronic resource]. – Access mode: <https://www.intelrealsense.com/>
30. Gazebo Simulator Documentation [Electronic resource]. – Access mode: <https://gazebo.org/>
31. ДСТУ 3008:2015. Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлення.
32. ДСТУ 8302:2015. Бібліографічне посилання. Загальні положення та правила складання.
33. Закон України «Про охорону праці» [Електронний ресурс]. – Режим доступу: <https://zakon.rada.gov.ua/>
34. Кодекс законів про працю України [Електронний ресурс]. – Режим доступу: <https://zakon.rada.gov.ua/>
35. Методичні рекомендації до виконання кваліфікаційних робіт бакалавра КПІ ім. Ігоря Сікорського.

ДОДАТОК А Лістинг програми

cv_obstacle_publisher.py

```
import cv2
import rclpy
from rclpy.node import Node
from visualization_msgs.msg import Marker, MarkerArray
from builtin_interfaces.msg import Duration

class CVObstaclePublisher(Node):

    def __init__(self):
        super().__init__('cv_obstacle_publisher')

        self.publisher_ = self.create_publisher(
            MarkerArray,
            'cv_obstacle_markers',
            10
        )

        self.cap = cv2.VideoCapture('/dev/video0', cv2.CAP_V4L2)

        if not self.cap.isOpened():
            self.get_logger().error("Camera not found")
            return

        self.timer = self.create_timer(0.1, self.process_camera)

    def make_marker(self, marker_id, ns, marker_type, x, y, z, sx, sy, sz, r, g, b, a):
        marker = Marker()

        marker.header.frame_id = "map"
        marker.header.stamp = self.get_clock().now().to_msg()

        marker.ns = ns
        marker.id = marker_id
        marker.type = marker_type
        marker.action = Marker.ADD

        marker.pose.position.x = x
        marker.pose.position.y = y
        marker.pose.position.z = z
        marker.pose.orientation.w = 1.0

        marker.scale.x = sx
        marker.scale.y = sy
        marker.scale.z = sz

        marker.color.r = r
        marker.color.g = g
        marker.color.b = b
        marker.color.a = a

        marker.lifetime = Duration(sec=0, nanosec=300000000)

        return marker
```

```

def add_puma_model(self, marker_array):
    marker_array.markers.append(
        self.make_marker(100, "puma560", Marker.CYLINDER,
                        0.0, 0.0, 0.15,
                        0.5, 0.5, 0.3,
                        0.0, 0.2, 1.0, 1.0)
    )

    marker_array.markers.append(
        self.make_marker(101, "puma560", Marker.SPHERE,
                        0.0, 0.0, 0.4,
                        0.25, 0.25, 0.25,
                        0.0, 0.2, 1.0, 1.0)
    )

    marker_array.markers.append(
        self.make_marker(102, "puma560", Marker.CUBE,
                        0.35, 0.0, 0.65,
                        0.7, 0.12, 0.12,
                        0.6, 0.6, 0.6, 1.0)
    )

    marker_array.markers.append(
        self.make_marker(103, "puma560", Marker.SPHERE,
                        0.7, 0.0, 0.65,
                        0.22, 0.22, 0.22,
                        0.0, 0.2, 1.0, 1.0)
    )

    marker_array.markers.append(
        self.make_marker(104, "puma560", Marker.CUBE,
                        0.95, 0.0, 0.95,
                        0.12, 0.12, 0.65,
                        0.6, 0.6, 0.6, 1.0)
    )

    marker_array.markers.append(
        self.make_marker(105, "puma560", Marker.SPHERE,
                        0.95, 0.0, 1.3,
                        0.18, 0.18, 0.18,
                        0.0, 0.2, 1.0, 1.0)
    )

    marker_array.markers.append(
        self.make_marker(106, "puma560", Marker.CUBE,
                        1.2, 0.0, 1.3,
                        0.35, 0.08, 0.08,
                        0.1, 0.1, 0.1, 1.0)
    )

def process_camera(self):
    ret, frame = self.cap.read()

    if not ret:
        self.get_logger().warning("No frame received")
        return

    frame = cv2.resize(frame, (640, 480))
    height, width = frame.shape[:2]

```

```

hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

lower_red_1 = (0, 180, 120)
upper_red_1 = (5, 255, 255)

lower_red_2 = (175, 180, 120)
upper_red_2 = (180, 255, 255)

mask1 = cv2.inRange(hsv, lower_red_1, upper_red_1)
mask2 = cv2.inRange(hsv, lower_red_2, upper_red_2)

thresh = mask1 + mask2

small_kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (5, 5))
thresh = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, small_kernel)

join_kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (35, 35))
thresh = cv2.dilate(thresh, join_kernel, iterations=1)
thresh = cv2.morphologyEx(thresh, cv2.MORPH_CLOSE, join_kernel)

contours, _ = cv2.findContours(
    thresh,
    cv2.RETR_EXTERNAL,
    cv2.CHAIN_APPROX_SIMPLE
)

marker_array = MarkerArray()
self.add_puma_model(marker_array)

obstacle_count = 0

for contour in contours:
    area = cv2.contourArea(contour)

    if area < 300:
        continue

    x, y, w, h = cv2.boundingRect(contour)

    cx = x + w // 2
    cy = y + h // 2

    camera_origin_x = 1.2
    camera_origin_y = 0.0
    camera_origin_z = 1.3

    map_x = camera_origin_x + ((cx - width / 2) / width) * 1.5

    map_y = camera_origin_y + (
        0.3 + (1.0 - min(area / 50000, 1.0)) * 1.2
    )

    map_z = camera_origin_z - (
        (cy - height / 2) / height
    ) * 1.2

    size = min(max(area / 20000, 0.15), 0.6)

    marker_array.markers.append(
        self.make_marker(

```

```

        obstacle_count,
        "cv_obstacles",
        Marker.CUBE,
        map_x,
        map_y,
        map_z,
        size,
        size,
        size,
        1.0,
        0.0,
        0.0,
        1.0
    )
)

estimated_distance = map_y

cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 2)

cv2.putText(
    frame,
    f"Obstacle {obstacle_count} | Distance: {estimated_distance:.2f} m",
    (x, y - 10),
    cv2.FONT_HERSHEY_SIMPLEX,
    0.6,
    (0, 0, 255),
    2
)

obstacle_count += 1

self.publisher_.publish(marker_array)

cv2.imshow("Obstacle Detection", frame)
cv2.imshow("Threshold", thresh)
cv2.waitKey(1)

self.get_logger().info(f"Detected obstacles: {obstacle_count}")

def main(args=None):
    rclpy.init(args=args)

    node = CVObstaclePublisher()

    rclpy.spin(node)

    node.cap.release()
    cv2.destroyAllWindows()

    node.destroy_node()
    rclpy.shutdown()

if __name__ == '__main__':
    main()

```

ДОДАТОК Б Презентація



Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Навчально-науковий інститут атомної та теплової енергетики
Кафедра інженерії програмного забезпечення в енергетиці

Побудова карти перешкод для робота- маніпулятора PUMA 560 на основі комп'ютерного зору з візуалізацією в RViz

Виконала: Кушнір Катерина Вікторівна

Керівник: Доцент, к.т.н. Варава Іван Андрійович

2026

АКТУАЛЬНІСТЬ ТЕМИ

Актуальність роботи — полягає в необхідності своєчасного виявлення та візуалізації перешкод у робочій зоні робота-маніпулятора для запобігання зіткненням, підвищення безпеки роботи роботизованих систем та оптимізації планування траєкторії руху маніпулятора.

Особливо актуальним є використання методів комп'ютерного зору та систем візуалізації для побудови карти перешкод у реальному часі.

Мета роботи — розробити програмний продукт для побудови та візуалізації карти перешкод робота-маніпулятора PUMA 560 на основі методів комп'ютерного зору з використанням ROS2 та RViz.



Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

2

ПОСТАНОВКА ЗАДАЧІ

Для досягнення поставленої мети необхідно вирішити такі завдання:

- Провести аналіз існуючих систем побудови карт перешкод для роботів-маніпуляторів
- Виконати проектування програмної системи
- Розробити модуль комп'ютерного зору для виявлення перешкод
- Реалізувати побудову карти перешкод
- Розробити візуалізацію робочого простору в RViz
- Виконати інтеграцію модулів у середовище ROS2
- Провести тестування програмного продукту



Функціональні та нефункціональні вимоги

Функціональні:

- отримання відеопотоку;
- виявлення перешкод;
- визначення координат;
- побудова карти;
- візуалізація в RViz.

Нефункціональні:

- робота в реальному часі;
- масштабованість;
- модульність;
- сумісність з ROS2;
- простота розгортання.



Аналіз існуючих систем побудови карт перешкод

Система	Тип сенсора	ROS2	Візуалізація	Точність	Вартість
Intel RealSense	RGB-D камера	Так	RViz	Висока	Висока
LiDAR Mapping Systems	LIDAR	Так	RViz	Дуже висока	Дуже висока
Stereo Vision Systems	Стереокамери	Так	RViz	Висока	Середня
Запропонована система	RGB камера	Так	RViz	Середня	Низька



Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

БІБЛІОТЕКИ І ФРЕЙМВОРКИ

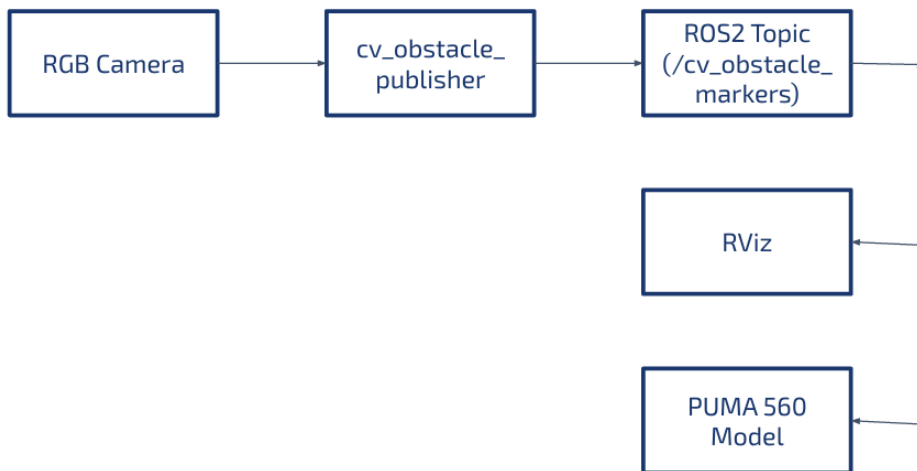
Використанні технології:



Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

6

Діаграма Компонентів



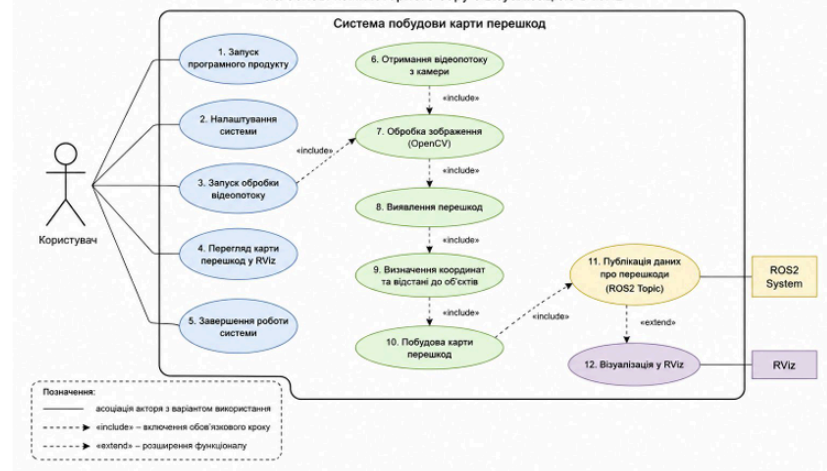
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»



ПРОЕКТУВАННЯ СИСТЕМИ

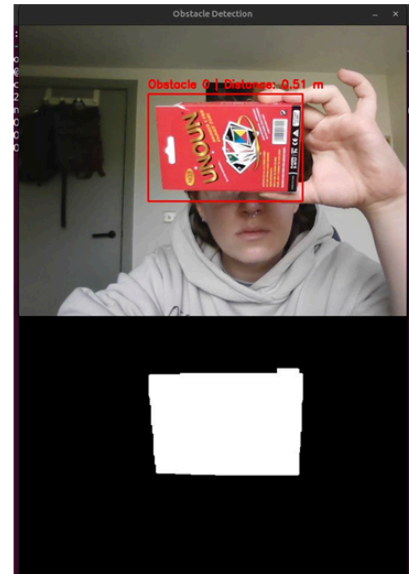
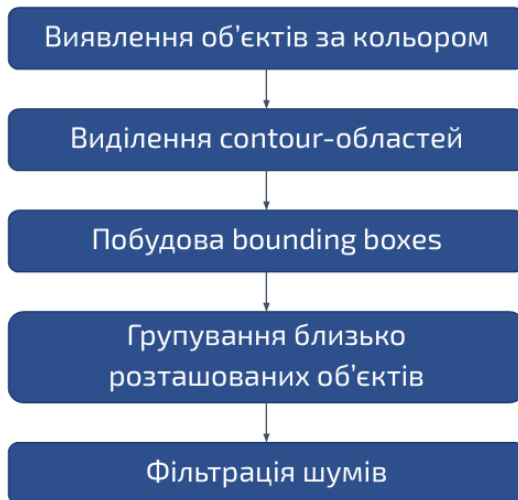
Use Case діаграма роботи програмного продукту

Побудова карти перешкод для робота-маніпулятора PUMA 560
на основі комп'ютерного зору з візуалізацією в RViz



Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Алгоритм виявлення перешкод за допомогою OpenCV:

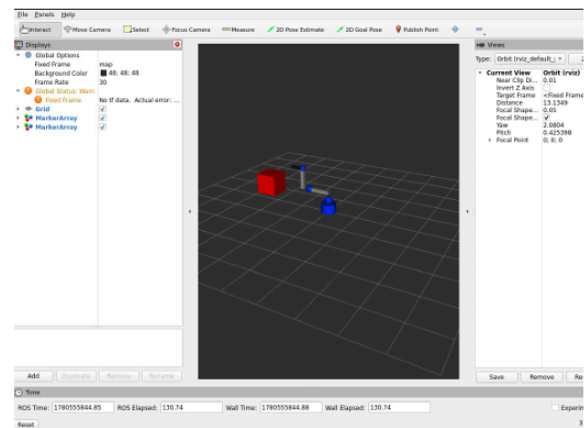


Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

9

Результат візуалізації перешкод у RViz

- Відображення перешкод у вигляді 3D-маркерів
- Оновлення карти у реальному часі
- Візуалізація робочої зони PUMA 560
- Відображення положення obstacle objects



10

Оцінка відстані до об'єктів

Чим більша площа об'єкта в кадрі,
тим ближче він знаходиться до камери.



Використаний підхід:

- аналіз площі contour-об'єкта
- визначення положення об'єкта в кадрі
- оцінка відносної глибини

Реалізовано:

- Оцінка приблизної глибини сцени
- Визначення просторового положення перешкод
- Конфігурація «камера на кінцевому ефекторі»

$$D \propto \frac{1}{A}$$

де:

- D — приблизна відстань до об'єкта;
- A — площа контуру.



Результати тестування програмного продукту

ID	Вимога	Метод перевірки	Результат
FR-01	Отримання відеопотоку з камери	Запуск RGB камери	PASSED
FR-02	Виявлення перешкод засобами OpenCV	Тестування на реальних об'єктах	PASSED
FR-03	Визначення координат перешкод	Аналіз положення маркерів у RViz	PASSED
FR-04	Оцінка відстані до об'єктів	Зміна положення об'єкта перед камерою	PASSED
FR-05	Передача даних через ROS2 Topic	Моніторинг ROS2 повідомлень	PASSED
FR-06	Візуалізація карти перешкод у RViz	Перевірка відображення маркерів	PASSED
FR-07	Оновлення карти в реальному часі	Динамічне переміщення об'єкта	PASSED
FR-08	Відображення моделі PUMA 560	Перевірка моделі в RViz	PASSED



ВИСНОВКИ

- Реалізовано програмний продукт для побудови карти перешкод робота-маніпулятора PUMA 560
- Створено систему обробки відеопотоку на основі методів комп'ютерного зору
- Реалізовано виявлення перешкод у робочій зоні маніпулятора за допомогою OpenCV
- Виконано визначення приблизних координат та відстані до об'єктів
- Реалізовано передачу даних через ROS2 topics
- Створено 3D-візуалізацію моделі маніпулятора та карти перешкод у RViz
- Реалізовано оновлення карти перешкод у реальному часі
- Програмний продукт протестовано та підготовлено для подальшого розвитку й інтеграції з роботизованими системами



Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

14



Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute"