Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

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

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Другий (магістерський)
(рівень вищої освіти)

«Розробка програмного забезпечення для побудови 3D моделі навколишнього середовища мобільного робота»
(тема)

Виконав: студент	2 курсу, гр. КТРСм-22-1
	Акопов М.Г.
	(прізвище, ініціали)
Спеціальність	151 Автоматизація
та комп'ютерно-інтегровані технології	
освітньої	Комп'ютеризовані та
програми	робототехнічні системи
	(код і повна назва напряму)
Тип програми	освітньо-професійна
	(повна назва освітньої програми)
Керівник	доц. Максимова С.С.
	(посада, прізвище, ініціали)

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

2024 р.

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

Дата: 23.01.2024 Підпис: 

Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
Кафедра КІТАР
Рівень вищої освіти Другий (магістерьский)
Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
Тип програми освітньо-професійна
Освітня програма Автоматизація та комп`ютерно-інтегровані технології
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри КІТАР _____
(підпис)
«____»_____20__ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Акопову Михайлу Георгійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка програмного забезпечення для побудови 3D моделі навколишнього середовища мобільного робота
затверджена наказом по університету від 03.11.2023р. № 1287 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 24.01.2024 р.
3. Вихідні дані до роботи
3.1 Середовище моделювання Fusion360
3.2 Операційна система роботів
3.4 Забезпечити побудову хмари точок за допомогою лідара
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
4.1 Вступ
4.2 Аналіз технічного завдання
4.3 Розробка програмного забезпечення для побудови 3D моделі навколишнього середовища мобільного робота
4.4 Вибір апаратних засобів
4.7 Вказівки щодо заходів з безпечної експлуатації
4.8 Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) Демонстраційний матеріал представлений у форматі презентації PowerPoint (*.ppt) – с. формату А4

6. Консультанти розділів роботи

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

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз технічного завдання	21.09.2023	Виконано
2	Провести аналіз існуючих рішень машинного зору	05.10.2023	Виконано
3	Провести розробку алгоритму роботи	19.10.2023	Виконано
4	Провести вибір апаратних засобів для модуля машинного зору	10.11.2023	Виконано
5	Розробити програмний засіб для побудови 3D моделі навколишнього середовища мобільного робота	11.12.2023	Виконано
6	Провести розрахунки, пов'язані з охороною праці	20.12.2023	Виконано
7	Оформлення пояснювальної записки	27.12.2023	Виконано
8	Подання роботі до ЕК	12.01.2024	

Дата видачі завдання 03.11.2023р

Студент — Акопов М.Г.

(підпис) — (прізвище, ініціали)

Керівник роботи — доц. Максимова С.С.

(підпис) — (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 46 с., 2 табл., 29 рис., 2 дод., 17 джерел.

АВТОМАТИЗАЦІЯ, АЛГОРИТМ, МОТОР-РЕДУКТОР, МОДУЛЬ КЕРУВАННЯ, ІДЕНТИФІКАЦІЯ ОБ’ЄКТІВ.

Мета роботи – підвищення ефективності виробництва шляхом створення тривимірної карти простору для навігації мобільного робота.

Об’єкт розробки – процес ідентифікації об'єктів.

Предмет розробки – модуль ідентифікації об'єктів для роботизованої платформи.

Для досягнення мети проаналізовано існуючі програмно-технічні методи реалізації машинного зору та ідентифікації об’єктів. Розроблено схему розташування компонентів, необхідних для ідентифікації, відстеження перешкод та будування 3D моделі навколишнього середовища мобільного робота.

ABSTRACT

Explanatory note: 46 p., 2 tabl., 29 pic., 2 applications, 17 sources.

AUTOMATION, ALGORITHM, GEAR MOTOR, CONTROL MODULE, OBJECT IDENTIFICATION.

The purpose of the work is to increase production efficiency by creating a three-dimensional map of space for the navigation of a mobile robot.

The object of development is the process of building a three-dimensional map of space.

The subject of development is the development of software for building an environment map.

To achieve the goal, the existing software and technical methods of implementing machine vision and object identification were analyzed. A diagram of the location of the components necessary for identification, tracking of obstacles and construction of a 3D model of the mobile robot's environment has been developed.

ЗМІСТ

Перелік скорочень..... 9

Вступ..... 10

1 Аналіз технічного завдання..... 11

1.1 Історія розвитку технології машиного зору..... 11

1.2 Огляд існуючих рішень детекції об’єктів..... 12

1.2.1 Structured light камери..... 12

1.2.2 Time of Flight камери..... 13

1.2.3 Depth from Stereo камери..... 15

1.2.4 Light Field камери..... 16

1.2.5 Камери на lidar технологіях..... 18

1.3 Висновки за розділом 1..... 23

2 Вибір компонентів та програмного середовища для побудови 3D простору 22

2.1 Вибір компонентів автоматизованого модуля 22

2.1.1 Вибір обчислювального модуля..... 22

2.1.2 Лідар RPLIDAR A1..... 24

2.1.3 SD Картка..... 26

2.2 Вибір середовища та методів розробки..... 28

2.2.1 Операційна система Linux Ubuntu (18.04.6 – Bionic Beavel)… 28

2.2.2 ROS фреймворк для роботизованих систем..... 29

2.2.3 Хмара точок та її використання..... 33

2.2.4 CAD система Fusion 360..... 36

2.3 Інструмент симуляції та візуалізації Gazebo, Rviz, Rqt..... 40

2.3.1 Rviz……. 40
2.3.2 Gazebo……. 42
2.3.1 Rqt……. 45
2.4 Висновки за розділом 2…….. 47
3 Розробка програмного забезпечення для побудови 3D моделі навколишнього середовища мобільного робота……..... 48
3.1 Основні відомості щодо структури ROS……..……..… 48
3.2 Налаштування фреймворку, створення робочого простору, та ініціалізація проекту……..……...…...… 50
3.3 Висновки за розділом 3……...…...… 57
Висновоки…….. 58
Перелік джерел посилання……. 59
Додаток А Апробація наукових результатів..……...… 61
Додаток Б Лістинг програми…… 65
Додаток В Презентація……. 96

ПЕРЕЛІК СКОРОЧЕНЬ

НДР – науково-дослідна робота;

ПЗ – програмне забезпечення;

ПК – персональний комп’ютер.

ПЗ – програмне забезпечення;

ТЗ – технічне завдання;

RGB – red, green, blue;

RGB-D – red, green, blue, depth;

ROS – robot operating system.

ВСТУП

В сучасному світі високих технологій, де звичним стало будування одних машин іншими, вклад машинного зору майже неможливо переоцінити. Майже не можливо уявити конвеєрну лінію сучасного автомобільного заводу без багатофункціональних маніпуляторів. У галузі будування безпілотних транспортних засобів машинний зір є безальтернативним. Саме тому виконана у цій роботі розробка автоматизованого модуля побудови навколишнього середовища мобільно робота є актуальною.

Мета роботи – підвищення ефективності виробництва шляхом створення тривимірної карти простору для навігації мобільного робота.

Об’єкт розробки – процес ідентифікації об'єктів.

Предмет розробки – модуль ідентифікації об'єктів для роботизованої платформи.

Для виконання мети роботи необхідно:

– провести аналіз наявних рішень, існуючих рішень машинного зору, RGB-D камер;

– проаналізувати принципи дії, переваги та недоліки апаратних модулів;

– провести вибір середовища розробки;

– налагодити програмне забеспечення для побудови 3D моделінавколишнього середовища мобільного робота.

Робота виконана згідно [1–4].

Принципи дії, переваги та недоліки апаратних модулів було мною розглянуто на конференції сьомої міжнародної науково-технічної конференції “ Виробництво & Мехатронні Системи 2023”, у вигляді тез доповідей [5].

1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ

1.1 Історія розвитку технологій машинного зору

З початку активного розвитку робототехніки наприкінці 20-го століття, індустріязробила величезний крок від незграбного Wabot-1 зразка 1973 року, до неймовірно спритних виробів Boston Dynamics. Як це стало можливо?

Областю що на мою думку дала найбільший поштовх для такого розвитку є машинний зір.

У цій роботі розглянемо як програмні, так і технічні засоби що дозволяють автоматизованим платформам отримувати інформацію про оточення в повному обсязі. А також як отримані дані можуть буті використані.

Перші спроби змусити комп'ютер "бачити" належать до початку 60-х років 20 століття. Однак, лише останніми роками у зв'язку з синергетичним ефектом від таких технологій, як машинне та глибоке навчання (Machine/Deep Learning), штучний інтелект (Artificial Intelligence) та аналіз великих даних (Big Data), технології комп'ютерного (машинного) зору стали знаходити все більше застосувань у різних галузях промисловості та повсякденному житті людей.

Ринок та діапазон застосування комп'ютерного зору в останні 10-15 років значно розширилися.

1.2 Огляд існуючих рішень побудови карти глибини

1.2.1 Structured light камери

Основна ідея вкрай проста. Ставимо поруч проектор, який створює, наприклад, горизонтальні (а потім вертикальні) смужки та поруч камеру, яка знімає картину зі смужками, проілюстровано на рис. 1.1.

Оскільки камера і проекторзміщені один щодо одного, то смужки також будуть зміщуватися пропорційно відстані до об'єкта. Вимірюючи це зсунення ми можемо розраховувати відстань до об'єкта. Найбільш широко відомий цей принцип вимірювання глибини став, коли у 2010 році компанія Microsoft випустила сенсор глибини MS Kinect. Частіше за все проектор працює в інфрачервоному спектрі.

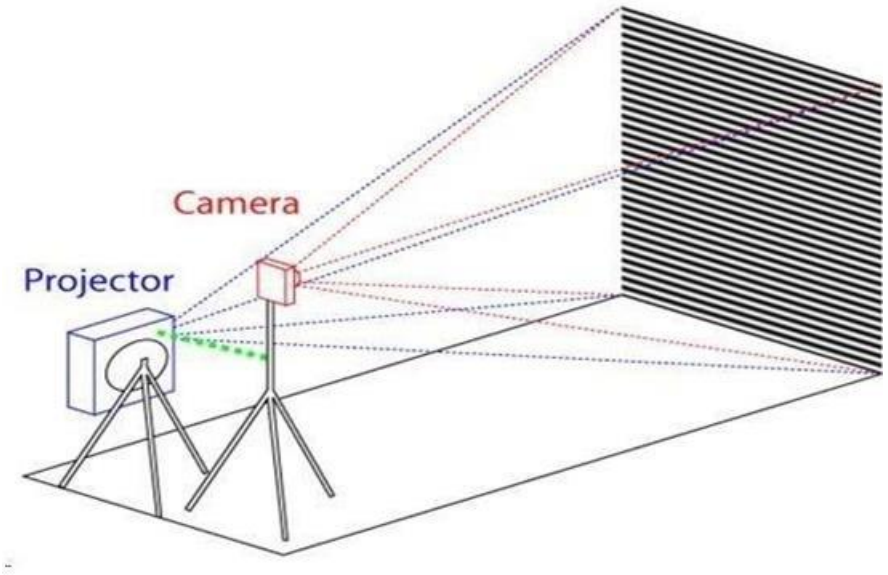


Рисунок 1.1 – Проста схема утворення проекційної сітки

Очевидно, що інфрачервоний сенсор має проблеми. По-перше, сонце нам засвічує наш відносно слабкий проектор, тому на вулиці такі камери не працюють. Навіть у тіні, якщо поруч біла стіна будівлі освітлена сонцем, у вас можуть бути великі проблеми з ID Face. Рівень шумів у Kinect також зашкалює, навіть коли сонце закрите хмарами.

Інша велика проблема – це відображення та відзеркалення. Оскільки інфрачервоне світло також відбивається, то блискучі поверхні буде проблематично ідентифікувати.

Переваги:

– розповсюджена система;

– помірна ціна;

– отримання карти глибини.

Недоліки:

– не ефективно на вулиці, чи у освітленому сонці приміщенні.

Підсумок:

– може бути використана на підприємствах без прямого сонячного освітлення. Компроміс ціни/продуктивності.

1.2.2 Time of Flight камери

Наступний спосіб отримання глибини є цікавішим. Він заснований на вимірі round-trip затримки світла (ToF – Time-of-Flight). Як відомо, швидкість сучаснихпроцесорів висока. За один такт процесора на 3 ГГц світло встигає пролетіти лише 10 сантиметрів. Або 10 тактів на метр. Відповідно встановлюємо імпульсне джерело світла та спеціальну камеру. Фактично нам потрібно виміряти затримку, з якої світло повертається до кожної точки (рис. 1.2).

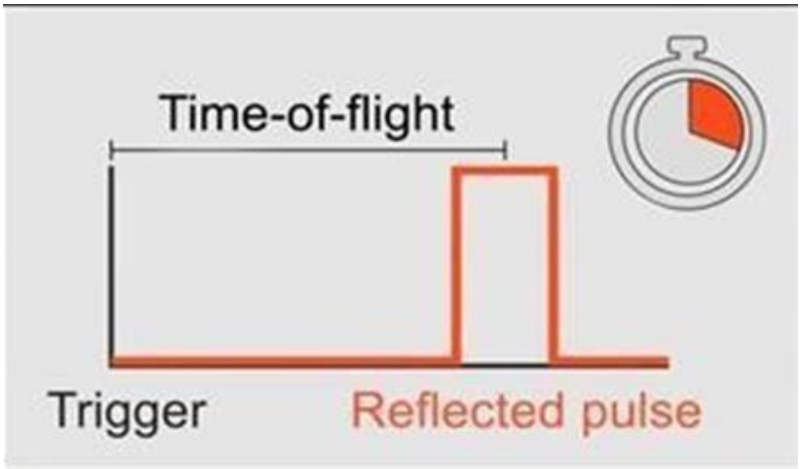


Рисунок 1.2 – Графічний приклад заміру повернення відбитого світлового імпульса

Або, якщо у нас кілька сенсорів з різним часом накопичення заряду, то, знаючи зсув часу щодо джерела для кожного сенсора і знятої яскравості спалаху, ми можемо розрахувати зсув, зображено на рис. 1.3, і відповідно, відстань до об'єкта, причому збільшуючи кількість сенсорів – збільшуємо точність.

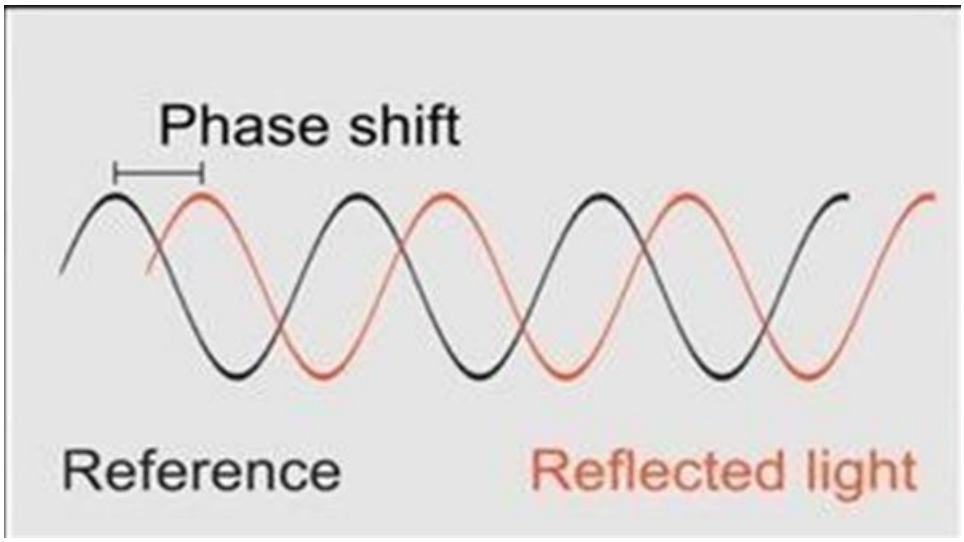


Рисунок 1.3 – Метод зсуву

Переваги:

– відносно велика точність вимірів.

Недоліки:

– так як джерело світла як правило інфрачервоне, маємо ті ж недоліки що і Structured light камери.

Підсумок:

– мало чим відрізняється від попередньої технології, має ті ж самі особливості використання.

1.2.3 Depth from Stereo камери

Побудова карти глибини зі стерео добре відома і застосовується вже понад 40 років. Головна перевага подібних камер – сонячне світло їм не тільки не заважає, а й навпаки, робить їх результати кращими, як наслідок – активне застосування таких камер для всіляких вуличних задач. Приклад побудови карти глибини за допомогою стерео зображено на рисунку 1.4.

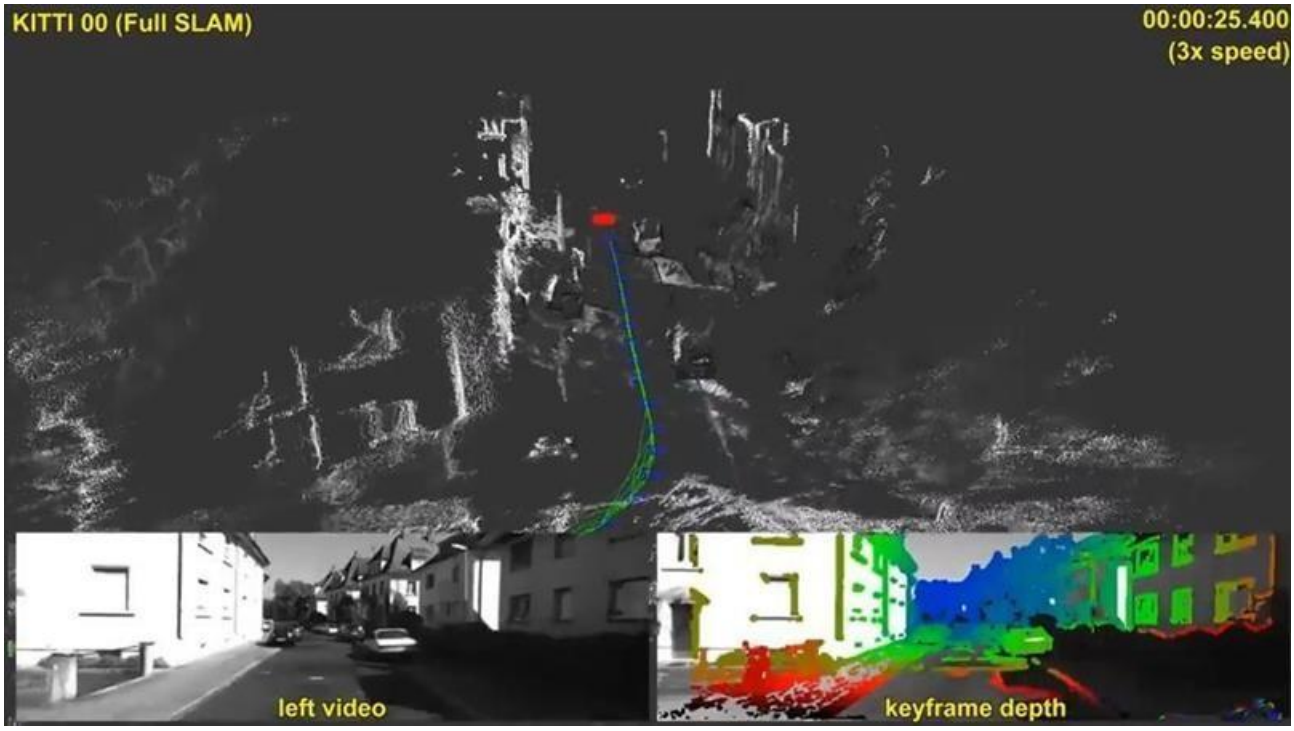


Рисунок 1.4 – Приклади звичайного зображення, і depth from stereo

Помітні поліпшення у переведення побудови глибини зі стерео на новий рівень влила, звичайно, тема автономних автомобілів. З 5 розглянутих способів побудови відео глибини тільки двом – цьому і наступному (стерео і пленоптиці) не заважає сонце і не перешкоджають сусідні автомобілі. При цьому пленоптика в рази дорожча та менш точна на великих дистанціях. Скластися далі може по-різному, прогнози будувати складно, але в даному випадку варто погодитися з

Ілоном Маском – у стерео з усіх 5 способів найкращі перспективи.

Але цікаво, що, схоже, ще сильніший вплив на розвиток теми побудови глибинизі стерео вплинуть не безпілотні автомобілі (яких випускається поки що невелика кількість), а набагато масові пристрої, де вже зараз будується карта глибини по стерео, а саме… Правильно! Смартфони!

Роки чотири тому пішов просто бум «двооких» смартфонів, в якому відзначилися буквально всі бренди, бо якість фотографій, зроблених з однією камерою, і якість фотографій, зроблених з двома кардинально відрізнялася, а з точки зору підвищення ціни смартфона це було не настільки суттєво.

Переваги:

– сонячне світло, ІК джерел світла не впливають на роботу системи.

Недоліки:

– необхідні порівняно більші апаратні можливості для побудови зображення.

Підсумок:

– глибина зі стерео щодо вартості обладнання один з найдешевших спосіб отримання глибини, оскільки камери нині недорогі і продовжують швидко дешевшати. Складність у тому, що подальша обробка набагато ресурсомістка, ніж у інших способов.

1.2.4 Light Field камери

Тема пленоптики (від латинського plenus – повний і optikos – зоровий) або світлових полів поки що порівняно слабо відома широким масам, хоча професіонали їй почали займатися вкрай щільно.

Основна ідея спробувати зафіксувати в кожній точці не просто світло, а двомірний масив світлових променів, зробивши кожен кадр чотиривимірним (рис. 1.5).

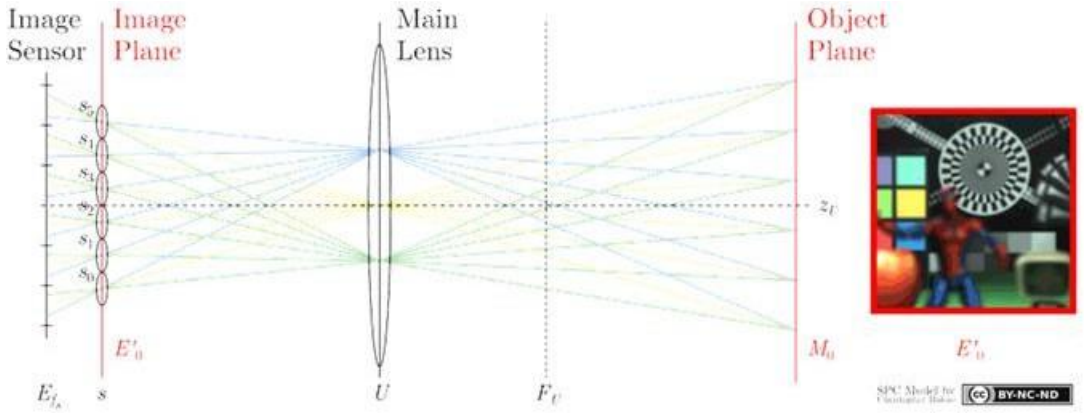


Рисунок 1.5 – На практиці використовуєтьямасив мікролінз

Все це мало б відносно теоретичний інтерес, якби Google у Pixel 2 не реалізував пленоптику, накривши лінзою 2 пікселі.

Як наслідок, утворилася мікростереопара, що дала можливість замірити глибину, до якої Google додала нейромереж, і вийшло взагалі чудово (рис. 1.6).

Рисунок 1.6 – Приклад карти глибини отриманий за допомогою камери Google Pixel 2

Переваги:

– дає змогу отримати карту глибини у високій якості.

Недоліки:

– надто велика ціна.

Підсумок:

– перспективний напрямок для споживчого ринку камер. Проте для робототехніки така якість зображення не є необхідною.

1.2.5 Камери на lidar технологіях

Взагалі, лазерні далекоміри міцно увійшли до нашого життя, коштують недорого і дають високу точність. Перші лідери (від LIDaR – Light Identification Detection and Ranging), побудовані як зв'язки подібних пристроїв, що обертаються навколо горизонтальної осі, першими почали використовувати військові, потім тестували в автопілотах машин. Там вони виявили себе досить добре, що викликало потужний сплеск інвестицій у область. Спочатку лідари оберталися, даючи кілька разів на секунду картину (рис. 1.7).

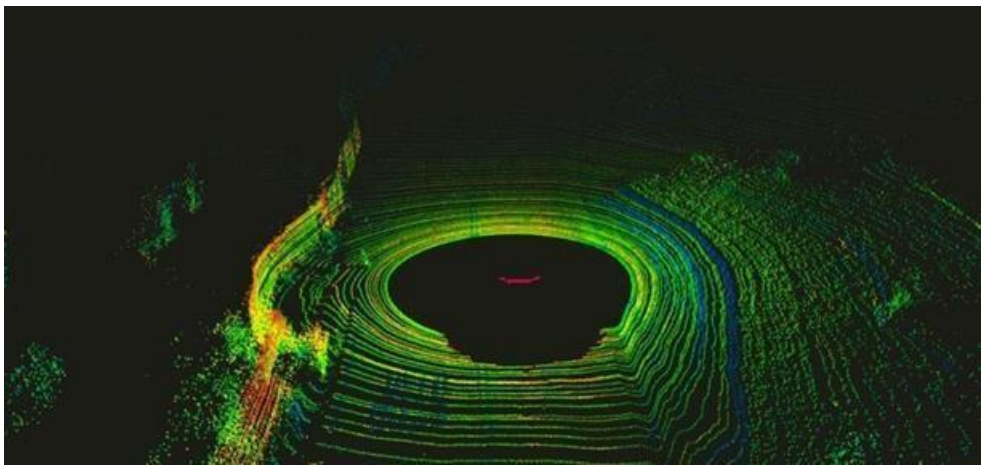


Рисунок 1.7 – Приклад отриманих даних від сенсорів лідара

Основною проблемою таких приладів є їх відносно невелика надійність, що пов’язана з обертанням механізму. Проте за останні роки в цьому напрямку було зроблено деякі модернізації.

Зокрема, відносно нещодавно відразу в кількох виробників з'явилися так звані Solid State Lidar (рис. 1.8), у яких в принципі немає частин, що рухаються, які показують кардинально вищу надійність, особливо при трясці, нижчу вартість і т.д. Що важливо для нас, то це те, що Solid State Lidar дає прямокутну картинку, тобто. по суті починає працювати як «звичайна» камера глибини.

Рисунок 1.8 – Solid State LIDAR відInnovizOne

Приклад вище дає відео приблизно 1024x256, 25 FPS, 12 біт компоненту. Подібні лідари монтуватимуть під решітку радіатора, як це зображено на рис. 1.9, оскільки пристрій добре гріється.

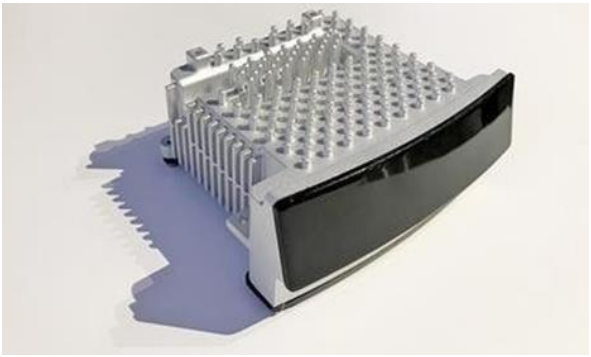


Рисунок 1.9 – Варіант компоновки лідара з пасивним охолодженням

Лідари монтують у різні схеми залежно від вартості комплекту та потужності бортової системи, якою потрібно буде всі ці дані обробляти. Відповідно, змінюються і загальні характеристики автопілота. Як наслідок, дорожчі машиникраще переноситимуть курні дороги і легше «ухилятися» від машин, що в'їжджають на перехрестях збоку, дешеві – лише допоможуть зменшити кількість (чисельних) аварій у пробках.

Якщо говорити про лідари як про камери, варто сказати ще одну важливу особливість, яка істотна при використанні solid-state лідарів. Вони за своєю природою працюють як камери з бігучим затвором, що вносить відчутні спотворення при зйомці рухомих об'єктів (рис. 1.10).

Рисунок 1.10 – Приклад спотворення зображення аналогічний RollingShutter, вінже бігучий затвор

1.3 Висновки за розділом 1

У першому розділі було проведено аналіз існуючих рішень отримання карти глибини, Найбільшу увагу приділено лідарним системам, для з’ясування оптимальної сфери їх використання. У процесі було виявлено існування недоліку лідарних систем, а саме – ефект RollingShutter.

Ця особливість, разом з низьким FPS вимагає адаптації алгоритмів обробки, але в будь-якому випадку завдяки високій точності, у тому числі на велики відстанях у лідарів конкурентів немає. Підведемо підсумки.

Переваги:

– найбільш висока точність, у тому числі найкраща з усіх методів на великих відстанях;

– краще працюють на рівних поверхнях, на яких стерео перестає працювати.

Недоліки:

– високе енергоспоживання;

– відносно низька частота кадрів. Rolling shutter, і необхідність компенсувати його ефект при обробці.

Підсумок:

– завдяки великій точності можуть використовуватися у гібридних сенсорах у автопілотах машин.

Згідно з переліченими вище перевагами – для подальшої розробки було обрано лідар.

2 ВИБІР КОМПОНЕНТІВ ТА ПРОГРАМНОГО СЕРЕДОВИЩА ДЛЯ ПОБУДОВИ 3D ПРОСТОРУ МОБІЛЬНОГО РОБОТА

2.1 Вибір компонентів автоматизованого модуля

2.1.1 Вибір обчислювального модуля

Raspberry Pi – це одноплатний комп'ютер, який був створений з метою стимулювання навчання програмування та забезпечення доступності обчислювальних ресурсів. За свою коротку історію Raspberry Pi став не лише популярним інструментом освіти, а й важливим інструментом для наукових досліджень. Давайте розглянемо еволюцію Raspberry Pi та її внесок у галузі науки.

Історія Raspberry Pi почалася на початку 21 століття, коли група вчених та інженерів із University of Cambridge Computer Laboratory висунула ідею створення доступного комп'ютера для навчання програмуванню. Їхньою метою було забезпечити доступ до обчислювальних ресурсів для широкого кола людей, особливо в освітніх закладах.

У 2008 році команда почала розробляти концепцію недорогого комп'ютера з мінімальними характеристиками, які можна використовувати у навчальних цілях. Цей процес призвів до створення першого Raspberry Pi, Model A та Model B, які були представлені у 2012 році. Ці маленькі, але потужні комп'ютери стали популярними серед ентузіастів та студентів.

З моменту свого випуску Raspberry Pi став важливим інструментом освіти. Він допомагає учням та студентам освоїти основи програмування, електроніки та робототехніки. Розроблені програми та проекти, які використовують Raspberry Pi, стали частиною навчальних програм у багатьох навчальних закладах по всьому світу.

Raspberry Pi доступний у багатьох моделях, що відрізняються за процесорною потужністю, обсягом оперативної пам'яті та іншими параметрами. Звичайні характеристики включають процесори ARM, HDMI-порт, USB-порти і слот для microSD-карти. Це надає користувачам безліч опцій для розробки та застосування. Ознайомитися з характеристиками плати можна у таблиці (таблиця 2.1).

Таблиця 2.1 – Характеристики обраної плати

Параметр	CM4
Процесор	Cortex-A72 (Arm v8) 64-bit SoC @ 1.5GHz
Графічне ядро	VideoCore 6 graphics
Оперативна пам’ять	4 Гігабайтів
Постійна пам’ять	8-16-32 Гігабайтів eMMC
Інтерфейси	2 USB-a, 2 HDMI, 2MIPI DSI, 2 MIPI SCI-2, PCI Express 2.0, Ethernet PHY,

Згодом Raspberry Pi також отримав визнання у науковій спільноті завдяки своїй доступності та універсальності. Вчені почали використовувати Raspberry Pi у різних галузях наукових досліджень, таких як астрономія, біологія, метеорологія та інші. Його компактні розміри та низьке енергоспоживання роблять його ідеальним інструментом для створення датчиків та систем збору даних.

З часом Raspberry Pi пройшов через кілька генерацій, кожна з яких пропонувала покращені характеристики та функціональність. Покращені процесори, більший обсяг оперативної пам'яті та додаткові порти дозволили Raspberry Pi стати ще потужнішим та універсальним інструментом для наукових досліджень.

Історія Raspberry Pi свідчить про те, як невеликий одноплатний комп'ютер,

створений для освіти, міг стати важливим інструментом у наукових дослідженнях. Його доступність та універсальність дозволяють вченим та студентам у всьому світі втілювати свої ідеї в життя та робити внесок у різні галузі науки. Для виконання роботи було обрано Compute module 4 та Carrier board. Із характеристиками CM4 можна ознайомитись у таблиці (таблиця 2.1).

Raspberry Pi Compute Module 4 – це платформа у вигляді несучої плати з інтерфейсами периферії, і обчислювального модуля.

Обчислювальний модуль – мозок платформи, мікрокомп’ютер з необхідними для обчислень і зберігання інформації модулями (рисунок 2.1).

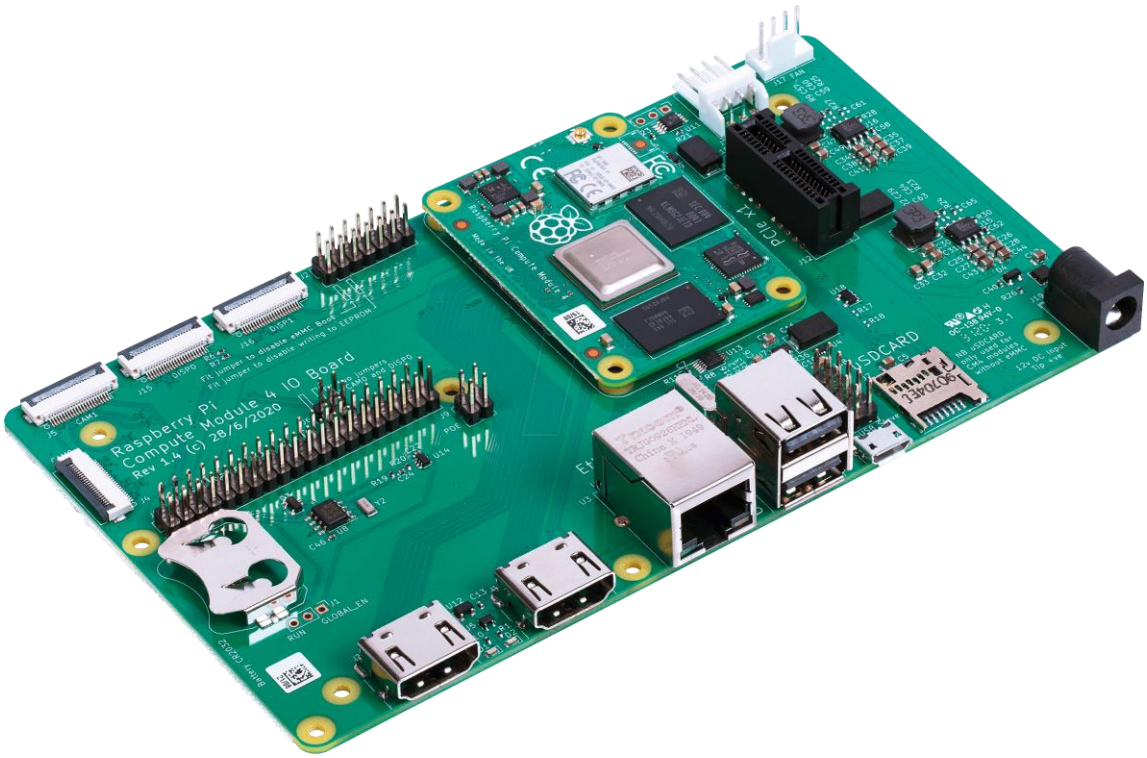


Рисунок 2.1 – Плата IO Carrier Board та Raspberry Pi Compute Module 4

2.1.2 Лідар RPLIDAR A1

(Lidar – Light Detection and Ranging), який використовує лазерне випромінювання для вимірювання відстаней до об'єктів. Він має 360-градусну

омнідирекційну здатність, що дозволяє отримувати дані про навколишнє середовище з усіх боків.

Цей лідер може вимірювати відстань до об'єктів на відстані 12 метрів з точністю до 1 сантиметра. Він працює з частотою сканування 5000 разів на секунду і може збирати дані в режимі реального часу.

RPLIDAR A1 Laser Ranging Sensor є компактним і легким датчиком, що робить його ідеальним для використання в робототехніці, автономних транспортних засобах, системах безпілотного керування та інших подібних проектах. Він також має вбудований спосіб обробки даних, що дозволяє отримувати чисті дані в реальному часі без необхідності використовувати додаткові ресурси для обробки даних (рисунок 2.2).

Рисунок 2.2 – Лідар RPLIDAR A1

2.1.3 SD Картка

Всі пристрої Raspberry Pi включають роз'єм для карт пам'яті SD або microSD, щоб допомогти користувачам вирішити цю проблему.

SD-карти (Secure Digital cards) є типом флеш-пам'яті, який широко використовується для зберігання та передачі даних в електронних пристроях. У робототехніці SD-карти відіграють важливу роль у зберіганні програмного забезпечення, даних сенсорів, карт та іншої інформації.

SD-карти доступні в різних ємностях, починаючи від кількох мегабайт до кількох терабайт. Існують різні класи та швидкості SD-карток, що дозволяє вибирати відповідну картку залежно від вимог щодо швидкості передачі даних.
SD-карти стійкі до впливу вібрацій, температурних змін та механічних впливів, що робить їх надійними для використання у різних умовах робототехніки.

SD-картки використовуються для багатьох задач, зокрема зберігання програмного забезпечення. SD-картки використовуються для зберігання операційних систем, керуючого програмного забезпечення та драйверів, що забезпечує гнучкість та легкість оновлення програмних компонентів.

Зберігання карт і мапінгу. Роботи в робототехніці використовують SD-карти для зберігання карт навколишнього середовища, інформації про шляхи та інші дані, необхідні для навігації.

Запис Даних Сенсорів: Дані сенсорів, такі як зображення з камер, дані з лідарів та акселерометрів, записуються на SD-картки для подальшого аналізу та обробки.

Файли конфігурації: SD-картки можуть містити конфігураційні файли, які визначають параметри роботи робота, такі як швидкість руху, параметри виявлення перешкод та інші налаштування.

SD-карти є важливим елементом у системах робототехніки, забезпечуючи надійне та гнучке зберігання даних, необхідних для функціонування та розвитку роботів. Ефективне використання SD-карт сприяє покращенню продуктивності та функціональності робототехнічних систем.

Початкові моделі Raspberry Pi A і Raspberry Pi B підтримуються SD-карти.

Починаючи з моделі «B+» (2014 р.), потрібна картка пам'яті microSD. Мінімальна необхідна ємність становить 8 ГБ. За замовчуванням Raspberry Pi підтримує до 32 ГБ пам'яті, однак для роботи з цими пристроями можна відформатувати велику ємність. Враховуйте, що для встановлення офіційної ОСRaspbian вам знадобиться картка microSD ємністю не менше 8 ГБ, тоді як Raspbian Lite потрібно мінімум 4 ГБ. Raspbian – це рекомендована ОС від Raspberry Pi Foundation, хоча ви можете запускати багато різних операційних систем, включаючи різні дистрибутиви Linux.

Для проекту була обрана картка Kingston High Endurance microSD (рис. 2.3).

Рисунок 2.3 – Картка Kingston High-Endurance microSD

Так як зазвичай цю картку використовують для систем безпеки та камер

спостереження, картка підійде і для систем реального часу. Карти пам'яті High-Endurance microSD компанії Kingston випускаються із ємностями від 32 ГБ до 128 ГБ. Для пректу досить молодшої моделі з 32 гб пам’яті.

2.2 Вибір середовища та методів розробки

2.2.1 Операційна система Linux Ubuntu (18.04.6 – Bionic Beavel)

Ubuntu – це повністю безкоштовна та відкрита, проста в освоєнні операційна система (ОС), розроблена британською компанією Canonical Ltd. на базі дистрибутива Linux та Unix-подібної ОС Debian. Випущена у трьох редакціях: Desktop для настільних ПК та ноутбуків, Server для серверів та Core для інтернету речей. Дистрибутив Ubuntu було обрано через сумісність з фрейсворком ROS, а також через його популярність, що спрощує подолання програмних несправностей. У свою чергу версія 18.04.6 з найменуванням Bionic Beavel обрана за схожим принципом. Більш нові версії ОС підтримують більш нові версії ROS, на які у свою чергу не дуже багато навчальних матеріалів. (Офіційну документацію не беремо до уваги). Саме цю операційну систему було обрано через сумісність, тобто фреймворк на інших ОС може працювати не так стабільно. В якості платформи для ОС використовуватиметься Oracle VM Virtual Box. Таке рішеня було прийняте через необхідність тестування різних версій ОС на початковому єтапі. Заливка операційної системи на Rasberry значно повільніша ніж на десктопі. До того ж, програма дозволяє використовувати декілька віртуальних машин з різними версіями ОС без необхідності їх перезавантаження (рис. 2.4).

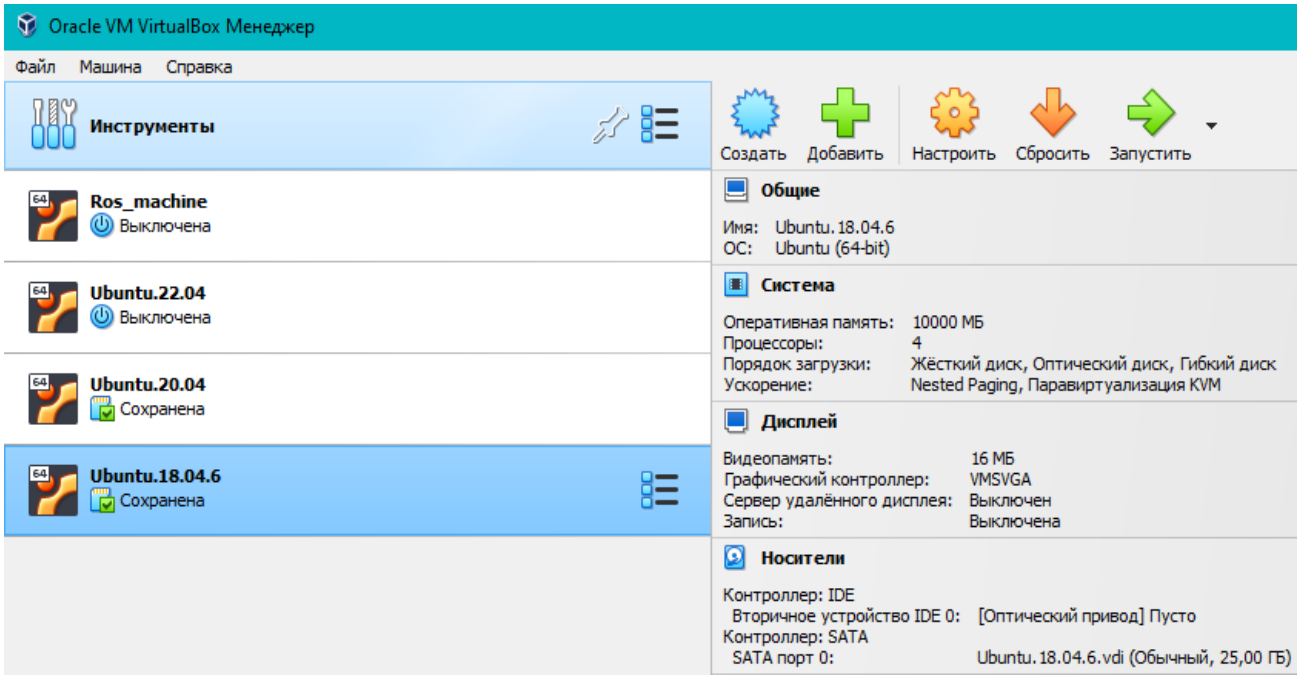


Рисунок 2.4 – Список використаних версій ОС

2.2.2 ROS фреймворк для роботизованих систем

Robot Operating System (ROS) (рис. 2.4). є гнучкою і потужною платформою для розробки програмного забезпечення в галузі робототехніки. У цій статті розглянемо історію виникнення та розвитку ROS, починаючи з його зародження та закінчуючи перспективами використання у сучасній робототехніці.

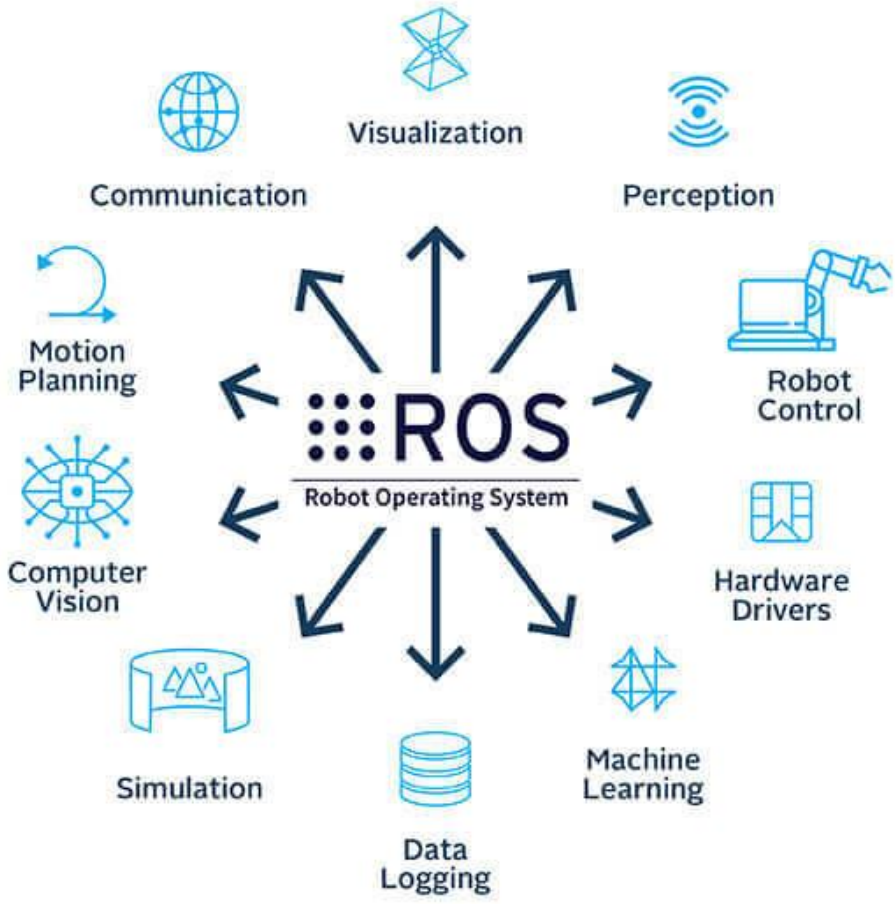


Рисунок 2.5 – Концепція ROS

Історія ROS почалася в Stanford AI Lab, коли професор Morgan Quigley та його студенти Ендрю Нг та Віллонг Цао усвідомили необхідність уніфікованої та відкритої платформи для програмування роботів. У 2007-2008 роках вони розпочали роботу над проектом, який згодом перетворився на ROS. ROS був офіційно анонсований у 2009 році, і його вихідний код було викладено у відкритий доступ. Ця дія стала ключовим моментом, оскільки вона відкрила шлях для активної участі спільноти у розробці та покращенні системи. Перші версії ROS надавали базові інструменти та бібліотеки для управління роботами, що дозволило дослідникам та розробникам зосередитися на більш високорівневих завданнях.

З 2011 року ROS зазнав істотних змін та доповнень, включаючи нові бібліотеки та інструменти, такі як RViz для візуалізації та Gazebo для симуляції. У

цей час також розпочалася стандартизація ROS, що сприяло сумісності та переносимості коду між різними роботами та системами. Це створило умови подальшого поширення ROS в промисловості робототехніки. У період з 2015 по 2018 роки ROS почав активно впроваджуватись у промислові рішення. Багато компаній та дослідні групи почали використовувати ROS для розробки інноваційних рішень у галузі автономних транспортних засобів, медичної робототехніки та виробничої автоматизації. У цей час співтовариство ROS також активно обмінювалося знаннями та досвідом, що сприяло його подальшому розвитку.

Сьогодні ROS продовжує розвиватися і адаптуватися під вимоги робототехніки, що змінюються. Нові версії системи впроваджують передові технології, такі як машинне навчання та комп'ютерний зір. ROS 2, випущений у 2017 році, надає більш сучасну архітектуру, підтримує реальний час та забезпечує підвищену стабільність. В останні роки ROS став невід'ємною частиною розробки автономних систем, включаючи автономні автомобілі та безпілотні дрони. Інтеграція ROS дозволяє розробникам використовувати широкий спектр готових бібліотек та інструментів для сприйняття навколишнього середовища, прийняття рішень та управління рухом, що є важливим для успішної реалізації автономних рішень. ROS 2, наступне покоління системи, було введено для вирішення деяких обмежень попередньої версії, включаючи обмеження реального часу та підтримку різних типів обладнання. Стандартизація в рамках ROS 2 сприяє легкості інтеграції та обміну інформацією між різними системами та платформами, що робить його привабливішим для широкого кола розробників та інженерів.

Службові роботи, призначені для виконання різних завдань у сферах обслуговування, логістики та виробництва, дедалі успішніше впроваджують ROS у свої системи. Це забезпечує ефективне сприйняття навколишнього середовища, координацію дій та взаємодію з людьми. ROS стає ключовим компонентом для розробки складних та інтелектуальних службових роботів, що відкриває нові перспективи у сфері автоматизації та обслуговування. Robot Operating System продовжує еволюціонувати та залишатися центральним елементом у розробці

робототехнічних додатків. Від ініціативи Stanford AI Lab до широкого використання в промисловості та академії, ROS продемонстрував свою здатність адаптуватися до мінливих вимог сфери робототехніки. Перспективи його застосування в автономних системах, в індустрії службових роботів та інших галузях підтверджують його важливу роль у формуванні майбутнього робототехнічних технологій та комп'ютерного зору.

Комп'ютерний зір (Computer Vision, CV) – це область штучного інтелекту, пов'язана з аналізом зображень та відео. Вона включає набір методів, які наділяють комп'ютер здатністю «бачити» і витягувати інформацію з побаченого. Системи складаються з фото- або відеокамери та спеціалізованого програмного забезпечення, яке ідентифікує та класифікує об'єкти. Вони здатні аналізувати образи (фотографії, картинки, відео, штрих-коди), а також особи та емоції.

ROS підтримує паралельні обчислення, має гарну інтеграцію з популярними бібліотеками C++, такими як OpenCV, Qt, Point Cloud Library та ін., і вона може працювати на одноплатних комп'ютерах, таких як Raspberry Pi або BeagleBone Black, а також з мікроконтролерними платформами, наприклад, Arduino. Для роботи з робототехнічними засобами використовуються ноди, різновид багатофайлової системи.

Поняття ноди відноситься до найменшої "робочої" одиниці використовуваної в ROS. Можна провести аналогію з однією програмою, що виконується. ROS рекомендує створити одну ноду для кожного завдання, що дозволить легко використовувати її в інших проектах. При запуску нода реєструє інформацію про себе на майстрі (назва ноди, типи повідомлень, що обробляються). Зареєстрована нода може взаємодіяти з іншими нодами (отримувати та надсилати запити). Важливо, що обмін повідомленнями між нодами працює без участі майстра (з'єднання між нодами відбувається на пряму). Майстер забезпечує лише єдиний простір імен для вирішення питання, куди підключитися до конкретної ноди. Адреса запуску ноди береться зі змінної оточення «ROS_HOSTNAME», яка повинна бути визначена до запуску. Порт встановлюється довільне унікальне значення.

Для виконання роботи було обрано версію ROS Melodic Morenia. Не дивлячись на те що вже досить великий проміжок часу існує версія ROS 2 Iron Irwini, навчальних матеріалів окрім як англійською майже немає. Окрім цього у процесі вибору та тестування фреймворку, було виявлено що старіші версію ROS є більш популярними у звичайних користувачів. Тобто навіть у англомовних форумах тем присвячених ROS 2 – значно менше.

Як попередньо було описано, в Robot Operating System інтегрована бібліотека «Point Cloud Library», що необхідна для роботи з хмарою точок.

2.2.3 Хмара точок та її використання

Хмара точок (Point Cloud) є тривимірним просторовим уявленням навколишнього об'єкта, представленим набором тривимірних точок у просторі (рис. 2.6). У робототехніці це стало важливим інструментом для сприйняття навколишнього середовища роботом. У цій статті ми розглянемо історію виникнення та еволюції використання хмари точок у робототехніці, починаючи з перших кроків у цьому напрямі.

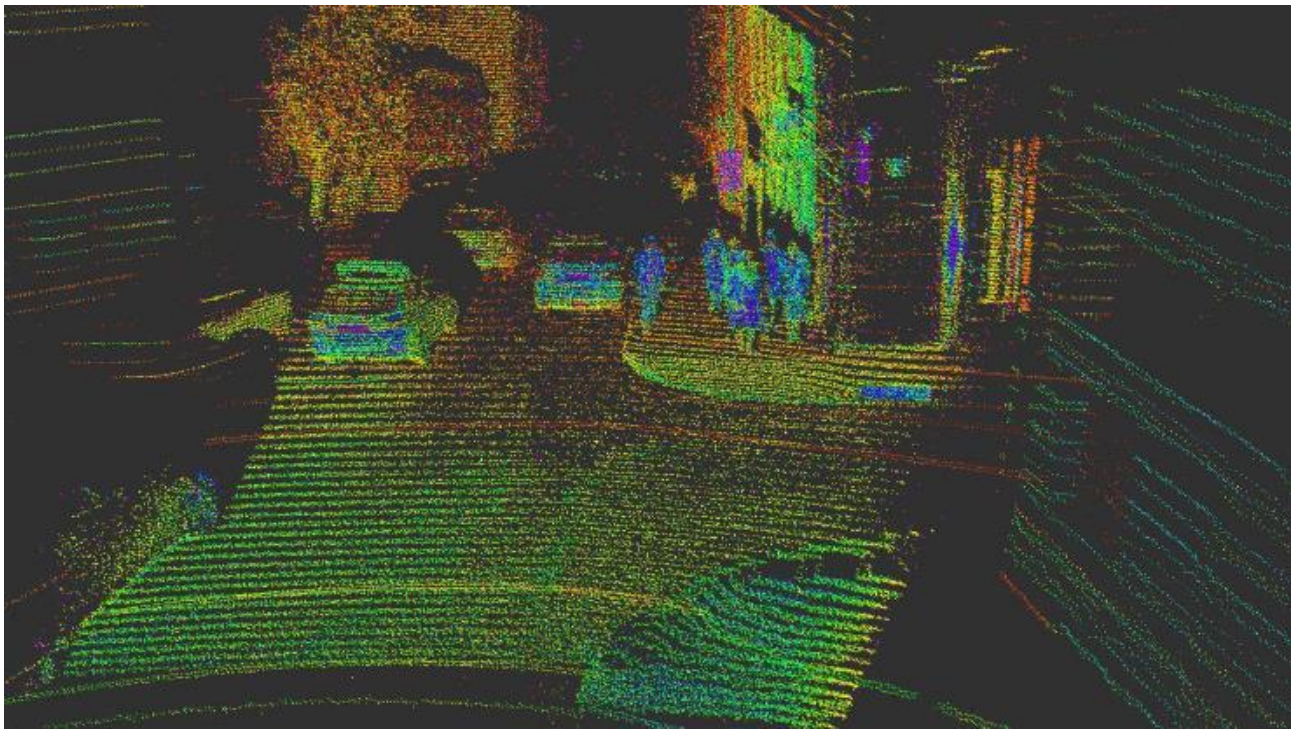


Рисунок 2.6 – Хмара точок отримана з лідара безпілотного автомобіля

Ідея використання тривимірних точок для сприйняття навколишнього середовища роботом почала формуватися у 1980-1990-х роках. Дослідники стали усвідомлювати, що для точного сприйняття та навігації робота необхідно мати більш складні та точні дані про навколишнє середовище, ніж це надавали традиційні датчики.

У цей час почали активно застосовуватися методи лазерного сканування та стереогляду для створення хмар точок. Лазерні сканери та стереокамери дозволяли роботам отримувати більш детальне уявлення навколишнього середовища, створюючи точкові хмари, що відображали структуру об'єктів у просторі.

З розвитком технологій сенсорів, таких як RGB-D камери та LiDAR, а також покращенням алгоритмів обробки даних, роботи стали здатними генерувати більш точні та багаті хмари точок. Це уможливило більш точне та надійне сприйняття навколишнього середовища, що стало критично важливим у різних галузях робототехніки, таких як автономні автомобілі, дрони та службові роботи.

З появою автономних транспортних засобів та роботів, хмари точок стали ключовим елементом для забезпечення надійної та безпечної навігації у різноманітних середовищах. Автономні автомобілі, наприклад, використовують хмари для розпізнавання перешкод, дорожніх знаків та інших елементів дорожньої інфраструктури. Сьогодні хмари точок залишаються активно досліджуваною областю у робототехніці. З розвитком технологій та підвищенням обчислювальної потужності, очікується, що хмари точок будуть використовуватися в ще ширшому спектрі додатків, включаючи покращене розуміння навколишнього середовища, маніпуляції об'єктами та взаємодію з ними.

Хмари точок у робототехніці стали ключовим елементом для забезпечення роботам здатності сприймати та взаємодіяти з навколишнім середовищем. Їх історія, починаючи з перших ідей у 1980-1990-х роках і закінчуючи сучасними технологіями, свідчить про безперервний розвиток та важливість цього підходу для досягнення автономності та ефективності роботів.

Хмара точок – це необроблені дані що отримуються завдяки триангуляції з великої кількості знімків, 3Д сканів лідара, чи з інформації пікселів стерео зйомки (карти глибини). Це дані, на основі яких проводиться оцінка вимірювань та створюється модель тривимірного об’єкту або простору. Технічно хмара точок є базою даних, в якій містяться точки, розташовані в тривимірній системі координат. Однак, з точки зору типового робочого процесу, єдино важливим є той факт, що хмара точок є точнішим цифровим записом об'єкта або простору, збережену у вигляді дуже великої кількості точок, що покривають поверхні об'єкта. Нажаль з одними лише даними скану поверхні отримати повноцінну карту – майже неможливо. На це впливають декілька факторів:

– шум. Усі сенсори мають певні похибки, які перетворюються на різні артефакти. Наприклад, ізольовані точки, які висять у повітрі там, де їх бути не повинно (рис. 2.7);

– наявність інших об'єктів. При скануванні невеликої площі з метою подальшої побудови 3Д моделі, наприклад пам’ятнику, можуть заважати зайві об’єкти. (При скануванні великих площ – не так актуально).

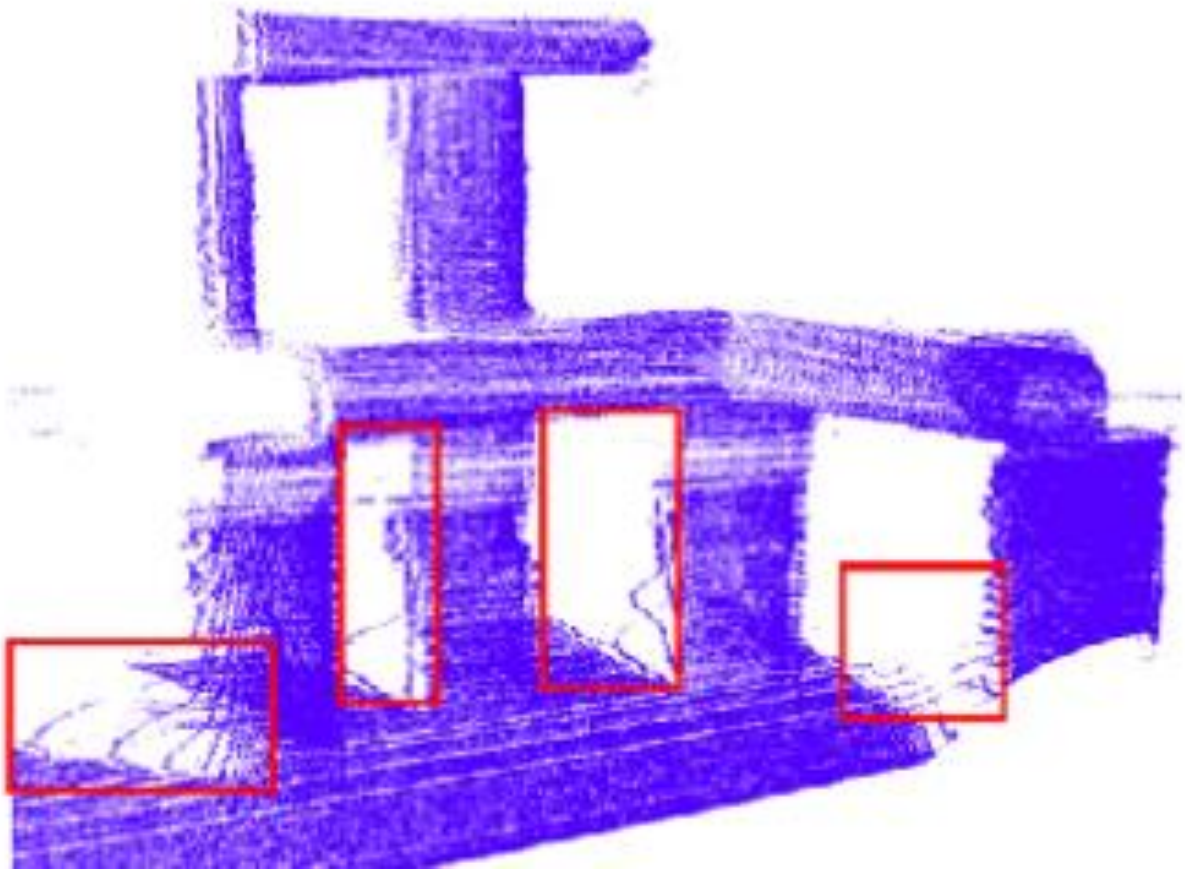


Рисунок 2.7 – Шум у хмарі точок

Для того, щоб виділити досліджуваний об'єкт і вирішити ряд озвучених вище проблем, попередньо необхідно обробити хмару точок у три етапи: видалення шумових точок зі скана, видалення опорних поверхонь, сегментація об'єктів, що залишилися.

2.2.4 CAD система Fusion 360

Autodesk Fusion 360 - інноваційна платформа для комп'ютерної допомоги в проектуванні (CAD), проектування віртуального середовища та віртуального виробництва. У цій статті ми розглянемо історію розвитку Fusion 360, починаючи з його появи, і розглянемо різноманітні сфери застосування цього інструменту в сучасній науці та інженерії.

Fusion 360 був вперше представлений Autodesk у 2012 році як революційний інструмент, що поєднує у собі можливості 3D-моделювання, засіб проектування, аналізу та віртуального виробництва. Його технологія була спрямована на зняття обмежень, що стояли перед інженерами та дизайнерами, надаючи інтегроване рішення для повного циклу розробки продукту. У наступні роки Autodesk активно розвивала Fusion 360, запроваджуючи технології хмарних обчислень. Це дозволило користувачам працювати з проектами у реальному часі, обмінюватися даними та спільно працювати над проектами у віртуальному середовищі. Інтеграція з хмарними технологіями зробила Fusion 360 потужним інструментом для колективної розробки та управління проектами. Додаткові можливості Fusion 360 були додані у вигляді інструментів для віртуального тестування та аналізу. Користувачі тепер можуть проводити аналіз міцності, теплопередачі та інших параметрів ще на етапі проектування, що суттєво знижує кількість ітерацій у процесі розробки та покращує якість кінцевого продукту.

Fusion 360 не обмежується лише проектуванням. В останні роки інструмент інтегрував можливості віртуального виробництва. Це дозволяє користувачам створювати керуючі програми для верстатів із числовим програмним керуванням (CNC) і навіть моделювати виробничі процеси у 3D-просторі.

З розвитком Інтернету речей (IoT) Fusion 360 знайшов своє місце у цій галузі. Інженери використовують Fusion 360 для проектування розумних пристроїв, датчиків та інших IoT продуктів. Інтеграція з хмарними обчисленнями забезпечує можливість керування та моніторингу цих пристроїв у режимі реального часу.

Autodesk Fusion 360, спочатку представлений як інструмент для 3D-моделювання та проектування, (рис. 2.8) зазнав помітних змін, ставши інтегрованою платформою для проектування та інженерії. У цій статті розглянемо історію розвитку Fusion 360, звертаючи увагу на його роль у сфері робототехніки та різноманітні галузі його застосування. З моменту появи Fusion 360 став привабливим інструментом для інженерів і дизайнерів, що працюють в області робототехніки. У період з 2015 по 2017 роки Autodesk активно інтегрувала функціональності, специфічні для робототехніки, надаючи можливості

моделювання та проектування різних механізмів та мобільних платформ. З розвитком робототехніки стало зрозумілим, що віртуальне тестування відіграє ключову роль у розробці. Autodesk впровадила у Fusion 360 інструменти для віртуального тестування робототехнічних механізмів. Це включало аналіз механічної міцності, динаміки руху та взаємодії з навколишнім середовищем. Зі зростанням інтересу до безпілотних систем та дронів, Fusion 360 знайшов свою нішу у розробці цих пристроїв. Інженери використовують платформу для моделювання та тестування різних типів дронів, а також для інтеграції та оптимізації їх систем керування та компонентів.

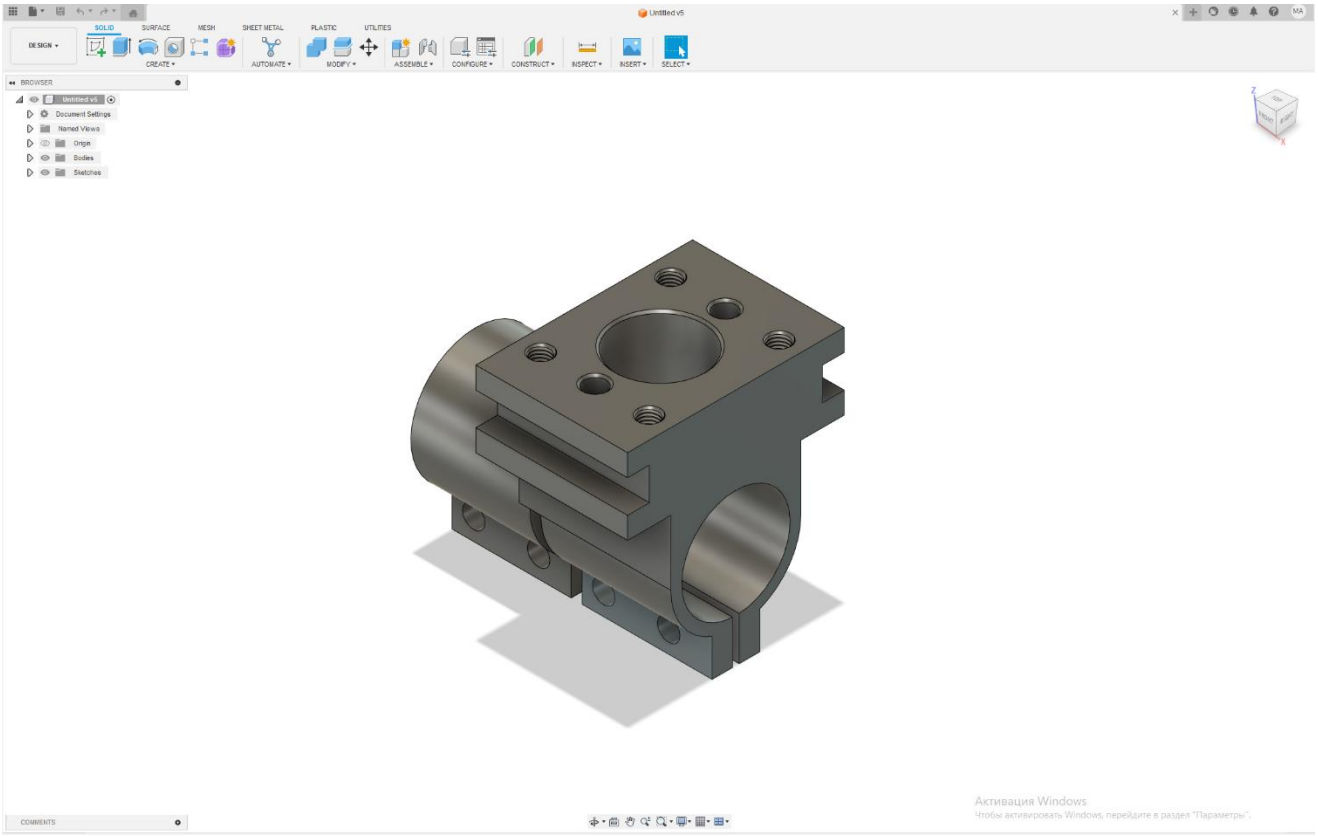


Рисунок 2.8 – Середа модел.вання Fusion360

Fusion 360 успішно інтегрувався у сферу робототехніки, надаючи інженерам та дослідникам потужний інструмент для проектування, тестування та моделювання. Його широкі можливості, інтеграція з хмарними технологіями та

постійний розвиток роблять Fusion 360 ключовим компонентом у розробці інноваційних робототехнічних систем.

Для того щоб умовно позначити мобільного робота в симуляції Gazebo 3D, необхідна модель цього робота у розширенні .xacro (.urdf). URDF – (Universal Robotic Description Format), використовується для опису роботів у ROS. Є по суті, діалектом XML. Ця мова представляє робота як сукупність ланок (link), зчленувань (joint), сенсорів та низки допоміжних параметрів. Файли такого розширення можна отримати завдяки перетворенню через окремий плагін моделей формату .f3d (Fusion360) або .sldprt (SolidWorks). Для роботи був обраний Fusion360 через знайомість інтерфейсу (рис. 2.9).

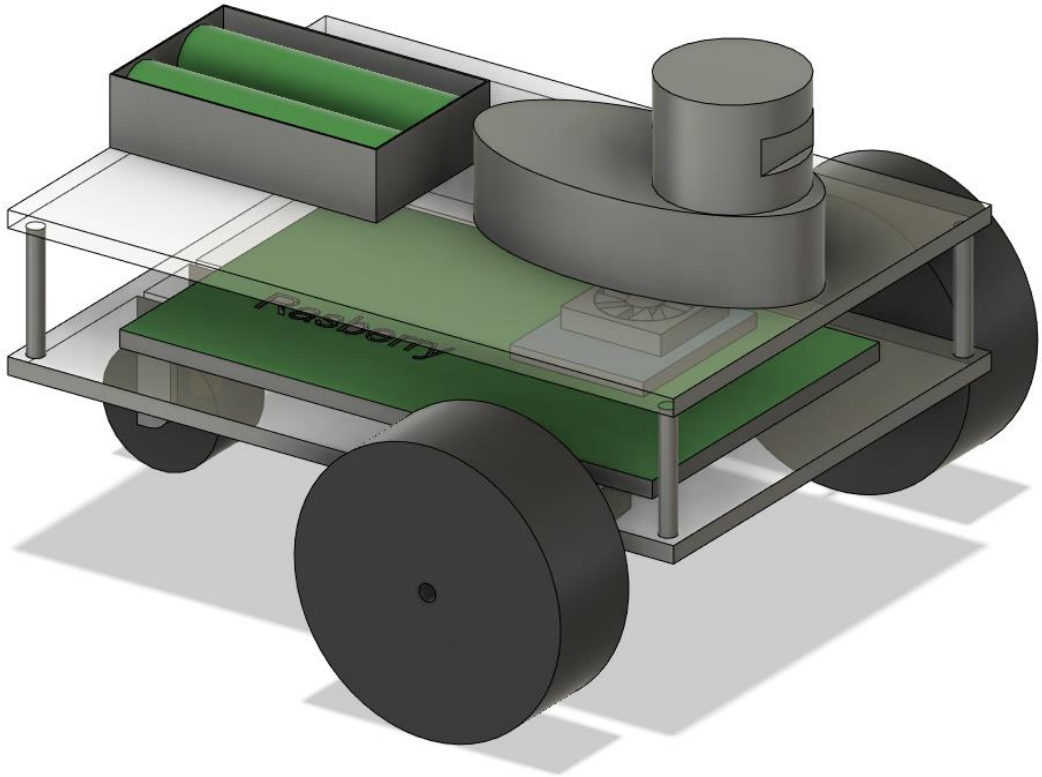


Рисунок 2.9 – Умовна модель робота виконана у Fusion360

2.3 Інструментарії симуляції та візуалізації Gazebo, Rviz, Rqt

2.3.1 Rviz

Rviz, або ROS Visualization, є потужним інструментом у системі Robot Operating System (ROS) для візуалізації даних і стану роботів у реальному часі. У цій статті розглянемо історію появи Rviz, його розвиток у рамках ROS та області його застосування у робототехнічних дослідженнях та розробках.

Rviz був уперше представлений у 2008 році як інструмент для візуалізації даних у ROS. Це сталося невдовзі після випуску ROS, і Rviz був розроблений з метою надати розробникам та дослідникам зручний спосіб візуалізації інформації про роботи та їхнє оточення. На початку свого шляху Rviz надавав базові засоби для візуалізації координатних систем, хмар точок та інших даних. У наступні роки, з 2010 по 2012, Rviz став активно розвиватися відповідно до потреб розробників у ROS. Нові можливості включали підтримку різних типів датчиків, візуалізацію траєкторій, відображення хмар точок з використанням кольорів для кодування даних, а також взаємодія з інтерфейсами управління.

У період з 2013 по 2015 роки Rviz став ще більш гнучким завдяки інтеграції з різними бібліотеками та інструментами. Це включало візуалізацію результатів роботи алгоритмів обробки зображень, відображення 3D-моделей роботів та їх оточення, а також підтримку віртуальної реальності. З 2016 року Rviz продовжив розвиватися у напрямку більш ефективної та зручної візуалізації складних даних. Це включало поліпшення у відображенні багатовимірних даних, інтеграцію з сенсорами глибини, а також більш просунуті можливості візуалізації для поліпшення розуміння динаміки руху роботів.

В останні роки Rviz став невід'ємною частиною дослідницьких та розробних процесів у робототехніці. Розробники використовують його для візуалізації та налагодження своїх алгоритмів, а також для навчання та освіти у сфері робототехніки.

На сьогоднішній день Rviz охоплює широкий спектр областей застосування у ROS. Це включає в себе:

– Навігація та планування: Rviz використовується для візуалізації світів та траєкторій роботів, що полегшує налагодження та покращення алгоритмів навігації.

– Обробка зображень та комп'ютерний зір: Розробники можуть візуалізувати результати обробки зображень, що полегшує розуміння алгоритмів комп'ютерного зору.

– Тестування та налагодження: Rviz надає можливість у реальному часі відстежувати стан робота, що спрощує тестування та налагодження програмного забезпечення.

– Навчання та освіта: У навчальних програмах та лабораторних заняттях Rviz допомагає студентам краще розуміти принципи робототехніки, надаючи візуальне представлення даних та дій робота.

Rviz продовжує відігравати важливу роль в екосистемі ROS, надаючи розробникам та дослідникам засіб для візуалізації та аналізу даних у реальному часі. З його постійним розвитком та широким спектром застосування, Rviz залишається важливим компонентом у дослідницьких та розробних процесах, сприяючи прогресу в галузі робототехніки.

Інструмент Rviz дозволяє в реальному часі візуалізувати на 3D-сцені всі компоненти робототехнічної системи – системи координат, частини, що рухаються, показання датчиків, зображення з камер (рисунок 2.10).

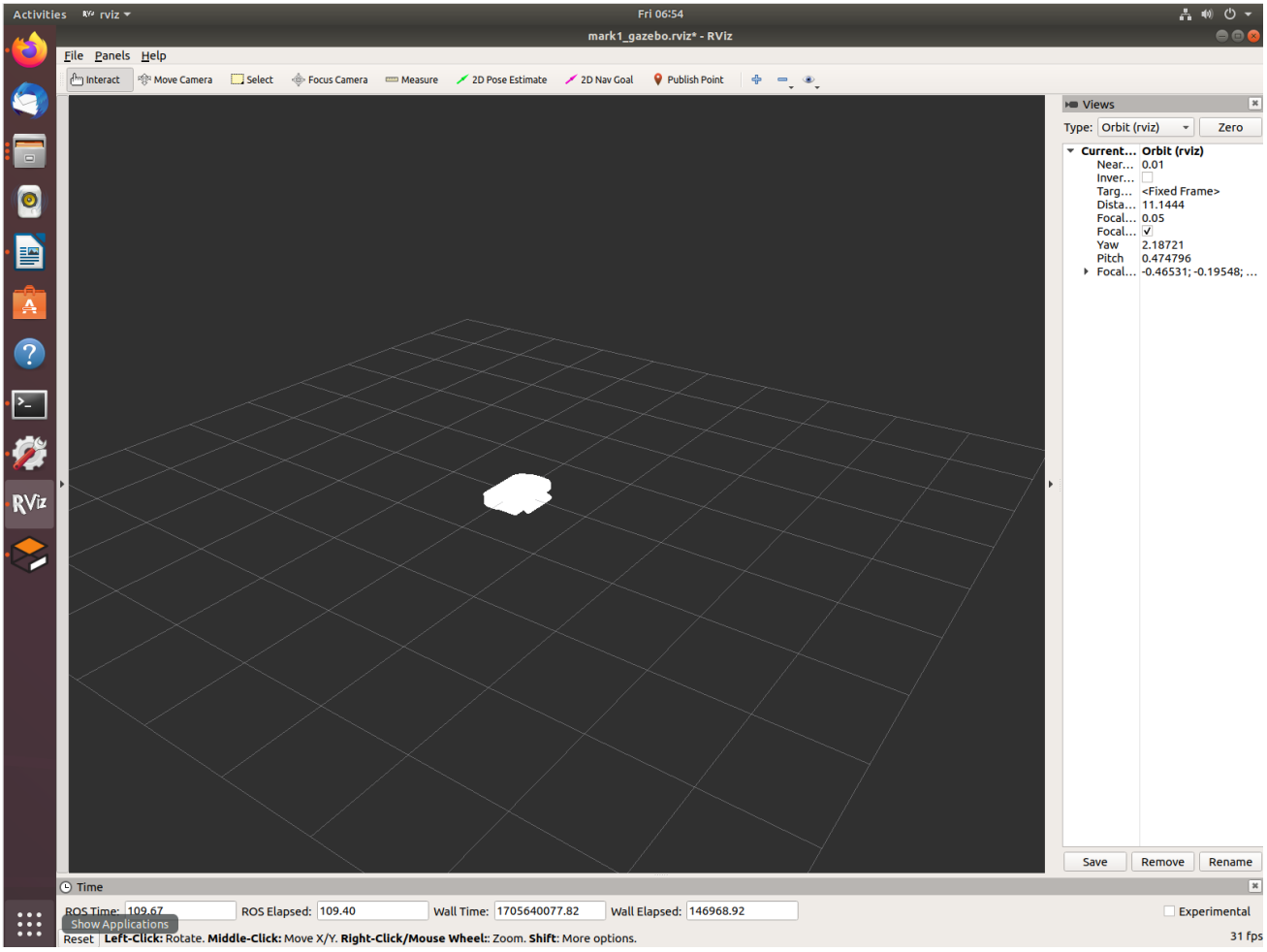


Рисунок 2.10 – Вікно візуалізації Rviz

Основною функцією що виконує цей інструмент у роботі є відображення побудованної моделі віртуального простору.

2.3.2 Gazebo

Gazebo 3D, що розробляється некомерційною організацією OSRF (Open Source Robotics Foundation), має низку переваг у порівнянні з іншими робототехнічними симуляторами. По-перше, він безкоштовний та має відкритий код. По-друге, він дуже популярний серед світової робототехнічної спільноти та є офіційним симулятором змагань DARPA (рисунок 2.11).

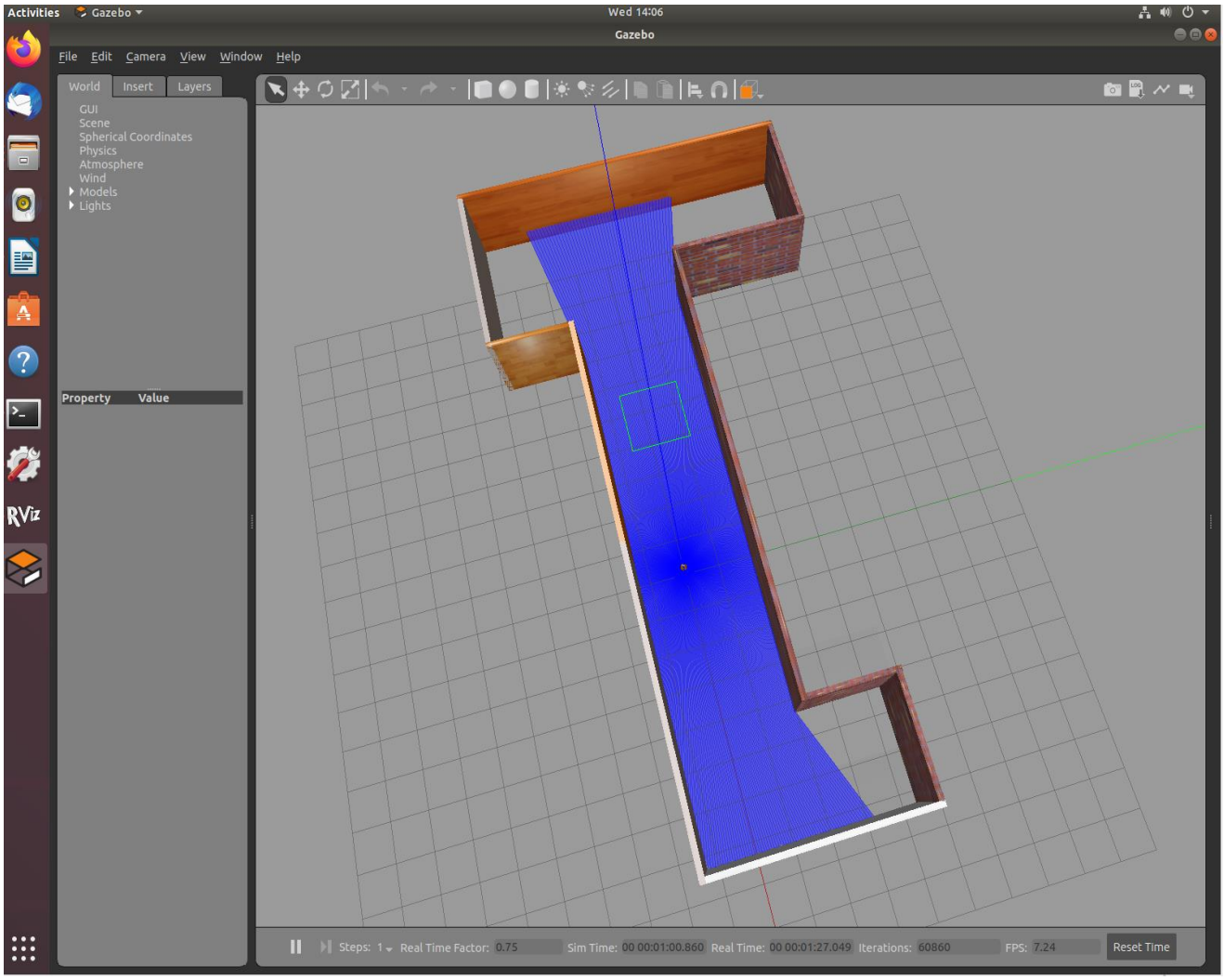


Рисунок 2.11 – Симуляція в Gazebo 3D

Gazebo 3D є потужним симуляційним середовищем у робототехніці, розробленим для інтеграції з Robot Operating System (ROS). У цій статті ми розглянемо історію появи Gazebo, його роль у розробці та тестуванні робототехнічних програм, а також перспективи використання цього середовища в майбутньому.

Історія Gazebo почалася в 2002 році, коли автор Andrew Howard почав розробляти середовище для симуляції багатьох робототехнічних аспектів. У 2004 році вийшла перша версія Gazebo, що надала дослідникам та розробникам інструмент для перевірки та налагодження своїх робототехнічних алгоритмів у симульованому середовищі. Gazebo став широко використовуватися в робототехніці, і в 2011 році було ухвалено рішення щодо його інтеграції з ROS. Це

рішення стало ключовим моментом, який забезпечив єдиний засіб для симуляції та управління роботами в контексті операційної системи для роботів. Інтеграція з ROS дозволила розробникам ефективніше тестувати та розробляти свої робототехнічні рішення.

З 2014 року Gazebo продовжив розвиватися, надаючи все більш складні засоби моделювання та симуляції. Можливості фізичного моделювання, освітлення та датчиків були вдосконалені, роблячи середовище ще ближчими до реальної фізичної взаємодії роботів з навколишнім середовищем. Ці покращення зробили Gazebo ще більш привабливим для широкого кола дослідників та інженерів. Gazebo став незамінним інструментом у галузі досліджень та освіти у робототехніці. Студенти та вчені можуть використовувати це середовище для вивчення алгоритмів, тестування стратегій управління та розробки нових робототехнічних програм. Це також сприяло поширенню Gazebo в академічному середовищі та його використанню в курсах робототехніки. З появою автономних систем, таких як безпілотні автомобілі та дрони, Gazebo став незамінним інструментом для розробки та тестування алгоритмів та систем керування. Віртуальне моделювання в Gazebo дозволяє розробникам створювати складні сценарії та тестувати реакцію своїх систем у різних умовах, що суттєво знижує ризики та витрати у процесі розробки.

Перспективи Gazebo 3D у робототехніці дуже обнадійливі. З розвитком технологій симуляції, покращенням інтерфейсів та розширенням можливостей фізичного моделювання, Gazebo залишається ключовим інструментом для інновацій у робототехніці. Перспективи включають більш реалістичні середовища, підтримку для нових типів датчиків та інтеграцію з технологіями штучного інтелекту для створення ще розумніших і адаптивніших роботів. В останні роки Gazebo почав активно розширювати свій функціонал для більш точного та гнучкого моделювання роботів. Додавання підтримки більш складних типів датчиків, таких як камери з глибинним баченням, радари та лідери, дозволяє дослідникам детальніше вивчати поведінку своїх роботів у різних сценаріях.

З розвитком автономних автомобілів та безпілотних дронів, Gazebo став

незамінним інструментом для розробників. Віртуальне моделювання Gazebo дозволяє створювати складні сценарії, включаючи взаємодію з іншими об'єктами і ситуації на дорозі або в повітрі. Це сприяє більш надійному тестуванню та налагодженню систем керування перед реальними експериментами. Зі зростанням інтересу до машинного навчання Gazebo також став платформою для інтеграції з цими технологіями. Дослідники використовують Gazebo для навчання нейронних мереж за умов віртуального середовища, що полегшує перетворення отриманих моделей для використання на фізичних роботах. При погляді в майбутнє видно, що Gazebo 3D продовжить свій розвиток, стаючи ще гнучкішим і потужнішим симуляційним середовищем. Перспективи включають підтримку більш складних фізичних моделей, поліпшену візуалізацію та можливості взаємодії з іншими симуляторами. Важливим напрямком розвитку також є інтеграція з технологіями віртуальної реальності (VR), що дозволить дослідникам ще глибше занурюватися у віртуальні сценарії та проводити більш точні експерименти.

Gazebo 3D продовжує відігравати ключову роль у сучасній робототехніці, надаючи розробникам та дослідникам потужний інструмент для симуляції та тестування робототехнічних рішень. Від початку своєї історії до сьогодні Gazebo продемонстрував свою важливість в освіті, дослідженнях і промисловості, а його перспективи свідчать про те, що він залишиться ключовим компонентом у формуванні майбутнього автономних систем та робототехнологій.

Симулятор Gazebo на відміну від Rviz дозволяє відлагодити систему ще до моменту фізичної побудови мобільного робота, тобто знімати показники різноманітних датчиків та сканерів. До того ж, у такій симуляції повністю прораховуються фізичні параметри, що є вкрай важливим для мобільних платформ з роботичними кінцівками.

2.3.3 Rqt

RQT (ROS Qt-based GUI Toolkit) виділяється серед інструментів ROS як інтегрована бібліотека, що призначена для зручної розробки графічних

інтерфейсів користувача (GUI). У цій статті ми розглянемо історію появи RQT, його еволюцію та різноманітні галузі застосування у робототехнічних дослідженнях та розробках.

RQT був уперше представлений у співтоваристві ROS у 2012 році. У цей час ROS ставала все більш популярною серед розробників робототехніки, і потреба в інструментах візуалізації та управління зростала. RQT створювався як набір інструментів, заснованих на бібліотеці Qt для створення графічних інтерфейсів, інтегрованих безпосередньо в робоче середовище ROS. Протягом наступних кількох років RQT продовжував розвиватися, додаючи нові інструменти та можливості. У цей період були представлені різні віджети та розширення для RQT, дозволяючи розробникам створювати більш складні та функціональні інтерфейси. Це включало можливості візуалізації даних, управління параметрами роботів і інструменти для налагодження. RQT активно інтегрувався з іншими інструментами ROS, такими як RViz (ROS Visualization), Gazebo і навіть інструментами для аналізу даних. Це покращило взаємодію між різними компонентами системи ROS та надало розробникам більш досконалі інструменти для аналізу та контролю своїх робототехнічних систем. RQT став важливим інструментом в освітніх програмах робототехніки. З його допомогою студенти можуть вивчати та розуміти принципи управління та візуалізації роботів у реальному часі. Це полегшує процес навчання та допомагає студентам краще розуміти внутрішні механізми робототехнічних систем.

Rqt – це програмна основа ROS, яка реалізує різноманітні інструменти GUI у формі плагінів. У rqt можна запускати всі існуючі інструменти графічного інтерфейсу як закріплювані вікна. Інструменти все ще можуть працювати традиційним автономним методом, але rqt полегшує керування всіма різноманітними вікнами на екрані одночасно. У роботі використовується команда rqt_gragh, що виводить інформацію про поточні ноди та топіки (рисунок 2.12).

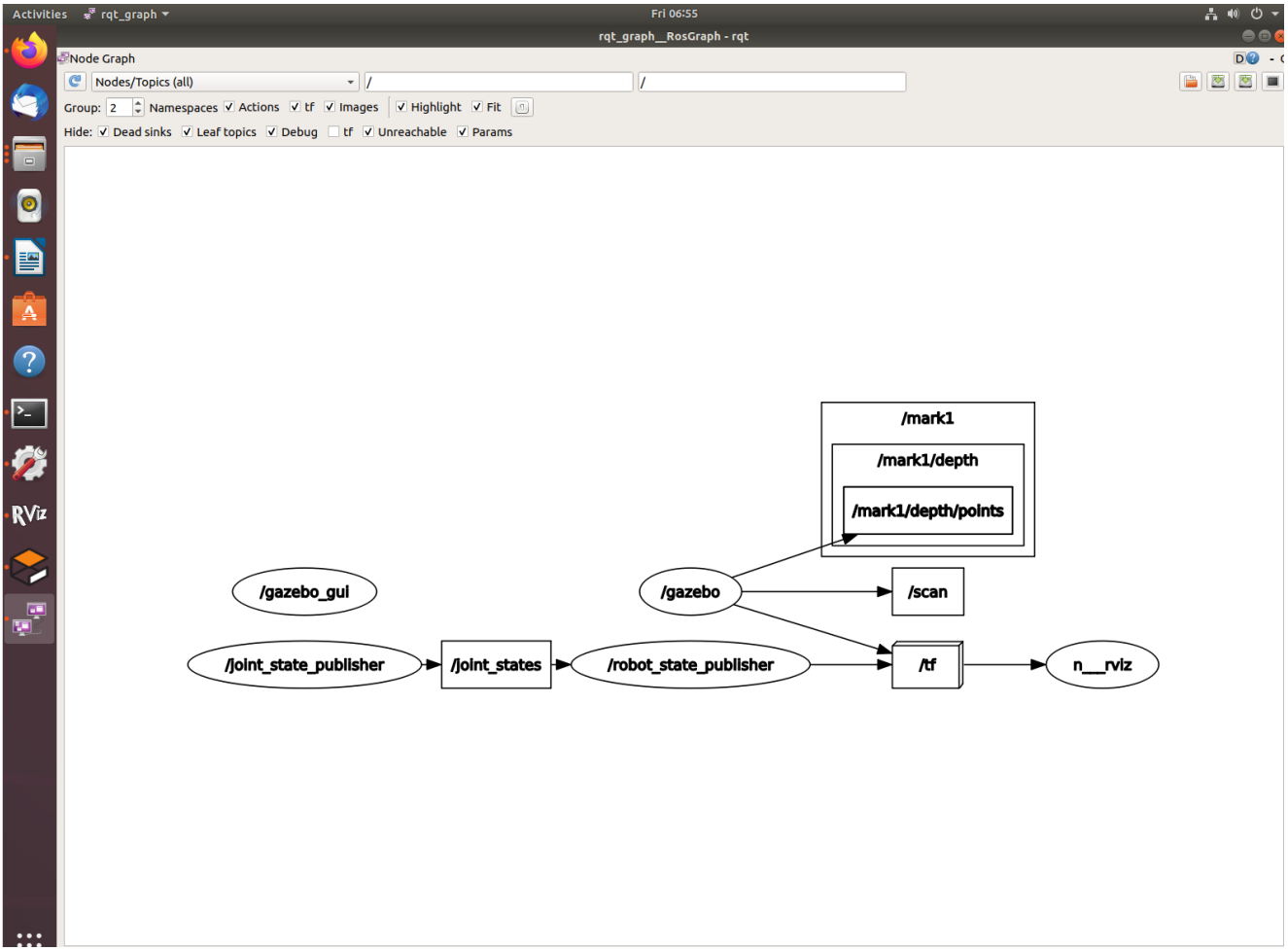


Рисунок 2.12 – Сполучення між вузлами та темами в проекті

Таким чином rqt плагін це потужний інструмент відладки та багтрекінгу проекту. Завдяки йому можна побачити чи не побачити роботу потрібних у проекті вузлів.

2.4 Висновки за розділом 2

У розділі 2 проаналізовано інструменти та програмні засоби, необхідні для подальшої розробки. Проведено вибір апаратних елемнтів, що будуть симульовані далі. Серед існуючих та сумісних операційних систем була обрана оптимальна версія згідно з більшим обсягом даних до неї. Знайдено та надано загальну інформацію щодо фреймворку та його сфери використання, інструментарію, тощо.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕСПЕЧЕННЯ ДЛЯ ПОБУДОВИ 3D МОДЕЛІ НАВКОЛИШНЬГО СЕРЕДОВИЩА МОБІЛЬНОГО РОБОТА

3.1 Основні відомості щодо структури ROS

Робототехнічна операційна система (ROS) надає гнучку та масштабовану інфраструктуру для розробки програмного забезпечення для роботів. Однією з ключових концепцій у ROS є "вузли" (nodes) і "топіки" (topics), які відіграють важливу роль в організації та обміні даними в системі. Давайте розглянемо кожен із цих елементів докладніше.

Вузол є незалежним процесом, що виконує певне завдання в рамках робота. Кожен вузол може бути написаний різними мовами програмування та виконуватись на окремому обчислювальному пристрої. Вузли виконують різні завдання, такі як керування моторами, обробка даних сенсорів, візуалізація інформації та інші завдання, необхідні функціонування робота. Вузли взаємодіють між собою через топіки, сервіси та параметри, забезпечуючи гнучкість та модульність у розробці.

Топік є механізмом для передачі даних між вузлами в ROS. Це асинхронний канал, яким вузли можуть публікувати інформацію та підписуватися її у. Топики використовуються для обміну повідомленнями, такими як дані сенсорів, зображення з камер, команди руху та інша інформація, яка потрібна для спільної роботи вузлів. Вузол, який відповідає за обробку даних з лазерного датчика, може публікувати інформацію про відстань до об'єктів на топіці "laser_scan". Інші вузли можуть підписуватись на цей топік для використання цих даних.

Вузли можуть публікувати дані на топіки та підписуватись на топіки, щоб отримувати інформацію від інших вузлів.

Взаємодія здійснюється через публікацію та підписку на повідомлення, визначені в ROS, такі як стандартні повідомлення для зображень, геометричних

даних і т.д. Це забезпечує легкість додавання нових компонентів до системи без необхідності зміни всієї архітектури. Дослідження вузлів і топиків у ROS дозволяє покращити модульність, розширюваність та перевикористання програмного забезпечення для роботів. Описані концепції надають ефективний механізм створення складних систем роботів, де різні компоненти можуть взаємодіяти між собою задля досягнення спільної мети.

ROS заснований на архітектурі графів, тобто обробка даних відбувається в програмних вузлах або нодах (Node). Вузли можуть обмінюватися інформацією за моделлю публікуючий – підписник. Одна і та сама пара вузлів може бути взаємопідписана, що відрізняється від звичної моделі Master-Slave в IOT. Більш того, вузли можуть працювати окремо, багатопоточно.

Topic, або тема, виконує функцію передачі та сортування інформації між вузлами. Можна сказати що вузли використовують теми для комунікації (рисунок 3.1).

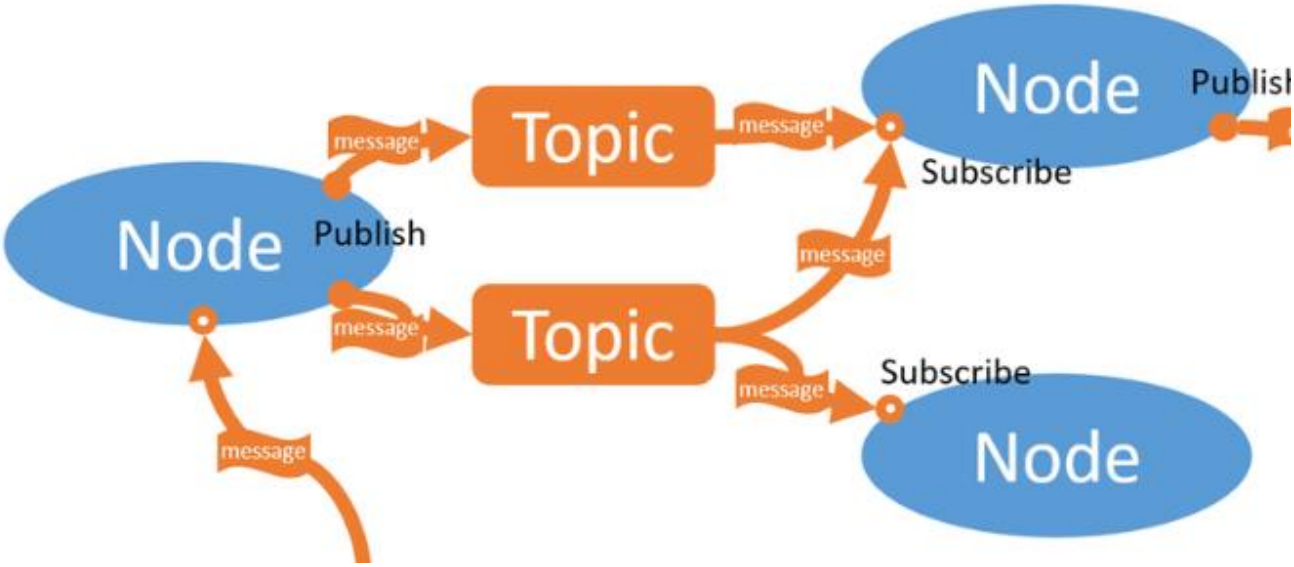


Рисунок 3.1 – Приклад сполучення між вузлами та темами

3.2 Налаштування фреймворку, створення робочого простору, та ініціалізація проекту

Після успішної установки ROS та необхідних пакетів за інструкцією з офіційного сайту, потрібно створити робочу директорію наступними командами:

Додаємо перемінні середи:

source /opt/ros/melodic/setup.bash

Створюємо робочу директорію catkin workspace – catkin_ws, у ній папку src – source. Використовуємо лінукс команду mkdir – make directory:

mkdir -p ~/catkin_ws/src

cd – лінукс команда для зміни директорії. Входимо в робочу директорію:

cd ~/catkin_ws/

catkin – базовий набір інструментів побудови пакетів в ROS. В даному випадку команда catkin_make збирає пакет. Без побудови пакету через catkin, тобто зробивши звичайну папку, у пакеті не буде файлу CMakeLists.txt, що у свою чергу буде необхідно при ініціалізації цього пакету:

catkin_make

Після побудови середовища встановлюємо джерело для файлів типу.sh:

source devel/setup.bash

Заходимо в папку src:

cd src

Створюємо проект mark1 командою catkin_create_pkg:

catkin_create_pkg mark1

В папці пакету mark1 створюємо директорію джерела:

cd mark1

mkdir src

cd src

У папці джерела створюємо 5 пакетів, де 2dnav – стек навігації, config – налаштування та виконавчий файл проекту, control – пакет управління рухом, description – опис робота формату .xacro для симулятора gazebo, gazebo – пакет опису простору у одноіменному симуляторі:

catkin_create_pkg mark1_2dnav

catkin_create_pkg mark1_config

catkin_create_pkg mark1_control

catkin_create_pkg mark1_description

catkin_create_pkg mark1_gazebo

Після конвертації моделі з Fusion360 маємо наступні папки – meshes та urdf (рисунок 3.2).

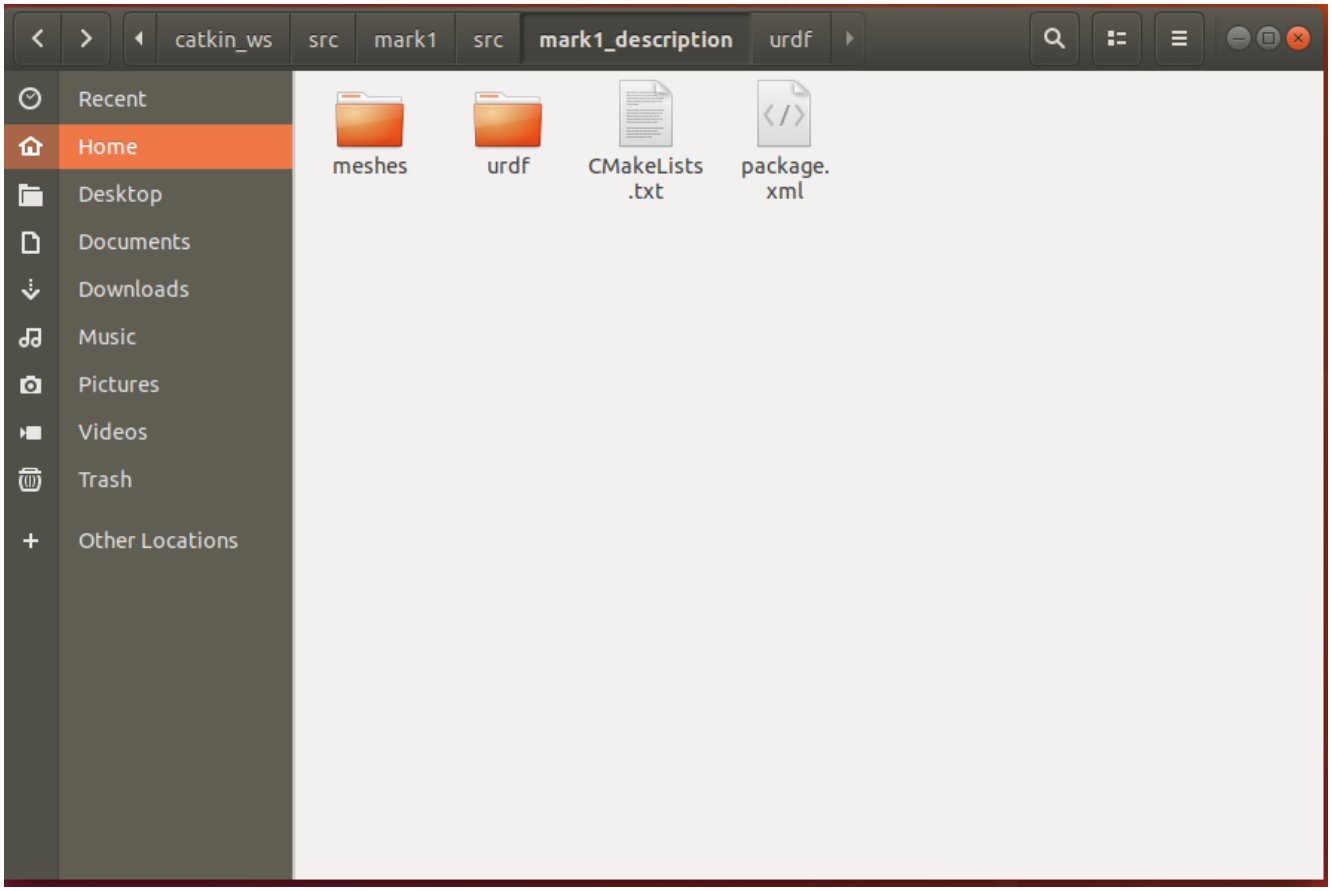
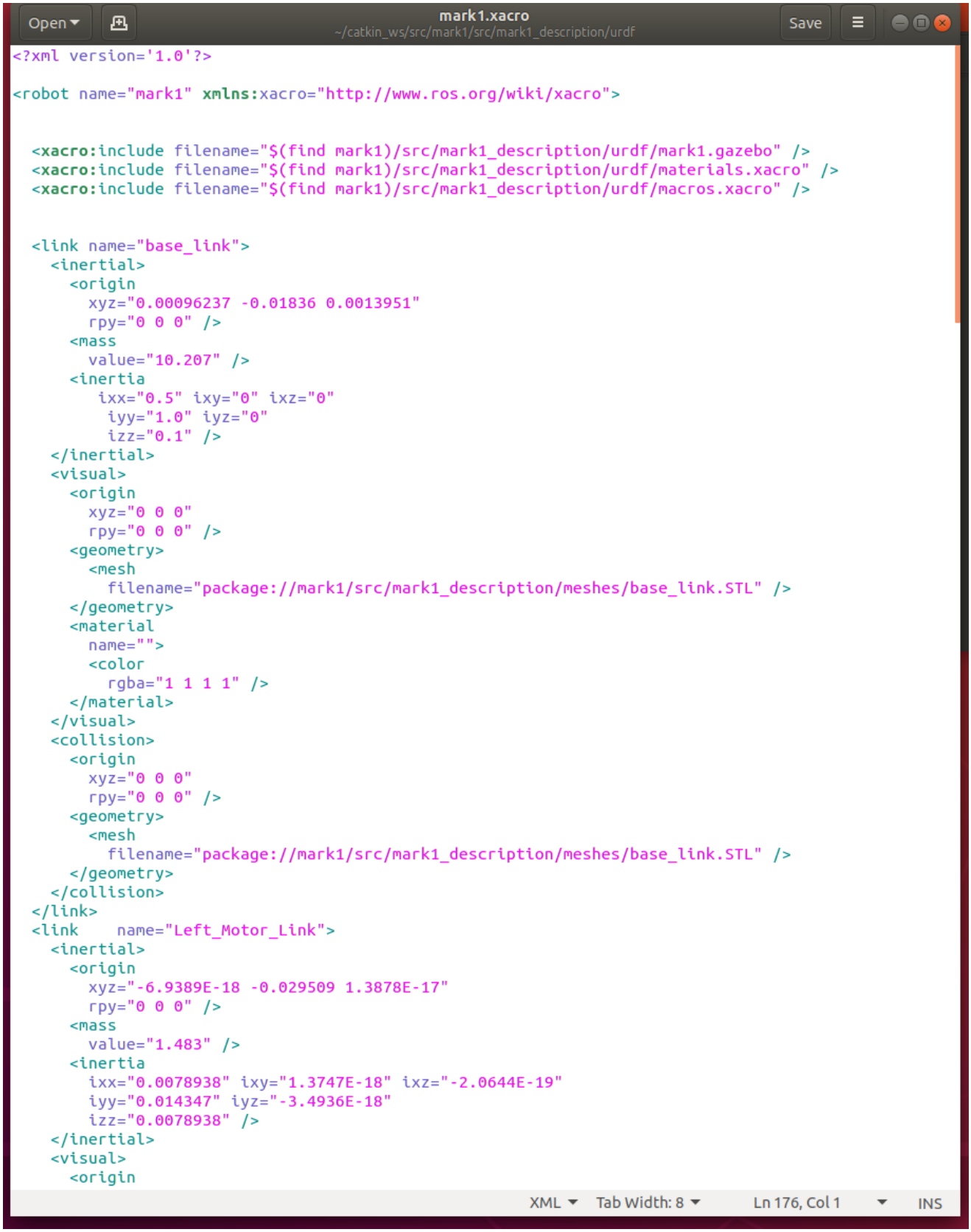


Рисунок 3.2 – Вміст пакету discription

У середині маємо xacro та stl файли, що є описом параметрів моделі. Її розмірів, ваги, типу з’єднань, тощо. Найбільш цікаво розглянути саме mark1.xacro файл (рисунок 3.3).




```
<?xml version='1.0'?>

<robot name="mark1" xmlns:xacro="http://www.ros.org/wiki/xacro">


  <xacro:include filename="$(find mark1)/src/mark1_description/urdf/mark1.gazebo" />
  <xacro:include filename="$(find mark1)/src/mark1_description/urdf/materials.xacro" />
  <xacro:include filename="$(find mark1)/src/mark1_description/urdf/macros.xacro" />


  <link name="base_link">
    <inertial>
      <origin
        xyz="0.00096237 -0.01836 0.0013951"
        rpy="0 0 0" />
      <mass
        value="10.207" />
      <inertia
         ixx="0.5" ixy="0" ixz="0"
          iyy="1.0" iyz="0"
          izz="0.1" />
    </inertial>
    <visual>
      <origin
        xyz="0 0 0"
        rpy="0 0 0" />
      <geometry>
        <mesh
          filename="package://mark1/src/mark1_description/meshes/base_link.STL" />
      </geometry>
      <material
        name="">
        <color
          rgba="1 1 1 1" />
      </material>
    </visual>
    <collision>
      <origin
        xyz="0 0 0"
        rpy="0 0 0" />
      <geometry>
        <mesh
          filename="package://mark1/src/mark1_description/meshes/base_link.STL" />
      </geometry>
    </collision>
  </link>
  <link    name="Left_Motor_Link">
    <inertial>
      <origin
        xyz="-6.9389E-18 -0.029509 1.3878E-17"
        rpy="0 0 0" />
      <mass
        value="1.483" />
      <inertia
        ixx="0.0078938" ixy="1.3747E-18" ixz="-2.0644E-19"
        iyy="0.014347" iyz="-3.4936E-18"
        izz="0.0078938" />
    </inertial>
    <visual>
      <origin
```

Рисунок 3.3 – Вміст файлу mark1.xacro

Група рядків що позначена link name характеризує конкретне з’єднання, його

позицію відповідно до початку системи координат, вагу. У mesh вказується залежність до stl файла що і описує меші моделі. Тобто фізичні параметри і форма моделі зберігаються окремо.

Окрім пакету description необхідно створити, та заповнити згідно документації пакет config, у якому у тому числі знаходитиметься файл mark1_gazebo.launch (рисунок 3.4).

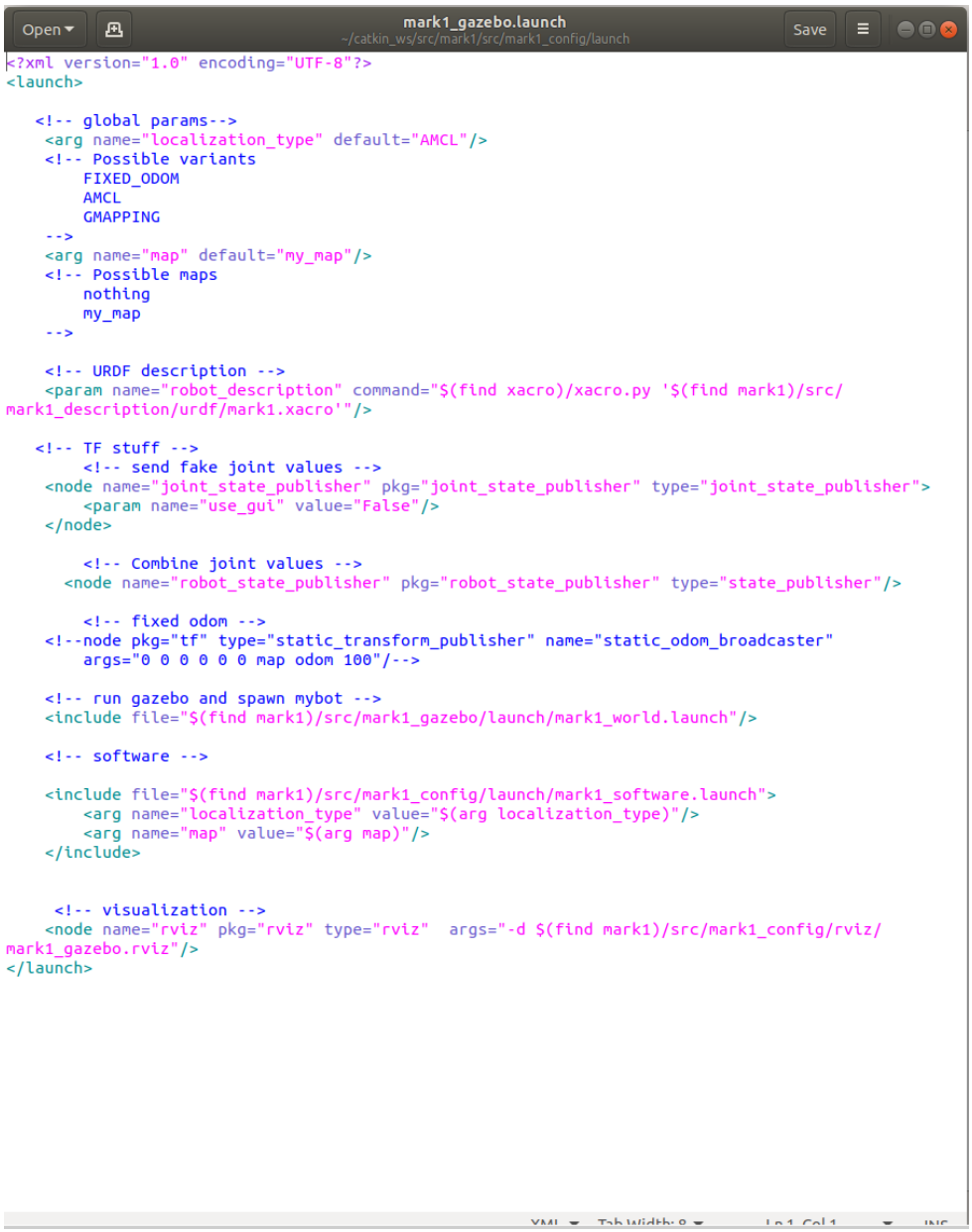


Рисунок 3.4 – Вміст файлу mark1_gazebo.launch

Вище названий файл є виконавчим, тобто саме за допомогою нього проект

запускатиметься в симуляторі gazebo. Тут і підтягуються файли software, world і xacro.

Для запуску Robot Operating System – використовуємо команду roscore, ядро ros (рисунок 3.5).

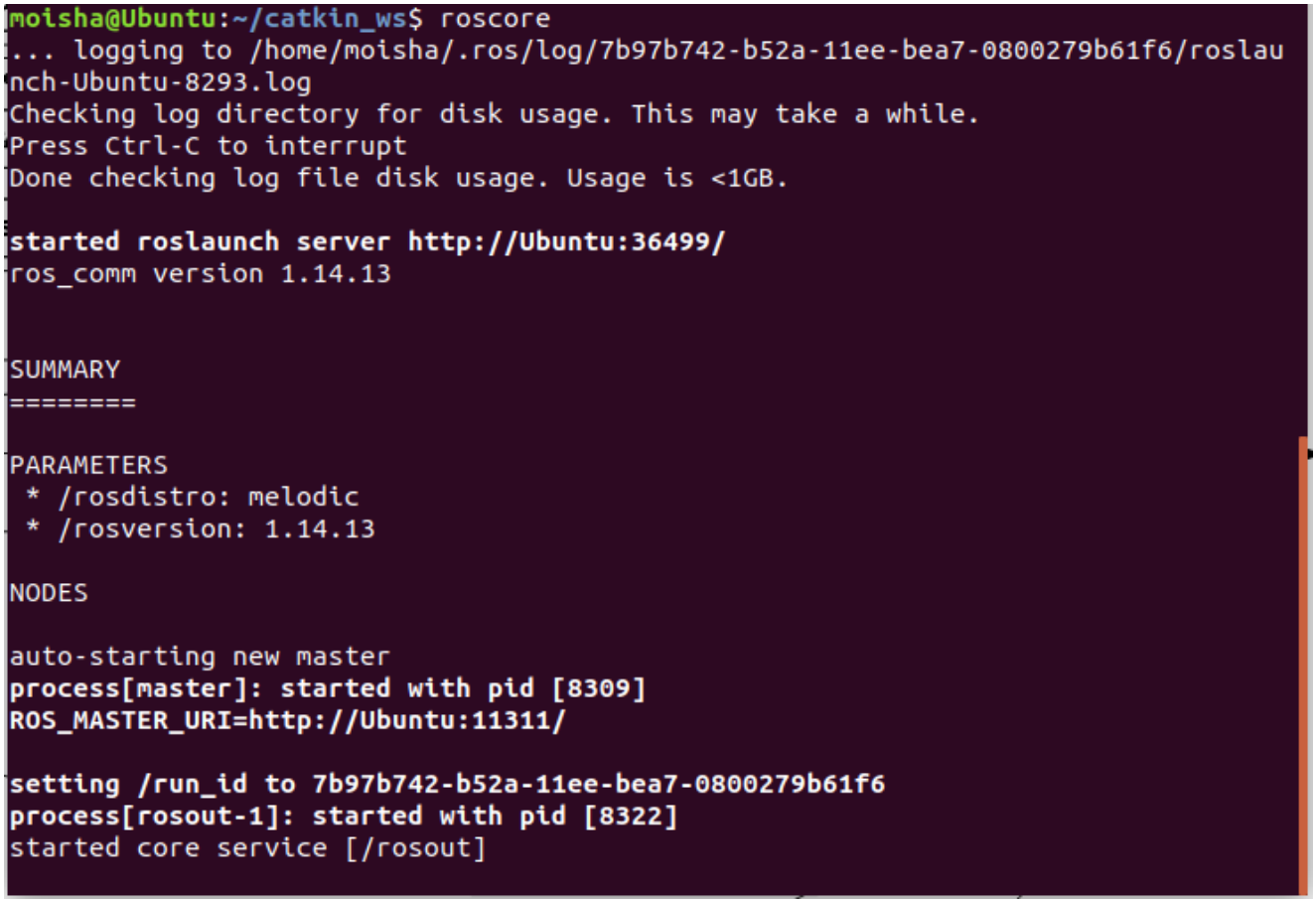


Рисунок 3.5 – Старт ядра ROS

У іншому вікні терміналу після встановлення джерела .sh, ініціюємо файл mark1_gazebo.launch проекту mark1 (рисунок 3.6).

```
moisha@Ubuntu:~/catkin_ws$ roslaunch mark1 mark1_gazebo.launch
... logging to /home/moisha/.ros/log/7b97b742-b52a-11ee-bea7-0800279b61f6/roslau
nch-Ubuntu-24679.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

xacro.py is deprecated; please use xacro instead
started roslaunch server http://Ubuntu:45255/
```

Рисунок 3.6 – Ініціалізація проекту

Одразу після завантаження, відкривається вікно симулятора Gazebo, і вікно візуалізації Rviz. (рисунок 3.7) і (рисунок 3.8) відповідно.

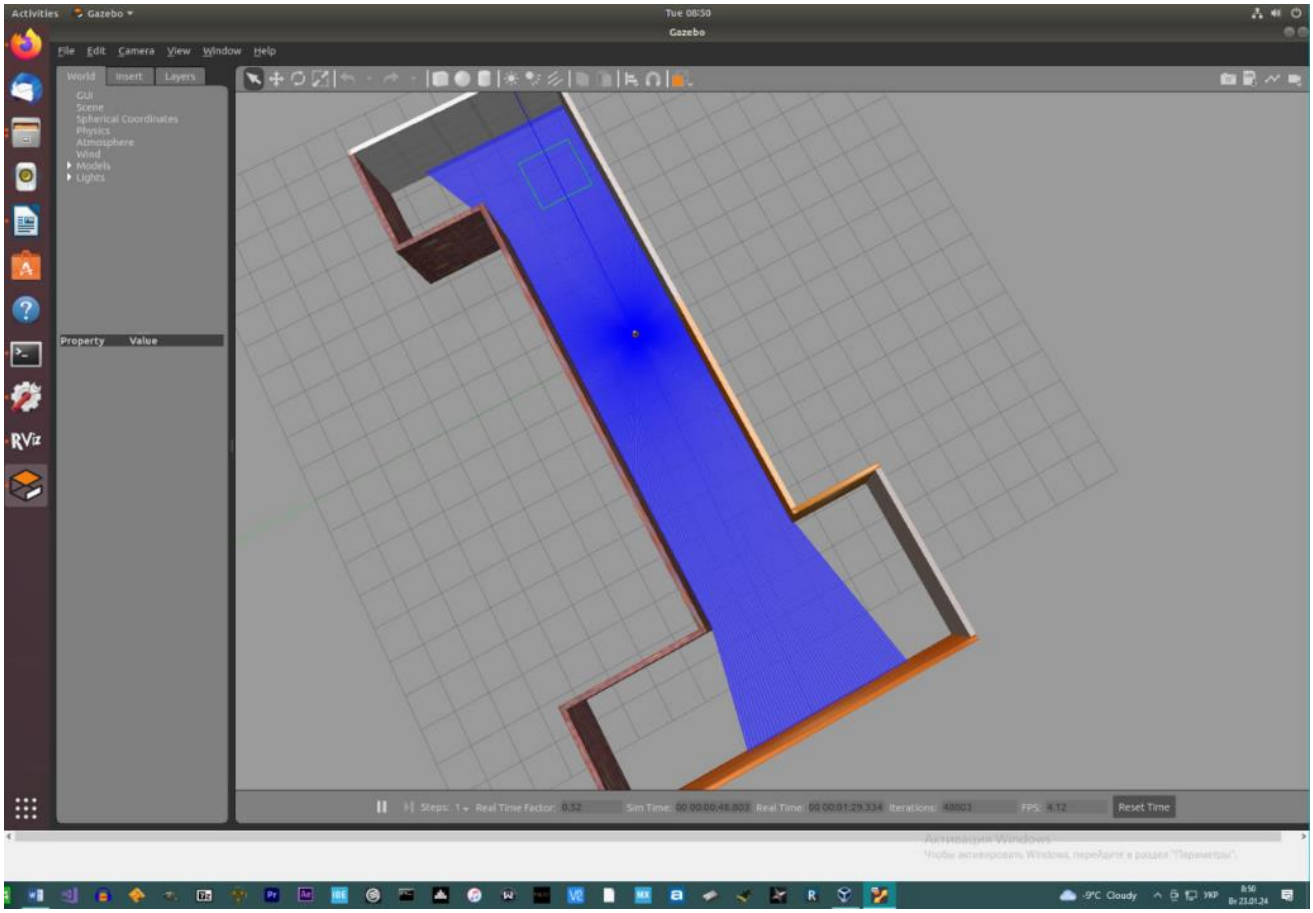


Рисунок 3.7 – Вікно симулятора Gazebo

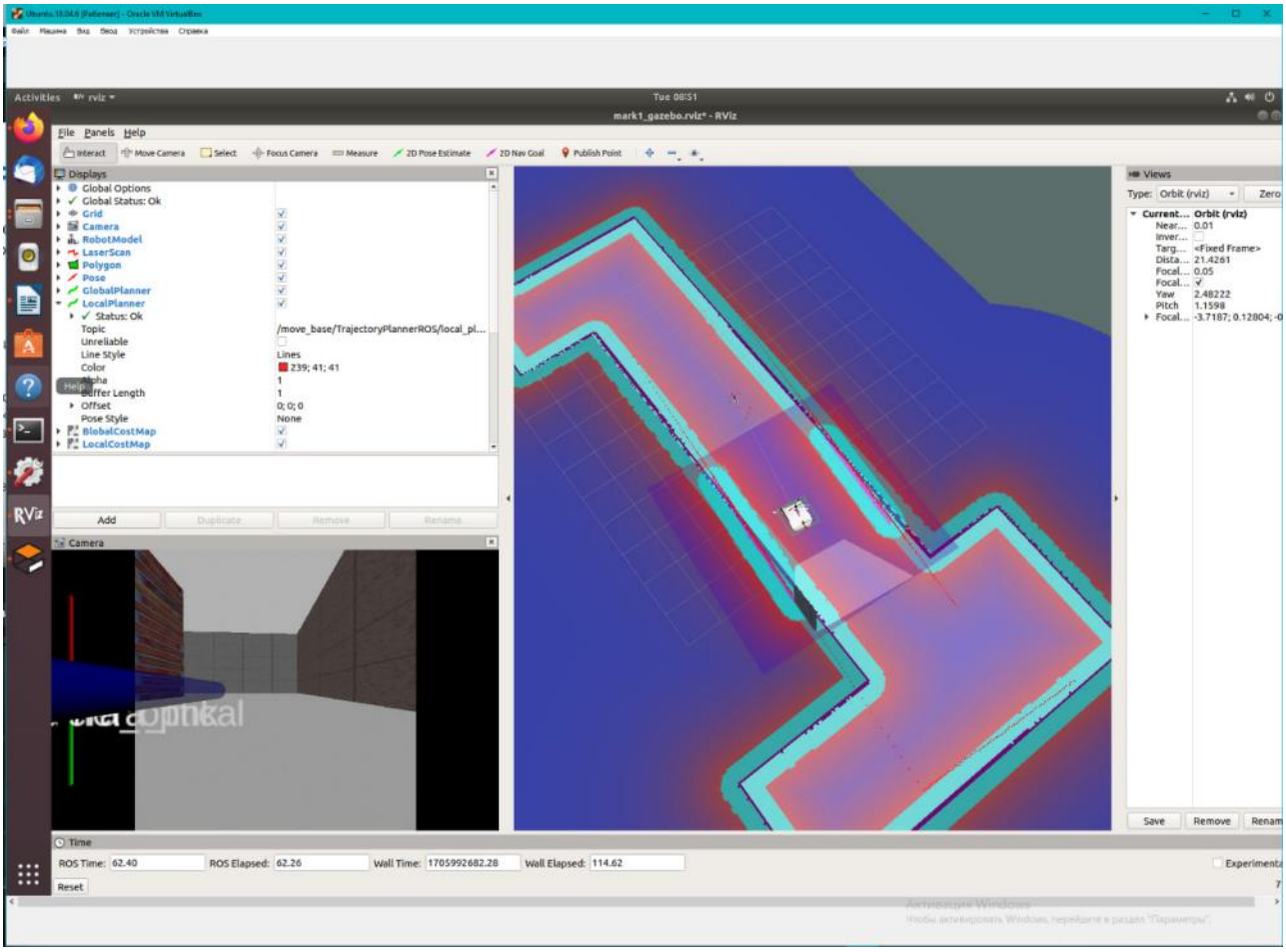


Рисунок 3.8 – Вікно візуалізації Rviz

Тим часом як симулятор прораховує колізію моделі і скан лідара, візуалізіція відображає оброблене зображення з побудованою картою місцевості.

На рисунках можна побачити побудовану модель 3D простору. Модель побудована згідно даних отриманих з лідару. Отримана модель зберігається у файлі my_map.yaml і в подальшому може бути використана для навігації завдяки навігаційному стеку 2dnav.

3.3 Висновки за розділом 3

У розділі 3 було розроблено програмне забеспечення для побудови 3D моделі навколишнього середовища мобільного робота. Для виконання задачі було проаналізовано наявні потужності фреймфорку ROS першої версії. Обрано сумісну операційну систему. Використано знання отримані в процесі навчання для побудови моделі мобільного робота у CAD системі Fusion360. Було протестовано можливість навігації за отриманою моделлю наволишнього середовища. Отримано результати, зшідно яким така можливість існує, проте точність позіціонування не є задовільною і потребує подальшої доробки.

Загалом у якості висновку можна підкреслити великі перспективи використання ROS у галузі робототехніки для потреб підприємств та ринку споживачів.

ВИСНОВКИ

Створення програмного забезпечення для побудови 3D моделі навколишнього середовища мобільного робота є актуальним завданням з численних причин, які враховують технологічний, економічний та соціальний контексти. Програмні засоби керування дозволяють оптимізувати роботу промислових роботів, зменшуючи час виконання завдань і підвищуючи загальну продуктивність виробництва. Завдяки програмному забезпеченню, мобільні роботи можуть автоматично виконувати широкий спектр завдань, що варіюються від монотонних і рутинних до складних і трудомістких, що призводить до збільшення ефективності виробництва. Програмне керування дозволяє забезпечити високу точність виконання роботами, що може бути критичним у виробничих процесах, де важлива якість і точність. Використання програмних засобів дозволяє оптимізувати використання ресурсів, таких як енергія та матеріали, зменшуючи витрати та вплив на навколишнє середовище. Інтеграція продуманих програмних рішень управління може сприяти підвищенню безпеки промислового виробництва, виключаючи або мінімізуючи ризики аварій і травматичних ситуацій. Загалом, створення програмного забезпечення для побудови 3D моделі навколишнього середовища мобільного робота є важливим етапом для покращення конкурентоспроможності підприємств, забезпечення стабільності та розвитку виробничого сектору.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

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

2. Дипломне проектування для студентів усіх форм навчання спеціальностей 151 «Автоматизація та комп'ютерно-інтегровані технології»: навч. посібник / І. Ш. Невлюдов, А. О. Андрусевич, О. В. Токарєва,Г. В. Пономарьова. Київ, 2018. 320 с.

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

4. Невлюдов І. Ш. Інтелектуальне проектування технології роботизованого збирання: моногр. / І. Ш. Невлюдов, А. М. Цимбал, С. С. Мілютін. – Харків: нац. ун-т радіоелектроніки. – Харків : ХНУРЕ, 2010 – 206 с.

5. Akopov M. and Maksumova S. (2023), «Selection of Sensors for Building a 3D Model of the Mobile Robot's Environment», VII International Conference MANUFACTURING & MECHATRONIC SYSTEMS pp. 33-34 [Електронний ресурс] – Режим доступу: https://openarchive.nure.ua/bitstreams/550a3f42-eeb4-4555-95b1-ab22a699e4bd/download

6. Цимбал О. М. Технології програмування: Visual C++:навч. посібник / О. М. Цимбал ; МОН України; НМЦВО, ХНУРЕ –Харків : ХНУРЕ, 2006.–336 с.

7. Офіційний сайт RoboRealm – [Електронний ресурс] – Режим доступу: www/ URL:https://www.roborealm.com/help/Object_Recognition.php

8. Arduino Nano – [Електронний ресурс] – Режим доступу: www/ URL:

https:// doc.arduino.ua/ru/hardware/Nano

9. Двигун з редуктором 1:48 (двох-осьовий) – Режим доступу: www/ URL: https://arduino.ua/prod662-motor-s-redyktorom-148-dvyh-osevoi

10.Драйвер двигунів L293D – [Електронний ресурс] – Режим доступу: www/ URL: https://myrobot.ru/stepbystep/el_driver.php

11. Драйвер двигуна на L293D модуль – [Електронний ресурс] – Режим доступу: www/URL:https://electronoff.ua/good/drayver-dvigatelya-na-l293d-modul.ph

12. Офіційний сайт Kingston – [Електронний ресурс] – Режим доступу: www/ URL:https://www.kingston.com/ua/memory-cards/high-endurance-microsd-card

13. Офіційний сайт TinkerCAD – [Електронний ресурс]. – Режим доступу: www/ URL: https://www.tinkercad.com/dashboard.

14. Ян Ерик Солем – Програмування машинного зору мовою

15. Python.PID regulator – [Електронний ресурс]. – Режим доступу: www/ URL: https://forum.arduino.cc/t/pid-control-on-2-wheels-robot/534233.

16. Лекційний матеріал № 2–3 з дисципліни "Гнучкі комп'ютеризовані робототехнічні системи та технології їх програмування" для студентів усіх форм навчання напряму 6.050202 "Автоматизація та комп’ютерно – інтегровані технології" / упоряд. : В. В. Євсеев, Д. В. Гурін ; М-во освіти і науки України, ХНУРЕ. – Харків : ХНУРЕ, 2022. – 18 с.

17. Module 'cv2.cv2' has no attribute 'COLOR' – [Електронний ресурс] – Режим доступу: www/ URL:https://stackoverflow.com/questions/65633635/module-cv2-cv2-has-no-attribute-color

18. AttributeError: module 'cv2' has no attribute 'VideoCapture' – [Електронний ресурс] – Режим доступу: www/URL: https://stackoverflow.com/questions/66622649/attributeerror-module-cv2-has-no-attribute-videocapture