Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

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

АТЕСТАЦІЙНА РОБОТА

Пояснювальна записка

Другий (магістерський)
(рівень вищої освіти)

Розроблення комп'ютерно- інтегрованої системи управління транспортним засобом на основі системи комп'ютерного зору

(тема)

Виконав:
студент 2 курсу, групи КІТПВм-20-1

Приходько Максим Олександрович
(прізвище, ініціали)

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

Тип програми Освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп'ютерно-інтегровані технологічні процеси і виробництва
(повна назва освітньої програми)

Керівник проф. Цимбал О.М.
(посада, прізвище, ініціали)

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

2021р.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет АКТ
Кафедра КІТАМ
Рівень вищої освіти другий (магістерський)
Спеціальність 151 Автоматизація та комп’ютерно-інтегровані технології
Тип програми Освітньо-професійна
Освітня програма Комп’ютерно-інтегровані технологічні процеси і виробництва
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри КІТАМ _____
(підпис)
«____»_____2021_ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Приходько Максиму Олександровичу
(прізвище, ім’я, по батькові)

1. Тема роботи Розроблення комп'ютерно- інтегрованої системи управління транспортним засобом на основі системи комп’ютерного зору
затверджена наказом по університету від 08.11.2021 р. № 1697Ст
2. Термін подання студентом роботи до екзаменаційної комісії 16.12.2020 р.
3. Вихідні дані до роботи Raspberry Pi 4, модуль камери Intel RealSense D435i, Платформа Pololu Zumo, Ноутбук, Підключення до мережі інтернет
4. Перелік питань, що потрібно опрацювати в роботі
4.1 Детекція маркерів Aruco;
4.2 Вимірювання відстані до маркерів;
4.3 Реалізація інструкцій руху платформи;
4.4 Висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) Демонстраційний матеріал представлений у форматі презентації PowerPoint (*.ppt) – 9с. формату А4
6. Консультанти розділів роботи

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

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примі т-ка
1	Аналіз завдання	05.10.2021	
2	Огляд літератури за темою роботи	11.10.2021	
3	Аналіз існуючих рішень	17.10.2021	
4	Моделювання та підбір компонентів	6.11.2021	
5	Розробка алгоритму роботи	25.11.2021	
6	Оформлення атестаційної роботи	05.12.2021	
7	Подання роботи в ЕК	16.12.2021	

Дата видачі завдання 08.11.21

Студент _____ Приходько М.О.
(підпис) (прізвище, ініціали)

Керівник роботи _____ проф. Цимбал О.М.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 87 с., 1 табл., 74 рис., 1 дод., 17 джерел.

АНАЛІЗ ЗОБРАЖЕНЬ, КАМЕРА ГЛИБИНИ, МОДЕЛЮВАННЯ, ЕКСПЕРИМЕНТ.

Об’єкт дослідження – автоматизована система керування транспортними засобами на основі системи комп’ютерного зору.

Предмет дослідження – автоматизована платформи.

Мета атестаційної роботи – розробити систему автоматичного керування транспортним засобом на основі комп’ютерного зору.

Методи дослідження – математичні методи розрахунку, емпіричний метод спостереження, експеримент, аналіз даних.

В роботі проведено аналіз існуючих рішень систем автоматизації транспортних засобів, їх складових та областей застосування. Розглянуто особливості роботи алгоритмів розпізнавання та аналізу зображень, аналізу глибини на зображенні. Спроектовано та розроблено програмно-апаратний комплекс, що дозволяє тестувати алгоритми керування транспортним засобом, та реалізує основу для робота під склади підприємств.

Для визначення трудомісткості роботи було проведено необхідні розрахунки, що наведені у розділі охорони праці.

Результати магістерської атестаційної роботи було опробовано у доповіді на спеціалізованій конференції.

ABSTRACT

Explanatory note: 87 p., 1 tables, 74 figures, 1 app., 17 sources.

IMAGE ANALYSIS, DEPTH CAMERA, DIAGNOSIS, MODELING, EXPERIMENT.

The object of research – automated vehicle control system based on computer vision system.

The subject of research is the automated platform.

The purpose of the certification work is to develop a system of automatic control of the vehicle based on computer vision.

Research methods – mathematical methods of calculation, empirical method of observation, experiment, data analysis.

The analysis of existing solutions of vehicle automation systems, their components and areas of application is carried out in the work. Features of work of algorithms of recognition and the analysis of images, the analysis of depth on the image are considered. The software and hardware complex is designed and developed, which allows to test the algorithms of vehicle control, and implements the basis for work under the warehouses of enterprises.

Necessary work was carried out to determine the complexity of the work calculations given in the section of labor protection.

The results of the master's attestation work were tested in reports at a specialized conference.

ЗМІСТ

ПЕЛІК СКОРОЧЕНЬ

ГГц – гігагерц;

ІЧ – інфрачервоний;

ПЗС – прилад із зарядовим зв'язком;

ADAS – advanced driver assistance system;

CAN – controller area network;

CNN – convolutional neural network;

CPU – central processing unit;

FOV – field of view;

FPGA – field-programmable gate array;

GNSS – global navigation satellite system;

GPS – global positioning system;

GPU – graphics processing unit;

IMU – inertial measurement unit;

LVDS – low-voltage differential signaling;

PWM – pulse-width modulation;

SLAM – simultaneous localization and mapping;

ToF – time-of-flight camera.

ВСТУП

На сьогоднішній день комплексні автоматизовані системи керування транспортними засобами стали універсальним інструментом для вирішення широкого спектру проблем, наприклад автоматизація промислових складів, доставка вантажів, самокерований комерційний транспорт. За рахунок використання автоматизованих систем керування транспортними засобами збільшилась безпека і точність транспортних засобів основаних на подібних системах.

В умовах швидкого розвитку та автоматизації транспортних засобів найбільш актуальним виникає питання впровадження систем автоматичного керування транспортними засобами. Для задоволення потреб галузі та підвищення ефективності та якості керування розробляються більш точні та швидкі алгоритми і системи аналізу зображення.

Об’єкт дослідження – автоматизована система керування транспортними засобами на основі системи комп’ютерного зору.

Предмет дослідження – принципи роботи та вилучення чинників керування з зображення.

Актуальність роботи – системи автоматизації транспорту набувають все більшу популярність у різноманітних сферах, від роботів для роботи на складах, до комерційного транспорту. Тому тематика автоматизованого керування набирає все більшого попиту на ринку автоматизованих систем.

Мета роботи – розробити систему автоматичного керування транспортним засобом на основі комп’ютерного зору.

Завдання роботи – аналіз та огляд існуючих систем автоматизації транспортних засобів, огляд та порівняння алгоритмів та існуючих засобів розпізнавання та ідентифікації оточення, розробка програмного забезпечення для пересування тестової платформи на основі маркерів Aruco [1].

Атестаційна робота виконується за методичними вказівками [2] та згідно ДСТУ 3008-15 [3].

1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ ТА АНАЛОГІЧНИХ РІШЕНЬ

Метою даної магістерської атестаційної роботи є розробка комп'ютерно-інтегрованої системи управління транспортним засобом на основі системи комп’ютерного зору, що базується на експериментальному дослідженні алгоитмів аналізу та обробки зображення отриманного з камер. Для цього необхідно проаналізувати предметну область, розглянути питання пов’язані з процесом обробки та аналізом об’єктів на зображенні, розробити засоби керування транспортним засобом. Необхідно провести аналіз існуючих систем, виділити їх особливості і підібрати пристрої які будуть використані при розробці власної системи.

Для використання в промислових цілях випущена велика кількість систем керування транспортними засобами, кожна з них має як свої плюси, так і мінуси.

Також різниться і призначення систем, наприклад вони вже застосовуються у комерційному вантажному транспорті, автономних роботах для огляду території (наприклад робот Spot від компанії Boston Dynamics), сільськогосподарський техніці та автомобілях. Завдяки інтеграції подібних систем вдається досягти високої точності та надійності, зменшити потрібність у великій кількості персоналу, у випадку з вантажним транспортом досягти більш економного та екологічного водіння.

В якості існуючих засобів систем керування транспортними засобами були вибрані продукти популярних виробників в цій сфері.

1.1 Система керування транспортним засобом компанії Cognitive Pilot

Однією з існуючих систем керування на основі візуальної інформації е комплекси для автоматизації сільськогосподарської техніки від компанії Cognitive Pilot [4] (рисунок 1.1). Використання системи керування компанії

Cognitive Pilot дозволило оптимізувати маршрути роботи техніки задля економії пального та збільшити кількість готову продукції за рахунок аналізу зон обробки. Сама система також дозволяє розпізнавати перешкоди, класифікувати їх та приймає рішення, що робити: об'їжджати, гальмувати або, поки є час, попереджати оператора.

Рисунок 1.1 – Зовнішній вигляд системи

Дані отримані з камери обробляються комп'ютерним модулем з нейронною мережею, завдання якої розпізнавати зони роботи для техніки та будувати на їх основі маршрути (рисунок 1.2) [4].

Ось які завдання виконує дана система:

– керує комбайном враховуючи особливості росту культури, і по тому як пройшли комбайни до нього;

– дозволяє працювати машинам зі змішаного парку, як машинам з автопілотом так і на ручному керуванні. Для автопілоту немає жодної різниці;

– дуже чітко утримує відстань між проходами, забезпечуючи мінімум необроблених ділянок;

– шукає перешкоди, класифікує їх та приймає рішення що робити: об'їжджати, гальмувати або, поки є час, попереджати оператора;

– підтримує оптимальну швидкість у конкретній ситуації.

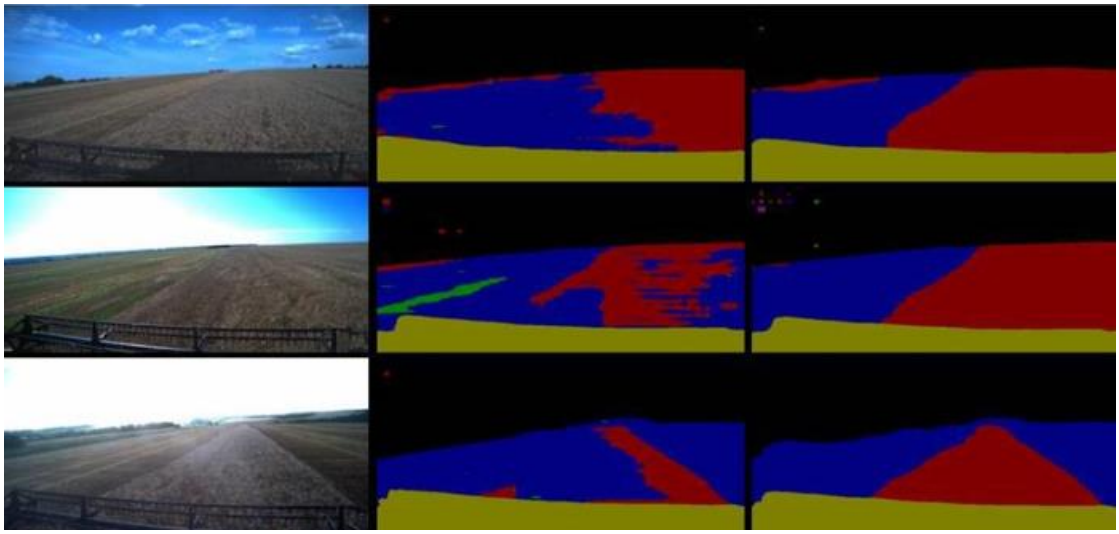

Рисунок 1.2 – Результат розмітки зображення

Оброблені данні для керування технікою передаються в CAN шину транспортного засобу, до якої підключені всі виконуючі механізми.

1.2 Компанія TuSimple

Окрім сільськогосподарської техніки системи автопілота використовують і у комерційному транспорті, так наприклад стартап TuSimple [5] розробив систему керування вантажним автомобілем, в рамках тестування якого без втручання водія вдалося подолати транспортний маршрут протяжністю 1500 км за 14 годин 6 хвилин, тоді як ця ж відстань фура під керуванням водія проїжджає приблизно за 24 години. Сама система автопілоту включає в себе 12 камер та 3 лідара, які дозволяють аналізувати ситуацію на дорозі (рисунок 1.3).

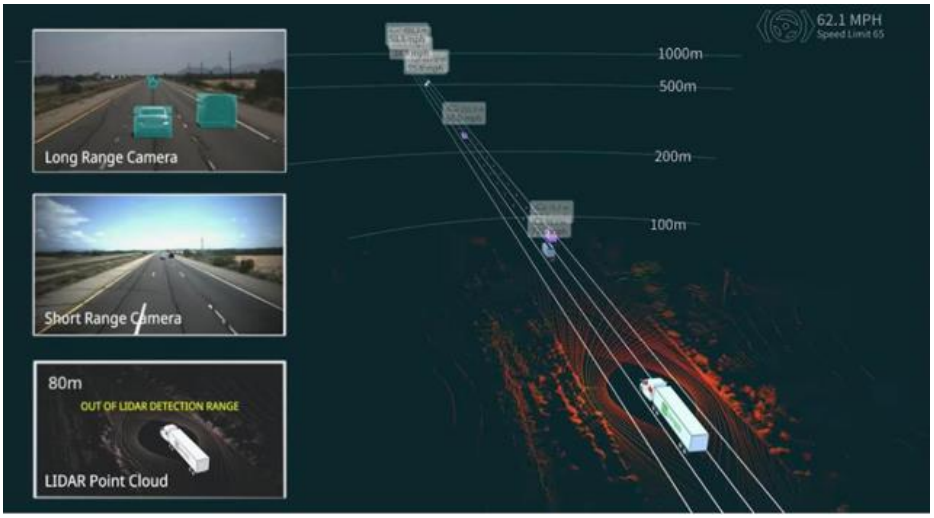

Рисунок 1.3 – Аналіз ситуації на дорозі

З використанням системи автоматичного керування компанії TuSimple вдалося досягти покращення якості вантажоперевезення, поліпшення якості керування, зменшення вартості та зниження забруднення оточуючої середи.

1.3 Компанія Tesla

Окрім перерахованих прикладів високим попитом користується системи автопілотів сучасних автомобілів, які дозволяють частково зняти навантаження з водія транспортного засобу та збільшити безпеку пересування та уникнення аварійних ситуацій. Лідером в цій сфері є компанія Tesla [6] з їх системами автопілоту [6].

Сам автопілот використовує 8 камер, набор ультразвукових сенсорів (рисунок 1.4), працюючих на близькій дистанції та радару якій знаходиться за бампером та випромінює та приймає радіохвилі (діапазон частот 76–77 ГГц).

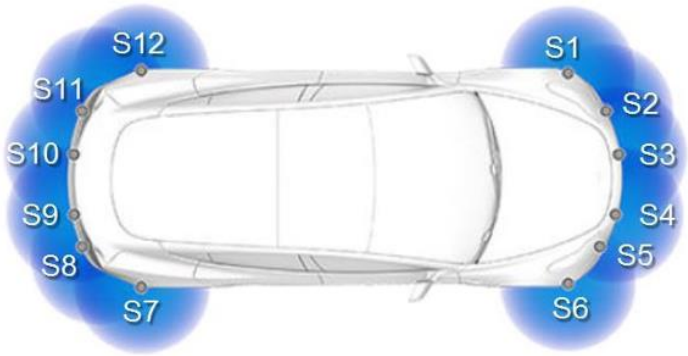

Рисунок 1.4 – Масив парктроників

Цілі перед радаром (рисунок 1.5) відображають сигнал залежно від їх розмірів та форми. Оскільки вантажівки відбивають більше, ніж велосипеди, відбиті сигнали можна використовувати для базової ідентифікації та класифікації об'єктів. Радар може відстежувати та класифікувати об'єкти одночасно для кількох цілей (рисунок 1.6). Використовуючи опосередковане відображення, радар також може виявляти транспортні засоби, що йдуть попереду транспортного засобу. Це дає можливість більш ранньої реакцію на уповільнення руху транспортного засобу що знаходиться попереду. Відносні швидкості радарних цілей визначаються з використанням принципу ефекту Доплера.

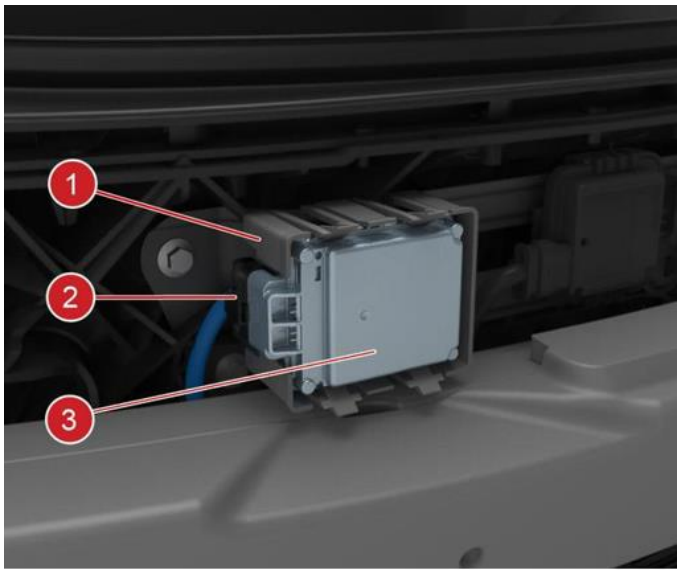

Рисунок 1.5 – Розміщення блоку радару

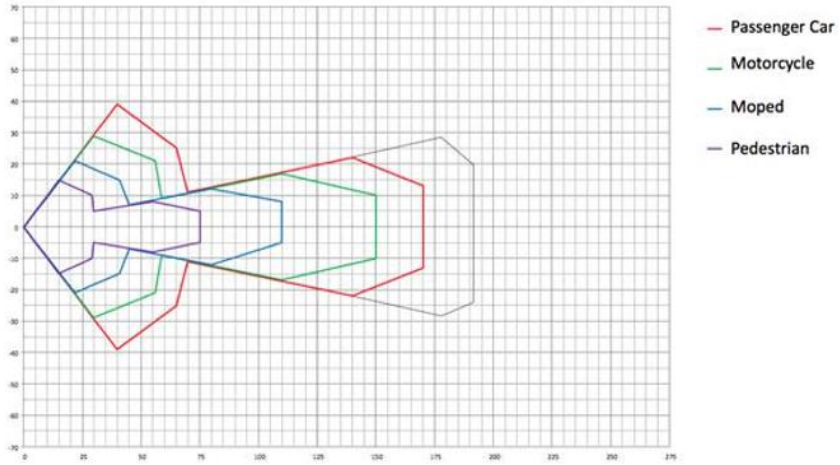

Рисунок 1.6 – Класифікація об’єктів радаром

Застосування масиву камер дозволяє аналізувати оточуючу ситуацію на дорозі, а за допомогу радару проводиться аналіз перешкод та інших транспортних засобів попереду, задля визначення їх розташування та швидкості (рисунок 1.7).

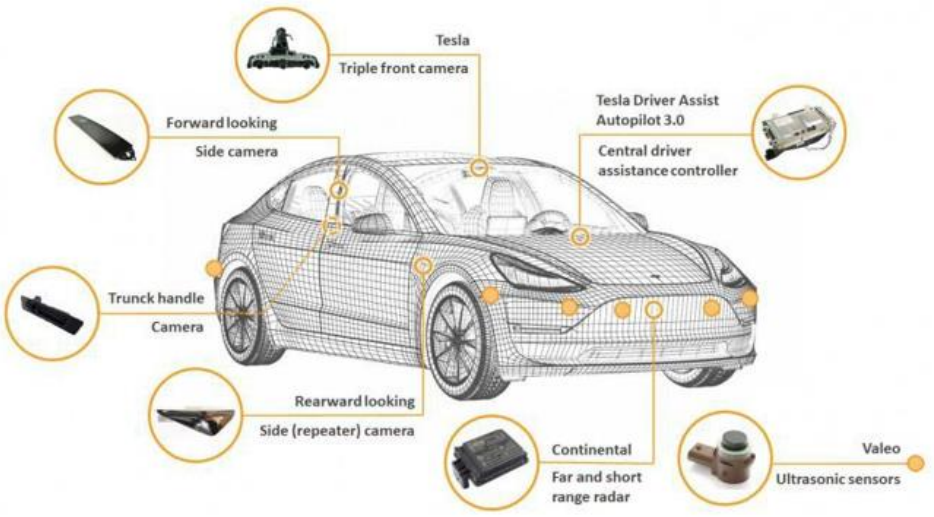

Рисунок 1.7 – Загальне розміщення сенсорів автопілоту

Отримані данні передаються по шинам CAN та LVDS до центрального комп’ютера (рисунок 1.8), задача якого проводити сбір та аналіз даних, з подальшим вилученням сигналів керування для двигунів та рульової рейки [6].

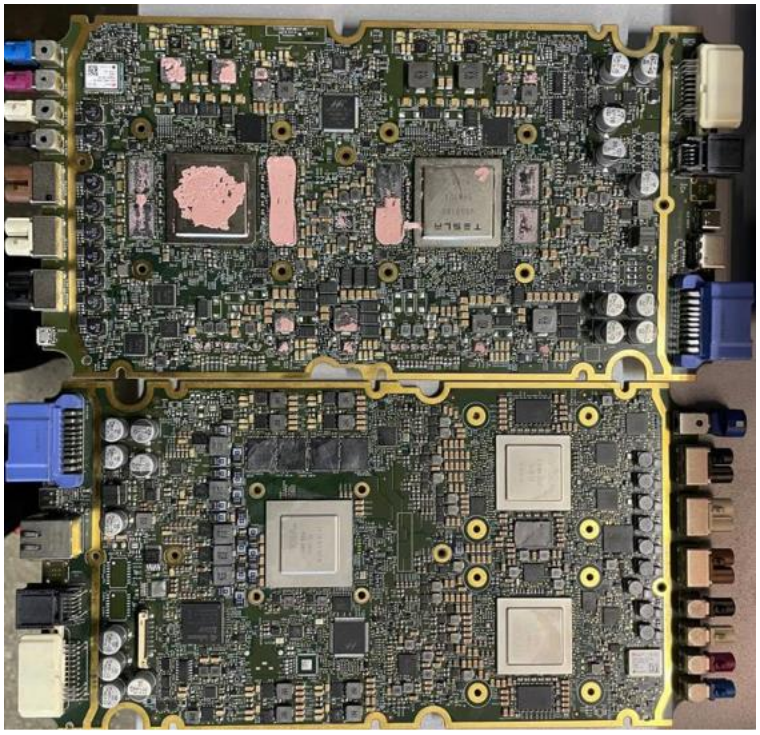

Рисунок 1.8 – Плати двох поколінь автопілотів

Задача нейронних мереж в комп’ютері автопілоту полягає в обробці сирих даних з камер, побудові 3-х вимірної карти оточення (рисунок 1.9), аналізу об’єктів (дорожніх знаків, розмітки, інших учасників руху, перешкод та іншого). В результаті обробки цих даних будується оптимальній маршрут руху з дотриманням віх правил безпеки руху.

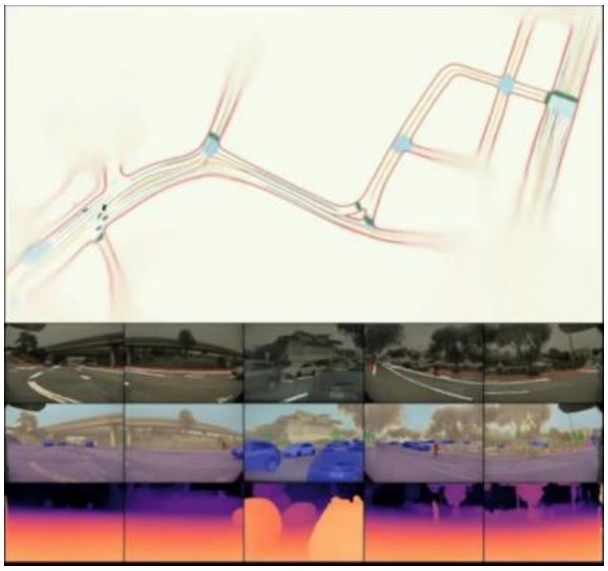

Рисунок 1.9 – Карта оточення автомобіля

Процес обробки візуальної інформації можна поділити на такі основні етапи:

– спочатку зображення відправляються в модуль виправлення, який приймає зображення та калібрує їх, переводячи у віртуальну виставу. Цей крок значно покращує продуктивність автопілота, оскільки робить зображення більш схожими один на одного, коли нічого не відбувається, що дозволяє мережі легше порівнювати зображення і фокусуватися на основних компонентах, які не є частиною типового фону;

– потім оброблені зображення надсилаються в першу мережу під назвою RegNet [7]. RegNet - лише оптимізована версія CNN зі згортковою нейронною мережею. Дана модель має багато виходів, кожен із яких повідомить вам загальну інформацію про загальну картину, чи є у ній автомобіль, дорожній знак тощо;

– у результаті злиття двох вищеописаних методів з’являється можливість отримати загальну інформацію про склад оточення, та розміщення всіх його елементів.

Завдяки вищеописаних методів аналізу виникає можливість реалізації наступного функціоналу для автопілоту:

– зміна смуги руху (рисунок 1.10) це дуже серйозний маневр на дорозі, оскільки існують так звані «сліпі» зони. Тому автопілот може автоматично перебудуватися у обрану водієм смугу, вибравши оптимальний момент для безпечного здійснення маневру;

Рисунок 1.10 – Зміна смуги руху

– автоматичне паркування (рисунок 1.11). Труднощі з паралельним паркуванням виникають у багатьох автовласників, але електроніка в Tesla може зробити все самостійно. Варто згадати, що ця функція використовується в багатьох автомобілях інших марок, тому інновацією тут і не пахне. Але хочеться наголосити, що система автоуправління може припаркувати транспортний засіб навіть на досить тісному паркуванні, з чим зазвичай не можуть впоратися інші авто;

Рисунок 1.11 – Автоматичне паркування

– безпека пішоходів. Система від розробників Tesla також здатна розпізнавати пішоходів. Якщо автопілот вважає, що існує високий ризик зіткнення, він автоматично задіяє гальмівну систему авто;

– самонавчання. Характерною особливістю системи є те, що вона постійно вдосконалюється, причому це відбувається в режимі реального часу. Ця система вміє самонавчатися під час використання електрокара. Принцип роботи функції дуже нескладний - під час експлуатації автономної системи відбувається збір інформації та передача на сервери розробників з метою подальшого аналізу даних. Зв'язок блоку автопілоту (рисунок 1.12) з виконавчими механізми (рульовою рейкою, гальмами, моторами) реалізований на основі CAN шини, яка має змішану топологію (рисунок 1.13). Для збільшення точності також використовуються блок GPS.

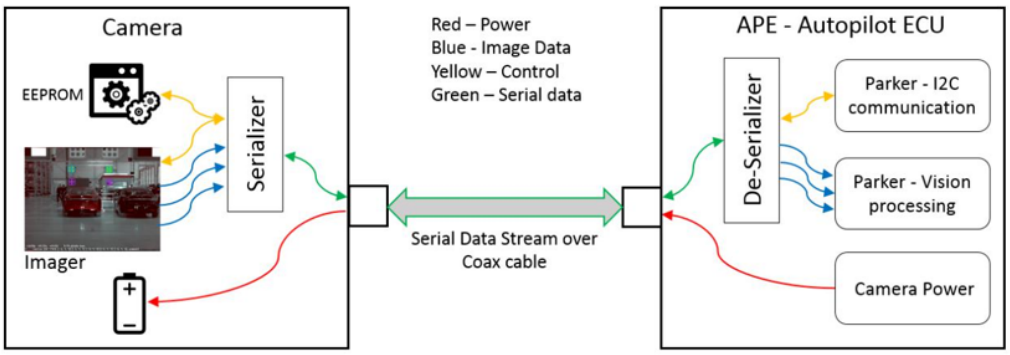

Рисунок 1.12 – Програмна топологія обробки зображення

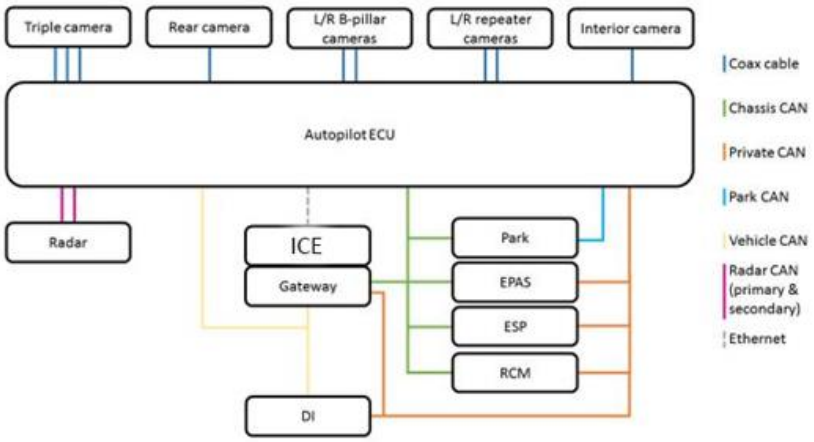

Рисунок 1.13 – Топологія мережі CAN шини

В результаті аналізу зрозуміло що системи автоматичного керування траннпортом набирають все більшу популярність, дозволяючи покращувати якість та безпеку керування транспортними засобами. Але основним мінусом існуючих систем все ще є висока ціна на такі системі, необхідність високоточніх лідарів та камер.

1.4 Компанія CommaAi

Одним з рішень автоматизації транспортного засобу є OpenSource система – Openpilot від компанії CommaAi (рисунок 1.14). Openpilot — це напів-автоматизована система водіння для транспортних засобів (рисунок 1.15) з відкритим програмним забезпеченням, яка була розроблена компанією CommaAi.

Рисунок 1.14 – Система керування автомобілем від компанії CommaAi [9]

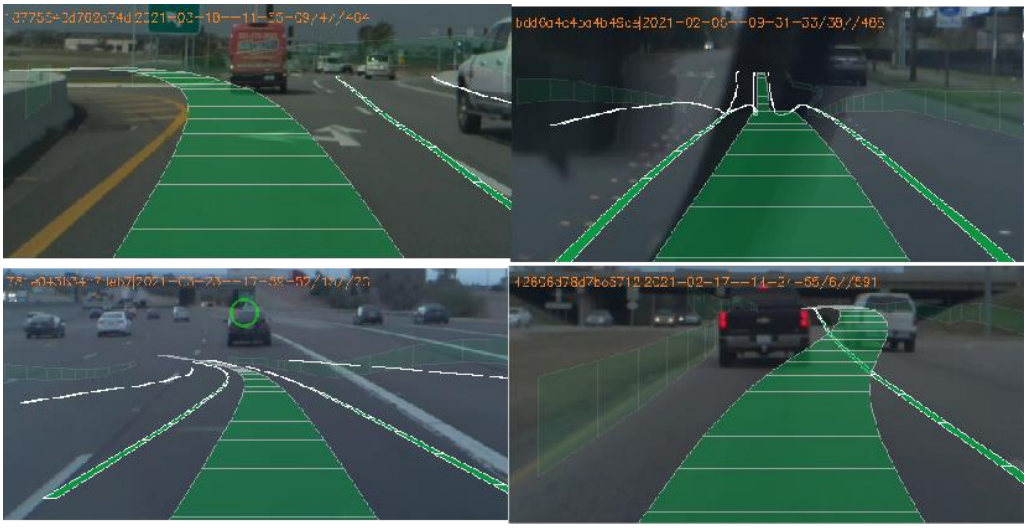

Рисунок 1.15 – Приклад розмітки дорожнього покриття

Система Openpilot працює як заміна для системи допомоги водію, оригінального виробника обладнання, з метою вдосконалення зорового сприйняття людиною дорожньої обстановки задля зниження навантаження на водія та зменшення вірогідності аварійних ситуацій. Ця система дозволяє користувачам модифікувати свій автомобіль використовуючи збільшену обчислювальну потужність, покращені датчики та постійно оновлювані функції допомоги водіям, які вдосконалюються за допомогою даних що подаються користувачами.

Принцип дії системи Openpilot обснований на зчитуванні даних з різних датчиків (камери, IMU, датчики кута повороту керма, GPS приймачі та інші) (рисунок 1.16), після отримання дані оброблюються, фільтруються та передаються у якості вхідних даних у велику нейронну мережу. В середені нейронної мережі дані перетворюються в команди керуванням системами транспортного засобу.

Рисунок 1.16 – Перша версія автопілоту

Сам Openpilot оснований на 5 основних бібліотеках:

– opendbc : Бібліотека для інтерпретації трафіку шини CAN;

– panda : Інтерфейс CAN шини автомобіля;

– cereal : специфікація обміну повідомленнями для роботизованих систем;

– laika : бібліотека обробки GNSS сигналів;

– rednose : бібліотека фільтрів Калмана, використувується для візуальної одометрії та системи SLAM.

Особливості даного рішення:

– автоматичне центрування у смузі руху. openpilot використовує машинне навчання , навчене за діями водія користувача, щоб визначити найбезпечніший шлях на дорозі. На дорогах без розмітки, алгоритм шукає найбільш безпечний шлях;

– адаптивний круїз контроль. openpilot підтримує безпечну дистанцію від транспортного засобу попереду. Він здатний керувати рухом в заторах без втручання користувача. Він використовує дані про кривизну доріг і обмеження швидкості OpenStreetMap, щоб дозволити сповільнювати на крутих поворотах і встановлювати бажану швидкість автомобіля на поточне обмеження швидкості;

– моніторинг стану водія. openpilot розпізнає обличчя водія; якщо водій відволікається, openpilot попереджає водія. Якщо водій відволікається більше ніж на шість секунд, openpilot уповільнює автомобіль до зупинки та попереджає користувача звуковим сигналом;

– допомога при зміні смуги руху. openpilot використовує модель для зміни смуги руху, коли користувач включає покажчик повороту, для підтвердження зміни смуги потрібно штовхнути кермо. openpilot також взаємодіє з монітором сліпих зон на певних марках і моделях, щоб блокувати зміну смуги руху в разі виявлення автомобіля монітором сліпих зон;

– оновлення програмного забезпечення та алгоритмів роботи. openpilot отримує оновлення програмного забезпечення по бездротовій мережі через Wi–Fi або стільниковий зв'язок.

1.5 Висновки розділу

У першому розділі були розглянуті аналогічні рішення на ринку, в результаті розгляду яких можна зробити висновок, що вони мають високу вартість за рахунок використаних датчиків, та низьку гнучкість в аспекті розробки нових систем. Було розглянуто технічне завдання для виконання даної атестаційної роботи, було складено список вимог, яким повинен відповідати результат виконання даної роботи.

Виходячи з цього, основними задачами, що мають бути виконані для побудування такої системи є:

– розробка критеріїв оцінки точності системи;

– затвердження чинників керування, які будуть вираховуватися;

– розробка процесів та операцій, що будуть складати процедуру розробки алгоритму кервуання;

– розробка компонентів системи та складання моделі;

– експериментальне дослідження з метою перевірки розроблених алгоритмів.

2 ТЕХНІЧНІ ЗАСОБИ РЕАЛІЗАЦІЇ ПРОЕКТУ

Системи комп’ютерного зору представляють собою комплекси, які складаються з пристроїв для отримання зображення (відеокамери), пристроїв для обробки отриманої графічної інформації та програм для контролю і адміністрування системою. Сама технологія комп’ютерного зору відноситься до методології створення штучних систем, які отримують і аналізують інформацію з зображень. Зображення може потрапляти в систему обробки безліччю шляхів, наприклад відеопослідовність, тривимірні данні з медичних та промислових сканерів, зображення з фотокамер та інші.

2.1 Загальний огляд систем комп’ютерного зору

Типові рішення для систем комп’ютерного зору складаються з наступних компонентів:

– одної або декількох цифрових або аналогових камер з підібраною під задачу оптикою;

– програмному забезпеченню для обробки зображення з камер у цифровий вигляд;

– комплекс для аналізу та обробки даних (наприклад ПК або сервер);

– програмне забезпечення машинного зору;

– інформаційні канали для виведення даних та звітів про отримані результати.

Використовуючи підхід багаторівневої системи аналізу та обробки зображення можливо створювати системи для повсякденного використання:
В системи відеоспостереження;

– системи контролю за транспортом;

– системи управління автоматизованими процесами (автономні транспортні засоби, промислові роботи, автономні промислові лінії);

– системи для моделювання і створення комп'ютерних моделей фізичних об'єктів та навколишнього середовища;

– системи для взаємодії обладнання з персоналом;

– системи організації інформації для індексації баз даних та зображень.

Варто відзначити що область комп’ютерного зору є досить новою в сучасному розумінні. Комп’ютерний зір саме як область почав розвиватися в 70-х роках 20 століття, хоча напрацювання в цієї сфері були і раніше. Раніше основною проблемою малого поширення комп’ютерного зору була відсутність потужних комп’ютерів які були здатні обробляти великі масиви інформації. Проте самі дослідження у сфері комп’ютерного зору переважно проводилися в рамках інших наукових досліджень, тому важко було сформувати конкретні проблеми даної галузі. Досить багато рішень існували у вигляді практичних напрацювань, та не були втілені в реальність. Але з іншої сторони все більше рішень і розроблених технологій наразі знайшли своє застосування в сучасних комерційних продуктах і широко використовуються на практиці. В практичному використанні даних технологій ми можемо помітити використання вузькоспеціалізованих методів програмування комп’ютерів для кожної задачі, які гуртуються на досить узагальнених принципах комп’ютерного зору.

В якості прикладів використання комп’ютерного зору в сучасні часи можна виділити також розпізнавання образів та об’єктів з застосуванням навчання, яке вже можна віднести до області штучного інтелекту. Виходячи з цього можна віднести комп’ютерний зір до предметної області штучного інтелекту.

При розробці систем комп’ютерного зору є також дуже важливим питання оптимізації алгоритмів і роботи комп’ютера під час виконання і обробки задач комп’ютерного зору. Тому для систем комп’ютерного зору створюються нові методи програмної та апаратної оптимізації, ціллю яких є збільшення швидкості комп’ютера та зменшити використання ним ресурсів,

що є надзвичайно важливим для швидкої та ефективної роботи досліджуваної дисципліни.

2.2 Застосування систем комп’ютерного зору

Комп’ютерний зір в сучасному розумінні має переважно промислову специфіку, у сферах роботизації, оптичної перевірки, сортування і контролю. Для забезпечення алгоритму інформацією використовуються різноманітні датчики та камери. Тому при використанні подібних систем у промисловості вирішуються та оптимізуються багато проблем і процесів. Наприклад с допомогою алгоритму розпізнавання і відеокамери можливо дуже ефективно контролювати якість на виробництві, коли необхідно автоматично перевіряти кінцевий продукт, напівфабрикат чи сировину на наявність браку та дефектів. Комп’ютерний зір в рамках виробництва також може бути застосований для роботи автоматичних маніпуляторів в цілях вимірювання та пошуку орієнтації роботів що переміщують вироби та предмети (рисунок 2.1).

Рисунок 2.1 – Аналіз продуктів на виробничій лінії

Військова справа в сучасні часи є однією з передових сфер застосування комп’ютерного зору. Сучасні наробки у сфері широко використовуються для

аналізу і визначення дислокації ворожих сил, воєнної техніки та військових. Свою ефективність комп’ютерний зір показує при запуску ракет, систем нічного та теплового зору, систем автоматичного наведення та слідкування за ціллю (рисунок 2.2).

Рисунок 2.2 – Система спостереження за ціллю

Сучасні наробки у сфері також широко використовуються для керування безпілотними апаратами [8] (рисунок 2.3). Можливість систем аналізу керування безпілотними літальними апаратами дає можливість отримувати максимальну кількість інформації про ворогів та поле бою ще до початку бойових дій, що може бути використано для прийняття важливих стратегічних рішень.

Рисунок 2.3 – Безпілотний летальний апарат

Комп’ютерний зір в сучасному розумінні має переважно промислову специфіку, у сферах роботизації, оптичної перевірки, сортування і контролю транспортних засобів. Для забезпечення алгоритму інформацією використовуються різноманітні датчики та камери. Тому при використанні подібних систем у керуванні транспортними засобами вирішуються та оптимізуються багато проблем і процесів. Наприклад с допомогою алгоритму розпізнавання і відеокамери можливо дуже ефективно контролювати розміщення складських роботів відносно перешкод (рисунок 2.4)[10].

Рисунок 2.4 – Роботи для роботи на складі

Орієнтування у просторі даних роботів реалізовано за рахунок двох камер (рисунок 2.5), які реалізовують розпізнавання реперних точок зі штрих-кодами товарів та платі обробки інформації на основне FPGA.

Рисунок 2.5 – Сенсори для аналізу маркерів

Автомобільна промисловість з покращенням алгоритмів розпізнавання також почала впроваджувати систему автопілоту у транспортні засоби. При чому в залежності від потужності та типу алгоритмів для обробки зображення система автопілоту може бути як повністю безпілотною, так і в вигляді помічника водія в залежності від рівня автономності. В залежності від можливостей розробленої системи керування можливо виділити 6 рівнів автономності за класифікацією SAE International:

– 0 рівень: повністю ручне керування з можливістю попередження про небезпечні ситуації на дорозі;

– 1 рівень це адаптивний круїз-контроль та контроль рулювання або гальмування транспортного засобу;

– 2 рівень може контролювати і рулювання, і гальмування, але, як і рівнем раніше, лише за певних обставин: наприклад, на шосе водієві доводиться втручатися;

– 3 рівень автономності не потребує постійного втручання водія, але водій все ще повинен брати керування на себе в важких умовах;

– система з 4 рівнем бере на себе повний контроль, дозволяючи водієві відпочити, але тільки якщо для цього створені всі умови, наприклад, є високо деталізовані тривимірні карти, щоб система з точністю до кількох сантиметрів знала, де знаходиться. Більшість розробників намагаються створити системи саме цього рівня;

– рівень 5 передбачає повну автоматизацію, у цій гіпотетичній ситуації перед кріслом водія навіть немає керма.

За допомогою систем комп'ютерного зору автомобілі можуть визначати своє місце положення, аналізувати дорожню ситуацію поруч з собою і уникати потенційних аварійних ситуацій. Окрім використання систем у роботі автомобіля їх також використовують при його виробництві, задля збільшення швидкості і якості виробництва (рисунок 2.6).

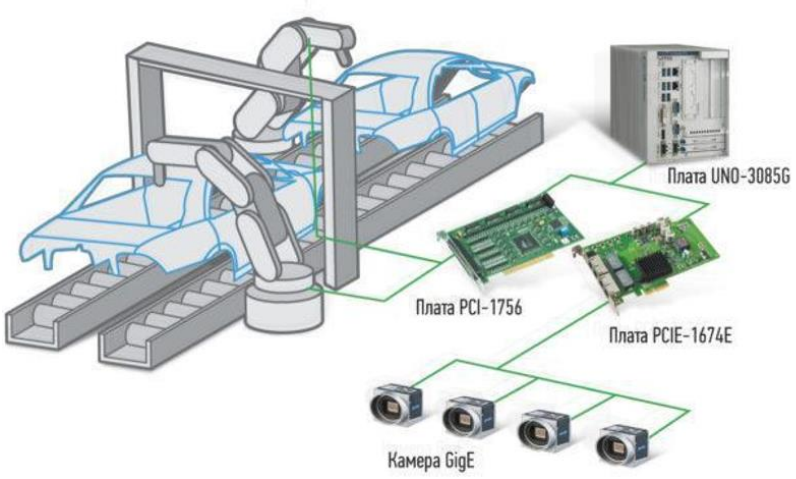

Рисунок 2.6 – Аналіз виробів на виробничій лінії

Окрім вже описаного, комп'ютерний зір може використовуватися в цілях пошуку людей в рятувальних операціях, автоматичного керування безпілотними роботами, в сферах спостереження, тощо.

Основною задачею всіх алгоритмів комп'ютерного зору є визначення знаходження об'єкту, процесу чи дії на зображенні чи відео. У порівнянні з людським зором, який легко справляється з даною задачею, комп'ютеру, в деяких ситуаціях, не завжди легко визначити окремі об'єкти на зображенні.

На основі широкого поширення систем комп’ютерного зору в сучасному світі можна зробити висновки про їх користь, що підтверджується графіком з використання систем комп’ютерного зору в різні часи (рисунок 2.7).

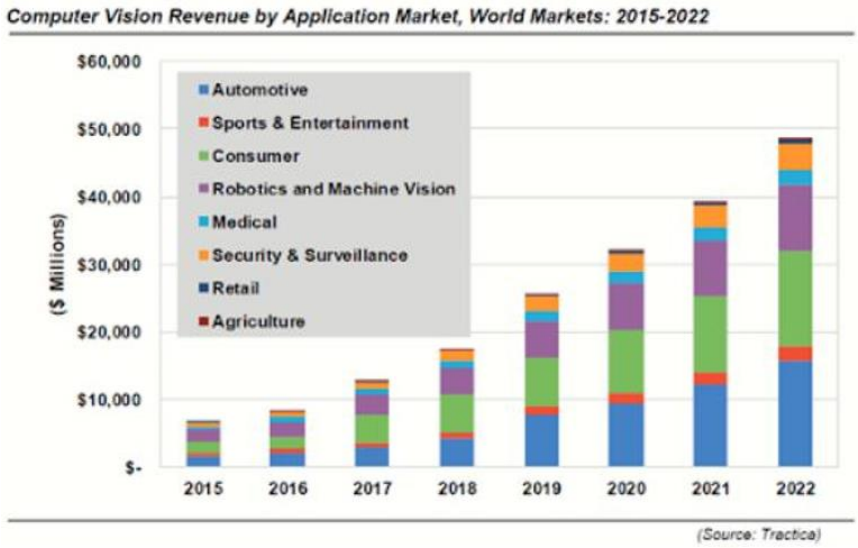

Рисунок 2.7 – Аналіз виробів на виробничій лінії

2.3 Типи сенсорів для отримання інформації про оточення

Поняття систем керування на основі аналізу оточення в сучасному розумінні виходить за рамки обробки зображень, так як дозволяє отримувати відповідну інформацію із зображення і приймати рішення на основі обробленої інформації. Іншими словами комп’ютерний зір при своєму максимальному розвитку намагається сприймати інформацію з фото або відео так, як це робить людське око. Тому в сучасних системах керування на основі комп’ютерного зору зображення проходить через наступні кроки: отримання зображення з фото або відеокамери, обробка зображення (зміна яскравості, контрасту, експозиції), отримання необхідної інформації, прийняття рішень для керування. В саму систему відеодані можуть бути представлені у вигляді багатьох форм, таких як відеозображення, зображення з масивів камер, лідарів, або тривимірних моделей з різноманітних сканерів.

Як можна побачити, функції комп’ютерного зору досить різноманітні, проте серед них можна визначити певні спільні моменти для всіх систем.

Виробники оснащують сучасні автомобілі широким набором розширених функцій контролю та датчиків. Системи попередження та уникнення зіткнень, монітори сліпих зон, система допомоги при утриманні в смузі, попередження про виїзд із смуги руху та адаптивний круїз-контроль – це приклади встановлених функцій, які допомагають водіям та автоматизують певні завдання водіння, роблячи водіння безпечнішим і легшим. Лідари, радари, ультразвукові датчики та камери мають свої власні нішеві набори переваг і недоліків. Високо або повністю автономні транспортні засоби, як правило, використовують кілька сенсорних технологій для створення точної мапи оточення транспортного засобу на великі та короткі відстані в різних погодних умовах і умовах освітлення. Крім того, що технології доповнюють одна одну, важливо також мати достатнє перекриття, щоб збільшити резервування та підвищити безпеку. Sensor fusion - це концепція використання декількох сенсорних технологій для створення точної та надійної карти навколишнього середовища навколо автомобіля. Ультразвукові хвилі страждають від сильного загасання в повітрі більше ніж на кілька метрів; тому ультразвукові датчики в основному використовуються для виявлення об'єктів на малому радіусі.

Автоматичне керування транспортним засобом об’єднує в собі велику кількість методів керування та аналізу оточення за рахунок сенсорів (камери, лідари, радари, парктроніки) (рисунок 2.8).

Рисунок 2.8 – Приклади рішень радару та 3D камери

Одним із методів який використовується у автоматичному керуванні є виявлення та утримання смуги руху – важливий метод, необхідний для створення розумного автомобіля. Важливо контролювати рух автомобіля у смузі, на якій рухається автомобіль. Основні методики роботи даної системи можуть основуватися на алгоритмах Трансформації Хафа для вилучення ознак, SVM для машинного навчання разом із виділенням лінії смуги руху, її краю та області дозволенного переміщення транспортного засобу. Точність методів аналізу відеозображення на основі цього методу може досягати 90 %. Але використаний алгоритм має низьку точність при зміні освітлення, що не є універсальнім для керування транспортним засобом. Тому використання алгоритмів на базі нейронних мереж YOLO і CNN для розпізнання смогу руху набули більшу популярність, в порівняні з першим методом. Система на основі нейронної мережі була здатна виявляти смуги, об’єкти та надавати пропозиції водієві з точністю до 99 %.

Важливими даними у роботі алгоритму автоматичного керування транспортним засобом є данні відстані до оточуючих об’єктів, та їх швидкості руху. Задля отримання таких даних використовуються різноманітні системи (лідари, радари, стерео камери, парктроніки). Кожен із засобів виявлення дистанції має як свої переваги, так і недоліки.

2.4 Лідари

Лідар (рисунок 2.9) і радар мають широкий спектр загальних і додаткових функцій, які можуть розпізнавати оточення, а також вимірювати швидкість об’єкта.

Рисунок 2.9 – Лідар компанії Velodyne [11]

Технології вимірювання даних інструментів є схожими, тому для аналізу існуючих засобів можемо провести порівняння по деяким критеріям:

– критерії діапазону. Лідар і радар можуть виявляти об'єкти на відстанях від кількох метрів до понад 200 м. Лідар має труднощі з виявленням об’єктів на близькій відстані. Радар може виявляти об'єкти від менше метра до більше 200 м;

– роздільна здатність. Завдяки здатності лідару колімувати лазерне світло і його короткому довжині хвилі від 905 нм до 1550 нм, за допомогою лідару можливою досягти роздільну здатність інфрачервоного світла порядку 0,1 градуса. Це дозволяє отримувати тривимірну характеристику об’єктів у сцені з надзвичайно високою роздільною здатністю без значної обробки даних. З іншого боку, довжина хвилі радара (4 мм для 77 ГГц) не може розрізняти невеликі особливості об’єктів, особливо коли відстань до них є високою;

– поле зору (FOV). Твердотільні лідари і радари мають чудовий горизонтальний FOV, тоді як механічні системи лідарів з їх обертанням на 360 градусів мають найширший FOV з усіх передових технологій допомоги водієві ADAS. Лідар має кращий вертикальний FOV, ніж радар. Лідар також має перевагу над радаром у кутовій роздільній здатності (як за азимутом, так і за висотою), що є однією з ключових властивостей, необхідних для кращої класифікації об’єктів;

– погодні умови. Однією з найбільших переваг радарів є їхня стійкість до дощу, туману та снігу. За таких погодних умов продуктивність лідару зазвичай погіршується. Використання ІЧ-хвилі 1550 нм допомагає лідарам досягти кращої продуктивності в несприятливих погодних умовах;

– вартість і розмір. Радарні системи стали масовими в останні роки, що робить їх дуже компактними та доступними. Оскільки лідар набув популярності, його вартість різко впала, а ціни впали з приблизно 50000 доларів США до 10000 доларів США. Деякі експерти прогнозують, що вартість модуля лідару впаде до 200 доларів США до 2022 року.

Принцип роботи лідару полягає у випроміненні імпульсів лазерного випромінювання зі швидкістю до 150000 імпульсів в секунду. В більшості лідарів використовують короткі імпульси лазерного випромінювання, з високою точністю фіксуючи моменти їх передачі і прийому відгуків (відбитих сигналів), щоб обчислити відстані до об'єктів в навколишньому просторі. Після об'єднання серії таких вимірювань з інформацією про місцезнаходження і орієнтації транспортного засобу, створюється результуюча тривимірна сцена цікавить області простору (рисунок 2.10).

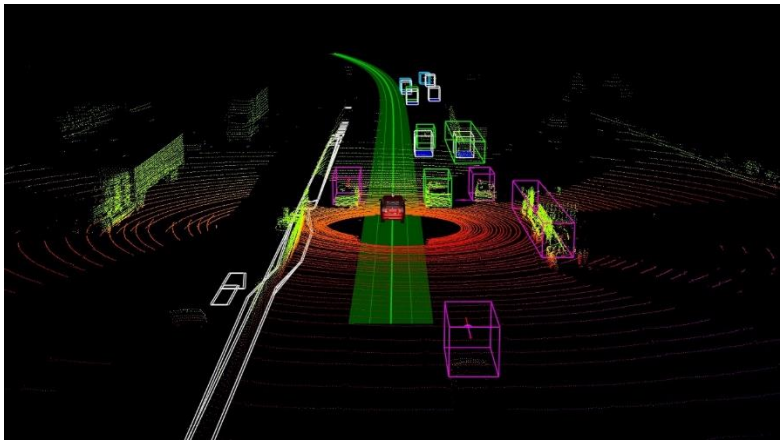

Рисунок 2.10 – Хмара точок зібрана лідаром

Найчастіше ця сцена зберігається у вигляді масиву координат (x, y, z), званого хмарою точок. При роботі точний час передачі і прийому лазерного променя зберігається в ОЗУ обчислювальної системи та використовуються для розрахунку часу, проведеного світлом в дорозі, і, отже, відстані до об'єкта, який відбив промінь. В лідарах зазвичай використовується один з двох режимів, що визначають метод вимірювання відстані: режим безперервної хвилі або імпульсний режим. У системах з імпульсною модуляцією, також відомих як час прогонові системи, лазером випромінюються поодинокі світлові імпульси з високою частотою проходження. Вимірюється час, що минув с моменту випромінювання імпульсного сигналу до моменту потрапляння відбитого променю в приймач. Відстань до точки поверхні об'єкта, в якій сталося відображення лазерного променю, може бути обчислено за формулою (2.1):

$$D = 0{,}5 \cdot c \cdot t, \tag{2.1}$$

де D – шукана відстань до точки об’єкту;

c – швидкість світла;

t – повне час проходження світлом шляху до точки відображення і назад.

У системах з безперервним випроміненням лазер випромінює безперервний сигнал, до якого потім застосовується амплітудна модуляція

(наприклад синусоїдальна). В цьому випадку час проходження світлом повного шляху від передавача до приймача буде прямо пропорційно зсуву фаз в ізлучённом і прийнятому сигналах. Для зниження невизначеності може бути використана багатотонова синусоїдальна модуляція. Також в системах з безперервною хвилею використовується альтернативний метод – з лінійною частотною модуляцією. У таких системах переданий і прийнятий сигнали змішуються, а для демодуляції і отримання інформації, що міститься в несучій частоті, використовується когерентний приймач. Потрібно відзначити, що в рівняннях визначення відстані до об’єкту передбачається, що детектор протягом часу вимірювання стаціонарен. Для випадків з рухомим детектором необхідно буде внести в рівняння відповідні поправки.

2.5 Стереокамери

Інший підхід аналізу відстані до об’єктів використовує принципи стереокамер. У традиційному стереобаченні дві камери, зміщені горизонтально одна від одної, використовуються для отримання двох різних поглядів на сцену, подібно до людського бінокулярного зору. Порівнюючи ці два зображення, можна отримати інформацію про відносну глибину у вигляді карти невідповідності, яка кодує різницю горизонтальних координат відповідних точок зображення. Значення в цій карті диспропорції обернено пропорційні глибині сцени у відповідному місці пікселя (рисунок 2.11).

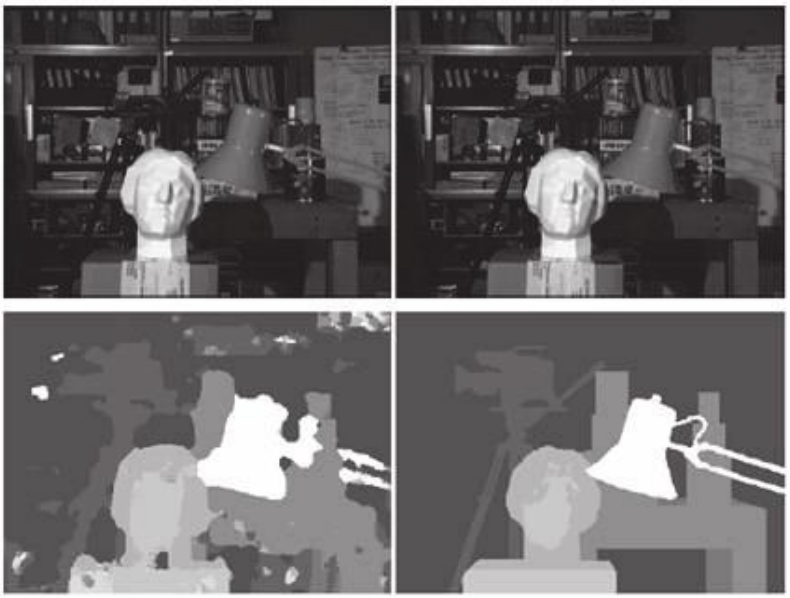

Рисунок 2.11 – Зображення глибини отримане з двох камер

У системі комп’ютерного зору потрібно виконати кілька етапів попередньої обробки:

– зображення має бути неспотвореним, тому необхідно усунути його дисторсію. Це гарантує, що спостережуване зображення відповідає проекції ідеальної камери-обскури;

– зображення повинні відображатися на спільну площину, щоб дозволити порівняння пар зображень, процесс відомий як ректифікація зображення;

– інформація міри, який порівнює два зображення, зведена до мінімуму. Це дає найкращу оцінку положення об’єктів на двох зображеннях і створює мапу невідповідності;

– за бажанням, отримана карта диспропорції проектується в 3D хмару точок. Використовуючи параметри камер, хмару точок можна обчислити таким чином, щоб вона забезпечувала вимірювання у відомому масштабі.

Але використання системи стерео зору потребує калібровки камер між собою задля досягнення високої точності роботи. Камери різні, і неможливо ідеально вирівняти їх, щоб створити ідеальні карти глибини. Таким чином, метод програмного калібрування використовується двома камерами, які

роблять кілька фотографій об’єкта. Найчастіше використовується друкована шахову дошку. Метод программного калібрування аналізує ці фотографії за допомогою виявлення країв, щоб знайти точки перетину на зображенні шахової дошки, щоб створити загальну область огляду двох камер. Після збору та збереження зображень виконується калібрування. Калібрування працює над визначенням загальної області перегляду камер шляхом завантаження всіх пар, які знімають і зберігають із попереднього процесу, а також обчислюють матриці корекції. Спочатку алгоритм калібрування намагається знайти шахову дошку на фотографії. Потім він виявляє точки перетину чорних і білих квадратів на шаховій дошці, щоб знайти відповідні точки в кожній камері, а потім будує калібровані зображення (рисунок 2.12).

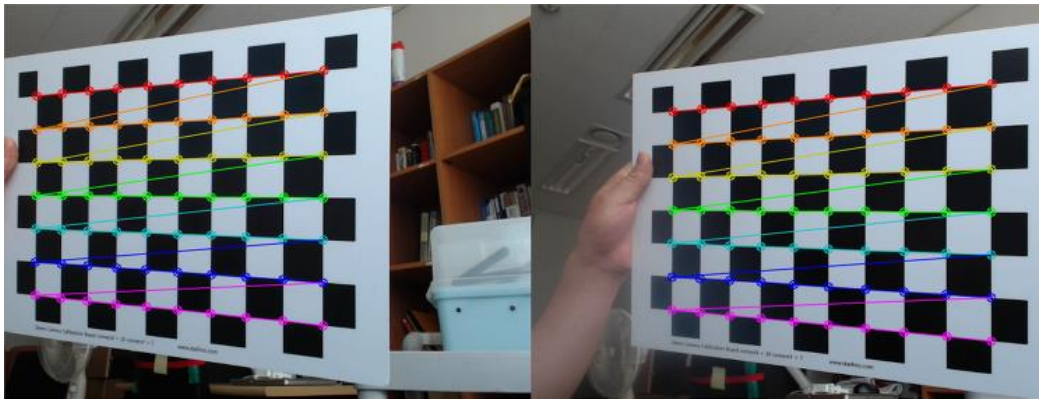

Рисунок 2.12 – Зображення глибини отримане з двох камер

Цей крок калібрування відноситься до якості калібрування, що призводить до хорошого або поганого результату карти невідповідності. Налаштування карти глибини: об’єднує попередні кроки, завантажуючи зображення, які були збережені разом із результатами калібрування. Потім він представляє карту глибини та інтерфейс для точного налаштування. Під час налаштування карти глибини відеокадри збираються з кожної камери після застосування до неї результатів калібрування, щоб було легко встановити відповідні параметри карти диспропорцій відповідно до результатів калібрування, що допомагає точнішій карті диспропорцій.

2.6 Гібридні камери глибини

Окрім використання стереопари задля визначення глибини зображення можуть використовуватися гібридні системи, які складаються з однієї або декількох камер ІК діапазону, та проектора сітки крапок.

Однією з представників гібридних сенсорів глибини можна виділити камеру Intel RealSense D435i (рисунок 2.13) [12].

Рисунок 2.13 – Камера глибини Intel RealSense D435i

Камера глибини Intel RealSense серії D435i використовує стереобачення для обчислення глибини. Реалізація стереобачення складається з лівого, правого і додаткового інфрачервоного проектора. Інфрачервоний проектор проектує невидимий статичний ІЧ-шаблон для покращення точності глибини в сценах з низькою текстурою.

Принцип роботи даної камера полягає у реалізації стереозору, який моделює зір людини за допомогою двох або більше камер для захоплення 2D-зображень під різними кутами. Ці зображення називають зображеннями диспропорції. Інформацію про глибину можна розрахувати на основі невідповідності та зсуву об’єктів на двох зображеннях. Однак цей метод передбачає складні обчислення, що спричиняє повільну швидкість вимірювання.

Метод проекції структурованого малюнку набуває все більшої уваги завдяки його високій швидкості та точності вимірювання глибини. У порівнянні зі стереобаченням, структуроване світло потребує наявності проектора світлового паттерну (рисунок 2.14). Проектор випромінює на об’єкти зображення, зокрема одноточковий візерунок, однолінійний візерунок та кодований малюнок. Відображення візерунка фіксується камерою. Інформацію про глибину можна розрахувати на основі тріангуляції.

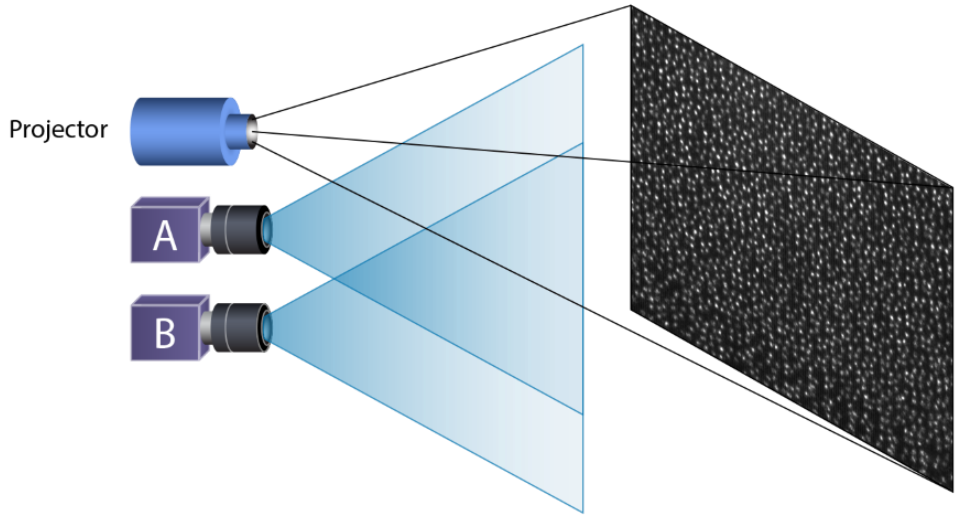

Рисунок 2.14 – Схема проекції паттерну

Інтерференція видимого світла та калібрування проектора є двома ключовими проблемами в методі структурованого освітлення. У цьому методі проектори важко калібрувати, оскільки вони не мають малий кут випромінення на малій дистанції. Деякі дослідження розглядали проектор як зворотну камеру, тоді як інші зосереджені на оцінці рівняння площин світлових смуг, що випромінюються проектором. Однак обидва питання залежать від визначення параметрів камери. Таким чином, основним недоліком методу структурованого освітлення є накопичення помилок у калібруванні проектора та камери.

Щоб вирішити ці проблеми, спочатку використовується інфрачервоне структуроване світло як джерело паттерну (рисунок 2.15), оскільки довжина хвилі інфрачервоного світла відрізняється від довжини хвилі видимого світла. По-друге розроблюється простий і ефективний структурований світловий шаблон, а також відповідний алгоритм кодування та декодування.

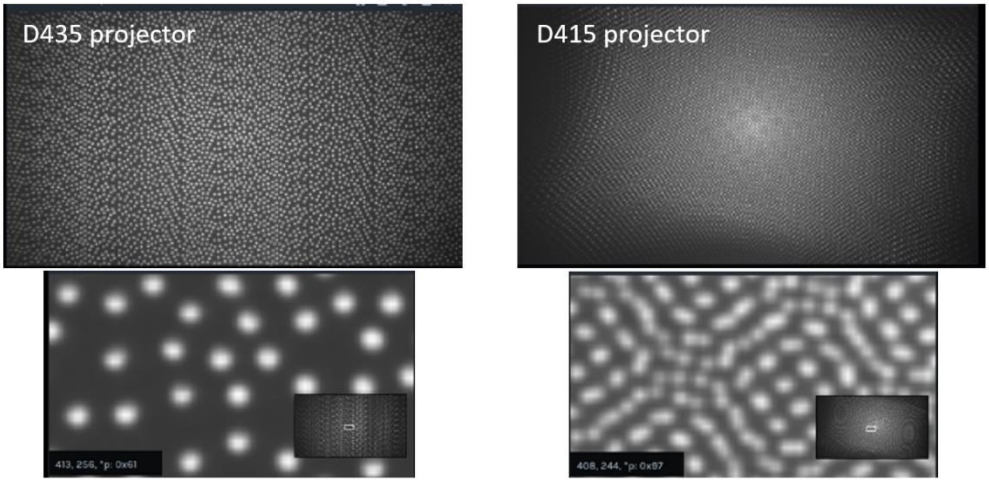

Рисунок 2.15 – Різновиди паттернів

Відповідно до різних типів проектованих світлових шаблонів, методи структурованого світла можна класифікувати на одноточковий шаблон, однолінійний шаблон і структуроване світло. Метод одноточкового шаблону отримує інформацію про глибину шляхом точкового сканування всього зображення. Таким чином, складність обчислень різко зростає зі збільшенням розмірів вимірюваного об'єкта. Цей метод демонструє практичність вимірювання глибини на основі методу структурованого світла.

У порівнянні з методом структурованого світла з одноточковим шаблоном, метод структурованого світла з одноточковим візерунком може отримати інформацію про глибину лише шляхом сканування одновимірного об’єкта. Таким чином, час обробки і складність обчислень значно зменшуються. Однак однорядковий метод вимагає багаторазового сканування об’єкта для отримання інформації про глибину.

Для скорочення часу вимірювання запропоновано метод структурованого світла з кодованим малюнком (рисунок 2.16). Цей метод дозволяє отримати дані про глибину за один кадр. Крім того, останнім часом метод активно вивчався через його високу точність.

Рисунок 2.16 – Результат отримання паттерну ІЧ камерою

Схематичний вигляд системи камер і апаратної структури показано на рисунку 2.17. Система складається з інфрачервоного лазерного випромінювача, дифракційної решітки та ПЗС-камери. Для підвищення точності може використовуватися дві і більше камери. Роздільна здатність ПЗС-камери становить 640x480. Лазерний діод випромінює один промінь і розбивається на декілька променів за допомогою дифракційної решітки. ПЗС-камера фіксує шаблон і співвідносить його з еталонним шаблоном. Еталонний шаблон розташований на площині відомої глибини і зберігається в пам'яті комп'ютера.

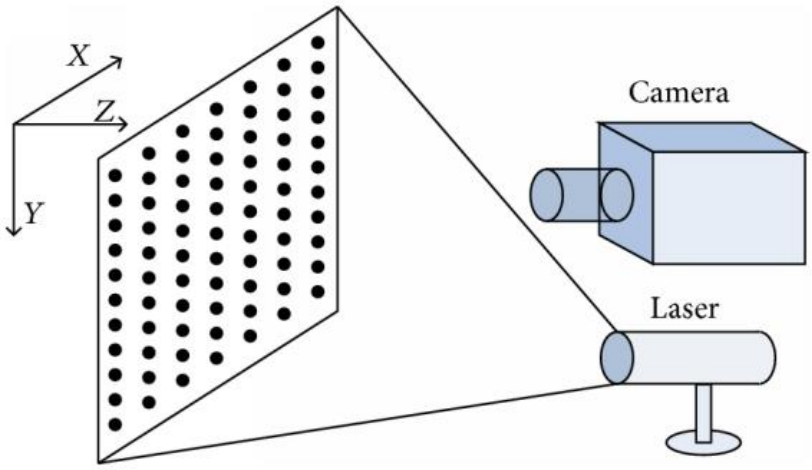

Рисунок 2.17 – Система отримання глибини об’єкту на основі однієї камери

Лазер випромінює точковий матричний малюнок, спроектований на сцену. ПЗС-камера фіксує шаблон і співвідносить його з еталонним шаблоном. Потік алгоритму вимірювання глибини проілюстрований на рисунку 2.18. Цей потік алгоритму включає розрахунок зміщення пікселя, лінійну підгонку параметрів системи, вимірювання глибини та тривимірну реконструкцію.

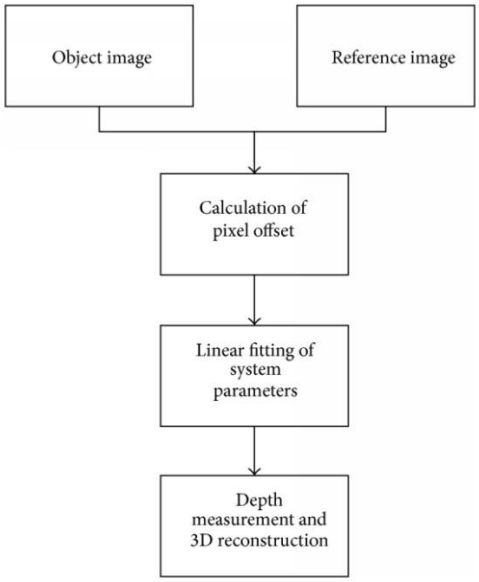

Рисунок 2.18 – Хід роботи алгоритму аналізу глибини

Розподіл інтенсивності світла крапок на необробленому зображенні нерівномірний, а зображення темне з поганим контрастом. Щоб вирішити ці

проблеми, використовується алгоритм попередньої обробки зображення, включаючи зменшення шуму, покращення та бінаризацію зображення.

В результаті обробки отриманого зображення отримано зміщення кожного пікселя між відповідними точками, на основі цих даних проводиться обчислення відстані до кожної точки на зображенні. Після цього ми можемо отримати тривимірні координати кожного з лазерних спеклів на зображенні об’єкта. 3D-сцену можна реконструювати на основі тріангуляризації Делоне (рисунок 2.19).

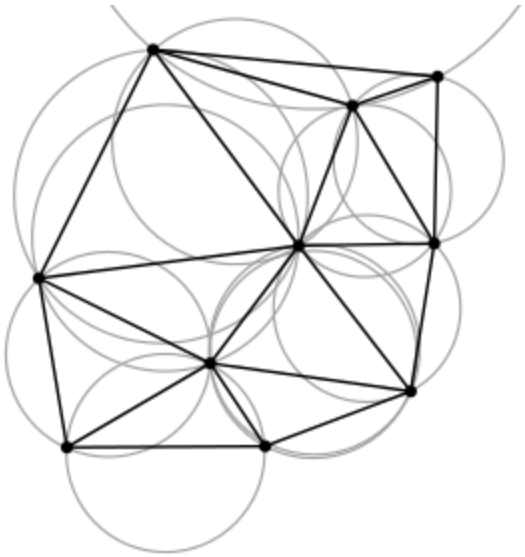

Рисунок 2.19 – Візуалізація тріангуляризації Делоне

Тріангуляція Делоне для множини точок P на площині — це така тріангуляція DT(P), що жодна точка множини P не знаходиться всередині описаних довкола трикутників кіл в множині DT(P). Тріангуляція Делоне дозволяє якомога зменшити кількість малих кутів. Цей спосіб тріангуляції був винайдений Борисом Делоне в 1934 році.

Базуючись на визначенні Делоне, коло описане навколо трикутника утворене трьома точками з вихідної множини точок називається пустим, якщо воно не містить вершин трикутника інших ніж ті три, що його задають (інші точки допускаються тільки на периметрі кола, але не всередині).

Умова Делоне стверджує, що мережа трикутників є тріангуляцією Делоне, якщо всі описані кола трикутників пусті. Це є початкове визначення для двовимірного простору. Його можна використовувати для тривимірного простору, якщо використовувати описані сфери замість описаних кіл.

Для множини точок на одній лінії тріангуляції Делоне не існує (фактично, поняття тріангуляції для такого випадку невизначене). Для чотирьох точок на одному колі (наприклад прямокутник) тріангуляція Делоне має два випадки, тобто можна розділити цей чотирикутник двома способами, які задовольняють умови Делоне.

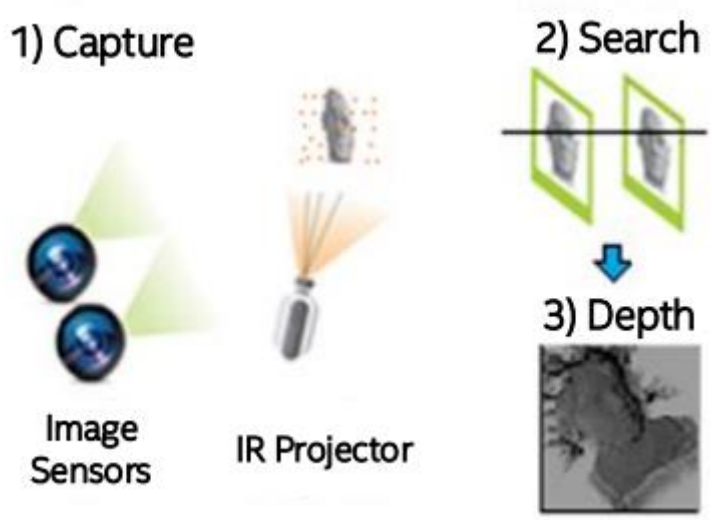

Рисунок 2.20 – Система отримання глибини об’єкту на основі двох камер

Ліва і правий камери (рисунок 2.20) знімають сцену і надсилають дані зображення в процесор обробки глибини зображення, який обчислює значення глибини для кожного пікселя зображення шляхом співвіднесення точок на лівому зображенні з правим зображенням і через зсув між точкою на зображенні. Значення пікселів глибини обробляються для створення кадру глибини. Об’єднання кадрів глибини створюють повноцінний відео потік з виміром глибини зображення.

Отримані завдяки сенсорам данні потребують подальшої обробки, задля вилучення з них умов та правил керування транспортним засобом в залежності

від задачі. Тому для обробки та аналізу даних використовується велика кількість алгоритмів. В результаті їх роботи можна отримати данні про положення у просторі транспортного засобу, о відстані та положенню навколишніх об’єктів, о шляхах руху та інші данні необхідні для автоматичного керування транспортним засобом.

2.7 Алгоритми обробки отриманої інформації

Одним з алгоритмів який активно використовується у автоматичному керуванні транспортним засобом є алгоритм SLAM [13].

SLAM – це метод, який використовується для автономних транспортних засобів, який дозволяє створювати карту та локалізувати транспортний засіб на цій карті одночасно. Алгоритми SLAM дозволяють транспортному засобу відображати та орієнтуватися у невідомих середовищах. Інженери використовують інформацію карти для виконання таких завдань, як планування шляху та уникнення перешкод.

SLAM був предметом технічних досліджень протягом багатьох років. Але завдяки значному покращенню швидкості обробки інформації комп’ютерами та доступності недорогих датчиків, таких як камери та лазерні далекоміри, SLAM тепер використовується для практичного застосування у все більшій кількості областей.

Розглянемо домашній робот-пилосос. Без SLAM він буде просто випадковим чином рухатися по кімнаті і, можливо, не зможе очистити всю поверхню підлоги. Крім того, такий підхід використовує надмірну потужність, тому батарея буде розряджатися швидше. З іншого боку, роботи з SLAM можуть використовувати таку інформацію, як кількість обертів колеса та дані з камер та інших датчиків зображення, щоб визначити своє місцеположення в даний час. Це називається локалізацією. Робот також може одночасно використовувати камеру та інші датчики, щоб створити карту перешкод у

своєму оточенні та уникнути очищення однієї і тієї ж області двічі (рисунок 2.21).

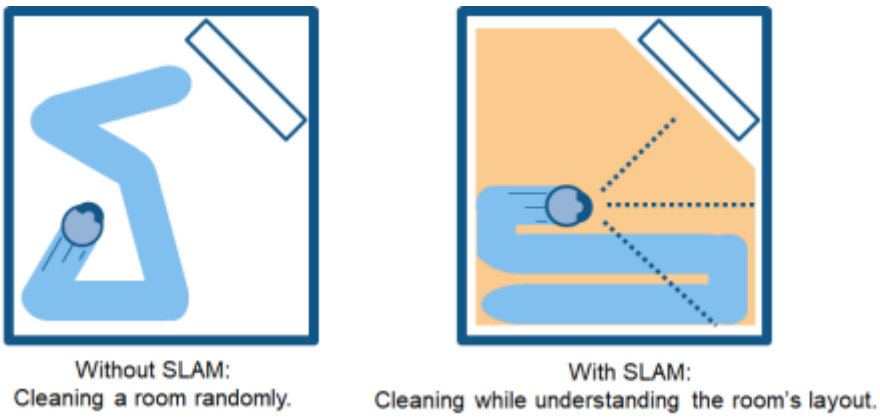

Рисунок 2.21 – Робот пилосос використовуючий SLAM

SLAM корисний у багатьох інших програмах, таких як навігація по парку мобільних роботів, для знаходження полиць на складі, паркування самокерованого автомобіля на порожньому місці або доставка посилок за допомогою дрона в невідомому середовищі.

Загалом, існує два типи технологічних компонентів, які використовуються для роботи алгоритму SLAM. Перший тип – це обробка сигналів датчиків, включаючи фронтальну обробку, яка значною мірою залежить від використовуваних датчиків. Другий тип – оптимізація аналізу положення, включаючи внутрішню обробку, яка не залежить від датчиків.

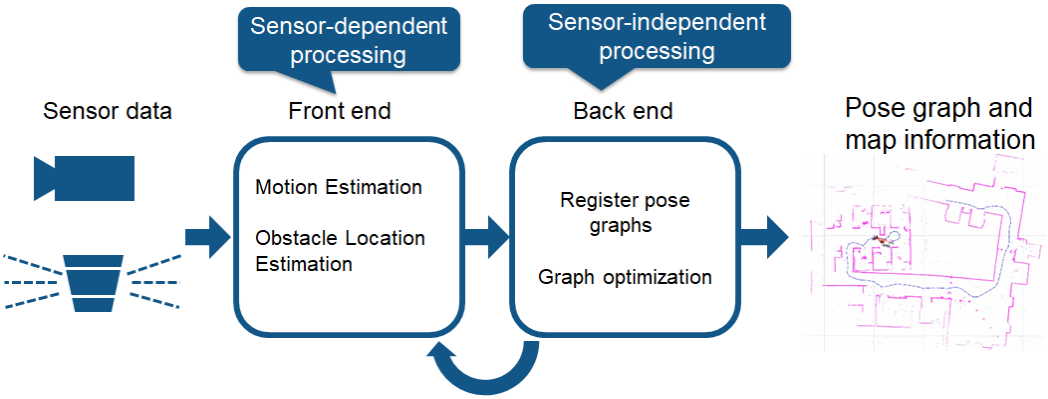

Рисунок 2.22 – Принцип роботи SLAM

Як випливає з назви, візуальний SLAM (або vSLAM) використовує зображення, отримані з камер та інших датчиків зображення. Visual SLAM може використовувати прості камери (ширококутні камери, камери «риб'яче око» та сферичні камери), складні камери для очей (стерео та мультикамери) і камери RGB-D (камери глибини та ToF).

Візуальний SLAM можна реалізувати за низькою ціною за допомогою відносно недорогих камер. Крім того, оскільки камери надають великий обсяг інформації, їх можна використовувати для виявлення орієнтирів (раніше виміряних положень). Виявлення орієнтирів також можна поєднати з оптимізацією на основі графіків, що забезпечує гнучкість реалізації SLAM (рисунок 2.23).

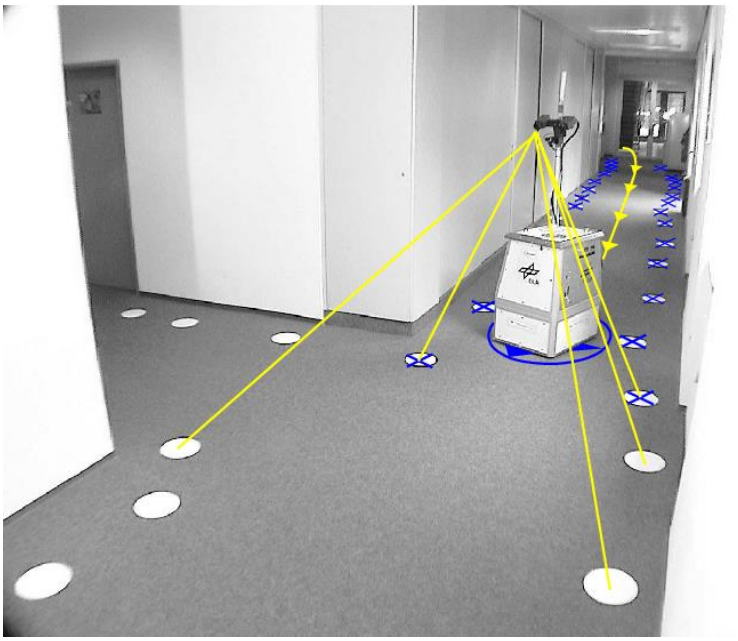

Рисунок 2.23 – Орієнтація робота у просторі

Монокулярний SLAM – це коли vSLAM [14] використовує одну камеру як єдиний датчик, що ускладнює визначення глибини. Задачу пошуку глибини зображення можна вирішити шляхом виявлення маркерів AR, шахових дошок або інших відомих об’єктів на зображенні для локалізації, або шляхом об’єднання інформації камери з іншим датчиком, таким як інерціальні вимірювальні модулі (IMU), які можуть вимірювати фізичні величини, такі як

швидкість та орієнтація. Технологія, пов’язана з vSLAM, включає структуру руху, візуальну одометрію та корегування положення транспортного засобу відносно оточуючих об’єктів.

Порівняно з камерами, ToF та іншими датчиками, лазери значно точніші та використовуються наприклад в високошвидкісних транспортних засобах, такими як самокеровані автомобілі та дрони. Вихідними значеннями лазерних датчиків зазвичай є 2D або 3D хмари точок . Хмара точок лазерного датчика забезпечує високоточні вимірювання відстані та дуже ефективно працює для побудови карт із SLAM. Як правило, рух оцінюється послідовно шляхом зіставлення хмар точок. Розрахований рух (пройдена відстань) використовується для локалізації транспортного засобу. Для співставлення хмари точок лідару використовуються алгоритми реєстрації, такі як ітераційні алгоритми найближчої точки і алгоритми перетворення нормального розподілу. 2D або 3D карти хмар точок можуть бути представлені у вигляді сіткової або воксельної карти (рисунок 2.24).

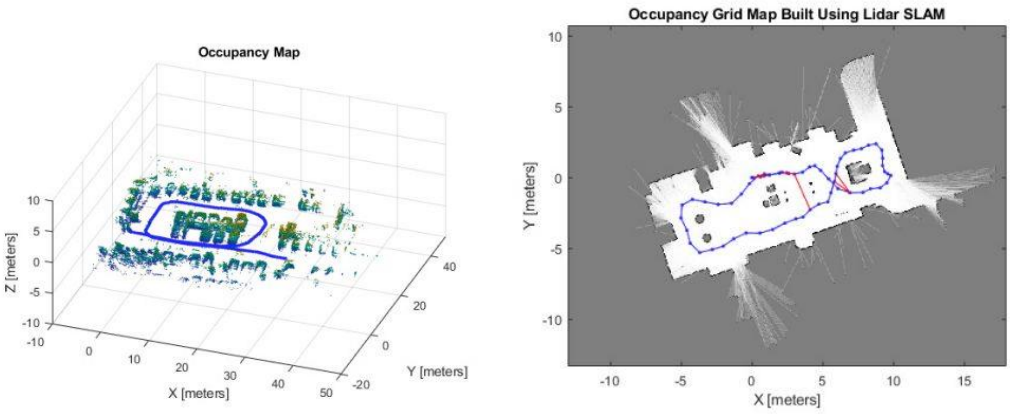

Рисунок 2.24 – 3D та 2D хмари точок

З іншого боку, хмари точок не настільки деталізовані, як зображення з камер, і не завжди забезпечують достатні характеристики для порівняння. Наприклад, у місцях, де мало перешкод, важко вирівняти хмари точок, і це може призвести до втрати сліду про місцезнаходження транспортного засобу. Крім того, узгодження хмар точок зазвичай вимагає високої обчислювальної

потужності, тому необхідно оптимізувати процеси для підвищення швидкості. Через ці проблеми локалізація для автономних транспортних засобів може включати об’єднання інших результатів вимірювань, таких як одометрія коліс, глобальна навігаційна супутникова система GPS та дані IMU. Для таких транспортних засобів, як складські роботи, зазвичай використовується 2D-лідар SLAM , тоді як SLAM з використанням 3-D-лідарних хмар точок можна використовувати для БПЛА та автоматизованого водіння.

Хоча SLAM використовується для деяких практичних застосувань, кілька технічних проблем перешкоджають застосуванню цього алгоритму в універсальних транспортних засобах. Але у кожної проблеми є рішення, яке може допомогти подолати проблему.

Одним з мінусів алгоритму є накопичення помилок положення у просторі, що спричиняє значні відхилення від фактичних значень.

SLAM оцінює послідовне переміщення, яке включає певну похибку. Помилка накопичується з часом, спричиняючи значні відхилення від фактичних значень. Це також може призвести до згортання або спотворення даних карти, що ускладнює подальше переміщення та навігацію у просторі. Візьмемо приклад проїзду по проходу квадратної форми. Оскільки помилка накопичується, початкова і кінцева точки робота більше не збігаються. Це називається проблемою замикання циклу. Подібних помилок оцінки положення не уникнути. Важливо виявити замикання циклів і визначити, як виправити або скасувати накопичену помилку (рисунок 2.25).

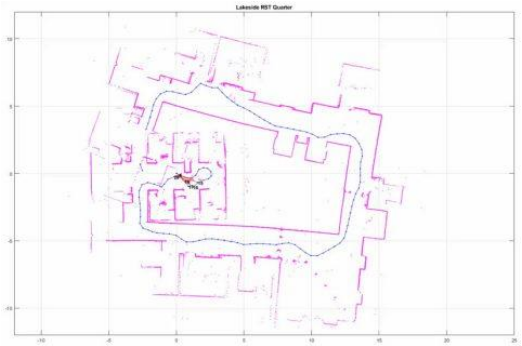

Рисунок 2.25 – Візуалізація похибки вимірювання

Один із рішень – запам’ятати деякі характеристики з раніше відвідуваного місця як орієнтир і мінімізувати помилку локалізації. Для виправлення помилок будуються графіки положення. Вирішуючи мінімізацію помилок як задачу оптимізації, можна отримати більш точні дані карти. Такий тип оптимізації називається пакетним корегування у візуальному SLAM.

Іншою проблемою є ситуація у якій із за помилки локалізації втрачено положення на карті. Картографування зображень і хмар точок не враховує характеристики руху робота. У деяких випадках цей підхід може генерувати розривні оцінки позиції. Наприклад, результат розрахунку показує, що робот, який рухається зі швидкістю 1 м/с, раптово стрибнув вперед на 10 метрів. Такого роду збої локалізації можна запобігти за допомогою алгоритму відновлення або об’єднання моделі руху з кількома датчиками для розрахунків на основі даних датчика.

Існує кілька методів використання моделі руху зі злиттям датчиків. Поширеним методом є використання фільтрації Калмана для локалізації. Оскільки більшість роботів з диференціальним приводом і чотириколісних транспортних засобів зазвичай використовують моделі нелінійного руху, часто використовуються розширені фільтри Калмана та фільтри частинок (локалізації Монте-Карло). У деяких випадках також можна використовувати більш гнучкі фільтри Байєса. Деякі часто використовувані датчики є інерціальними вимірювальними приладами, такими як IMU, інерційна навігаційна система, датчики акселерометра, гіроскопічні датчики та магнітні датчики). Колісні енкодери, прикріплені до транспортного засобу, часто використовуються для вимірювання одометрії.

Коли локалізація не вдається, рішення для відновлення реального положення є запам’ятовування орієнтира як ключового кадру з раніше відвідуваного місця. Під час пошуку орієнтиру використовується процес вилучення ознак, що дозволяє сканувати на високій швидкості. Деякі методи, засновані на ознаках зображення, включають пакет функцій і пакет візуальних

слоїв. Зовсім недавно глибоке навчання використовується для порівняння відстаней від функцій.

Проблема – високі обчислювальні витрати для обробки зображень, обробки хмари точок та оптимізації. Вартість обчислень є проблемою під час реалізації SLAM на апаратному забезпеченні транспортного засобу. Обчислення зазвичай виконуються на компактних і низькоенергетичних вбудованих мікропроцесорах, які мають обмежену обчислювальну потужність. Для досягнення точної локалізації важливо виконувати обробку зображень і узгодження хмари точок на високій частоті. Крім того, оптимізаційні обчислення, такі як замикання циклу, є процесами високого обчислення. Проблема полягає в тому, як виконати таку дорогу в обчислювальному відношенні обробку на вбудованих мікрокомп'ютерах.

Однією з контрзаходів є запуск різних процесів паралельно. Такі процеси, як вилучення ознак, що є попередньою обробкою процесу зіставлення, придатні для розпаралелювання (наприклад на графічних процесорах). Використання багатоядерних процесорів для обробки (рисунок 2.26), обчислення однієї команди з кількома даними і вбудованих графічних процесорів у деяких випадках можуть додатково підвищити швидкість. Крім того, оскільки оптимізацію графіка пози можна виконувати протягом відносно тривалого циклу, зниження його пріоритету та регулярне виконання цього процесу також може підвищити продуктивність.

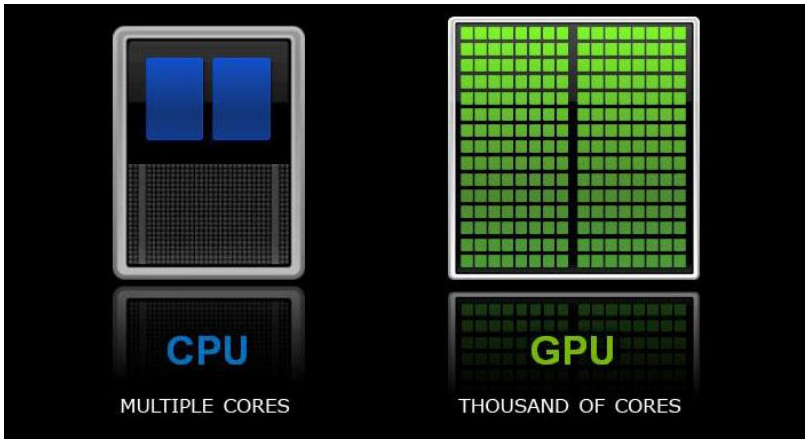

Рисунок 2.26 – Порівняння кількості ядер CPU та GPU

Окрім алгоритму навігації автоматизовані транспортні засоби можуть використовувати алгоритм для пошуку та орієнтування відносно смуги руху (наприклад систему круїзу на автомобілях, шляхи для переміщення автоматизованих роботів у складських приміщеннях).

Задача пошуку смуги руху для комп'ютерного зору не є проблемою, і вирішується у декілька етапів (рисунок 2.27):

– захоплення та декодування відеопотоку: після ініціалізації запису кожен кадр відео декодується (тобто перетворюється на послідовність зображень);

– перетворення зображення у відтінках сірого: відеокадри у форматі RGB, RGB перетворюється на відтінки сірого, оскільки обробка одноканального зображення швидше, ніж обробка триканального кольорового зображення;

– зменшення шуму: шум може створити помилкові краї, тому перш ніж йти далі, необхідно виконати згладжування зображення. Для виконання цього процесу використовується фільтр Гаусса;

– canny edge detector: він обчислює градієнт у всіх напрямках нашого розмитого зображення та відстежує краї із великими змінами інтенсивності;

– перетворення лінії Хафа: перетворення лінії Хафа – це перетворення, що використовується для виявлення прямих ліній. Тут використовується імовірнісне перетворення лінії Хафа, яке дає вихід у вигляді екстремальних значень виявлених ліній.

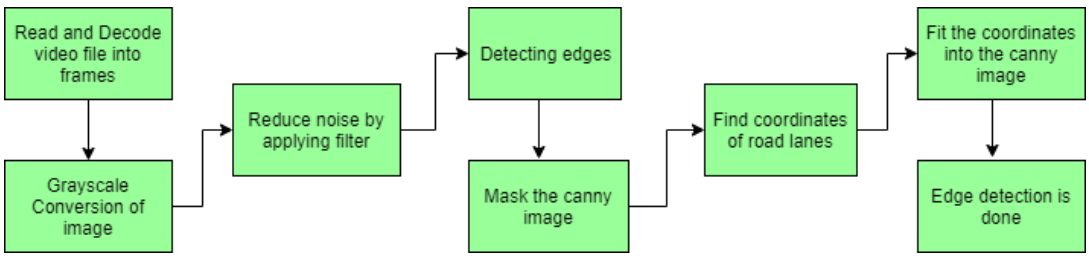

Рисунок 2.27 – Загальна схема обробки зображення для аналізу

2.8 Висновки по другому розділу

В даному розділі були розглянуті теоретичні відомості та визначення, що стосуються будови та принципу роботи систем автоматичного керування. Були розглянуті різноманітні типи систем, їх особливості роботи та області застосування. Були розглянуті принципи та алгоритми обробки отриманої інформації.

3 РОЗРОБКА ТЕСТОВОЇ ПЛАТФОРМИ

Для тестування та дослідження принципів розробки та побудови алгоритмів керування транспортними засобами на основі системи комп’ютерного зору була спроектована та розроблена рухома платформа (рисунок 3.1).

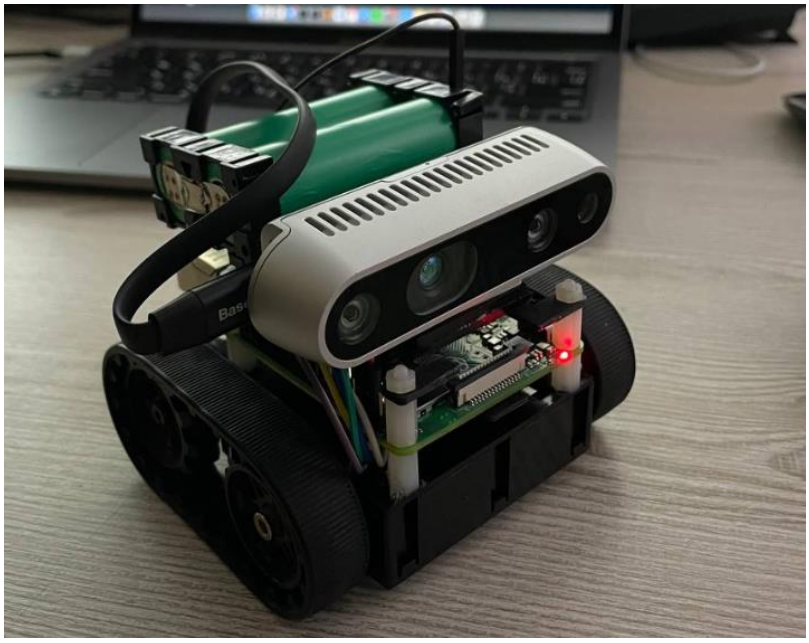

Рисунок 3.1 – Загальний вигляд платформи

За основу платформи було взято шасі Zumo [15] від компанії Pololu, завдяки малим розмірам та типу шасі «гусенична платформа» вдалось досягти компактності та універсальності для платформи.

У рух платформу приводять два коллекторні двигуни (рисунок 3.2) з редукторами 100:1, і наступними технічними параметрами:

– габарити 10 мм × 12 мм × 26 мм;

– вага 9,5 г;

– діаметр валу 3 мм;

– напруга живлення 8 В;

– струм без навантаження 150 мА;

– струм при навантаженні 1,5 А;

– кількість обертів на валу 330 об/хв.

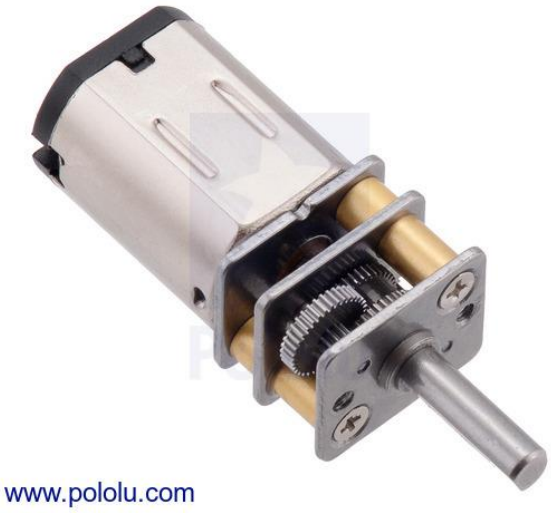

Рисунок 3.2 – Використані при розробці двигуни

Для керування моторами використовується двоканальний драйвер двигунів DRV8833 (рисунок 3.3) з наступними характеристиками:

– мікросхема драйвера DRV8833;

– мінімальна напруга живлення двигунів 2,7 В;

– максимальна напруга живлення двигунів 10,8 В;

– робоча напруга логічної частини від 2,7 В до 5,5 В;

– максимальний струм двигунів 0,5 А (середній) / 2,0 А (піковий);

– максимальна частота керування 50 кГц.

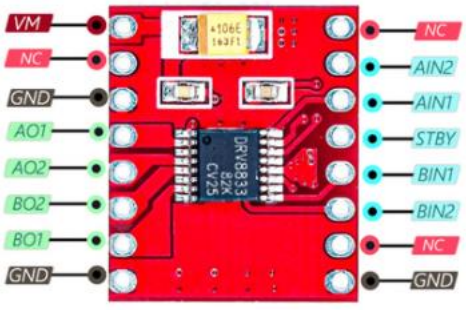

Рисунок 3.3 – Драйвер для контролю двигунів

Підключення драйвера реалізовано за схемою на рисунку 3.4.

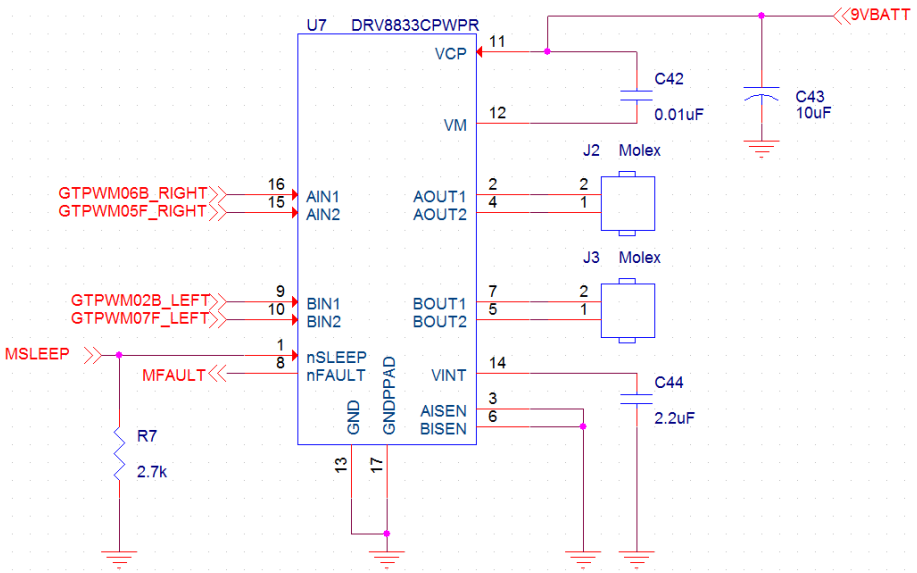

Рисунок 3.4 – Схема підключення драйверу

3.1 Обчислювальний модуль

Для керування драйвером і роботи керуючої програми використовується мікрокомп’ютер Raspberry Pi 4 (рисунок 3.5) [16].

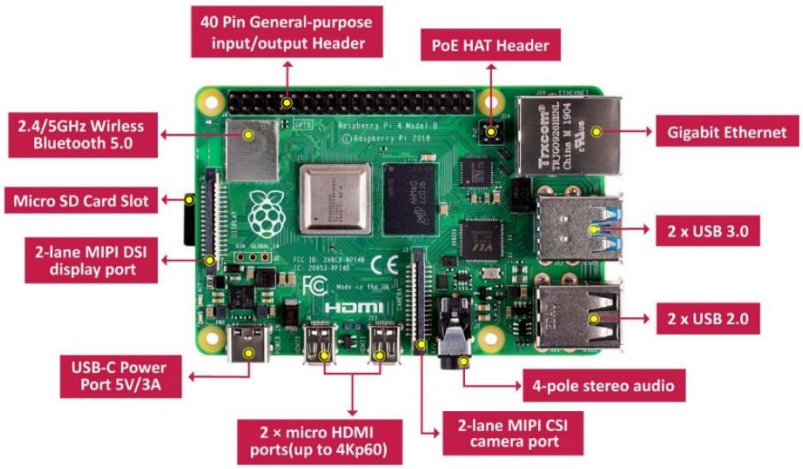

Рисунок 3.5 – Обчислювальна основа платформи

Вибір даної платформи обумовлений необхідністю підключення камери глибини по інтерфейсу USB3.0, та багатоядерним процесором для обробки та одночасного аналізу даних отриманих з камери глибини, тому технічні характеристики вибраної платформи наступні:

– процесор: 1,5 ГГц 64-бітний чотирьохядерний процесор ARM Cortex-A72;

– об'єм оперативної пам'яті: 8 ГБ LPDDR4 SDRAM;

– мережеві інтерфейси: габітний Ethernet з повною пропускною спроможністю інтерфейсу, двохдіапазонна бездротова мережа 802.11ac, Bluetooth 5.0;

– USB порти: два порти USB 3.0, Два порти USB 2.0;

– відеовихід: два micro-HDMI з підтримкою двох моніторів з роздільною здатністю до 4K;

– графічний інтерфейс: VideoCore VI, що підтримує OpenGL ES 3.x;

– апаратне декодування 4Kp60 відео HEVC;

– сумісність: повна сумісність з більш ранніми продуктами Raspberry Pi;

– роз'єм живлення: USB-C;

– напруга живлення: 5 В;

– додаткове живлення: Ethernet PoE Hat;

– максимальний споживаний струм: 3 А.

3.2 Сенсор глибини

В якості джерела для відеосигналу на дану платформу потрібно підібрати камеру з можливістю виведення як RGB зображення (задля розпізнавання маркерів, або кольорів ліній для руху), так і з можливістю побудови хмари точок, що будуть відповідати відстаням до об’єктів. Також камера повинна мати малі розмірі та вагу, та використовувати якомога менше струму для живлення, так як ємність акумуляторів на платформі не велика.

Виходячи з бажаних характеристик та вимог до камери був зроблений вибір на лінійку камер глибини RealSence від компанії Intel, а саме Intel RealSense D435i (рисунок 3.6).

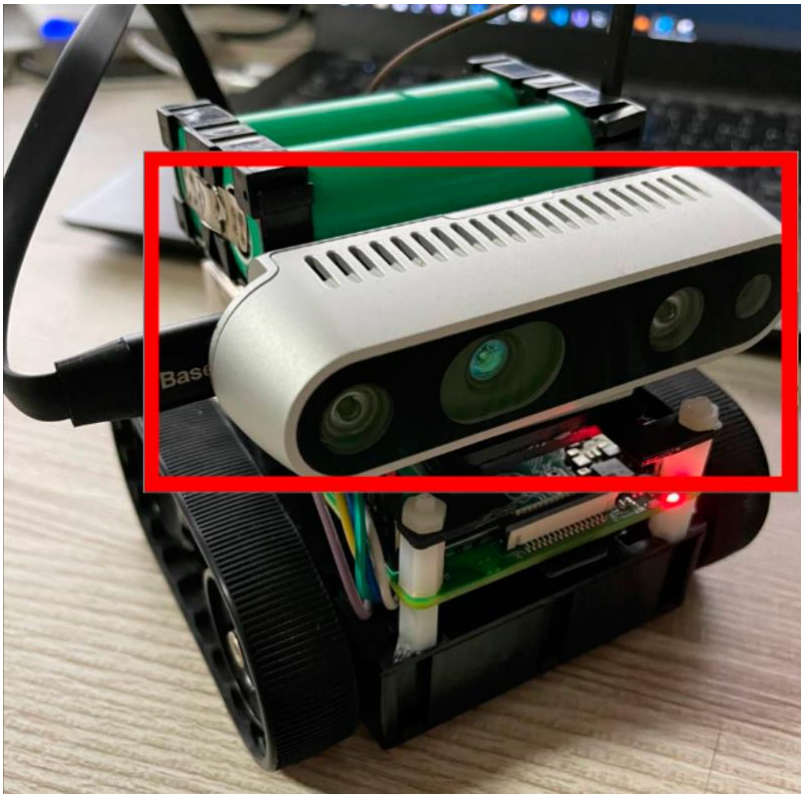

Рисунок 3.6 – Камера глибини на платформі

Сенсор має наступні технічні характеристики:

– область застосування: всередині та поза приміщеннями;

– технологія сенсора зображення: Global Shutter;

– максимальна дальність: 10 м;

– технологія визначення глибини: Active IR Stereo;

– мінімальна відстань: 0,105 м;

– кут огляду: 86° × 57° (±3°);

– вихідна роздільна здатність: до 1280x720;

– роздільна здатність RGB-сенсора: 1920x1080 90 кадр/с;

– частота RGB-сенсора: 30 кадр/с;

– кут огляду RGB-сенсора: 69,4° × 42,5° × 77° (±3°);

– процесор відеообробки: Intel RealSense Vision Processor D4;

– інтерфейси: USB 3.1 Gen 1;

– габаритні розміри: 90 мм x 25 мм x 25 мм.

Для зв’язку міні-комп’ютера та сенсору зображення використовується протокол USB2.0 з максимальною швидкістю 480 Мбіт/с.

В якості джерела живлення для платформи використовуються 2 акумулятори формату 18650, з ємністю в 3400 мА і номінальною напругою 3.7 В. У якості перетворювача напруги послідовно підключених акумуляторів був використаний DCDC перетворювач, з вихідною напругою 5 В та струмом 3 А (рисунок 3.7).

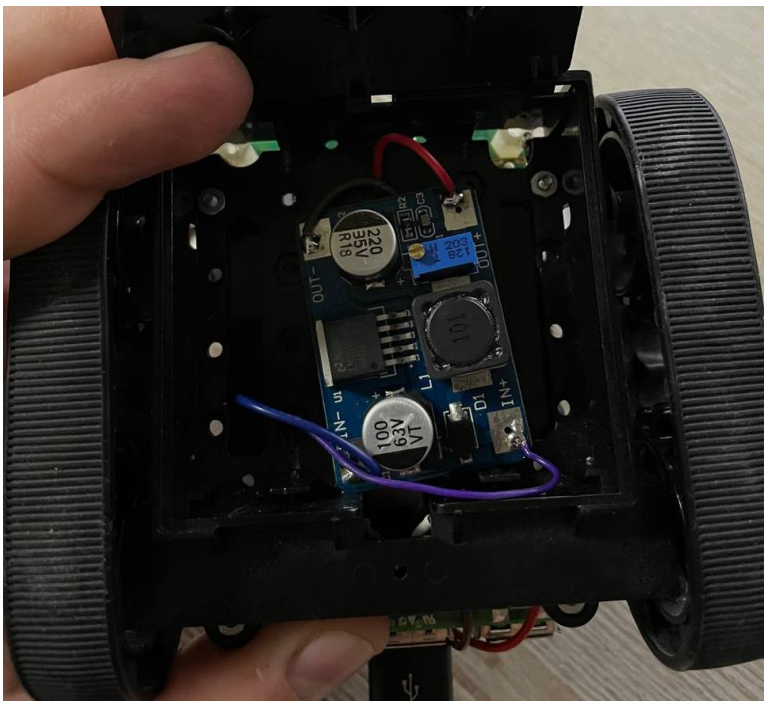

Рисунок 3.7 – DCDC перетворювач напруги

3.3 Інсталяція програмного забезпечення

Для розробки та роботи програмного забезпечення на мікрокомп’ютер повинно встановити операційну систему, в якості якої було вибрано Ubuntu 20.04.3 LTS, завдяки підтримки необхідних для 3д камери бібліотек. Задля

роботи камери потрібно встановити набір бібліотек librealsense за допомогою наступних команд:

а) оновлюємо бібліотеки для компіляції в системі:

```
sudo apt-get update && sudo apt-get dist-upgrade
sudo apt-get install automake libtool vim cmake libusb-1.0-0-dev libx11-dev xorg-dev libglu1-mesa-dev;
```

б) збільшуємо розмір swap файлу, тому що компіляція не завершиться вдачно з стандартним обємом swap файлу, який становить 100Мбайт:

```
CONF_SWAPSIZE=2048
sudo vi /etc/dphys-swapfile
sudo /etc/init.d/dphys-swapfile restart swapon -s;
```

в) створюємо udev правило, яке буде проводити настройку камери при її підключенні:

```
cd ~
git clone https://github.com/IntelRealSense/librealsense.git
cd librealsense
sudo cp config/99-realsense-libusb.rules /etc/udev/rules.d/
udevadm control --reload-rules && udevadm trigger;
```

г) встановлюємо шлях до бібліотеки для системи, та зберігаємо зміни у PATH:

```
export LD_LIBRARY_PATH=/usr/local/lib:$LD_LIBRARY_PATH
source ~/.bashrc;
```

ґ) встановлюємо бібліотеку protobuf:

```
cd ~
git clone --depth=1 -b v3.10.0 https://github.com/google/protobuf.git
cd protobuf
./autogen.sh
./configure
make -j1
```

```
sudo make install
cd python
export LD_LIBRARY_PATH=../src/.libs
python3 setup.py build --cpp_implementation
python3 setup.py test --cpp_implementation
sudo python3 setup.py install --cpp_implementation
export PROTOCOL_BUFFERS_PYTHON_IMPLEMENTATION=cpp
export PROTOCOL_BUFFERS_PYTHON_IMPLEMENTATION_VERSION=3
sudo ldconfig
protoc –version;
```

д) встановлюємо бібліотеку libtbb-dev:

```
cd ~
wget https://github.com/PINTO0309/TBBonARMv7/raw/master/libtbb-dev_2018U2_armhf.deb
sudo dpkg -i ~/libtbb-dev_2018U2_armhf.deb
sudo ldconfig
rm libtbb-dev_2018U2_armhf.deb;
```

е) компілюємо бібліотеку librealsense:

```
cd ~/librealsense
mkdir  build  && cd build
cmake .. -DBUILD_EXAMPLES=true -DCMAKE_BUILD_TYPE=Release DFORCE_LIBUVC=true
make -j4
sudo make install;
```

є) компілюемо та встановлюемо бібліотеку pyrealsense2 для мови Python:

```
cd ~/librealsense/build
cmake .. -DBUILD_PYTHON_BINDINGS=bool:true -DPYTHON_EXECUTABLE=$(which python3)
make -j1
```

sudo make install;

з) імпортуємо бібліотеку в PATH системи, задля її використанні у скриптах на мові Python:

export PYTHONPATH=$PYTHONPATH:/usr/local/lib;

и) проводимо тестування бібліотеки pyrealsense2.

Використовуючи Менеджер підключень до віддаленого робочого столу підключаємось до системи нашої тестової платформи.

Для підключення ми повинні знати IP адресу тестової платформи в нашій мережі, цю інформацію можна знайти наприклад в списку підключень роутера (рисунок 3.8).

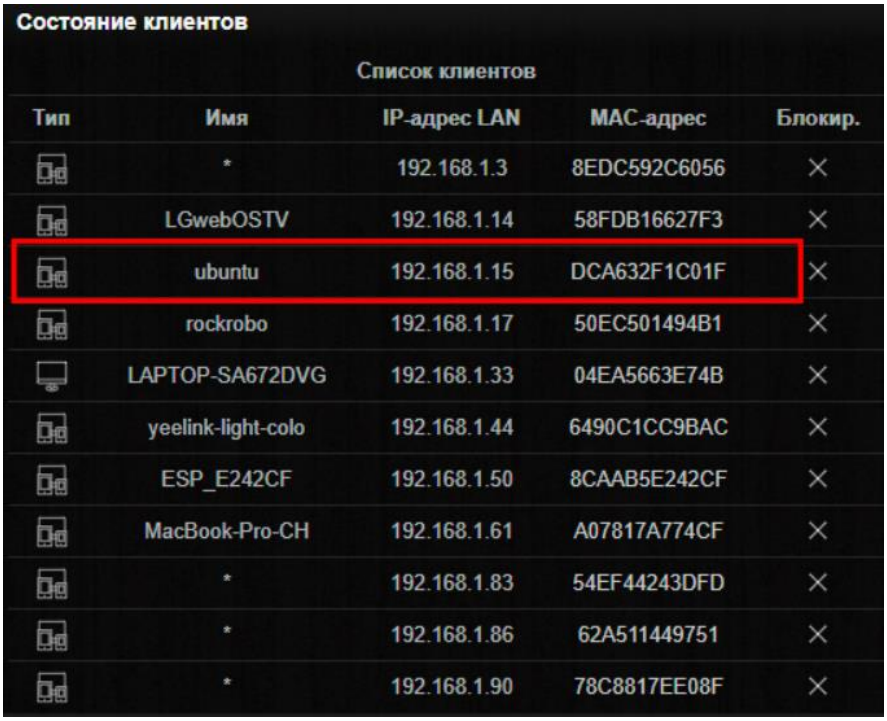

Рисунок 3.8 – Список пристроїв у локальній мережі

Використовуючи знайдену нами IP адресу та ім’я користувача ubuntu підключимося до віртуального робочого столу, використовуючи стандартні можливості системи Windows (рисунок 3.9).

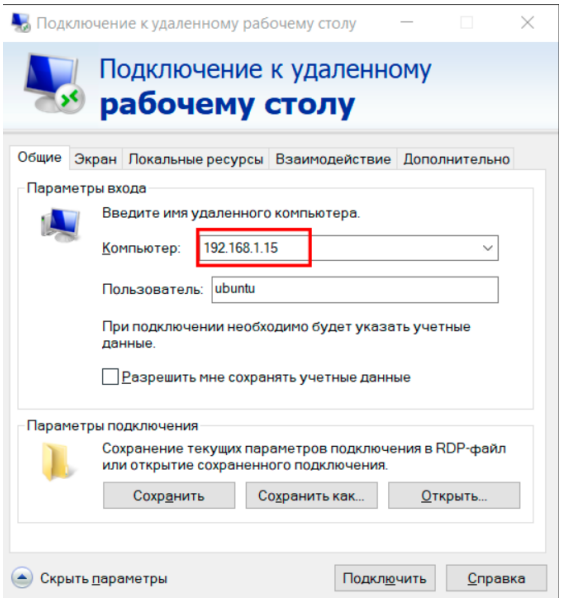

Рисунок 3.9 – Менеджер підключення до робочого столу

Відсутність помилок після імпорту бібліотеки свідчить про те що бібліотека успішно імпортувалася та готова до роботи (рисунок 3.10). Задля тесту 3д камери був написаний простий скрипт на мові Python (рисунок 3.11), задачею якого є отримання зображення з RGB та Depth сенсору камери, та виведення зображення на екран (рисунок 3.12).

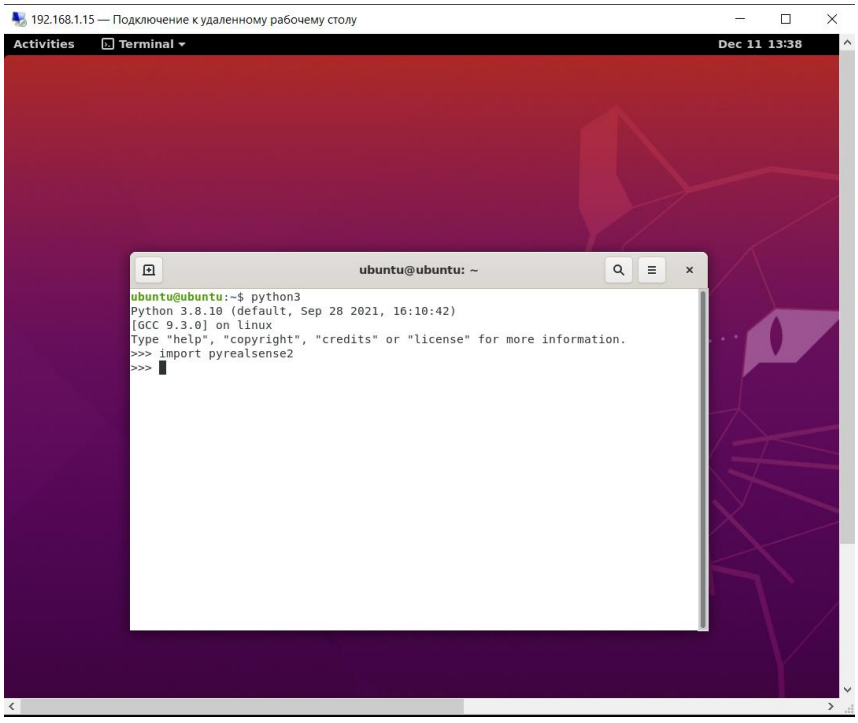

Рисунок 3.10 – Тестування встановленої бібліотеки

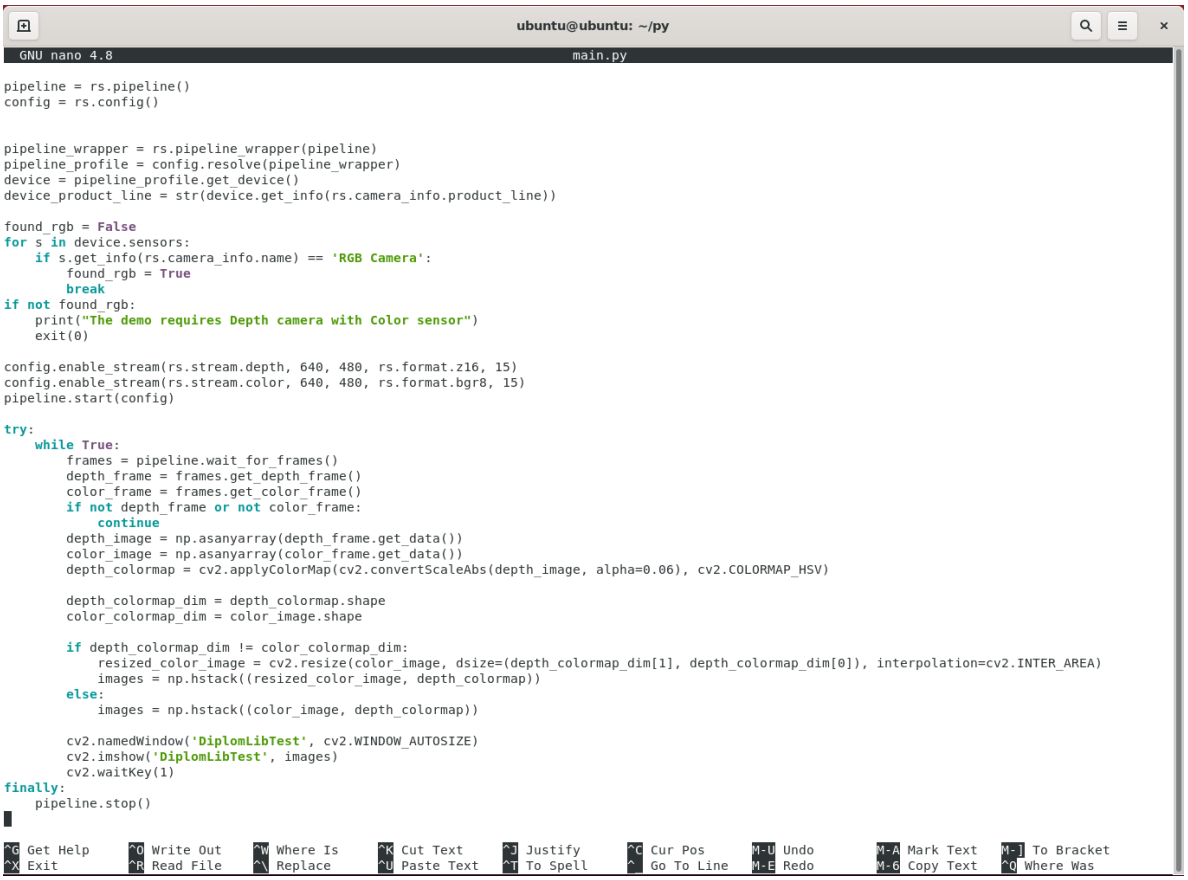

Рисунок 3.11 – Код для тестування сенсору глибини

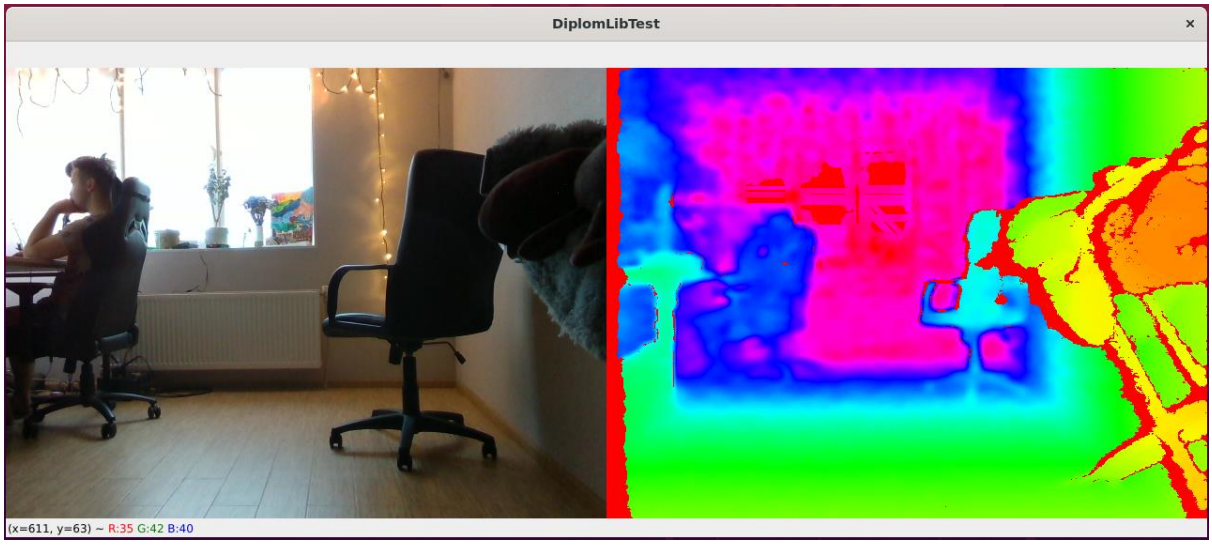

Рисунок 3.12 – Результат роботи коду для тестування

По результатам тестування можна зробити висновок що камера працює коректно та готова до роботи.

Для роботи двигунів платформи на плату драйвера потрібно подавати ШИМ сигнал, задля плавного регулювання числа обертів на двигунах. Управляючі контакти з драйверу заведені на GPIO виводи мікрокомп'ютера. Для роботи с контактами GPIO з коду використовується бібліотека RPi.GPIO, яку можна встановити командою: pip3 install RPi.GPIO.

Входи драйверів підключені до PWM виходів GPIO13 та GPIO19 (рисунок 3.13).

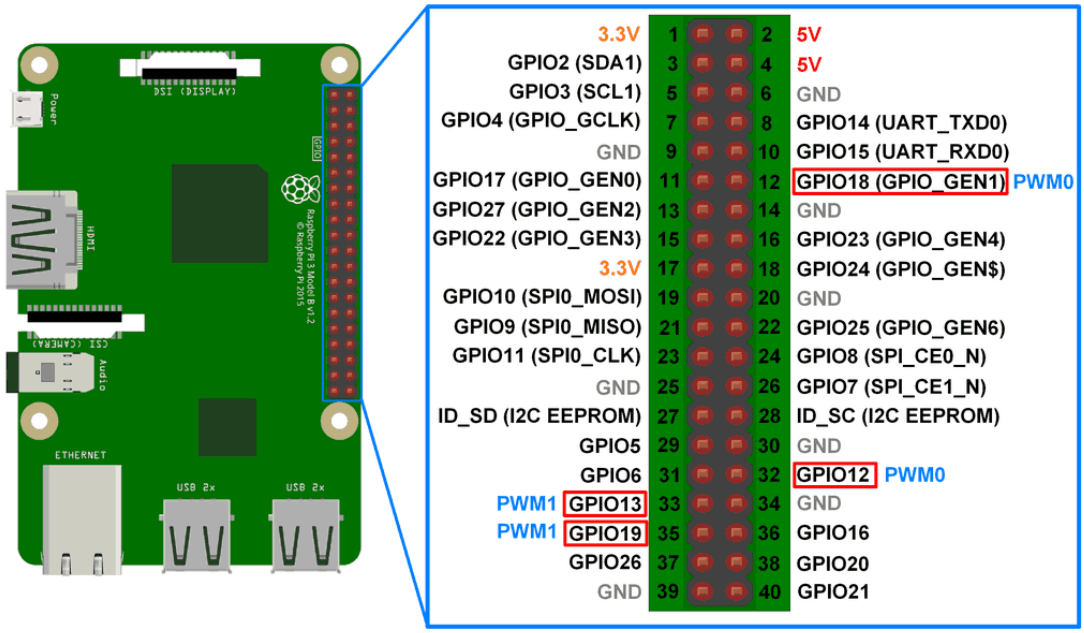

Рисунок 3.13 – Позначення GPIO виводів Raspberry Pi

3.4 Висновки по третьому розділу

В даному розділі були розглянуті компоненти тестової платформи для розробки алгоритмів керування, підібраний мікрокомп'ютер та камера глибини. Проведена інсталяція потрібних для роботи з сенсором глибини бібліотек, розроблений скрипт для тестування встановлених бібліотек та самої камери глибини.

4 ПРАКТИЧНА РЕАЛІЗАЦІЯ АЛГОРИТМУ

В якості завдання задля тестування зібраної платформи та роботи з алгоритмами керування буде розробка та написання алгоритму навігації роботизованої платформи на основі маркерів Aruco. При цьому маркерами будуть описуватися точки маршруту від бази платформи до вантажу (рисунок 4.1).

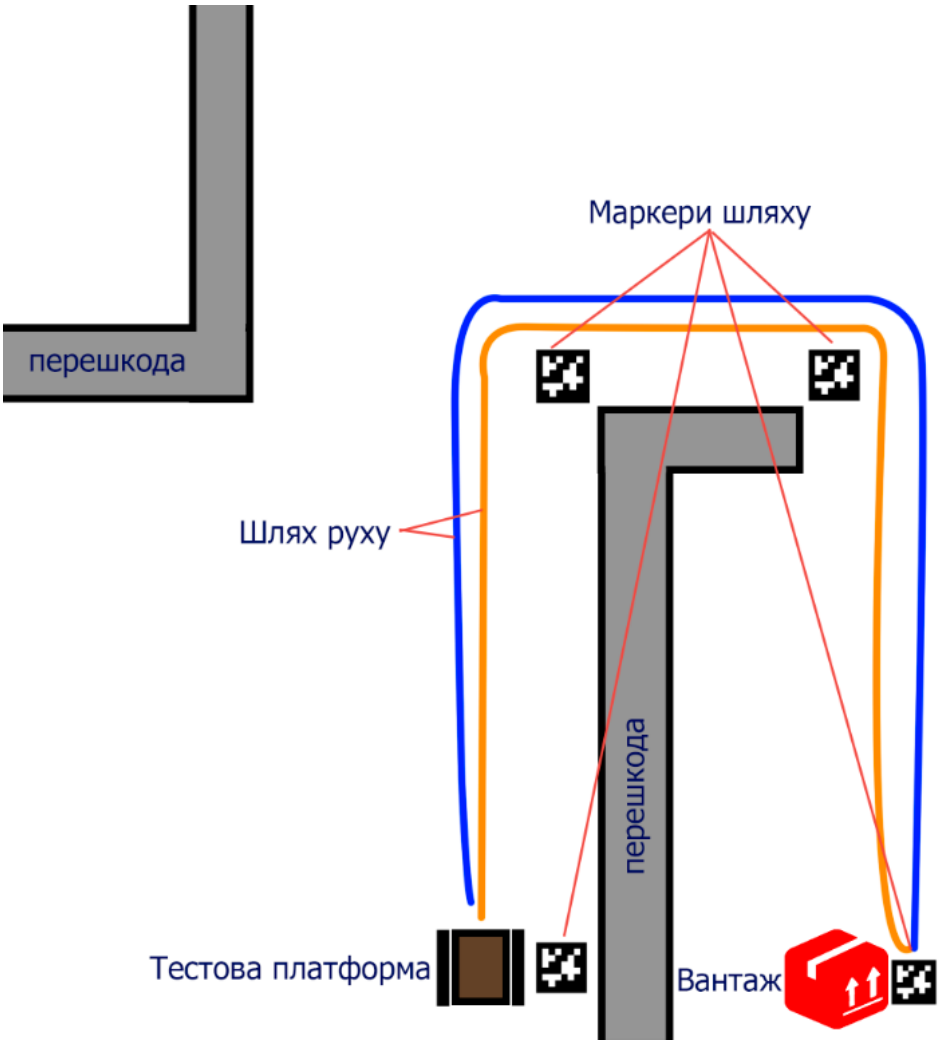

Рисунок 4.1 – Схема реалізації майданчика для тестування

4.1 Детекція маркерів Aruco

Першим кроком для виконання поставленого завдання є генерація та встановлення маркерів Aruco. В якості зображення маркерів була взята готова таблиця маркерів, які були роздруковані у вигляді вертикальних табличок.

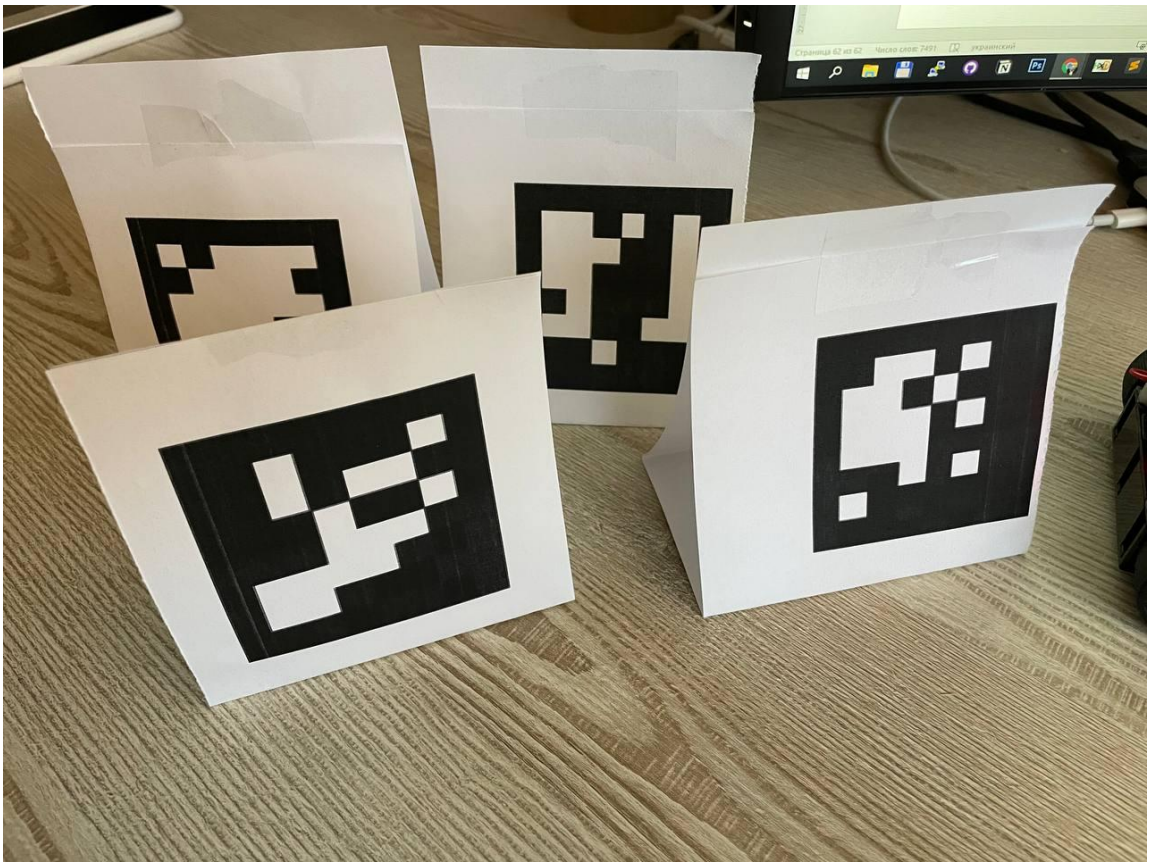

Рисунок 4.2 – Таблички з маркерами Aruco

Враховуючи наявність табличок з маркерами Aruco на данному єтапі потрібно написати та протестувати код детекції маркерів на зображенні. Перед написання алгоритму пошуку маркерів необхідно вивести на екран RGB зображення з камери глибини, використовуючи бібліотеку pyrealsence2 та opencv.

Код який виводить аналізує (рисунок 4.3) та виводить зображення приведений на рисунку 4.4.

```python
import pyrealsense2.pyrealsense2 as rs
import numpy as np
import cv2

pipeline = rs.pipeline()
config = rs.config()

pipeline_wrapper = rs.pipeline_wrapper(pipeline)
pipeline_profile = config.resolve(pipeline_wrapper)
device = pipeline_profile.get_device()
device_product_line = str(device.get_info(rs.camera_info.product_line))

found_rgb = False
for s in device.sensors:
    if s.get_info(rs.camera_info.name) == 'RGB Camera':
        found_rgb = True
        break

config.enable_stream(rs.stream.depth, 640, 480, rs.format.z16, 15)
config.enable_stream(rs.stream.color, 640, 480, rs.format.bgr8, 15)
pipeline.start(config)
```

Рисунок 4.3 – Код ініціалізації камери

```python
frames = pipeline.wait_for_frames()
depth_frame = frames.get_depth_frame()
color_frame = frames.get_color_frame()
if not depth_frame or not color_frame:
    continue
depth_image = np.asanyarray(depth_frame.get_data())
color_image = np.asanyarray(color_frame.get_data())
depth_colormap = cv2.applyColorMap(cv2.convertScaleAbs(depth_image, alpha=0.06),

depth_colormap_dim = depth_colormap.shape
color_colormap_dim = color_image.shape

if depth_colormap_dim != color_colormap_dim:
    resized_color_image = cv2.resize(color_image, dsize=(depth_colormap_dim[1],
    images = np.hstack((resized_color_image, depth_colormap))
else:
    images = np.hstack((color_image, depth_colormap))

cv2.namedWindow('DiplomLibTest', cv2.WINDOW_AUTOSIZE)
cv2.imshow('DiplomLibTest', images)
cv2.waitKey(1)
```

Рисунок 4.4 – Код виводу зображення на екран

Після отримання відеозображення з камери був написаний алгоритм пошуку маркерів на зображенні, та позначання їх прямокутниками (рисунок 4.5).

```python
def detectAruco(frame, depth_frame, raw_depth, blank_image):
    (corners, ids, rejected) = cv2.aruco.detectMarkers(frame, arucoDict, parameters=arucoParams)
    list_aruco = []
    list_y = []
    list_x = []
    list_id = []
    if len(corners) > 0:
        ids = ids.flatten()
        for(markerCorner, markerID) in zip(corners, ids):
            corners = markerCorner.reshape((4, 2))
            (topLeft, topRight, bottomRight, bottomLeft) = corners
            topRight = (int(topRight[0]), int(topRight[1]))
            bottomRight = (int(bottomRight[0]), int(bottomRight[1]))
            bottomLeft = (int(bottomLeft[0]), int(bottomLeft[1]))
            topLeft = (int(topLeft[0]), int(topLeft[1]))

            cv2.line(frame, topLeft, topRight, (0, 255, 0), 2)
            cv2.line(frame, topRight, bottomRight, (0, 255, 0), 2)
            cv2.line(frame, bottomRight, bottomLeft, (0, 255, 0), 2)
            cv2.line(frame, bottomLeft, topLeft, (0, 255, 0), 2)

            cX = int((topLeft[0] + bottomRight[0]) / 2.0)
            cY = int((topLeft[1] + bottomRight[1]) / 2.0)
            cv2.circle(frame, (cX, cY), 4, (0, 0, 255), -1)
            cv2.putText(frame, str(markerID), (topLeft[0], topLeft[1] - 15), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)
```

Рисунок 4.5 – Код розпізнавання маркерів

В результаті обробки відеозображення алгоритмом пошуку ми отримуємо наступне зображення (рисунок 4.6).

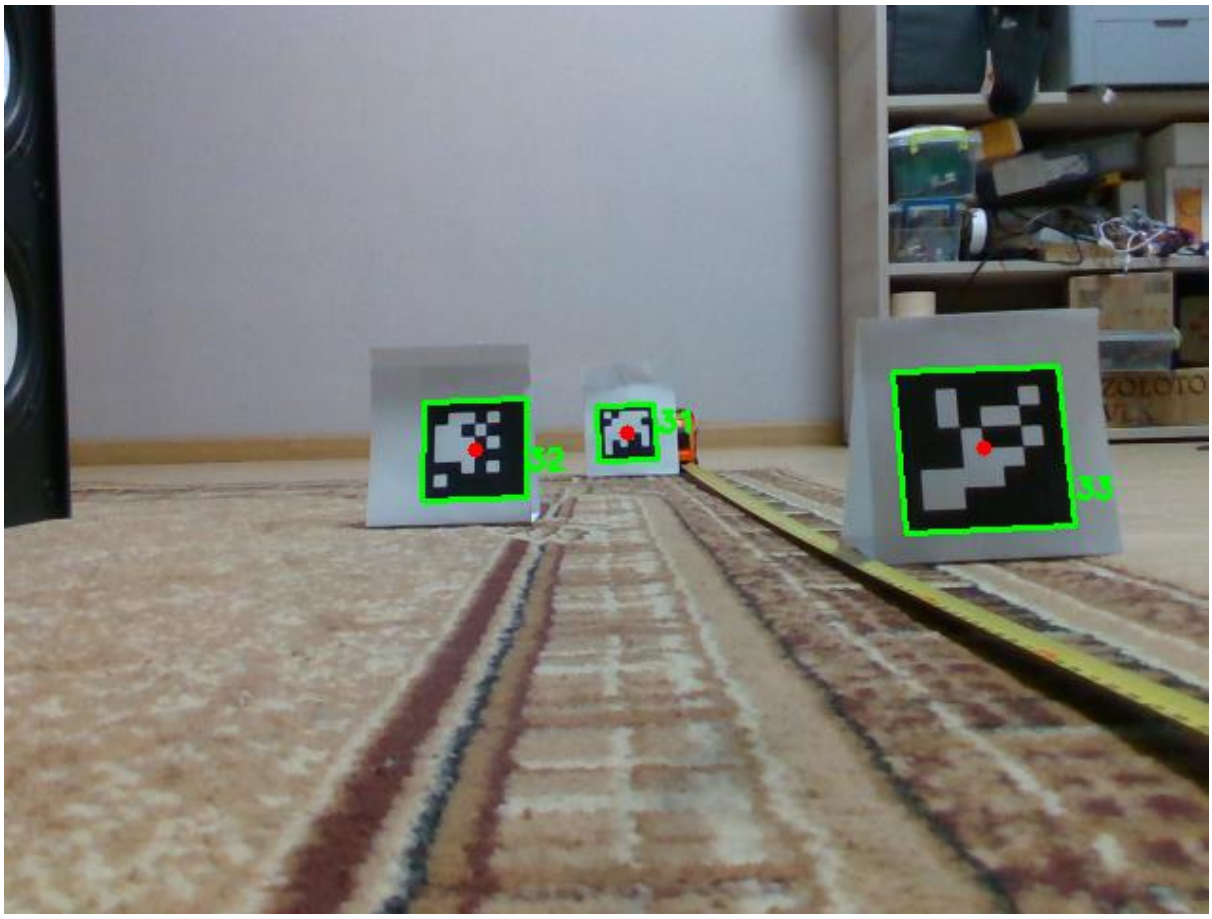

Рисунок 4.6 –Результат пошуку маркерів на зображенні

Окрім пошуку маркерів для руху платформи потрібно вимірювати відстань до кожного з маркерів, для розрахунку маршруту платформи. Оскільки ми використовуємо камеру глибини – нам достатньо активувати канал глибини, і отримувати інформацію про відстань до кожного пікселю функцією depth = raw_depth.get_distance(cX, cY), де cX та cY – координати точки, відстань до якої нам потрібно знайти.

В якості точки для знаходження відстані був вибраний центр кожного з маркерів.

4.2 Вимірювання відстані до маркерів

Для тестування точності алгоритму отримання відстані маркери Aruco були виставлені на рівній поверхні (домашньому килимі) на відстанях 50 см, 70 см та 130 см (рисунок 4.7).

Рисунок 4.7 – Стенд для тестування функції отримання дистанції до маркерів

Отримані відстані було вирішено наносити на зображення глибини, разом в контурами маркерів Aruco (рисунок 4.8).

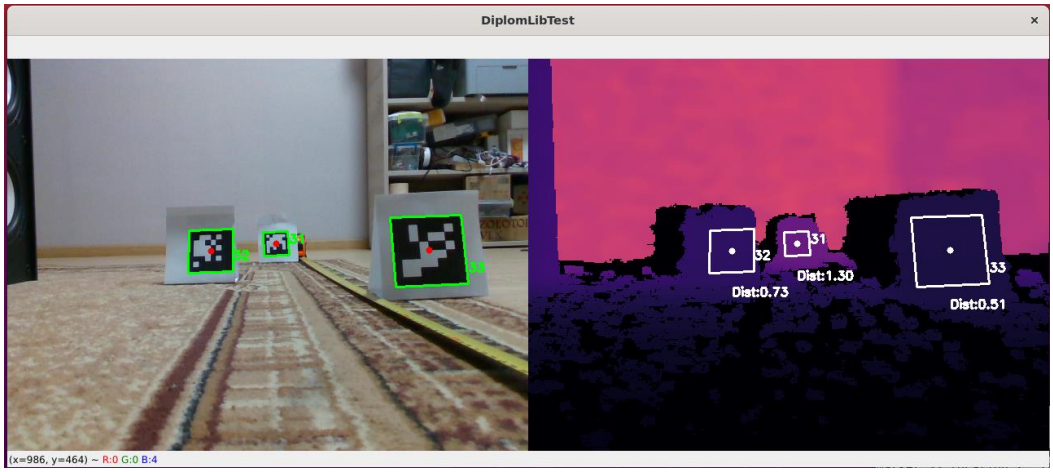

Рисунок 4.8 – Отримані значення відстані до маркерів

Отримані результати були занесені в таблицю 4.1 задля аналізу точності вимірювання.

Таблиця 4.1 – результати вимірювання відстані камерою

Реальна відстань, см	Відстань виміряна програмою, см	Похибка вимірювання, %
50	51	2 %
70	73	4 %
130	130	0 %

Похибку вимірювання було розраховано за формулами (4.1) та (4.2):

$$\Delta X = X - X_Д, \tag{4.1}$$

$$\delta = \frac{\Delta X}{X_Д} \times 100\ \%, \tag{4.2}$$

де ΔX – абсолютне значення похибки;

X – виміряне значення;

$X_Д$ – дійне значення;

δ – похибка вимірювання.

Середнє значення похибки становить (2 %+4 %)/2 = 2 %, що є досить точним, враховуючи розміри платформи 12 см x 10 см x 16 см.

Задля аналізу оточення та перешкод було побудовано 2D зображення оточення попереду платформи (рисунок 4.9). Опорною точкою по осі Y вважаємо найбільш віддалений маркер на зображенні, так як враховуючи перспективу зображення центр найбільш віддаленого маркеру буде знаходитися вище краю підлоги, враховуючи максимальну дистанцію роботи сенсора глибини (10 м).

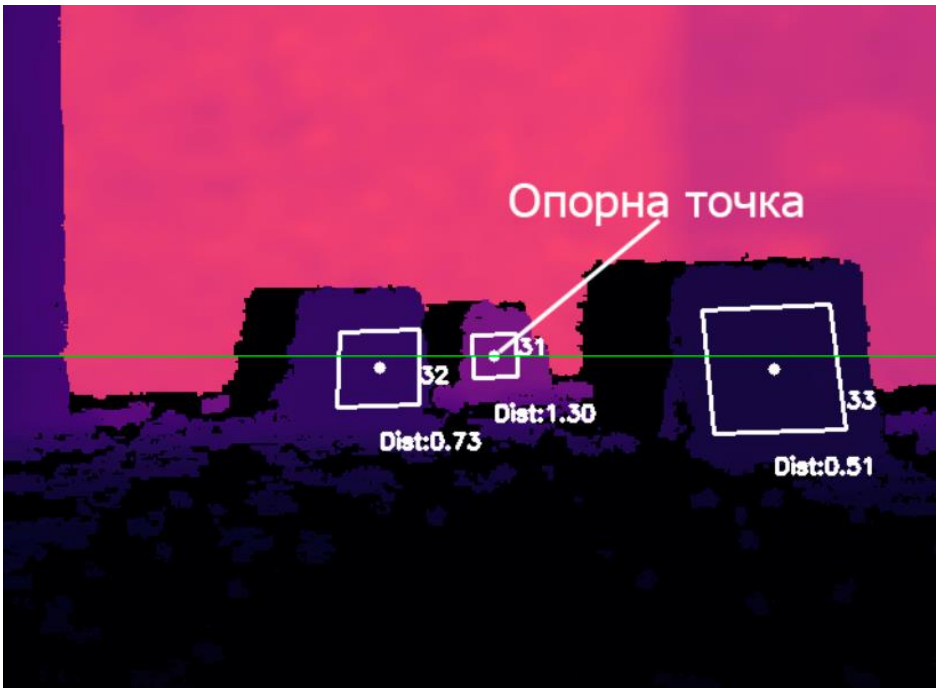

Рисунок 4.9 – Точка для горизонтальної лінії вимірювання глибини

На основі висоти опорної точки (її координати по осі Y) проводиться лінія, упродовж якої і аналізуються перешкоди попереду (рисунок 4.10).

Оскільки висота зображення 480 пікселів, а робоча дистанція 2-3 метри – потрібно масштабувати відстань, задля коректного відображення на зображенні. Перетворення відбувається за рахунок простої операції

множення: depth_y_mapped = int(depth_point * 150). У результаті перетворення реальна відстань 2 метри буде мати розмірність у 300 пікселів на зображенні.

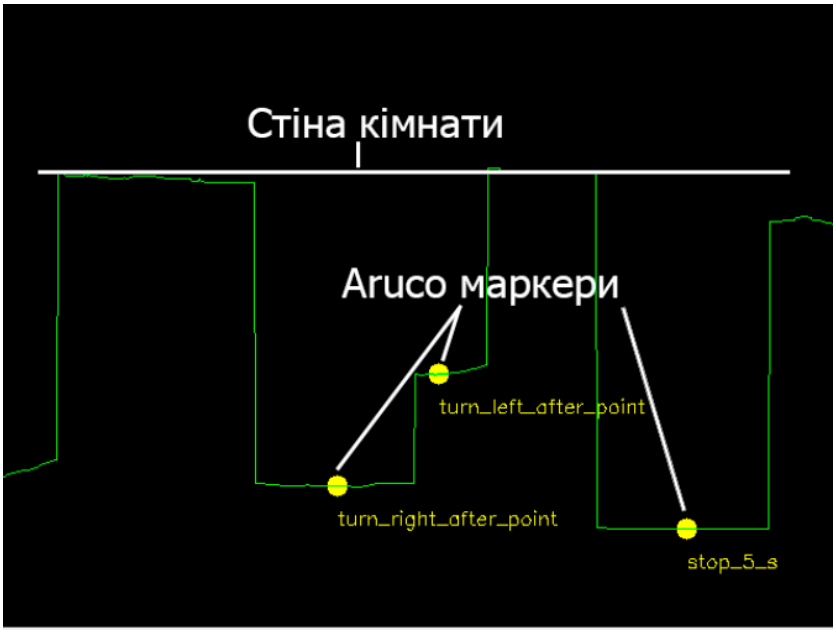

Рисунок 4.10 – 2D проекція відстані до об’єктів перед платформою

Загальний код малювання відстані до оточення має наступний вигляд (рисунок 4.11)

```
if len(list_aruco) > 0:
    min_point = 479
    for aruco_item in list_aruco:
        if min_point > aruco_item.y:
            min_point = aruco_item.y
    top_point = min_point
    cv2.line(depth_frame, (0, top_point), (639, top_point), (0, 255, 0), 1)
    print(top_point)
    depth_points = np.array([[0, 0]], np.int32)
    empty_array = np.empty((0, 2), int)
    old_y = 0
    for depth_x in range(639):
        depth_point = raw_depth.get_distance(depth_x, top_point)
        depth_y_mapped = int(depth_point * 150)
        y_point = 480 - depth_y_mapped
        if depth_y_mapped == 0:
            y_point = old_y
        old_y = y_point
        empty_array = np.append(empty_array, np.array([[depth_x, y_point]]), axis=0)
        for aruco_item in list_aruco:
            if depth_x == aruco_item.x:
                cv2.circle(blank_image, (depth_x, y_point), 8, (0, 255, 255), -1)
                cv2.putText(blank_image, markers_dict[aruco_item.id], (depth_x, y_point + 30),
                                        cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 255), 1)
    cv2.polylines(blank_image, [empty_array], False, (0,255,0))
```

Рисунок 4.11 – Код відображення відстані до об’єктів перед платформою

На даному етапі роботизована платформа має змогу розпізнавати тип та ID кожного з маркерів та аналізувати їх положення відносно себе.

4.3 Реалізація інструкцій руху платформи

Задля реалізації поставленої цілі потрібно скласти мікропрограму дій платформи в залежності від типу маркеру. Найпростіший спосіб програмування маршруту – описати послідовність дій при виявленні маркеру з потрібним ID. Наприклад це можна зробити у вигляді словника, де ключем буде ID маркера, а його значенням – потрібна дія (рисунок 4.12).

```
commands_list = { 31:"turn_r",
                  32:"turn_r",
                  33:"wait_5s_then_turn_r",
                  32:"turn_l",
                  31:"turn_l",
                  30:"stop"}
```

Рисунок 4.12 – Описання команд в залежності від маркеру

При даному лістингу команд шлях роботизованої платформи буде виглядати приблизно наступним чином (рисунок 4.13).

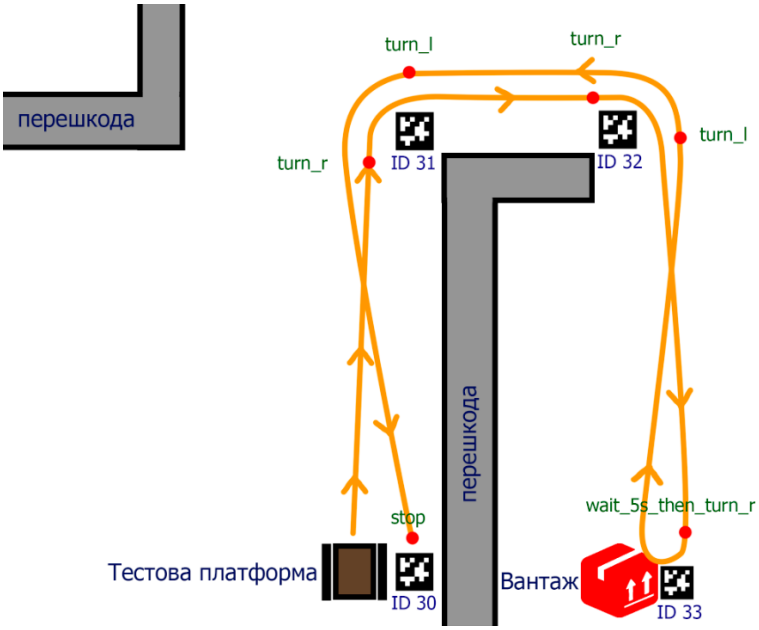

Рисунок 4.13 – Приблизний шлях руху платформи

Реалізація алгоритму виглядає наступним чином:

– платформа веде пошук першого маркеру, обертаючись з невеликою швидкістю;

– при детекції маркеру платформа коректує своє положення, намагаючись утримати маркер у центрі (+- 40 пікселів) (рисунок 4.14);

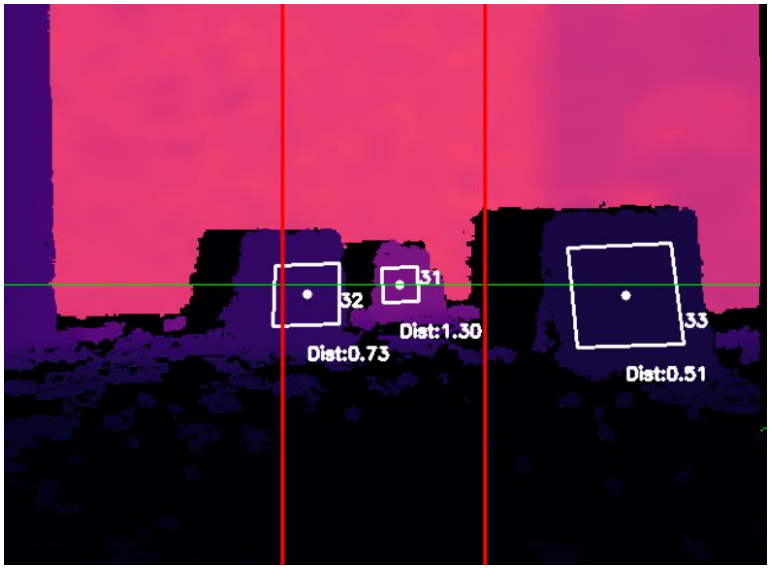

Рисунок 4.14 – Центр зображення, по якому корегується положення платформи

– платформа починає рухатися прямо, за 20 см до маркеру (камера глибини не може розпізнати відстань менше) вибирає наступне завдання зі списку, проїжджає вперед впродовж 0,5 с (проїхавши приблизно 10 см) та виконує завдання, наприклад розворот на право;

– при втрачанні маркера з поля зору платформа починає повільно обертатися, задля пошуку втраченого маркеру.

Код для операції пошуку, їзди до маркеру та поворотів представлений на рисунку 4.15.

```
def smallTurnL():
    pwmOutput_0.stop()
    pwmOutput_1.stop()
    pwmOutput_0.start(70)
    time.sleep(0.1)
    pwmOutput_0.stop()
    time.sleep(0.1)

def smallTurnR():
    pwmOutput_0.stop()
    pwmOutput_1.stop()
    pwmOutput_1.start(70)
    time.sleep(0.1)
    pwmOutput_1.stop()
    time.sleep(0.1)

def smallTurnSearch():
    runFowardStop()
    pwmOutput_1.start(50)
    time.sleep(0.3)
    pwmOutput_1.stop()
    time.sleep(0.1)
```

```
def allStop():
    pwmOutput_0.stop()
    pwmOutput_1.stop()

def runForward():
    pwmOutput_0.start(60)
    pwmOutput_1.start(60)

def smallForward():
    pwmOutput_0.start(60)
    pwmOutput_1.start(60)
    time.sleep(0.5)
    allStop()
```

Рисунок 4.15 – Функції руху платформи

Таким чином використовуючи набуті навички та розроблені алгоритми для роботизованої автономної платформи можна розробляти універсальні автоматизовані транспортні засоби для роботи у складських приміщеннях, на підприємствах та у інших галузях. При комбінації даного методу аналізу оточення та пошуку маркерів з методами навігації та аналізу маршрутів можна розробляти роботів доставщиків або автоматизовані платформи для моніторингу навколишньої території.

4.4 Виробнича санітарія в лабораторі

Роботи в лабораторії відносяться до робіт категорії 1а – легка фізична робота, яка виконується сидячи.

Оптимальні норми мікроклімату згідно з ДСН 3.3.6.042-99: в холодний період року – температура 22-24 °С; відносна вологість 40-60 %; швидкість руху повітря не більше 0,1 м/с. У теплий період року – температура 23-25 °С; відносна вологість 40-60 %; швидкість руху повітря не більше 0,1 м/с. Забезпечується за допомогою загальнообмінної вентиляції [17].

Шум та вібрація у робочому приміщенні. У приміщенні технічного відділу причинної шуму і вібрації являються апарати, прилади і устаткування: друкуючі пристрої, комп'ютери, вентилятори, кондиціонер та ін. При їхній роботі рівень вібрації не вище 33 дБ, рівень шуму не повинен перевищувати 50 дБА, що є нормою для даного виду діяльності відповідно до НПАОП 0.00-1.28- 2010 [17].

Заходи по забезпеченню встановлених норм: використання спеціальних шум-поглинаючих перегородок, застосування меблів, які сприяють зменшенню шуму і вібрації, установка апаратів і приладів на спеціальні амортизуючи підкладки.

Електробезпека. Для живлення устаткування (ПЕОМ, освітлювальні прилади) які є однофазними споживачами використовується трифазна мережа 380/220 В частотою 50 Гц з глухо заземленої нейтралі. Із цієї причини при роботі з електроприладами існує потенційна небезпека ураження людини електричним струмом, тому в правилах устрою електроустановок (згідно ПУЕ [18]) передбачені наступні заходи електробезпеки: конструктивні, схемноконструктивні й експлуатаційні.

Конструктивні – вимоги що забезпечують захист від доторкання персоналу до струмоведучих частин. ПЕОМ мають ступінь захисту IP 44. Прилади освітлення IP-23.

Схемно-конструктивним заходом захисту є занулення електрообладнання у приміщенні. Для користувача ПЕОМ важливим є дотримання правил безпеки експлуатації електрообладнання. Так, заборонено доторкатися до дротів та з’єднань при наявності напруги в мережі, а також 96 самостійно проводити ремонт електрообладнання. Усі питання щодо ремонту налагодження та інше, можуть виконувати тільки електрики та від повідні фахівці, які мають допуск до роботи із електрообладнанням певної категорії.

4.5 Висновки по четвертому розділу

В даному розділі були розроблені програмні засоби та алгоритми пошуку маркерів Aruco та їх ідентифікації, розроблено та протестована систему визначення відстані до перешкод відносно тестової платформи, яка виявилась дуже точною для камер глибини такого типу (похибка 2 %), та розробили спосіб задання завдань та маршрутів для роботи тестової платформи.

ВИСНОВКИ

В результаті виконання даної магістерської атестаційної роботи були розглянуті основні види систем автоматичного керування транспортними засобами. Розглянуто аналогічні рішення закордонних компаній. Були розглянуті сфери використання засобів автоматизації керування. Були представлені та проаналізовані деякі з сенсорів що присутні у багатьох системах.

Розроблено алгоритм керування тестовою платформою на основі камери глибини, яка орієнтується у просторі за рахунок маркерів Aruco.

Була змодельована та розроблена програмно-апаратна система, що реалізує розроблений алгоритм автоматичного керування та аналізу оточення.

Були проведені тести, в результаті яких була перевірена працездатність кожної окремої системи.

Таким чином, результаті виконання даної атестаційної роботи було отримано систему, що має ряд переваг перед існуючими аналогічними рішеннями, а саме: наявність програмної та апаратної реалізації аналізу оточення за рахунок камери глибини, наявність гнучкої та надійної основи для проектування нових систем.

У подальшому, для покращення роботи системи необхідно використовувати більш дорогі та ресурсо-доступні компоненти. Це потрібно для того щоб система могла більш швидше реагувати та обробляти зображення оточення, і на основі цієї інформації керувала транспортним засобом з більшою точністю.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Локализация по Aruco маркерам [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://habr.com/ru/post/482220/ – 12.10.2021.

2. ДСТУ 3008-15. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення [Текст]. – Введ. 2015-06-22. – К.: Держстандарт України, 2017. – 29 с.

3. Невлюдов І.Ш. Методичні вказівки з підготовки й оформлення кваліфікаційної роботи здобувачами другого (магістерського) рівня вищої освіти спеціальності 151 Автоматизація та комп'ютерно-інтегровані технології, освітньо-професійних програм: «Автоматизоване управління технологічними процесами», «Комп'ютерно-інтегровані технологічні процеси і виробництва», «Комп'ютеризовані та робототехнічні системи» / Упоряд. І. Ш. Невлюдов, Р. В. Артюх, Н. П. Демська, В. В. Євсєєв, О. І. Филипенко, О. М. Цимбал. – Харків: ХНУРЕ, 2021. – 50 с.

4. Cognitive Pilot (ООО «Когнитив Роботикс») [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://cognitivepilot.com/ – 07.11.2021.

5. TuSimple: Autonomous Trucking - A Better Path Forward [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://www.tusimple.com/ – 08.11.2021.

6. Artificial Intelligence & Autopilot | Tesla [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://www.tesla.com/AI – 11.11.2021.

7. RegNet or How to methodologically design effective networks [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://medium.com/analytics-vidhya/regnet-or-how-to-methodologically-design-effective-networks-c3560c1cf436 – 04.11.2021.

8. Беспилотный летательный аппарат [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://ru.wikipedia.org/wiki/%D0%91%D0%B5%D1%81%D0%BF%D0%B8%D0%BB%D0%BE%D1%82%D0%BD%D1%8B%D0%B9_%D0%BB%D0%B5%D1%82%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B0%D0%BF%D0%BF%D0%B0%D1%80%D0%B0%D1%82 – 05.11.2021.

9. Comma – introducing comma three [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://comma.ai/ – 07.11.2021.

10. Роботизация складов Amazon [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://vc.ru/story/163571-robotizaciya-skladov-amazon-privela-k-rostu-travm-na-50-i-usileniyu-nagruzki-na-lyudey – 07.11.2021.

11. Velodyne Lidar: Smart Powerful Lidar Solutions [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://velodynelidar.com/ – 03.11.2021.

12. Технология Intel® RealSense™ [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://www.intel.ru/content/www/ru/ru/architecture-and-technology/realsense-overview.html – 07.11.2021.

13. Implement Simultaneous Localization And Mapping (SLAM) with Lidar Scans [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://ch.mathworks.com/help/nav/ug/implement-simultaneous-localization-and-mapping-with-lidar-scans.html – 08.11.2021.

14. Система навигации и построения карт vSLAM [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://bestrobot.info/lifehack/sistema-navigacii-i-postroeniya-kart-vslam/ – 07.11.2021.

15. Zumo Robot [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://www.pololu.com/product/2506 – 07.11.2021.

16. Raspberry Pi 4 [Електронний ресурс]. – Електрон. текстові дані. – Режим доступу: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/ – 07.11.2021.

17. ПУЕ-2011. Правила устройства электроустановок [Текст]. – Введ. 2006-10-06. – К. : Об'єднання енергетичних підприємств «галузевий резервно-інвестиційний фонд розвитку енергетики», 2006. – 34 с.