Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

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

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Перший (бакалаврський)
(рівень вищої освіти)

Розробка системи автоматизації управління роботом секретарем
(тема)

Виконав:
студент 4 курсу, групи АКТСІ-20-3
Проценко Д.Є.,
(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 Системна інженерія
(повна назва освітньої програми)
Керівник проф. каф. КІТАР Євсеєв В.В.
(посада, прізвище, ініціали)

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

2024 р.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет АКТ
Кафедра КІТАР
Рівень вищої освіти перший (бакалаврський)
Спеціальність 151 Автоматизація та комп`ютерно-інтегровані технології
Тип програми Освітньо-професійна
Освітня програма Системна інженерія
(шифр і назва)

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

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Проценко Дмитру Євгенійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи: Розробка системи автоматизації управління роботом секретарем
Затверджена наказом № 545 Ст по університету від 03.06.2024
2. Термін подання студентом роботи до екзаменаційної комісії 14.06.2024 р.
3. Вихідні дані до роботи
3.1 Вибір мікрокомп'ютера;
3.2 Середовище розробки: Python IDE;
3.3 Бібліотеки: speech recognition, Adafruit_SSD1306, threading;
3.4 Мова програмування: Python;
3.5 Операційна ситсема: Raspbian OS.

4. Перелік питань, що потрібно опрацювати в роботі:
4.1 Вступ;
4.2 Аналіз сучасних роботів секретарей;
4.3 Розробка макета робота секретаря;
4.4 Розробка системи керування;
4.5 Висновки

4. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій Графічний демонстраційний матеріал в форматі PowerPoint(*.pptx) формату А4.

5. 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	Приміт-ка
1	Аналіз літератури по темі роботи	23.02.24 – 04.03.24	Виконано
2	Аналіз технічного завдання	10.03.24 – 20.03.24	Виконано
3	Аналіз сучасних роботів – секретарей	25.03.24 – 30.03.24	Виконано
4	Розробка макету	05.04.24 – 27.04.24	Виконано
5	Розробка програми керування роботом - секретарем	10.04.23 – 15.05.24	Виконано
6	Експериментальні підтвердження	17.05.24 – 28.05.24	Виконано
7	Розрахунки заземлення	29.05.24 – 02.06.24	Виконано
8	Подання роботи на перевірку Інтернет-сервісом	03.06.24 – 05.06.24	Виконано
9	Оформлення пояснювальної записки	05.06.24 – 08.06.24	Виконано
10	Подання роботи на рецензію	10.06.24	
11	Подання роботи на підпис зав. кафедри	12.06.24	
12	Подання кваліфікаційної роботи в ЕК	14.06.24	

Дата видачі завдання 17.02.2024

Студент _____ Проценко Д.Є.
(підпис) (прізвище, ініціали)

Керівник роботи _____ Євсєєв В.В.
(підпис) (прізвище, ініціал

РЕФЕРАТ

Пояснювальна записка: 73с., 23 рис., 2 дод., 25 джерел.

RASPBERRY PI ZERO W, PYTHON, SPEECH RECOGNITION, РОБОТ СЕКРЕТАР, РОЗПІЗНАВАННЯ ГОЛОСУ, АВТОМАТИЗАЦІЯ УПРАВЛІННЯ

Мета роботи – Розробка системи автоматизації управління мобільним роботом – секретарем, для покращення взаємодії користувача в рамкамах колоборативних роботів.

Об’єкт розробки – процес управління мобільним роботом.

Предмет розробки – моделі, алгорітмічне та програмне забезпечення голосового керування мобільним роботом.

В ході виконання кваліфікаційної роботи була розроблена система автоматизації системи керуванням роботом – секретарем. Проведено аналіз сучасних голосових асистентів, настільних роботів, апаратних рішень, методів розпізнавання голосових команд та розпізнавання навколишнього середовища заавдяки комбінації УЗ датчиків та ІЧ датчику ліній. Також були виявлені базові функції, якими повинен володіти робот – секретар.

Наступним етапом стала розробка макету робота секретаря на базі мікрокомп’ютера Raspberry Pi zero W із інстальованою на ній операційною системою Raspbian. Мовою програмування було обрано Python.

Для розробленої системи були проведені експериментальні дослідження на виявлення якості виконання голосових команд при різних уровнях зайвого шуму.

ABSTRACT

The explanatory note contains 73p., 23 drawings, 2 pp., 25 sources.

RASPBERRY PI ZERO W, PYTHON, SPEECH RECOGNITION, ROBOT SECRETARY, VOICE RECOGNITION, MANAGEMENT AUTOMATION

The purpose of the work is to develop a system for automating the management of a mobile robot - a secretary, to improve user interaction within the framework of collaborative robots.

The object of development is the process of managing a mobile robot.

The subject of development is models, algorithmic and software for voice control of a mobile robot.

In the course of the qualification work, a system for automating the control system of the robot secretary was developed. An analysis of modern voice assistants, desktop robots, hardware solutions, methods of recognizing voice commands and recognizing the environment using a combination of ultrasound sensors and IR line sensors was carried out. The basic functions that a robot secretary should possess were also identified.

The next stage was the development of a model of a secretary robot based on a Raspberry Pi zero W microcomputer with the Raspbian operating system installed on it. Python was chosen as the programming language.

For the developed system, experimental studies were conducted to identify the quality of voice command execution at different levels of excessive noise.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ІЧ – інфрачервоний;

ОС – операційна система;

РС – робот – секретар;

УЗ – ультразвук;

GPIO (англ. General-Gurpose Input/Output) – інтерфейс для зв’язку між компонентами;

HDMI (англ. High Definition Multimedia Interface) – інтерфейс для мультимедіа високої чіткості;

I2C (англ. Inter-Integrated Circuit) – послідовна асиметрична шина;

SCL (англ. Serial CLock) – лінія синхронізації;

SDI (англ. Serial Digital Interface) – лінія даних;

USB (англ. Universal Serial Bus) – послідовний інтерфейс для підключення периферійних пристроїв.

ВСТУП

На сьогоднішній день існує велика кількість комп’ютеризованих помічників. Їхнє існування спрощує планування завдань на майбутнє, та допомагає їх виконувати. Є багато прикладів таких помічників: Apple Siri, Google Assistant, Microsoft Cortana, Amazon Alexa, та інші. Вони допомагають планувати день, шукати, записувати та повідомляти необхідну інформацію. Завдяки додатковому обладнанню, можна керувати іншими речами: світло, телевізор та домашні роботи.

Домашні роботи – ще одні помічники, які спрощують наше життя. Вони можуть виконувати побутову справу, поки ви займаєтесь своїми речами. Тому на ринку існує велика кількість роботів для прибирання вдома: робот – пилосос, очисник плитки, вікон, басейну та інші.

Мета роботи – Розробка системи автоматизації управління мобільним роботом – секретарем, для покращення взаємодії користувача в рамкамах колоборативних роботів.

Об’єкт розробки – процес управління мобільним роботом.

Предмет розробки – моделі, алгорітмічне та програмне забезпечення голосового керування мобільним роботом.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– провести аналіз існуючих настільних роботів та голосових асистентів;

– розробити структурну схему макету;

– провести підбір елементної бази;

– розробити алгоритм розпізнавання голосу;

– розробити алгоритм розпізнавання команд;

– вибір методу розпізнавання оточення;

– розробити алгоритм пересування;

– провести експериментальні дослідження.

Кваліфікаційна робота виконана згідно ДСТУ 3008 – 15 [1] та керуючись

навчальним посібником з дипломного проекту [2] та методичними вказівками [3].

1 АНАЛІЗ СУЧАСНИХ РОБОТІВ СЕКРЕТАРЕЙ

1.1 Аналіз існуючих роботів секретарів

На сьогоднішній день роботів секретарів немає на ринку. Натомість, існують так звані асистенти. Це програмне забезпечення, яке допомагає шукати інформацію, робити нотатки ти виконувати прості задачі в межах їхнього функціоналу. Такими асистентами є: Siri від Apple, Google Assistant від Google, Alexa від Amazon, Cortana від Microsoft та інші.

Також, існують колонки від деяких компаній, в які вбудовані ці асистенти. Такі колонки можуть не тільки шукати контент, а і виводити його через динаміки або транслювати на телевізор. Такі асистенти можуть взаємодіяти із різними пристроями, до яких вони підключені: роботи пилососи, смарт пристрої та інші. Фото колонок зображено на рисунках 1.1 – 1.3.

Рисунок 1.1 – Apple HomePod [8]

Рисунок 1.2 – Google Home [9]

Рисунок 1.3 – Amazon Echo Dot [10]

Також, сьогодні набирають популярності настільни комунікаційні роботи. Їх задача – це розважати та комунікувати із користувачем. Ще такі роботи виконують пошук простої інформації, такої як погода, дата та час. Такими роботами є Emo robot AI, Vector Robot, Eilik та інші. Фото роботів зображено на рисунках 1.4 – 1.6.

Рисунок 1.4 – Emo robot AI [11]

Рисунок 1.5 – Vector Robot [12]

Рисунок 1.5 – Eilik [13]

Як можемо побачити, всі вони відрізняються між собою: Emo пересувається завдяки способу перекачування з ноги на ногу, Vector – завдяки гусеницям, а ось Eilik взагалі не пересувається, а стоїть на платформі.

Крім того, вони відрізняються функціоналом та способом взаємодії із оточенням. Emo не має рук, проте має яскравий великий екран, тому він показує свої емоції завдяки анімаціям та рухам ніг (перекачування, танцювання та інші рухи). Vector замість рук має підіймаючий гак, завдяки якому може не тільки підіймати якісь речі, а і контрольовано перевертати себе, для виразу своїх емоцій. Eilik жестикулює руками та зображує емоції на

своєму екрані.

Розвиток робототехніки йде великими кроками, і не виключено, що в найближчому майбутньому ми побачимо появу роботів-секретарів, які зможуть виконувати більш складні задачі. Такі роботи могли б включати в себе інтеграцію з календарями, управління електронною поштою, організацію зустрічей та навіть участь у відеоконференціях.

Важливу роль у цьому процесі відіграють наступні технології:

– штучний інтелект (ШІ) – для аналізу даних та прийняття рішень;

– машинне навчання (МН) – для постійного поліпшення якості виконання завдань;

– інтернет речей (IoT) – для інтеграції з іншими розумними пристроями;

– розпізнавання мови та обробка природної мови (NLP) – для кращого розуміння користувацьких команд.

Хоча повністю автономні роботи-секретарі ще не з'явилися на ринку, сучасні програмні асистенти та комунікаційні роботи вже здатні значно полегшити виконання багатьох повсякденних задач, демонструючи потенціал для подальшого розвитку в цьому напрямку.

1.2 Аналіз систем керування роботів секретарів

Сучасні РС працюють на базі мобільних платформ, що дозволяє їм пересуватися у просторі та виконувати свої завдання більш автономно. Для цього вони використовують різноманітні датчики, які допомагають орієнтуватися, уникати перешкод та визначати межі робочої поверхні.

Найбільш поширеним датчиком для орієнтування є ультразвуковий датчик HC-SR04, зображений на рисунку 1.6. Зазвичай його використовують для обминання перешкод.

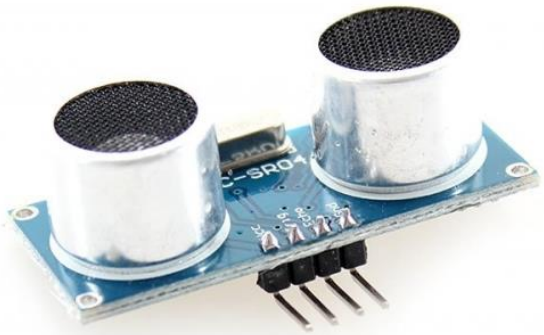

Рисунок 1.6 – Ультразвуковий датчик HC-SR04 [14]

Його принцип дії базується на використанні УЗ хвиль для вимірювання відстані до об'єктів. Датчик відправляє згенеровані імпульси із частотою 40 кГц та слухає луну. Ефективно виявляє рух з відривом до 4,5 метрів. Сліпа пляма знаходиться у двох сантиметрах перед датчиком. Живлення стандартне 5 вольт [14]. Завдяки цьому робота та налаштування датчику досить просте. Цей датчик є надзвичайно ефективним для виявлення перешкод і запобігання зіткненням, що робить його ідеальним для мобільних роботів.

Також можна додати додаткові датчики для визначення краю поверхні, по якому мобільна платформа пересувається. Це необхідно при проектуванні настільної мобільної платформи, щоб вона не падала зі столу. Одним з таких датчиків є KY-033 – датчик ліній, зображений на рисунку 1.7.

Рисунок 1.7 – Датчик ліній KY-033 на TCRT5000 [15]

Він може розрізняти чорний та білий кольори, для визначання ліній та слідкування по ним, та визначати відстань [15]. Тому через маленький розмір

цього датчику, його зручно використовувати при проектуванні мобільної платформи.

– для покращення орієнтації в просторі та взаємодії з оточенням РС можуть використовувати додаткові датчики та технології;

– камери та системи комп'ютерного зору – для розпізнавання об'єктів, облич та аналізу оточення;

– лідари (LIDAR) – для створення тривимірних карт місцевості та детекції перешкод;

– гіроскопи та акселерометри – для стабілізації та точного контролю руху.

Окрім апаратного забезпечення, РС потребують складного програмного забезпечення для ефективної роботи. Основні компоненти програмного забезпечення включають:

– системи навігації та маппінгу (SLAM) – для створення та оновлення карт оточення в режимі реального часу;

– алгоритми уникнення перешкод – для безпечного пересування в складних умовах;

– інтерфейси користувача – для зручного та інтуїтивного управління роботом.

Сучасні РС використовують широкий спектр датчиків та технологій для орієнтування в просторі та виконання своїх задач. УЗ датчики HC-SR04 та датчики ліній KY-033 є важливими компонентами для забезпечення безпечного та ефективного пересування. Поєднання апаратного та програмного забезпечення дозволяє створювати все більш розумні та автономні системи, які можуть значно полегшити виконання повсякденних завдань.

1.3 Аналіз базових функцій робота секретаря

Через те, що PC настільний, він може виконувати базові задачі, пов’язані із пошуком, записом та виводом інформації. Також робот асистент повинен комунікувати із іншими пристроями користувача. Наприклад вивід великої інформації не на самому роботі, а на телефон або комп’ютер. Таким чином користувач зможе зручно ознайомитись із наданою їй інформацією від робота.

Ще корисні функції, які повинна бути в PC – це функції календаря, годинника, будильника та таймера. Функція календар повинен записувати та зберігати події. При запиті, робот повинен виводити які будуть події в задану дату. Функція годинник при запиті повинна виводити дату та час на даний момент. Функція будильник повинен відтворювати звукове повідомлення, коли настане той час, який користувач зазначив під час налаштування цієї функції. Функція таймер повинна відтворювати звукове повідомлення, коли закінчиться той час, який було зазначено під час налаштування цієї функції. Також, використовуючи додаткові датчики, можна виводити додаткові данні, такі як: температуру, вологість, насичення кисню в повітрі та інші.

Роботи-секретарі є потужними інструментами, які можуть значно спростити та оптимізувати робочі процеси користувачів. Вони здатні виконувати широкий спектр завдань, починаючи від пошуку інформації та ведення календаря, до інтеграції з системами розумного будинку та іншими пристроями. Використання сучасних датчиків та передових технологій робить їх незамінними помічниками в повсякденному житті.

2 РОЗРОБКА МАКЕТА РОБОТА СЕКРЕТАРЯ

2.1 Розробка загальної структурної схеми

Першим етапом розробки будь-якої системи або виробу це розробка структури. Структура описує модулі процесів, апаратні рішення, програмні рішення та їх взаємозв’язок.

Розроблена структурна схема системи на базі Raspberry Pi zero W зображена на рисунку 2.1.

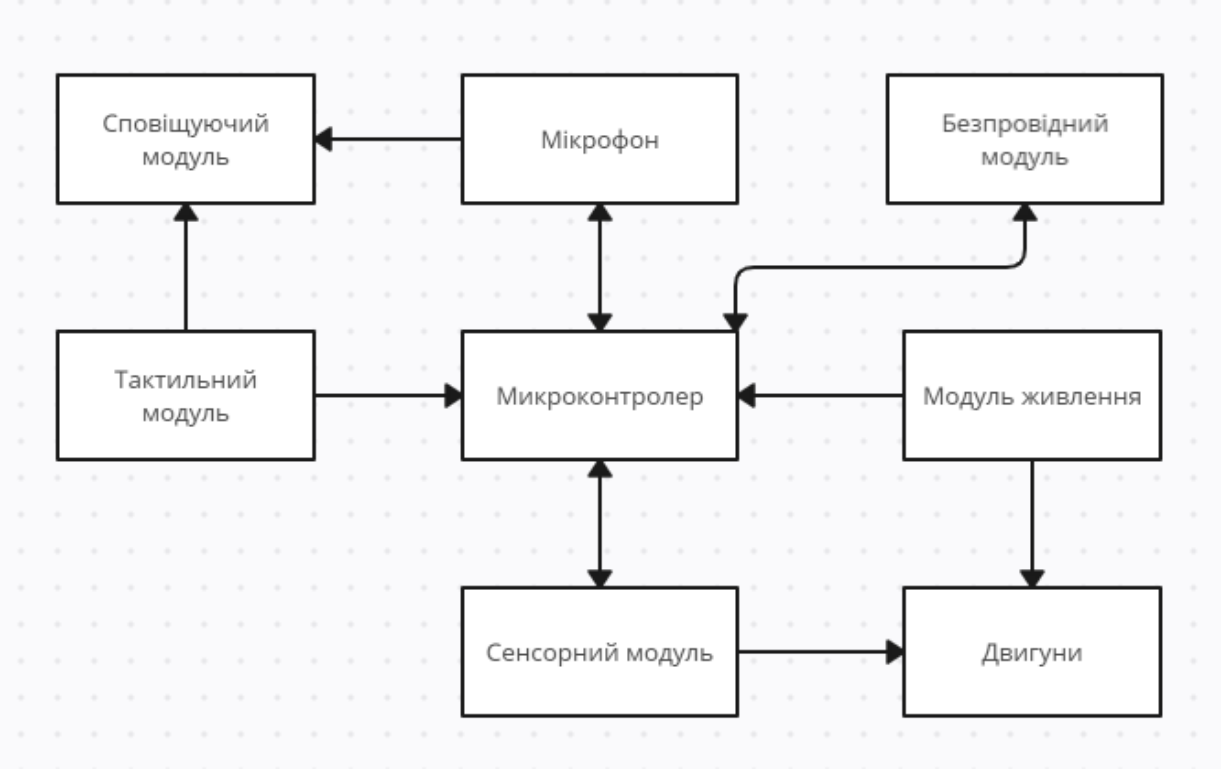

Рисунок 2.1 – Структурна схема системи

Система складається з наступних блоків:

– мікроконтролер – це апаратний модуль, на базі Raspberry Pi zero W, який призначений для виконання розрахунків, обробки данних та посилання сигналів на інші модулі, для виконання задачі;

– модуль живлення – апаратний модуль, призначений для живлення всіх інших модулів;

– сенсорний модуль – апаратний модуль, який використовується для сканування оточення та виявлення перешкод;

– двигуни – апаратний модуль, який використовується для переміщення мобільної платформи;

– безпровідний модуль – апаратний модуль, призначений для передачі інформації, та з'єднання із серверами для роботи бібліотеки Speech Recognition;

– мікрофон – апаратний модуль, який збирає звукову інформацію з оточення;

– тактильний модуль – апаратний модуль, який виконує розпізнавання доторкання в спеціальній зоні, для посилання сигналу та виконування запрограмованих на нього функцій;

– сповіщуючий модуль – апаратний модуль, який сповіщує користувачу інформацію через звук або дисплей.

Управління РС реалізовується на базі мови програмування Python.

Ці елементи формують основу системи на Raspberry Pi Zero W, яка може бути використана для різних застосувань, включаючи роботизовані системи, IoT пристрої та інші проекти з автоматизації і збору даних.

2.2 Аналіз та вибір апаратних модулів

Головним модулем для керування роботом являється мікроконтролер. Проаналізувавши ринок, було виявлено, що для поставленої задачі підходить мікрокомп'ютер Raspberry Pi zero W, зображений на рисунку 2.2.

Рисунок 2.2 – Raspberry Pi Zero W [16]

Характиристики Raspberry Pi Zero W:

- бездротова мережа: Wi-Fi 802.11 b/g/n;
- Bluetooth: BLE 4.1 на чіпі Cypress CYW43438;
- CPU: 1 ГГц процесор ARM11 на чіпі Broadcom BCM2835;
- ОЗУ: 512 МБ DDR2 ELPIDA;
- зовнішня пам'ять: слот карт пам'яті micro-SD;
- міні-HDMI: з роздільною здатністю на виході до 1080@p60;
- USB-порт: micro-USB з підтримкою On-The-Go (OTG);
- живлення: роз'єм USB micro-B для джерела на 5В/2А;
- GPIO: 40-контактний роз'єм;
- додаткові роз'єми: композитне відео, камера CSI.

Raspberry Pi Zero W це компактна версія старших аналогів. Незважаючи на розмір, плата має модуль Wi-Fi та Bluetooth, порти micro USB та mini-HDMI, що полегшує процес відлагудження не тільки на пряму а і через дистанційний доступ. Для роботи з платою необхідна карта пам’яті microSD на якій буде встановлена операційна система та джерело живлення [16].

Для орієнтування в просторі було застосовано УЗ датчик HC-SR04 та датчики ліній KY-033 на TCRT5000. Ця комбінація датчиків дозволить мобільний платформі оминати перешкоди та виявляти край поверхні, по якій пересувається платформа.

Для виявлення та запису навколишнього звуку, було обрано USB мікрофон MI-305 для Raspberry Pi, зображений на рисунку 2.3.

Рисунок 2.3 – USB мікрофон MI-305 для Raspberry Pi [17]

Цей мікрофон, попри малий розмір, вміє виявляти звук на потрібному рівні, не потребує додаткових драйверів для налаштування та підходить для проекту.

Для виводу зображення було обрано дисплей OLED S0.96” I2C із роздільною здатністю 128x64. Приклад дисплею зображено на рисунку 2.4.

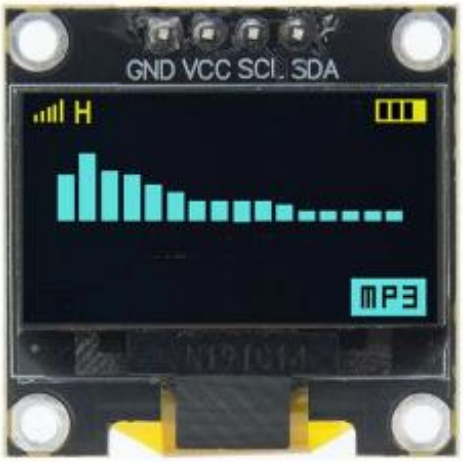

Рисунок 2.4 – Дисплей OLED S0.96” I2C [18]

Цей дисплей підключається через порти SCL та SDA, і працює із бібліотекою AdaFruit. Деякі специфікації дисплеїв можуть мати різні кольори пікселів. Завдяки цьому, можна вигадати різні сценарії використання дисплею [18].

Для руху робота були використані два двигуни редуктори. Зображення

двигуна можна побачити на рисунку 2.5.

Рисунок 2.5 – Двигун редуктор одно – осьовий [20]

В комплекті із двигунами входять колеса, які під’єднаються в спеціальний паз без додаткового закріплення.

Для керуванням цих моторів необхідно використовувати драйвер. Для цього було обрано драйвер на базі L293D, зображений на рисунку 2.5.

Рисунок 2.6 – Драйвер двигунів на базі L293D [21]

Драйвер має спеціальні зажими для проводів, які під’єднаються до двигунів. Це надасть більш надійне з’єднання та виключить випадковий розрив з’єднання [21].

Також в системі буде використана сенсорна кнопка, яка має буде виконувати важливу функцію – запуск основного алгоритму. Вона буде

працювати на зчитування дотику: якщо кнопка натиснута, тоді сигнал надсилається на плату. В протилежному випадку, кнопка сигнал не надсилає. Сенсорна кнопка, яка буде використовуватись в схемі, зображена на рисунку 2.7.

Рисунок 2.7 – Сенсорна кнопка [19]

2.3 Розробка схеми підключення

Головним елементом схеми є МК на базі Raspberry Pi Zero W, який виконує всі обчислення та керує іншими модулями. Для орієнтації у просторі використовуються УЗ датчики HC-SR04 та датчик ліній KY-033 на TCRT5000. Для переміщення використовуються одно-осеві мотори з редуктором 1:48, а для їхнього керування – драйвер на базі L293D. Схема підключення системи зображена на рисунку 2.5.

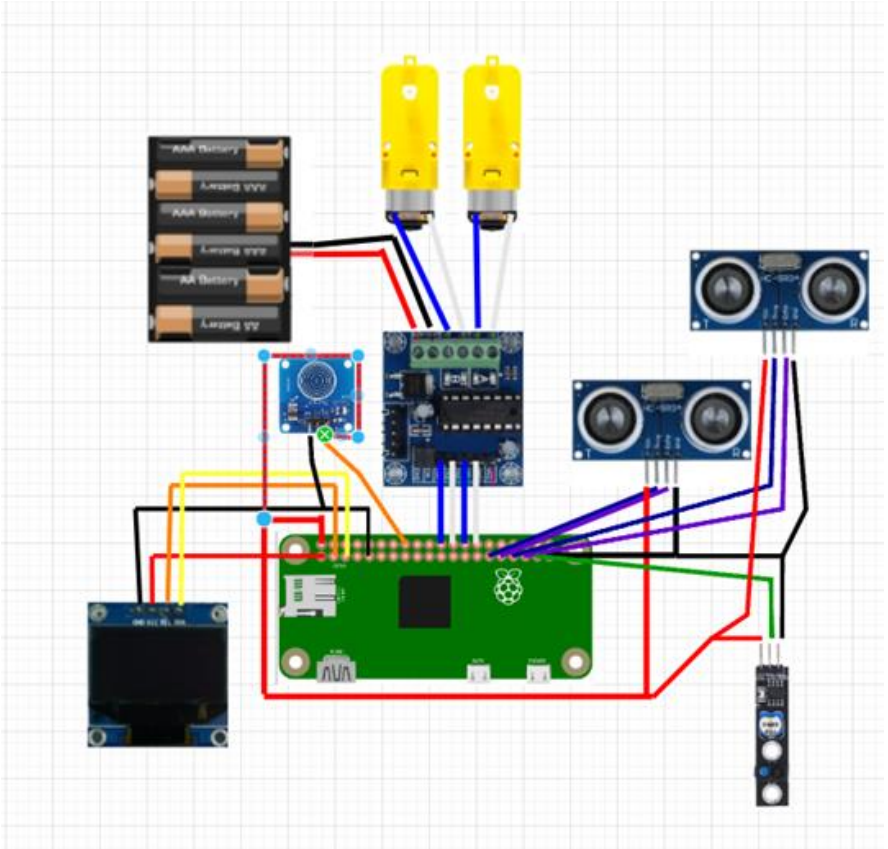

Рисунок 2.8 – Схема підключення

Для зручного з’єднування та живлення всіх компонентів була використана макетна плата.

Мотори редуктори під’єднані до драйверу за допомогою спеціальних зажимів. Це забезпечує надійне з’єднання та постійну проводимість сигналу на мотори. Для додаткового живлення моторів було додано блок для чотирьох акумуляторів типу АА. Кожний акумулятор видає напругу 1.5 V, тобто в сумі виходить 6 V. Цієї напруги вистачить для роботи моторів, та зменшить навантаження на основне джерело живлення усього робота.

Олед дисплей було під’єднано до GPIO 2 та 3, так як вони працюють із двонаправленими лініями зв’язку: SDA та SCL. Це дозволяє під’єднати до них пристрої, які працюють на базі I2C – асиметрична шина для зв’язку між інтегральними схемами електронних приборів. Живлення підключено напряму до плати, до піну напругою 3.3 V [18].

Сенсорна кнопка була підключена до вільного піну. Вона може

працювати в двох режимах – як кнопка або як перемикач. В режимі кнопки, сенсорна подає сигнал тільки тоді, коли користувач торкається спеціальної зони, яка реагує на дотик. В режимі перемикача, після дотику кнопка зберігає значення сигналу до тих пір, поки користувач не натисне на панель знов. Під’єднано до живлення напругою 5 V.

УЗ датчики під’єднуються за допомогою чотирьох пінів: 2 піна на живлення та два піна сигналів. Пін живлення під’єднано до джерела напругою 5 V. Піни сигналів під’єднуються до пінів echo та trig. Це пов’язано із методом роботи самого датчика УЗ: сигнал поступає на пін echo та випускає УЗ волну, котра відштовхується від об’єкту на її шляху, та повертається до датчику. В цей момент датчик зчитує час, за котре повертається волна та надсилає сигнал на пін trig.

Датчик ліній має 3 піни: два на живлення та один на вивід інформації. Датчик під’єднано до джерела живлення 5 V. Датчик зчитує інформацію, та надсилає її через третій пін. Також датчик має вбудований потенціометр, для регулювання дальності реагування.

2.4 Розробка макета робота

Головною частиною макету робота є основа і корпус, де будуть розташовані компоненти. Завдяки цьому, можна акуратно розвести проводи до компонентів, для більш зручного налаштування.

Для каркасу та корпусу макета PC був використан конструктор Lego, через простоту збірки робота та доступність деталей. Через особливість та різноманітність деталей, можна використовувати їх у різних сценаріях. Також можна розташовувати деталі таким чином, щоб не комплектні запчастини підходили за розміром. Завдяки цьому, в макет було додано деякі зручності для роботи та налагодження робота, а саме: роз’єм USB, роз'єм HDMI, кулер, та простір для розміщення джерел живлення.

Роз'єм HDMI дає можливість виводити зображення системи на зовнішній дисплей. Також можна контролювати процес виконання команд, через середу розробки алгоритму роботи робота.

Роз’єм USB потрібен для підключення зовнішньої периферії, наприклад для клавіатури або миші. Завдяки USB та HDMI налаштування робота можна проводити безпосередньо на самому роботі.

Так як головна плата знаходиться всередині корпусу, їй потрібно охолодження для запобігання тротлінгу. Тому з протилежної сторони від плати розташований маленький кулер. З іншої сторони розташовані отвори для видуву теплого повітря. Фото макета зображено на рисунку 2.9.

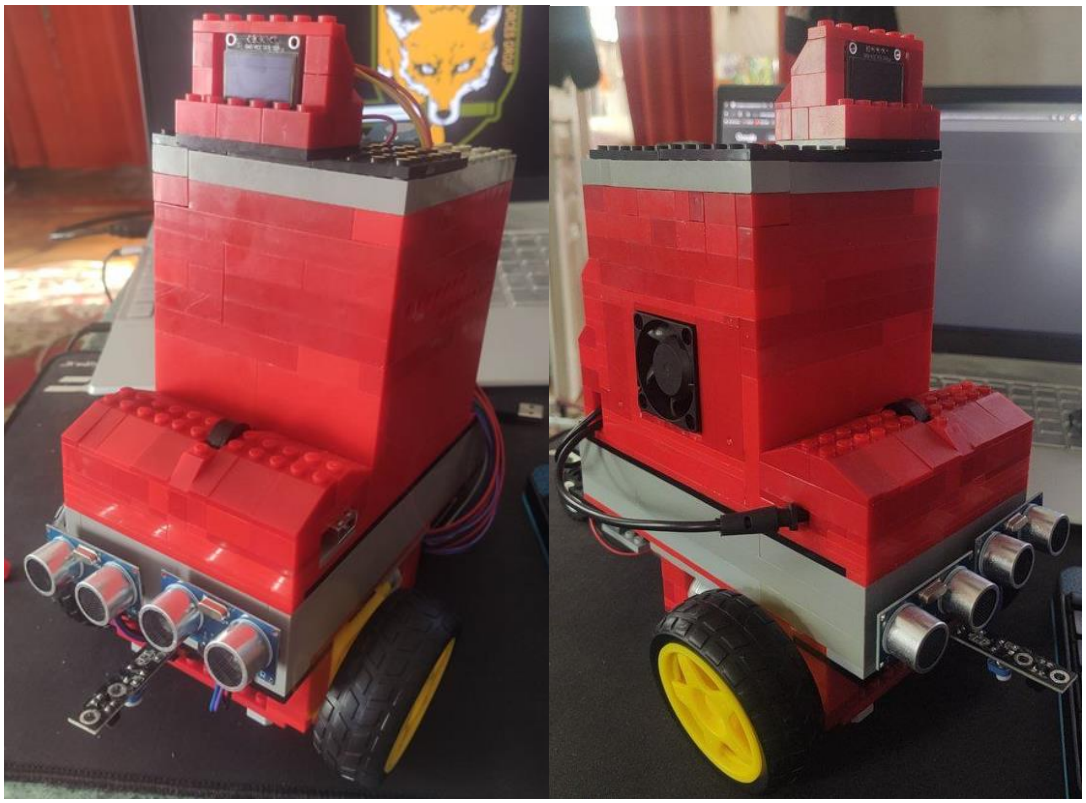

Рисунок 2.9 – Фото макета

2.5 Моделювання динамічної схеми мобільної платформи

Схема має такі блоки:

– $W_1(s)$ – перший УЗ датчик;

– $W_2(s)$ – другий УЗ датчик;

– $W_3(s)$ – ІЧ датчик лінії;

– $W_4(s)$ – МК;

– $W_5(s)$ – драйвер моторів;

– Output – двигуни.

Структурна схема мобільної платформи зображена на рисунку 2.10.

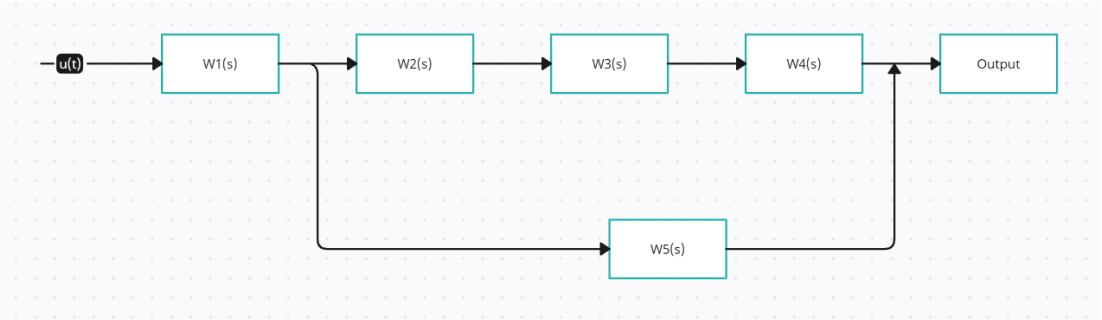

Рисунок 2.10 – Структурна схема мобільної платформи

Структурна схема дозволяє візуалізувати динаміку системи та зрозуміти значення блоків в роботі мобільної платформи.

Кожний блок має свою передавальну функцію. Вони предаставленні на формулах 2.1 – 2.4:

– блоки $W_1(s)$ та $W_2(s)$:

$$\frac{K_{us1,2}}{T_{us\ 1,2}s + 1} \tag{2.1}$$

– блок $W_3(s)$:

$$\frac{K_{line}}{T_{line}s + 1} \tag{2.2}$$

– блок $W_4(s)$:

$$K_p + \frac{K_i}{s} + K_d \cdot s \tag{2.3}$$

– блок $W_5(s)$:

$$K_{L293D} \tag{2.4}$$

– Output:

$$\frac{K_{motor}}{T_{motor}s + 1} \tag{2.5}$$

Для побудови графіків амплітудно-частотної характеристики (АЧХ) та фазово-частотної характеристики (ЛФЧХ) необхідно задати конкретні значення параметрів передавальних функцій. Маємо наступні значення для параметрів:

– K_{us1} = 1.5;
– T_{us1} = 0.05;
– K_{us2} = 1.5;
– T_{us2} = 0.05;
– K_{line} = 1.2;
– T_{line} = 0.05;
– K_p = 2;
– K_i = 1.5;
– K_d = 0.2;

– K_{L293D} = 1;

– K_{motoe} = 1.3;

– T_{motor} = 0.08.

Маючи дані, можемо побудувати графіки АЧХ та ЛФЧХ. Графіки зображані на рисунку 2.11.

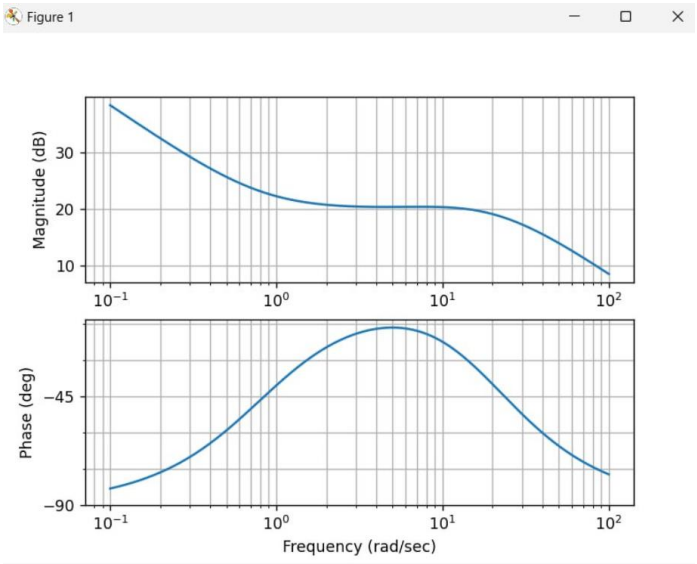

Рисунок 2.11 – Графіки АЧХ і ЛФЧХ відповідно

Аналізуючи графік АЧХ видно, що на низьких частотах амплітуда досить висока, але з ростом частоти амплітуда зменшується. Це означає, що система добре передає низькочастотні сигнали, але послаблює високочастотні сигнали.

На графіку ЛФЧХ можемо побачити, що фаза змінюється від -90 градусів на низьких частотах до приблизно -45 градусів на середніх частотах і знову зменшується до -90 градусів на високих частотах. Це показує, що на різних частотах система створює різний фазовий зсув.

3 РОЗРОБКА СИСТЕМИ КЕРУВАННЯ

3.1 Вибір середи розробки

МК Raspberry Pi Zero W може підтримувати різні ОС. Проте було виявлено, що для програмування РС вистачить ОС Raspbian – це збірка на базі Debian GNU/Linux створена спеціально для комп’ютерів Raspberry Pi [25].

Так як мовою програмування було обрано Python, вибір пав на дві середи програмування: Thonny та Python IDE.

Thonny має зручний інтерфейс, та має можливість завантажувати додаткові бібліотеки через внутрішній додаток. Проте, більшість потрібних для роботи додатків немає в переліку. Також при завантажуванні бібліотек через термінал, Thonny не бачить їх.

Python IDE має простий інтерфейс, проте добре працює із всіма бібліотеками і розпізнає їх, навіть якщо вони завантажені через термінал.

Отже, вибір Python IDE для розробки системи на Raspberry Pi Zero W є логічним кроком, особливо з урахуванням потреб у повній сумісності з бібліотеками та можливостями розширення функціоналу проекту.

3.2 Розробка загального алгоритму

Так як робот буде пересуватись, йому потрібно вміти об’їжджати перешкоди у різних випадках. Тому було розроблено алгоритм поведінки в різних ситуаціях: перешкода по центру робота, перешкода зліва від робота, перешкода справа від робота.

Якщо перешкода спереду від робота, він повинен оминути з будь якої сторони. Напрямок руху визначається випадково. Це до дасть до робота більш життєвості.

Якщо перешкода знаходиться зліва від робота, робот повинен оминути її з правої сторони, після чого рухається прямо. Так само і в ситуації, якщо перешкода з правої сторони, проте оминати він повинен з лівої сторони. Схема алгоритму зображено на рисунку 3.1.

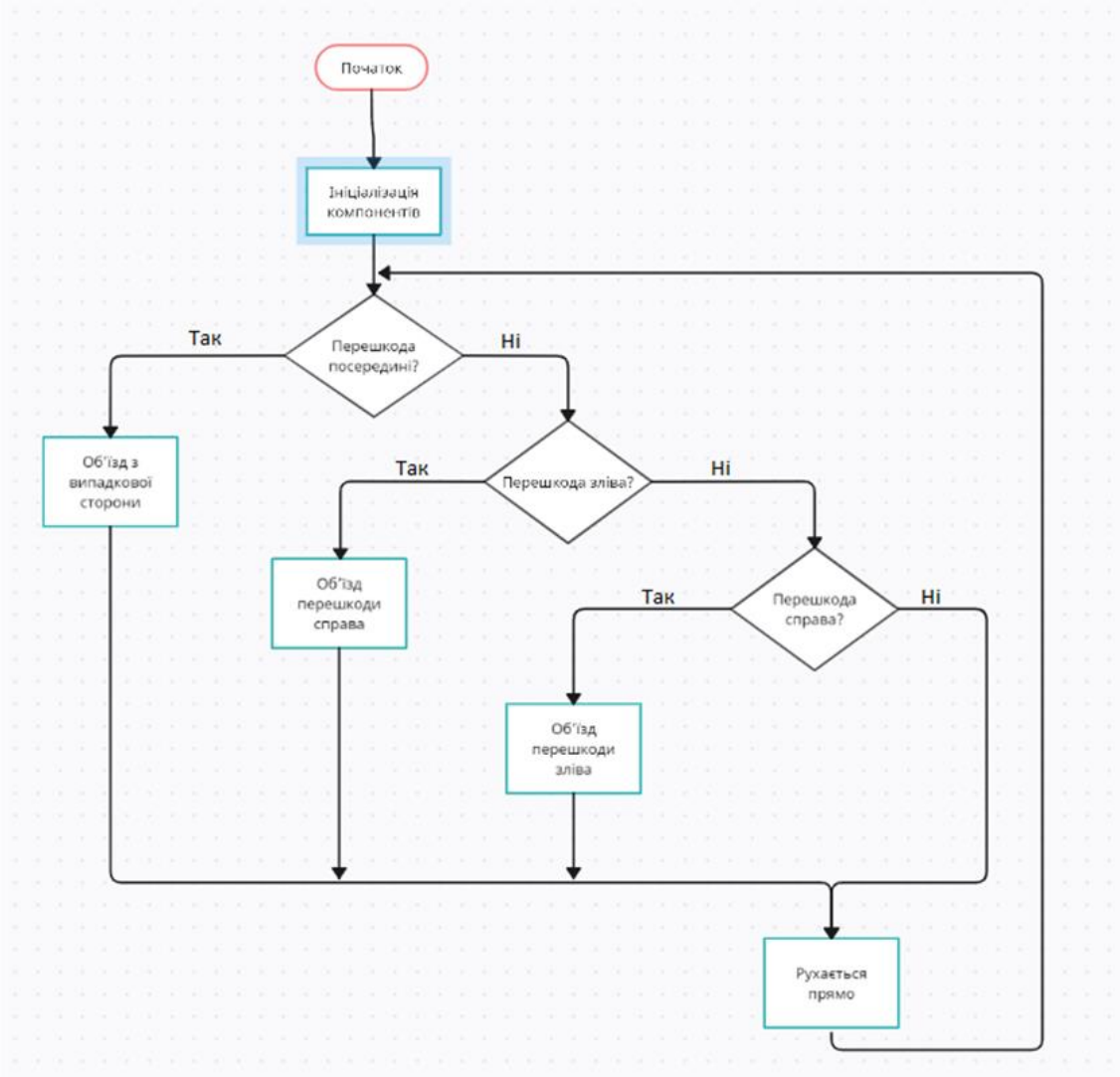

Рисунок 3.1 – Схема алгоритму руху робота

3.3 Розробка алгоритму обробки голосових команд

Основним методом зчитування команд в РС – це мікрофон. Тому було розроблено алгоритм обробки голосових команд.

Принцип дії такий: робот періодично записує звуки оточення і аналізує їх. Якщо в записі звуку не було знайдено ключову фразу для активації

подальшого алгоритму, робот заново запускає запис звуку оточення. Прикладом такої фрази може бути hello robot, після чого робот переходить в активний режим очікування команди. Якщо ж в записаному звуці було виявлено фразу, робот реагує на неї і чекає наступну команду. Це треба для того, щоб запобігти випадкових реагувань на команди, які були адресовані не роботу.

У випадку, коли після активації користувач каже неіснуючу команду або не каже зовсім нічого, робот повертається в стан очікування фрази для запуску головного алгоритму. Завдяки такому рішенню, робот буде повертатися в початковий стан очікування фрази, якщо користувач передумає давати команду.

Коли робот відреагував на фразу активації а потім і на команду з основного алгоритму, він виконує команду. Після виконання команди робот надає користувачеві зворотній зв'язок, наприклад, звуковим підтвердженням або світловим сигналом. Це дозволяє користувачеві бути впевненим, що команда була правильно зрозуміла і виконана.

Такий підхід до обробки голосових команд робить взаємодію з роботом інтуїтивною і зручною для користувача, забезпечуючи при цьому високу точність розпізнавання команд та безпеку даних.

Схема алгоритму розпізнавання голосових команд наведена на рисунку 3.2.

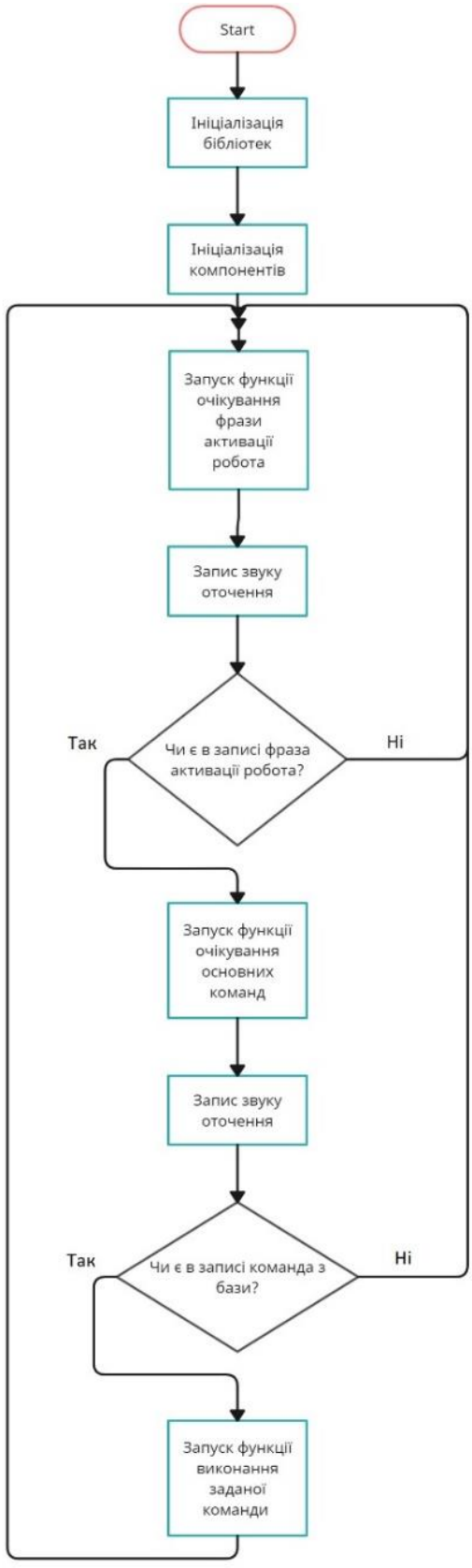

Рисунок 3.2 – Алгоритм розпізнавання голосових команд

3.4 Реалізація програмного забезпечення керуванням робота

Для реалізування пересування робота, роспізнавання голосу, виводу зображення на олед дісплей та роботи із повідомленнями для смартфону були застосовані додаткові спеціальні бібліотеки для роботи. За весь час розробки програмного керування РС, було задіяно такі бібліотеки:

```
import time
import speech_recognition as sr
import os
import socket
import random
import threading
import RPi.GPIO as GPIO
import Adafruit_GPIO.SPI as SPI
import Adafruit_SSD1306
from PIL import Image
from PIL import ImageDraw
from PIL import ImageFont
from pushbullet import Pushbullet
```

Бібліотека time потрібна для роботи із часом: визначення часу, секундомер і так далі. В коді бібліотека використовується для роботи із затримками між діями.

Speech_recognition потрібна для розпізнавання голосу, аналізу та його обробки. Завдяки цій бібліотеці, можна зчитувати звуки оточення, визначати слова. Бібліотека підтримує значну кількість мов, проте в ній відсутня українська, тому голосове керування реалізовано на англійській мові. Також однією з проблем даної бібліотеки – це неможливість роботи без підключення до інтернету [22].

Бібліотеки os та socket застосовуються в цій роботі для перевірки стану підключення до інтернету. Raspberry Pi zero W має вбудований Wi-Fi модуль, що є зручним для роботи, тому що не потрібно використовувати додаткові модулі. Також перевірка стану підключення до інтернету необхідна для роботи speech_recognition.

Якщо запустити програму без підключеного інтернету, функція розпізнавання команд просто не зможе проаналізувати голос і виведе помилку. Тому в коді реалізовано функцію перевірки. Приклад реалізації даної функції наведено нижче.

```
def check_internet_connection(host="8.8.8.8", port=53, timeout=3):
    try:
        socket.setdefaulttimeout(timeout)
        socket.socket(socket.AF_INET,
socket.SOCK_STREAM).connect((host, port))
        return True
    except Exception as ex:
        return False
if check_internet_connection():
    print("Інтернет з’єднання присутнє.")
    draw = ImageDraw.Draw(image)
    draw.rectangle((0,0,width,height), outline=0, fill=0)
    draw.text((20, 8),    'WI-FI CONNECTED', fill=255)
    disp.image(image)
    disp.display()
else:
    print("Інтернет з’єднання відсутнє.")
    draw = ImageDraw.Draw(image)
    draw.rectangle((0,0,width,height), outline=0, fill=0)
    draw.text((40, 8),    'NO WI-FI', fill=255)
```

```
disp.image(image)
disp.display()
```

Бібліотека random працює із випадковою генерацією чисел. В роботі вона використовується для вибіру руху робота. Це працює таким чином: генерується випадкове число від 1 до 5. Кожний рух має свій індекс, наприклад рух прямо має індекс один, рух ліворуч – 2 і так далі. Якщо під час перевірки датчиками оточення не виявлено перешкод, запускається функція random та визначається випадкове число. Після цього йде порівняння числа із індексами. Якщо число співпадає із індексом, то робот має рухатись в тому напрямку, який було зазначено у відповідному індексі. Приклад роботи бібліотеки наведено нижче:

```
random_number = random.randint(1, 5)
        if random_number == 1:
            GPIO.output(25, GPIO.HIGH)
            GPIO.output(8, GPIO.LOW)
            GPIO.output(7, GPIO.HIGH)
            GPIO.output(1, GPIO.LOW)
            time.sleep(0.5)
            GPIO.output(25, GPIO.LOW)
            GPIO.output(8, GPIO.LOW)
            GPIO.output(7, GPIO.LOW)
            GPIO.output(1, GPIO.LOW)
            time.sleep(5)
```

Threading дозволяє створювати та обробляти потоки одночасно, в незалежності один від одного. В програмі ця бібліотека використовується для паралельного керування пересуванням робота під час обробки голосових команд.

```
listen_thread = threading.Thread(target=move_robot)
listen_thread.start()
```

В зазначеному вище коді, ми створюємо потік, в який додаємо функцію move_robot, після чого визиваємо її. Таким чином, функція пересування робота не буде заважати функції роспізнаваня команд.

Бібліотека RPi.GPIO необхідна для роботи із Rapberry Pi та додатковими компонентами. Це пов’язано із тим, що надсилати сигнали на датчики та зчитувати сигнали з них можна через ніжки GPIO. Ця бібліотека дозволяє працювати із ніжками та підключати до них різні компоненти та датчики, а також надсилати та принімати сигнал від них. Нижче навелено приклад підключення та налаштування різних компонентів через GPIO.

```
GPIO.setmode(GPIO.BCM)
```

Ця команда визначає нумерацію ніжок GPIO. Таким чином при налаштуванні компонентів, треба звертатись до номерів контактів, визначених нумерацією мікросхем Broadcom, на відміну від номерів фізичних контактів у заголовку Raspberry Pi. Нумерацію пінів наведено на рисунку 3.3.

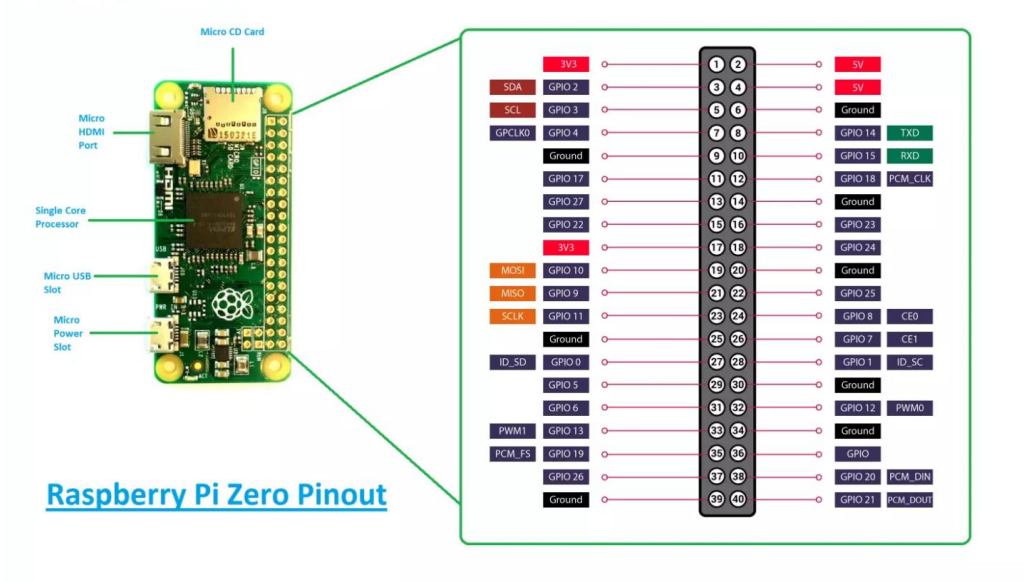

Рисунок 3.3 – Нумерація пінів GPIO [5]

```
GPIO.setwarnings(False)
```

Ця строка опціональна. Вона вимикає попередження при перезапуску програми у випадку, коли якісь компоненти ще були активні (наприклад двигуни)

```
GPIO.setup(25, GPIO.OUT)
GPIO.setup(8, GPIO.OUT)
GPIO.setup(7, GPIO.OUT)
GPIO.setup(1, GPIO.OUT)
GPIO.output(25, GPIO.LOW)
GPIO.output(8, GPIO.LOW)
GPIO.output(7, GPIO.LOW)
GPIO.output(1, GPIO.LOW)
```

В цьому фрагменті коду налаштовуються двигуни на вихід, тобто – на отримання сигналу. Також на їхні піни підключення ставиться значення LOW. Це потрібно для того, щоб мотори були в стані спокою, бо мотори працюють у випадку коли один контакт отримує позитивний заряд а протилежний – негативний. Можна також виставити на два контакти значення HIGH, але в такому випадку буде збільшена потреба в електроенергії з акумуляторів.

```
GPIO.setup(17, GPIO.IN, pull_up_down=GPIO.PUD_UP)
```

Ця строка відповідає за налаштування сенсорної кнопки. Так як кнопка надсилає сигнал на плату, вона підключена на вхід.

```
GPIO.setup(5, GPIO.OUT)
GPIO.setup(6, GPIO.IN)
GPIO.setup(13, GPIO.OUT)
```

GPIO.setup(19, GPIO.IN)

Тут наведено налаштування двох датчиків УЗ. Датчик УЗ має 2 піни для сигналів: перший пін відповідає за отримання сигналу, для посилання УЗ хвилі. Другий пін навпаки – відповідає за надсилання сигналу та реагує на відбиту хвилю, та надсилає сигнал на плату. Після підрахунку часу повернення сигналу, можна отримати приблизну дистанцію об’єкта.

GPIO.output(5, False)
GPIO.output(13, False)

Виставляється значення посилання УЗ хвилі на False, тобто – на вимкнено.

GPIO.setup(26, GPIO.IN)

Ця строка відповідає за налаштування датчику ліній.

Для виводу зображення для дісплею були застосовані бібліотеки Adafruit_GPIO.SPI, Adafruit_SSD1306 та PIL. Ці бібліотеки потрібні для підключення дисплею до пінів GPIO, та роботи із зображенням. Завдяки бібліотеці Adafruit_SSD1306 на дісплей можна виводити текст, анімації та фігури. Завдяки цьому із фігур було створено декілька анімацій очей робота, для повідомлення виконання функцій. Приклади лиць зображано на рисунку 3.4 [23].

Рисунок 3.4 – Анімація лиця робота

```
from pushbullet import Pushbullet
```

Ця бібліотека необхідна для роботи із додатком Pushbullet. Цей додаток надає можливість надсилати push повідомлення на смартфон та комп’ютер.

```
pb = Pushbullet("o.Rml8Oc48BcbdL*flVkVIxePM2qVnLgAK")
        print(pb.devices)
        dev = pb.get_device('Xiaomi Mi 9T')
        push = dev.push_note("Ваш Текст")
```

В цьому коді задається унікальний ідентифікаційний ключ, створений на вашому аккаунті. Далі визначається пристрій, на який ви хочете надіслати текст. Це може бути смартфон, комп’ютер або інший пристрій, на якому є можливість завантажити цей додаток. Наприклад для пристроїв від компанії Apple, програму Pushbullet неможливо завантажити, через відсутність підтримки операційних систем IOS, MacOS та iPadOS з боку розробника програмного забезпечення. Проте існують різні альтернативи, доступні для вище зазначених операційних систем, наприклад NTFY. В кінці ви наводите текст, який ви хочете щоб був надісланий на ваш пристрій. Розмір текста не має меж, можна відправляти як звичайні короткі повідомлення, так і великий текст.

Далі розглянемо цікаві функції:

```
def check_internet_connection(host="8.8.8.8", port=53, timeout=3):
    try:
        socket.setdefaulttimeout(timeout)
        socket.socket(socket.AF_INET,
socket.SOCK_STREAM).connect((host, port))
        return True
```

```
    except Exception as ex:
        return False
if check_internet_connection():
    print("Интернет соединение установлено.")
    draw = ImageDraw.Draw(image)
    draw.rectangle((0,0,width,height), outline=0, fill=0)
    draw.text((20, 8),    'WI-FI CONNECTED', fill=255)
    disp.image(image)
    disp.display()
else:
    print("Отсутствует интернет соединение.")
    draw = ImageDraw.Draw(image)
    draw.rectangle((0,0,width,height), outline=0, fill=0)
    draw.text((40, 8),    'NO WI-FI', fill=255)
    disp.image(image)
    disp.display()
```

Функція перевірки інтернет з’єднання. Якщо інтернет буде підключений, тоді на дисплей буде виведено текст 'WI-FI CONNECTED'. В протилежному випадку, якщо інтернет не буде підключений, буде виведено текст 'NO WI-FI'. Цей текст буде виведено до тих пір, поки не буде наявне з’єднання з інтернетом. Як тільки з’явиться інтернет, на дисплеї буде виведено текст 'WI-FI CONNECTED', і перевірка на з’єднання буде завершено.

```
def pressed_button():
    if GPIO.input(17) == True:
        print ('Button pressed')
        return wait_command()
    else:
        print ('Button don`t pressed')
```

Функція зчитування стану сенсорної кнопки. Якщо кнопка буде натиснута в момент зчитування, тоді буде запущено функцію wait_command().

```
def listen_for_hello_voxy():
    mic = sr.Microphone()
    with mic as source:
        r.adjust_for_ambient_noise(source)
        draw = ImageDraw.Draw(image)
        print('REC...')
        try:
            audio = r.listen(source, timeout=3)  # Adjust the timeout as needed
            text = r.recognize_google(audio)
            print(text)
            if 'hello voxy' in text or 'hello Vox' in text or 'hello boxing' in text or 'hello voxin' in text or 'hello foxy' in text or 'hello boxer' in text or 'hello voxing' in text or 'hello fox' in text:
                return wait_command()
            else:
                return False
        except sr.WaitTimeoutError:
            print("You didn't say anything.")
            return False
        except sr.UnknownValueError:
            print("Sorry, I could not understand what you said.")
            return False
        except sr.RequestError:
            print("Internet Connection Lost")
            return check_internet_connection()
```

Функція очікування голосової команди 'hello voxy' для запуску функції

wait_command(). Спочатку визначається мікрофон як пристрій для запису звуку. Після цього розпізнавач аналізує запис звуку, переводить його в текст та перевіряє його із базою. Справа в тому, що розпізнавач може перевести аудіо в текст некоректно, через схожість деяких слів. Тому шляхом експериментів було виявлено слова, які схожі на фразу 'hello voxy': 'hello Vox', 'hello boxing', 'hello voxin', 'hello foxy', 'hello boxer', 'hello voxing,' 'hello fox'. Завдяки цьому, можна збільшити ймовірність активації робота. Якщо команда була розпізнана, запускається функція wait_command(). Якщо команда не була роспізнана, функція запускається заново. Також в код включені ймовірні помилки:

– except sr.WaitTimeoutError Виникає якщо мікрофон не виявив ніякого звуку;

– except sr.UnknownValueError Виникає у випадку, коли користувач сказав невідому команду;

– except sr.RequestError Ця помилка виникає коли зникає інтернет з’єднання.

```
def move_robot():
    while x == 1 and y == 1:
        GPIO.output(5, True)
        time.sleep(0.00001)
        GPIO.output(5, False)
        while GPIO.input(6) == 0:
            pulse_start = time.time()
        while GPIO.input(6) == 1:
            pulse_end = time.time()
        pulse_duration = pulse_end - pulse_start
        distance1 = pulse_duration * 17150
        distance1 = round(distance1, 2)
        GPIO.output(13, True)
```

```
    time.sleep(0.00001)
    GPIO.output(13, False)
    while GPIO.input(19) == 0:
      pulse_start = time.time()
    while GPIO.input(19) == 1:
      pulse_end = time.time()
    pulse_duration = pulse_end - pulse_start
    distance2 = pulse_duration * 17150
    distance2 = round(distance2, 2)
```

Цей блок відповідає за рух робота та орієнтацію в просторі. Спочатку запускається датчик ліній і перевіряє, ч є попереду поверхня для пересування. Якщо її немає, робот від'їжджає назад. Далі запускаються датчики УЗ і перевіряють на наявність перешкод. Якщо перешкода по центру, тоді робот обирає випадкову стороно повороту. Якщо перешкода збоку, тоді робот об'їзджає її з протилежної сторони. У разі, якщо немає перешкод, робот обирає напрям руху випадково: він може поїхати прямо, повернутись ліворуч або праворуч, поїхати назад та просто стояти на місті. Це до дасть до робота трішки випадковості.

Також на початку всій функції здійснюється перевірка змінних x та y. Якщо вони будуть дорівнювати 1, робот буде рухатись, проте якщо одна з змінних не буде дорівнювати, тоді робот буде стояти.

```
def wait_command():
  x = 0
  mic = sr.Microphone()
  with mic as source:
    r.adjust_for_ambient_noise(source)
    draw = ImageDraw.Draw(image)
    draw.rectangle((0,0,width,height), outline=0, fill=0)
```

```
draw.ellipse((15, 10 , 55, 30), outline=1, fill=0)
draw.ellipse((72, 10 , 112, 30), outline=1, fill=0)
draw.ellipse((25, 15 , 45, 25), outline=1, fill=1)
draw.ellipse((82, 15 , 102, 25), outline=1, fill=1)
draw.rectangle((15,1,55,7), outline=0, fill=1)
draw.rectangle((72,1,112,7), outline=0, fill=1)
disp.image(image)
disp.display()
print('REC 2...')
try:
    audio = r.listen(source, timeout=5)  # Adjust the timeout as needed
    text = r.recognize_google(audio)
    print(text)
    if 'save event' in text or 'add event' in text or 'create event' in text:
        return save_event()
    elif 'check event' in text or 'show event' in text:
        return check_event()
    elif 'delete event' in text:
        return delete_event()
    elif 'stop' in text:
        y = 0
        return listen_for_hello_voxy()
    elif 'move' in text:
        y = 1
        return listen_for_hello_voxy()
    else:
        print('Error. unknow command')
        return listen_for_hello_voxy()
except sr.WaitTimeoutError:
    print("You didn't say anything.")
```

```
            return listen_for_hello_voxy()
        except sr.UnknownValueError:
            print("Sorry, I could not understand what you said.")
            return listen_for_hello_voxy()
        except sr.RequestError:
            print("Internet Connection Lost")
            return check_internet_connection()
```

wait_command() очікує на наступну команду. Принцип дії схожий на listen_for_hello_voxy(), проте має в собі більший перелік функцій.

По-перше, змінюється значення x на 0. Це необхідно для того, щоб виключити поміхи, які можуть виникнути при спробі розпізнання команди. При завершені функції wait_command(), змінна x знов буде дорівнювати 1.

По-друге, wait_command() має в собі більший перелік команд, а саме:

– save_event() #Функція збереження події;
– check_event() #Функція перевірки події в базі;
– delete_event() #Функція видалення події з бази.

В базі є команди для керування рухом робота. Це потрібно щоб користувач сам міг керувати станом руху.

Також в коді наведено виключені помилки, які можуть виникнути під час роботи. Помилки ідентичні тим, що були в listen_for_hello_voxy().

```
def save_event():
    mic = sr.Microphone()
    with mic as source:
        r.adjust_for_ambient_noise(source)
        print('Save event...')
        try:
            audio = r.listen(source, timeout=3)  # Adjust the timeout as needed
            text = r.recognize_google(audio)
```

```
            event = text
            with open("events.txt", "a") as file:
                file.write(f"{event}\n")
                print("loading")
                sleep(5)
                print("Event saved")
                pb=Pushbullet("o.Rml8Oc48BcbdL6flVkVIxePM2qVnLgAK")
                print(pb.devices)
                dev = pb.get_device('Xiaomi Mi 9T')
                push = dev.push_note("Event saved")
                return listen_for_hello_voxy()
```

Функція save_event() зберігає події в зовнішній файл. Користувач повинен назвати дату та подію, після чого подія зберігається в файлі. Також зберігання події супроводжується повідомленням на телефон. Додатково було налаштовано анімація екрану та помилки.

```
def check_event():
    mic = sr.Microphone()
    events_found = []  # To store all events found
    with mic as source:
        r.adjust_for_ambient_noise(source)
        print('Check event...')
        try:
            audio = r.listen(source, timeout=3)  # Adjust the timeout as needed
            text = r.recognize_google(audio)
            event = text
            with open("events.txt", "r") as file:
                for line in file:
                    if f"{event}" in line:
```

```
                events_found.append(line.strip())
        if events_found:
            pb = Pushbullet("o.Rml8Oc48BcbdL6flVkVIxePM2qVnLgAK")
            print(pb.devices)
            dev = pb.get_device('Xiaomi Mi 9T')
            for event in events_found:  # Push each found event
                push = dev.push_note("Event Notification", event)
                print(event)
            return listen_for_hello_voxy()
        else:
            print('No event')
            return listen_for_hello_voxy()
```

Ця функція перевіряє наявність події в файлі. Кристувач повинен назвати число і місяць, для більш точного виведення події. Знайдена подія або події надсилаються на телефон у вигляді повідомлення.

```
def delete_event():
    r = sr.Recognizer()
    mic = sr.Microphone()
    with mic as source:
        r.adjust_for_ambient_noise(source)
        print('Delete event...')
        try:
            audio = r.listen(source, timeout=3)
            text = r.recognize_google(audio)
            print(text)
            event = text.strip()
            with open("events.txt", "r") as file:
                lines = file.readlines()
```

```
            with open("events.txt", "w") as file:
                for line in lines:
                    if event in line:
                        print("Are you sure?")
                        confirm_audio = r.listen(source, timeout=5)
                        confirm_text = r.recognize_google(confirm_audio).lower()
                        if "yes" in confirm_text:
                            print("complete")
                            pb                                                    =
Pushbullet("o.Rml8Oc48BcbdL6flVkVIxePM2qVnLgAK")
                            print(pb.devices)
                            dev = pb.get_device('Xiaomi Mi 9T')
                            push = dev.push_note("Event deleted")
                            return listen_for_hello_voxy()
                        elif "no" in confirm_text or "cancel" in confirm_text:
                            print("complete")
                            print("process cancelled")
                            file.write(line)
                            return listen_for_hello_voxy()
                    else:
                        file.write(line)
                        return listen_for_hello_voxy()
            return check_internet_connection()
```

Функція delete_event() видаляє події з файлу. При введені дати система запитає, чи точно ви хочете видалити подію. Якщо так, то подія видалиться. Якщо ні, то процес видалення буде скасовано

3.5 Проведення експерементів

РС є невеликого розміру, тому експеременти з ним відбуваються в домашньому приміщені.

Було проведено два експерименти. Перший пов’язан із рухом РС, роспізнаванням перешкод завдяки УЗ датчикам та визначенням краю поверхні завдяки датчику ліній.

Принцип дії УЗ датчику полягає в тому, що датчик випускає УЗ хвилю, а потім фіксує її відлуніння від об’єкту. Таким чином можна визначити дистанцію до перешкоди таким чином:

– записати час початку випуску хвилі;

– записати час повернення хвилі;

– визначити час імпульса, визначивши різницю між початком і кінцем;

– за формулами 3.1 – 3.4 визначити дистанцію до об’єкту [6].

$$Speed = \frac{Distance}{Time} \tag{3.1}$$

$$34300 = \frac{Distance}{Time/2} \tag{3.2}$$

$$17150 = \frac{Disatnce}{Time} \tag{3.3}$$

$$17150 \cdot Time = Distance \tag{3.4}$$

Швидкість УЗ хвилі над рівнем моря дорівнює 343 м/с. Далі беремо час імпульсу, і ділемо його навпіл, бо нам потрібен час імпульсу, який доходить до об’єкта. І в кінці ми використовуємо стандартну формулу розрахунку

дистанціїї з інформаціїї про час та швидкість. Таким чином датчик досить чітко розпізнає дистанцію до об’єкту. Проте має невелику погрішність на ± 0,005 см що не є дуже кретичним в роботі.

Далі розглянемо принцип дії ІЧ датчику. Він має інфрачервоний світлодіод та фотодіод. Коли світло падає на поверхню, частина світла поглинається, а частина відбивається. Фотодіод реагує на відбите світло та генерує електроний сигнал. Дальність роботи ІЧ датчика, в залежності від виробника, може регулюватися завдяки потенціометра

Другий експеримент пов’язан із якістю розпізнавання голосових команд. Було виявлено, що сторонні звуки можуть заважати процесу зчитування та розпізнавання голосових команд, та зменьшує ймовірність розпізнавання приблизно до 15%. З 10 запросів було розпізнано лише 2, один з яких був не коректно оброблено. Тому було прийнято рішення зменьшити кількість зайвих звуків. Одним із головних джерел сторонніх звуків були двигуни для пересування робота. Після доработки коду програми, рухи робота стали більш короткими і мають більш великий інтервал між діями. А під час зчитування команд з основної функції, мотори взагалі вимикаються. Це потрібно для збільшення вирогідності розпізнавання команди. Після повторних замірів, з 10 запросів були обработані 9 команд, один з яких був обработан не коректно. Як ми бачимо, зменьшення зайвих звуків збільшило ймовірність розпізнавання команд до 85%.

3.6 Розрахунок заземлення

Заземлення допомагає захистити користувача та електронні компоненти від електричних ударів. Якщо відбудеться коротке замикання або інша електрична несправність, струм буде спрямований в землю, що зменшить ризик ураження електричним струмом. [2]

Також заземлення захищає обладнання від пошкоджень, які можуть виникнути через електричні перешкоди або стрибки напруги. Це продовжує

термін служби компонентів та знижує ймовірність їхнього виходу з ладу.

Застосування заземлення для РС може запобігти ститичним розрядам. Робот може накопичувати статичну електрику, особливо якщо він рухається по різних поверхнях. Заземлення допоможе уникнути раптових розрядів, які можуть пошкодити електронні компоненти. Також заземлення може підвищити стабільність датчиків. УЗ та ІЧ датчики можуть бути чутливими до електромагнітних перешкод. Заземлення допоможе забезпечити стабільність їхніх показників.

Для встановлення заземлення необхідно:

– перевірити схему підключення компнентів;

– використовувати якісні дроти та надійне з’єднання. Для цього в макеті використовується макетна плата та клеми;

– всі компоненти повинні мати спільну точку заземлення.

Таким чином, можна запобігти помилок в оброці сигналів датчиків, стабільну роботу МК та запобігти неісправностей системи.

ВИСНОВКИ

В ході виконання кваліфікаційної роботи була розроблена система автоматизації системи керуванням роботом – секретарем. Проведено аналіз сучасних голосових асистентів, настільних роботів, апаратних рішень, методів розпізнавання голосових команд та розпізнавання навколишнього середовища заавдяки комбінації УЗ датчиків та ІЧ датчику ліній. Також були виявлені базові функції, якими повинен володіти робот – секретар.

Метою роботи стало створення макету настільного робота-секретаря із функцією голосового керування з використанням доступних апаратних та програмних рішень. Завдяки цьому, в майбутньому можна покращити якість роботи та оброки голосових команд, а також додати нові функції змінюючи різні складові.

Наступним етапом стала розробка макету РС. Були обрані оптимальні компоненти для роботи запланованих функцій. Головним модулем для розрахунку та обробки всіх функцій був обран МК Raspberry Pi zero W із інстальованою на ній операційною системою Raspbian. Мовою програмування було обрано Python, а середою розробки Python IDE. Це пов’язано із тим, що для розпізнавання голосових команд була обрана біліотека Speech Recognition, яка здатна записувати навколишній звук, обробляти його та розпізнавати в ньому мову користувача та переводити його в текст. Для роботи із дісплеєм була використана бібліотека Adafruit_SSD1306. Ця бібліотека вміє працювати із дісплеями підключеними за допомогою технології I2C, та виводити на ньому різні графічні елементи, такі як: текст, фігури та може виводити просту анімацію завдяки математичним функціям. Всі компоненти були під’єднанні до МК за допомогою вбудованих в нього пінів GPIO.

Далі була розроблена програма для керування РС. Були розроблені функції для зберігання подій, перевірки на їх наявність в базі, та видалення з неї. Було додано можливість надсилати push повідомлення на телефон завдяки

додатку Pushbullet. Це надає можливість робота комунікувати із користувачем через звичайний повсякдений пристрій, який користувач завжди має при собі.

Були проведені експериментальні дослідження, на яких було виявленно погіршення якості розпізнавання голосових команд з причини надходження сторонніх звуків під час виконання функції. Через це, було переработано алгоритм пересування РС з додаванням затримок між діями робота. Завдяки цьому, якість обробки голосових команд підвищилась. Також у наслідку виявлення такої проблеми в конструкцію робота було додано сенсорну кнопку, як альтернативний спосіб активування РС.

ПЕРЕЛІК ПОСИЛАНЬ

1. 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.

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

3. Методичні вказівки з підготовки кваліфікаційної роботи бакалавра для здобувачів першого (бакалаврського) рівня вищої освіти спеціальності 151 Автоматизація та комп’ютерно-інтегровані технології освітньої програми «Системна інженерія» / Упоряд.: І.Ш. Невлюдов, О.М. Цимбал, О.В. Токарєва, А.І. Бронніков. – Харків: ХНУРЕ, 2022. – 66 с.

4. Невлюдов І.Ш. , Андрусевич А.О., Євсєєв В.В. Проектування мобільних роботів на базі одноплатних комп'ютерів (Raspberry Pi та мови Python 3.6) Харків ФОП Панов А.М.. 2020. Р.264. ISBN 978-617-7859-56-6

5. What is Raspberry Pi Zero? Pinout, Specs, Projects & Datasheet - The Engineering Projects. The Engineering Projects. URL: https://www.theengineeringprojects.com/2021/03/what-is-raspberry-pi-zero-pinout-specs-projects-datasheet.html (дата звернення: 12.04.2024)..

6. HC-SR04 Ultrasonic Range Sensor on the Raspberry Pi. The Pi Hut. URL: https://thepihut.com/blogs/raspberry-pi-tutorials/hc-sr04-ultrasonic-range-sensor-on-the-raspberry-pi (дата звернення: 15.05.2024).

7. SAI Conferences Call for Papers. URL: https://saiconference.com/Downloads/FTC2017/Proceedings/47_Paper_315-A_Prototyping_of_BoBi_Secretary_Robot.pdf (дата звернення: 13.03.2024).

8. SКолонка Apple HomePod 2 White (MQJ83). ≡ Навушники і

аксесуари Apple, купити навушники AirPods комплектом і по одному, доставка по всій Україні. URL: https://storepods.com.ua/kolonka-apple-homepod-2-white-mqj83/ (дата звернення: 03.05.2024).

9. Google Home | Wireless Bluetooth Smart Speaker | Voice Assistant – Купити на eBay PL (Польща) з Доставкою в Україну – Megazakaz. Megazakaz.com. URL: https://megazakaz.com/ua/pl/ebay/product/114736424401 (дата звернення: 03.05.2024).

10. Акустика портативна Amazon Echo Dot (5th Generation) Glacier White купити, ціна та відгуки в магазині MTA. mta.ua. URL: https://mta.ua/kolonki-dlya-telefoniv/807285-akystika-portativna-amazon-echo-dot-5th-generation-glacier-white?utm_source=google&utm_medium=cpc&utm_campaign=ROMAN_Pmax_Category-B-stock&gad_source=1&gclid=Cj0KCQjwpZWzBhC0ARIsACvjWRPduimildNXqPBdBfTdTWXUpbgRH-cMqhHfttgD3_mMZfAyj4uZqY4aAukSEALw_wcB (дата звернення: 03.05.2024).

11. EMO Robot, AI Desktop Pet, Living.AI. Robot Shop. URL: https://sk.robotshop.ro/en/products/emo-robot-ai-desktop-pet-living-ai (дата звернення: 04.05.2024)

12. Anki Vector Robot with Amazon Alexa Built-In Ukraine | Ubuy. Ubuy Ukraine. URL: https://www.u-buy.com.ua/en/product/6IJA20-vector-robot-by-anki-a-home-robot-who-hangs-out-helps-out-with-amazon-alexa-built-in (дата звернення: 04.05.2024)

13. ENERGIZE LAB Eilik Interactive Robot Pet, Perfect Ukraine | Ubuy. Ubuy Ukraine. URL: https://www.u-buy.com.ua/en/product/IK4M7YOPM-eilik-cute-robot-pets-for-kids-and-adults-your-perfect-interactive-companion-at-home-or-workspace-unique-gifts-for-girls-boys-blue (дата звернення: 04.05.2024).

14. Ультразвуковий датчик відстані HC-SR04 купити в Києві та Україні. Arduino в Україні. URL: https://arduino.ua/prod182-yltrazvykovoi-

datchik-rasstoyaniya-hc-sr04 (дата звернення: 10.05.2024).

15. Датчик лінії KY-033 на TCRT5000 купити в Києві та Україні. Arduino в Україні. URL: https://arduino.ua/prod2557-datchik-linii-ky-033-na-tcrt5000 (дата звернення: 10.05.2024).

16. Raspberry Pi Zero W - Raspberry Pi Zero WH. YADOM Essential Supplier of Industrial IOT Sigfox Lora M2M NBIOT. URL: https://yadom.eu/raspberry-pi-zero-w.html (дата звернення: 11.05.2024).

17. USB Міні мікрофон MI-305 для Raspberry PI купити в Києві та Україні. Arduino в Україні. URL: https://arduino.ua/prod3301-usb-mini-microphone-for-raspberry-pi (дата звернення: 10.05.2024).

18. Дисплей OLED S0.96" I2C 128x64 (жовто-синій) купити в Києві та Україні. Arduino в Україні. URL: https://arduino.ua/prod5356-displei-oled-s0-96-i2c-128x64-jelto-sinii (дата звернення: 11.05.2024).

19. Сенсорний датчик - кнопка купити в Києві та Україні. Arduino в Україні. URL: https://arduino.ua/prod1151-sensornii-datchik-knopka (дата звернення: 11.05.2024).

20. Двигун з редуктором 1:48 (одно-осьовий) купити в Києві та Україні. Arduino в Україні. URL: https://arduino.ua/prod3195-motor-s-redyktorom-148-odno-osevoi (дата звернення: 11.05.2024).

21. Драйвер L293D для робоплатформ и маломощных двигателей купити в Києві та Україні. Arduino в Україні. URL: https://arduino.ua/prod2608-modyl-draivera-motorov-na-l293d (дата звернення: 12.05.2024).

22. SpeechRecognition.PyPI.URL: https://pypi.org/project/SpeechRecognition/ (date of access: 12.04.2024).

23. GitHub - adafruit/Adafruit_SSD1306: Arduino library for SSD1306 monochrome 128x64 and 128x32 OLEDs. GitHub. URL: https://github.com/adafruit/Adafruit_SSD1306 (дата звернення: 15.05.2024).

24. threading Thread-based parallelism. Python documentation. URL: https://docs.python.org/uk/3/library/threading.html (дата звернення: 20.05.2024).

25. Огляд Raspbian OS - Безпека та відеоспостереження. Безпека та

відеоспостереження. URL: https://oxorona.com/raspbian-os-review/ (дата звернення: 02.05.2024).