การใช้การจดจำเสียงพูดในการควบคุมหุ่นยนต์ 4 ล้อ โดย ROS2 Kilted Kaiju บน Ubuntu 24.04

Speech Recognition for 4-Wheeled Mobile Robot Control using ROS2 Kilted Kaiju on Ubuntu 24.04

วุฒิพร มากมี[1,*], ดุลยวัต ปัญญารัตนศรีขจร[1], ชนมภัทร โตระสะ[1], เศรษฐกาล โปร่งนุช[1], อภิรักษ์ ธิตินฤมิต[1], กฤษฎา อิทธิโพธิรัตน์[1] และ ปฏิญญา สมานุหัตถ์[1]

[1]สาขาวิชาวิศวกรรมหุ่นยนต์ คณะวิศวกรรมศาสตร์และเทคโนโลยีอุตสาหกรรม มหาวิทยาลัยราชภัฏสวนสุนันทา

Wuttiporn Makmee[1,*], Dulyawat Panyarattanasrikachorn[1], Chonmapat Torasa[1], Sethakarn Prongnuch[1], Aphirak thitinaruemit[1], Kidssada Itthiphothirat[1] and Patinya Samanuhut[1]

[1] Robotics Engineering Department, Faculty of Engineering and Industrial Technology, Suan Sunandha Rajabhat University

66122544001@ssru.ac.th

บทคัดย่อ

งานวิจัยนี้มีวัตถุประสงค์เพื่อพัฒนาและทดสอบระบบควบคุมหุ่นยนต์ 4 ล้อด้วยคำสั่งเสียง โดยผสานรวมการทำงานของ Robot Operating System 2 (ROS2) รุ่น Kilted Kaiju บน Ubuntu 24.04 เข้ากับระบบฮาร์ดแวร์ควบคุมภายนอก(Off-board Control) วิธีการดำเนินงานประกอบด้วยสามส่วนหลัก 1) ROS2 Speech Recognition (SR) Node ทำหน้าที่รับคำสั่งเสียง "เดินหน้า", "ถอยหลัง", "เลี้ยวซ้าย", "เลี้ยวขวา" และแปลงเป็นรหัสคำสั่ง (Command Codes) 2) ROS2 Control Node ทำหน้าที่รับรหัสคำสั่งและส่งออกเป็นข้อมูลอนุกรม (Serial Data) ผ่านการเชื่อมต่อไร้สาย Bluetooth (HC-05) และ 3) ชุดควบคุมหุ่นยนต์ที่อาศัย Arduino Microcontroller Board ทำหน้าที่รับข้อมูลอนุกรมจาก Bluetooth (HC-05) module แล้วทำการแปลงรหัสคำสั่งนั้นไปควบคุมมอเตอร์ 4 ล้อผ่าน Driver Board L298N เพื่อให้หุ่นยนต์เคลื่อนที่ตามทิศทางที่ต้องการ ผลการทดลองแสดงให้เห็นว่าระบบที่ผสานระหว่างซอฟต์แวร์ ROS2 และฮาร์ดแวร์ Arduino Microcontroller Board สามารถตอบสนองต่อคำสั่งเสียงในการควบคุมทิศทางการเคลื่อนที่ของหุ่นยนต์ได้การออกแบบนี้พิสูจน์ให้เห็นถึงแนวทางการสร้างส่วนต่อประสานกับมนุษย์ด้วยเสียง (VUI) ที่มีประสิทธิภาพโดยใช้ ROS2 เป็นศูนย์กลางการประมวลผลคำสั่งที่ซับซ้อนและใช้ไมโครคอนโทรลเลอร์เพื่อจัดการงานควบคุมระดับต่ำ

คำสำคัญ: ROS2 Kilted Kaiju, การรู้จำเสียงพูด, การควบคุมด้วยเสียง, หุ่นยนต์สี่ล้อ

Abstract

This research focuses on developing and testing a voice-controlled 4-wheeled mobile robot system by integrating the Robot Operating System 2 (ROS2) Kilted Kaiju distribution on Ubuntu 24.04 with an off-board hardware control mechanism. The methodology involves three core components. Firstly, ROS2 Speech Recognition (SR) Node that captures voice commands "Forward," "Backward," "Turn Left," "Turn Right" and translates them into simple command codes. Second, a dedicated ROS2 Control Node receives these command codes and transmits them as serial data wirelessly via a Bluetooth HC-05 Bluetooth module. Lastly, the Robot Control Unit, which consists of an Arduino microcontroller, receives the serial data from the HC-05 and translates the command codes into logic signals to drive the 4-wheeled robot's motors via a L298N Driver Board. The experimental results demonstrate that the integrated system successfully provides responsive and accurate directional control of the robot based on voice commands. This design confirms a practical approach to building an effective Voice User Interface (VUI) for robotics, leveraging ROS2 for complex command processing while utilizing a microcontroller (Arduino) for low-level motor actuation tasks.

Keywords: ROS2 Kilted Kaiju, Speech Recognition, Voice Control, Four-Wheeled Robot

1. บทนำ

ปัจจุบันเทคโนโลยีระบบอัตโนมัติและวิทยาการหุ่นยนต์ (Robotics) มีบทบาทสำคัญอย่างยิ่งในการยกระดับประสิทธิภาพในอุตสาหกรรมต่าง ๆ การนำระบบอัตโนมัติมาใช้ในสภาพแวดล้อมที่มีการเปลี่ยนแปลงและมีมนุษย์เป็นศูนย์กลางมากขึ้น ทำให้จำเป็นต้องพัฒนารูปแบบปฏิสัมพันธ์ระหว่างมนุษย์กับหุ่นยนต์ (Human-Robot Interaction - HRI) ที่ใช้งานง่ายและแข็งแกร่ง การควบคุมด้วยเสียงถือเป็นโซลูชันที่น่าสนใจเป็นพิเศษ เนื่องจากช่วยให้สามารถสั่งงานแบบแฮนด์ฟรีและเพิ่มการเข้าถึงเมื่อเทียบกับอินเทอร์เฟซทางกายภาพ เช่น จอยสติ๊กหรือหน้าจอสัมผัส รูปแบบนี้ช่วยให้ผู้ปฏิบัติงานสามารถออกคำสั่งได้อย่างราบรื่นในขณะที่ยังคงรักษาสถานการณ์หรือมีส่วนร่วมในงานอื่น ๆ ที่ต้องใช้มือ [1-3] Robot Operating System 2 (ROS2) ได้กลายเป็นเฟรมเวิร์กมาตรฐานระดับโลกสำหรับการพัฒนาระบบหุ่นยนต์ที่ซับซ้อน เนื่องจากมีความสามารถในการสื่อสารแบบกระจาย (Distributed Communication) ที่มีประสิทธิภาพสูง และรองรับการทำงานแบบเรียลไทม์ [4,5] โดยเฉพาะอย่างยิ่ง ROS2 Kilted Kaiju ที่ถูกติดตั้งบนระบบปฏิบัติการ Ubuntu 24.04 (Noble) ซึ่งเป็นเวอร์ชั่นล่าสุดในช่วงเวลาของการพัฒนาโครงงาน การสื่อสารระหว่างมนุษย์กับหุ่นยนต์เป็นหัวใจสำคัญในการทำให้ระบบอัตโนมัติเข้าถึงได้ง่าย การควบคุมหุ่นยนต์ด้วยคำสั่งเสียงผ่านระบบ Speech Recognition (SR) จึงเป็นแนวทางที่มีประสิทธิภาพและเป็นธรรมชาติที่สุด แทนที่การควบคุม

ผ่านจอยสติ๊กหรือแอปพลิเคชันแบบดั้งเดิม โครงงานนี้จึงมุ่งเน้นการใช้เทคโนโลยี SR ภายใต้สภาพแวดล้อมของ ROS2 เพื่อสร้างช่องทางการสั่งการหุ่นยนต์เคลื่อนที่ 4 ล้อ ให้สามารถ เดินหน้า, ถอยหลัง, เลี้ยวซ้าย, และเลี้ยวขวา ตามคำสั่งของผู้ใช้งาน

2. วัตถุประสงค์ของงานวิจัย

เพื่อพัฒนาและบูรณาการ ROS2 Node สำหรับระบบรู้จำเสียงพูด (Speech Recognition) บนแพลตฟอร์ม ROS2 Kilted Kaiju เพื่อพัฒนาระบบสื่อสารแบบอนุกรม (Serial Communication) ผ่าน Bluetooth (HC-05) โดยใช้ ROS2 เป็นตัวกลางในการส่งคำสั่งควบคุมไปยังชุดควบคุม Arduino และ Driver Board เพื่อสร้างระบบควบคุมแบบ End-to-End ที่ทำให้หุ่นยนต์ 4 ล้อ สามารถเคลื่อนที่ตามทิศทางที่ผู้ใช้สั่งการด้วยเสียงได้อย่างถูกต้องและมีเสถียรภาพ

3. ขอบเขตการวิจัย

ระบบซอฟต์แวร์หลักทำงานบน ROS2 Kilted Kaiju ติดตั้งบน Ubuntu 24.04 (Noble) ผ่าน Virtual Machine (VM) บนคอมพิวเตอร์พกพา เน้นการรู้จำเฉพาะชุดคำสั่งเสียงที่จำกัด เช่น "เดินหน้า" และ "หยุด" สำหรับการควบคุมการเคลื่อนที่พื้นฐานเท่านั้น ระบบฮาร์ดแวร์ควบคุมระดับต่ำ (Low-Level Control) ใช้ Arduino ในการรับข้อมูลอนุกรมที่แปลงมาจาก ROS2 และส่งสัญญาณควบคุมไปยัง Driver Board เพื่อขับเคลื่อนมอเตอร์ DC 4 ตัวของหุ่นยนต์ การสื่อสารระหว่าง ROS2 (Notebook/VM) กับ Arduino ใช้วิธีการส่งข้อมูลอนุกรมผ่านโมดูล HC-05 Bluetooth

4. การดำเนินการวิจัย

วิธีการดำเนินงานของโครงงานนี้แบ่งออกเป็น 3 ขั้นตอนหลัก ได้แก่ 1) การเตรียมการทางกายภาพและการเชื่อมต่อ (ฮาร์ดแวร์), 2) การพัฒนาซอฟต์แวร์และการสร้าง Protocol, และ 3) การบูรณาการระบบเพื่อสั่งการหุ่นยนต์ การเตรียมการทางกายภาพและการเชื่อมต่อ (Hardware and Connectivity Setup) ขั้นตอนนี้เน้นการประกอบชุดหุ่นยนต์ และการตั้งค่าช่องทางการสื่อสารไร้สายระหว่างคอมพิวเตอร์กับ Arduinoติดตั้ง Ubuntu 24.04 (Noble) บน Virtual Machine (VM) และติดตั้งเฟรมเวิร์ก ROS2 Kilted Kaiju ติดตั้งไลบรารี Pyserial Speech Recognition (SR), และ Tkinter (สำหรับ GUI) ตามที่กำหนดในโค้ด speech_node.py และbt_bridge.py

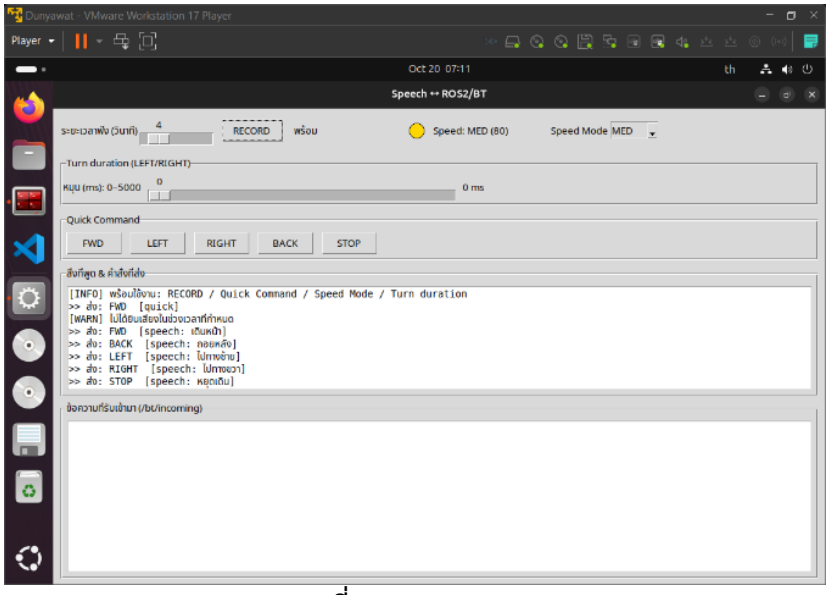
ดำเนินการประกอบโครงสร้างหุ่นยนต์ 4 ล้อ และติดตั้ง Arduino และ L298N Driver Board เดินสายสัญญาณควบคุมจากขา Digital/PWM ของ Arduino ไปยัง Driver Board และต่อสายมอเตอร์ 4 ตัวตามผังการควบคุมที่กำหนดใน Arduino.ino (L_IN1/2/EN, R_IN1/2/EN)เชื่อมต่อ HC-05 Bluetooth Module เข้ากับขา SoftwareSerial (D10, D11) ของ Arduino เพื่อเป็นช่องทางรับคำสั่ง

ทำการตั้งค่าการสื่อสาร Bluetooth (HC-05 Link) ดำเนินการจับคู่ (Pairing) HC-05 กับ Ubuntu VM สร้างช่องสัญญาณอนุกรมเสมือน (Virtual Serial Port) เช่น /dev/ttyS0 หรือ /dev/rfcomm0 เพื่อให้ ROS2 Node สามารถส่งข้อมูลผ่านพอร์ตนี้ได้ โดยพารามิเตอร์ Baud Rate ถูกกำหนดเป็น 9600 เพื่อให้สอดคล้องกับ BT.begin (9600) ในโค้ด Arduino

4.1 การพัฒนาซอฟต์แวร์และการสร้าง Protocol (Software Development and Protocol Design)

ขั้นตอนนี้เป็นการพัฒนาและวิเคราะห์ตรรกะของ ROS2 Node และ Arduino Code เพื่อให้เกิด Protocol

การสื่อสารที่สอดคล้องกันการพัฒนา ROS2 Node สำหรับ Speech และ BridgeSpeech GUI Node (ดังแสดงในรูปที่ 1) (speech_node.py) พัฒนาฟังก์ชัน map_speech_to_cmd เพื่อแปลงคำพูด (ภาษาไทย/อังกฤษ) เป็นรหัสคำสั่งภายใน สร้าง Logic การส่งคำสั่ง (send_cmd_with_turn) เพื่อแนบระยะเวลาการหมุน (ms) ไปกับคำสั่ง LEFT/RIGHT เมื่อผู้ใช้ตั้งค่าไว้ (เช่น ส่ง LEFT 1000) เผยแพร่คำสั่งที่ผ่านการเข้ารหัสไปยัง Topic **/**bt/outgoing Bluetooth Bridge Node (bt_bridge.py)สร้าง Node ที่รับข้อมูลจาก /bt/outgoing และส่ง Payload พร้อมตัวจบบรรทัด **(\n)** ไปยังพอร์ต Serial ที่เชื่อมกับ HC-05พัฒนา Loop การอ่านข้อมูล (_reader_loop) ที่รองรับการรับข้อมูลเป็นชุด และแยกคำสั่งเมื่อพบตัวจบบรรทัด เพื่อความน่าเชื่อถือของการสื่อสาร



รูปที่ 1 Speech GUI

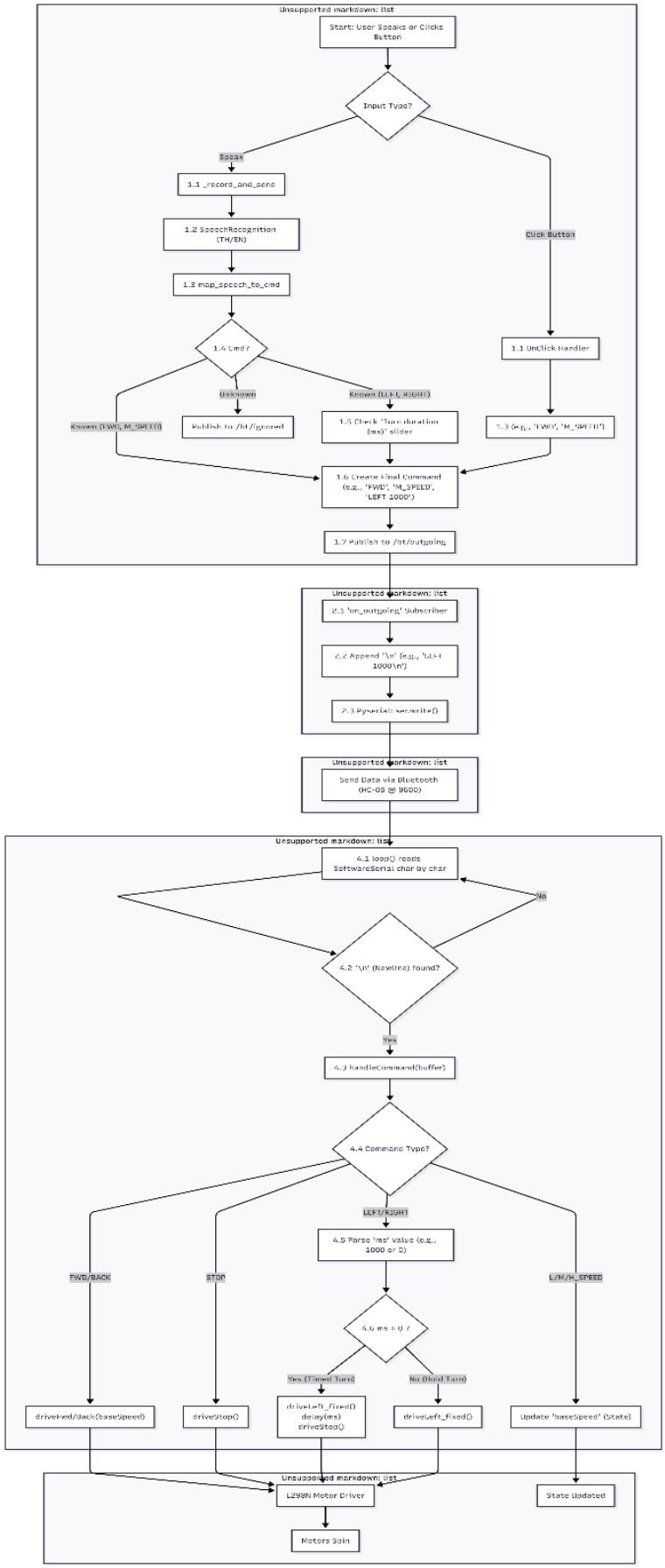
4.2 การพัฒนา Logic ควบคุมมอเตอร์บน Arduino (Command Interpreter)

การรับคำสั่งพัฒนาฟังก์ชัน loop() ให้ตรวจสอบการรับข้อมูลจาก SoftwareSerial BT และสร้างบัฟเฟอร์ข้อมูล (Buffer) เพื่อเรียก handleCommand() เมื่อรับครบหนึ่งบรรทัด (\n) การจัดการคำสั่ง (handleCommand) สร้างตรรกะควบคุมที่ยืดหยุ่นState Management คำสั่ง L_SPEED, M_SPEED, H_SPEED จะปรับค่าตัวแปร baseSpeed (int baseSpeed) Motor Actuation คำสั่ง FWD/BACK จะเรียกฟังก์ชัน driveFwd/driveBack(baseSpeed)Timed Turn Logic คำสั่ง LEFT/RIGHT ที่มีพารามิเตอร์เวลา (เช่น LEFT 1000) จะใช้ฟังก์ชัน delay(ms) และ driveStop() ทันที เพื่อให้หุ่นยนต์หมุนในระยะเวลาที่กำหนด
แผนภาพรวมของซอฟต์แวร์ (flowchart) ดังแสดงในรูปที่ 2

4.3 การออกแบบฮาร์ดแวร์และระบบไฟฟ้า(Hardware and Electrical Design)

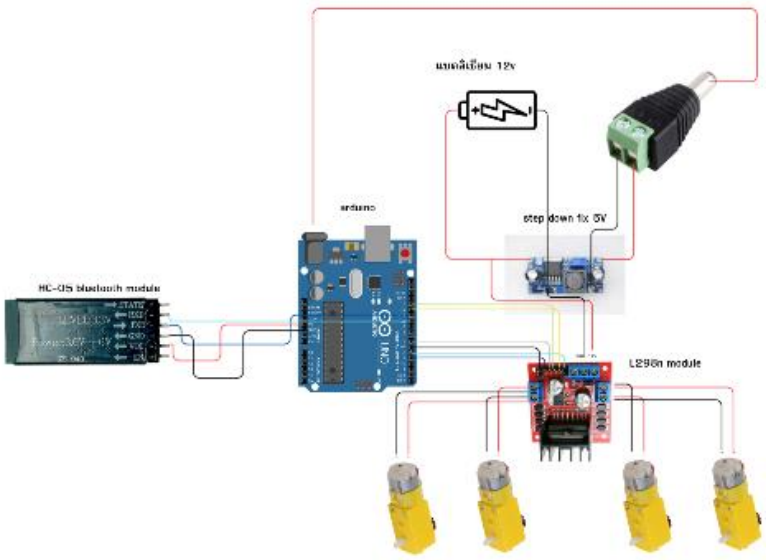
ขั้นตอนนี้เป็นการออกแบบและประกอบชุดหุ่นยนต์ พร้อมการจัดการระบบจ่ายพลังงานเพื่อให้ส่วนควบคุมและส่วนขับเคลื่อนทำงานได้อย่างเสถียรการออกแบบโครงสร้างและการประกอบ

การใช้โครงสร้างสำเร็จรูป เลือกใช้โครงสร้างหุ่นยนต์เคลื่อนที่ 4 ล้อสำเร็จรูป เพื่อเป็นแพลตฟอร์มหลักสำหรับการติดตั้งอุปกรณ์การติดตั้งมอเตอร์ ยึดมอเตอร์ DC ทั้ง 4 ตัวเข้ากับโครงสร้างและล้ออย่างมั่นคง การยึดชุดควบคุม ติดตั้ง Arduino (ตัวรับคำสั่ง), L298N Driver Board (ตัวขับมอเตอร์), และ HC-05 Bluetooth Module เข้ากับโครงสร้างอย่างมั่นคง การออกแบบระบบจ่ายพลังงานและวงจรระบบจ่ายพลังงานหลักใช้แบตเตอรี่ลิเธียมไอออน (Li-ion) ขนาด 12V โดยแยกการจ่ายไฟสำหรับส่วนขับเคลื่อนและส่วนควบคุมอย่างชัดเจน เพื่อป้องกัน



รูปที่ 2 แผนภาพรวมของซอฟต์แวร์ (flowchart)

สัญญาณรบกวน (Noise) จากมอเตอร์การจ่ายไฟและป้องกันกระแสเกิน ส่วนขับเคลื่อนใช้แบตเตอรี่ 12V จ่ายไฟเข้าที่ขา 12V ของ L298N Driver Board โดยตรง เพื่อให้ได้กำลังขับเคลื่อนสูงสุดสำหรับมอเตอร์ DCการจ่ายไฟส่วนควบคุมเชื่อมต่อ Step-Down DC-DC Converter โมดูล LM2596 เข้ากับแบตเตอรี่ 12V ปรับ Step-Down Converter ให้ได้แรงดันไฟฟ้าขาออก 5V ที่มีความเสถียรใช้ไฟ 5V ที่แปลงแล้วนี้เพื่อเลี้ยงชุดควบคุม Arduino ผ่านพอร์ตพาวเวอร์ซัพพลาย แผนภาพการเชื่อมต่อวงจรดังแสดงในรูปที่ 3



รูปที่ 3 แผนภาพการเชื่อมต่อวงจร

5. ผลการวิจัย

ผลการทดลองวิเคราะห์ตามโครงสร้างระบบที่พัฒนาขึ้น ซึ่งประกอบด้วย ROS2 Speech GUI Node (speech_node.py), ROS2 Bluetooth Bridge (bt_bridge.py), และชุดควบคุม Arduino (Arduino.ino) การวิเคราะห์การทำงานของ ROS2 Node และ Logic การแปลงคำสั่งระบบ Speech Control ทำงานโดย speech_node.py ซึ่งใช้ Google API ในการแปลงเสียง และ GUI (Tkinter) ในการควบคุมพารามิเตอร์การฟัง (ระยะเวลาฟัง และ Turn Duration) การส่งคำสั่งถูกเผยแพร่ไปยัง Topic /bt/outgoing ก่อนเข้าสู่ Bridge Node การแปลงคำสั่ง (Command Mapping)ฟังก์ชันmap_speech_to_cmd ใน speech_node.py ประสบความสำเร็จในการแปลงคำพูดเป็นรหัสควบคุมภายใน เช่น "เดินหน้า" เป็น \rightarrow "FWD" และมีการจัดการคำสั่งที่ซับซ้อน ได้แก่การปรับความเร็ว (Speed Preset) คำสั่งเช่น "ความเร็วสูง" จะส่งรหัส H_SPEED เพื่อไปปรับตัวแปร base Speed บน Arduinoการหมุนแบบมีระยะเวลา (Timed Turn) speech_node.py มี Logic พิเศษใน send_cmd_with_turn ซึ่งจะแนบค่าเวลา (ms) จาก GUI (ถ้าตั้งค่า > 0) ไปกับคำสั่ง LEFT/RIGHT เช่น ส่ง LEFT 1000 (ms)

5.1 การสื่อสารใน ROS2 และ Bridge Node

bt_bridge.py ทำหน้าที่เป็นผู้รับผิดชอบการสื่อสาร Serial โดยรันเป็น **Thread แยก** เพื่อให้สามารถรับและส่งข้อมูลพร้อมกันได้ (Bidirectional Communication) และมีการเพิ่มตัวจบบรรทัด (\n) ในโค้ด payload = (msg.data + self.eol).encode('utf-8') ซึ่งทำให้ Arduino สามารถประมวลผลคำสั่งได้อย่างแม่นยำ การวิเคราะห์

ตรรกะควบคุมระดับต่ำบน Arduinoโค้ด Arduino.ino ใช้ SoftwareSerial (D10, D11) เพื่อสื่อสารกับ HC-05 และมีฟังก์ชัน handleCommand() เป็นหัวใจหลักในการตีความคำสั่งที่ได้รับมา Logic การจัดการคำสั่ง (handleCommand)การจัดการ State คำสั่ง L_SPEED, M_SPEED, H_SPEED จะปรับค่าตัวแปร baseSpeed (int baseSpeed = 80) ซึ่งเป็นการจัดการ State ของหุ่นยนต์โดยไม่สั่งการมอเตอร์ทันทีMotor Actuation คำสั่ง FWD และ BACK จะเรียกฟังก์ชัน driveFwd/driveBack(baseSpeed) โดยใช้ค่าความเร็วล่าสุดที่ถูกตั้งไว้การหมุนแบบอัตโนมัติ เมื่อได้รับคำสั่งที่มีพารามิเตอร์เวลา (เช่น RIGHT 500) โค้ดจะเรียก driveRight_fixed() ด้วยความเร็วคงที่ TURN_SPEED (150), ใช้ delay(ms), และตามด้วย driveStop() ทันที ซึ่งเป็น Logic การควบคุมการหมุนที่มีความแม่นยำสูง ตารางที่ 1 แสดงผลการวัดผลและประสิทธิภาพของระบบรวมจากการทดลอง 50 รอบ (End-to-End Performance)

ตารางที่ 1 ผลการวัดและประสิทธิภาพของระบบรวมจาการทดลอง 50 รอบ

ตัวชี้วัด	ผลลัพธ์โดยเฉลี่ย	การวิเคราะห์ที่สอดคล้องกับโค้ด
ความแม่นยำ (SR)	80%-90%	ความสำเร็จขึ้นอยู่กับคุณภาพของไมโครโฟนและสภาพแวดล้อม (เสียงรบกวน) โดยตรง
เวลาหน่วง (Latency)	$\pm$ 0.4 วินาที	ความหน่วงสูงเกิดจากการใช้ recognize_google() ซึ่งเป็นการประมวลผล SR บนคลาวด์
ความยืดหยุ่นของ Control	สูงมาก	โค้ดรองรับการควบคุมความเร็ว (3 ระดับ), การหมุนแบบธรรมดา, และการหมุนแบบมีระยะเวลา (Timed Turn)

6. สรุปและข้อเสนอแนะ

งานวิจัยนี้ได้พัฒนาและบูรณาการระบบควบคุมหุ่นยนต์ 4 ล้อด้วยคำสั่งเสียงที่ยืดหยุ่นและมีความซับซ้อน โดยใช้ ROS2 Kilted Kaiju เป็นชั้นการประมวลผลอัจฉริยะ (Speech Recognition) และใช้ Arduino/L298N เป็นส่วนควบคุมทางกายภาพ การสื่อสารผ่าน HC-05 ด้วย Protocol ที่ออกแบบไว้ (การส่งรหัสคำสั่งสั้น ๆ พร้อม \n) ประสบความสำเร็จในการเชื่อมโยงสองระบบเข้าด้วยกันได้อย่างมีเสถียรภาพ การออกแบบฮาร์ดแวร์ที่ใช้แบตเตอรี่ 12V และ Step-Down Converter เพื่อแยกจ่ายไฟส่วนขับเคลื่อนและส่วน Logic ช่วยให้การทำงานของมอเตอร์มีความเสถียร ปัญหาและอุปสรรค เวลาหน่วง (Latency) จาก Cloud SR ปัญหาหลักคือความหน่วงที่สูง (เฉลี่ย 1.5 วินาที) ที่เกิดจากการประมวลผล Speech Recognition บนเซิร์ฟเวอร์ภายนอก (Google API) ทำให้ไม่สามารถควบคุมแบบเรียลไทม์ได้อย่างสมบูรณ์ ข้อจำกัดด้าน Bandwidth ของ Bluetooth แม้จะใช้รหัสคำสั่งสั้น ๆ แล้วแต่ความเร็วของ Bluetooth Serial (9600 Baud) ยังจำกัดการส่งข้อมูลที่ซับซ้อนหรือการส่ง Feedback ที่รวดเร็ว

ในอนาคตการพัฒนาควรเปลี่ยนไปใช้ SR แบบ Offlineควรเปลี่ยนไปใช้ไลบรารี SR ที่ประมวลผลแบบออฟไลน์ เช่น VOSK ใน ROS2 Node เพื่อขจัดปัญหา Latency ที่เกิดจากการสื่อสารอินเทอร์เน็ต อัปเกรดการสื่อสารด้วย ROS Protocol พิจารณาเปลี่ยนไปใช้การเชื่อมต่อ Wi-Fi (เช่น ESP32) เพื่อใช้ ROS Bridge ในการส่ง Message Type มาตรฐาน (เช่น geometry_msgs/msg/Twist) โดยตรง ซึ่งจะช่วยให้โค้ดฝั่ง Arduino ไม่ต้องมี Logic การตีความคำสั่งมากนัก และสอดคล้องกับเฟรมเวิร์ก ROS2 มากขึ้น การปรับปรุงระบบไฟฟ้า สำหรับงานที่มีภาระสูง ควรเพิ่ม Capacitor ขนาดใหญ่ที่ขา input ของ L298N เพื่อลด Noise ที่อาจส่งผลกระทบต่อ Step-Down Converter และ Arduino Logic

7. กิตติกรรมประกาศ

งานวิจัยนี้สำเร็จลุล่วงไปได้ด้วยดีด้วยความกรุณาอย่างยิ่งจาก ดร.เศรษฐกาล โปร่งนุช อาจารย์ประจำวิชา Software Development for Robot ที่ได้มอบหมายภารกิจทางวิชาการอันทรงคุณค่า ความรู้อันเป็นรากฐานที่ได้รับจากท่านไม่เพียงแต่ทำให้งานวิจัยนี้สำเร็จตามวัตถุประสงค์ของรายวิชาเท่านั้น แต่ยังเป็นจุดเริ่มต้นสำคัญที่ข้าพเจ้าสามารถนำองค์ความรู้ด้านระบบปฏิบัติการหุ่นยนต์ (ROS2) และการควบคุมฮาร์ดแวร์ไปประยุกต์และต่อยอด ในการศึกษาและการทำงานในอนาคตได้อย่างมั่นคง ข้าพเจ้าขอขอบพระคุณ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยราชภัฏสวนสุนันทา ที่ได้สนับสนุนทรัพยากร สถานที่ และอุปกรณ์ต่าง ๆ ซึ่งเอื้อให้การทดลองและการพัฒนาเทคโนโลยีในโครงงานนี้เป็นไปได้อย่างราบรื่น

8. เอกสารอ้างอิง

[1] Z. Wenzheng , G. Kruthika and Y. Fengpei, “Multimodal perception-driven decision-making for human-robot interaction: a survey,” *Frontiers in Robotics and AI,* Vol. 12, 2025.

[2] VA. Fara, SI. Petruc, R. Bogdan and M. Marcu, “A Comparison of Human Tracking Systems on a Mobile Robotic Platform,” *Sensors (Basel)*, Vol. 25(19):6172, Oct 2025.

[3] Á.-G. Salinas-Martínez, J. Cunillé-Rodríguez, E. Aquino-López and A.-I. García-Moreno, “Multimodal Human–Robot Interaction Using Gestures and Speech: A Case Study for Printed Circuit Board Manufacturing,” *J. Manuf. Mater. Process.* Vol. 8, 274, 2024.

[4] R. Sindhuja, R. Siva Dharshini and Mrs. Deepa. “Robotic Speech Recognition and Writing System,” *International Research Journal of Engineering and Technology (IRJET),* Vol. 07, Issue: 04, 2020.

[5] S. Dominykas, B. Matthias, S. Rijin, S. Ingo and A. Ayoub. “Robot System Assistant (RoSA): evaluation of touch and speech input modalities for on-site HRI and telerobotics,” *Frontiers in Robotics and AI*, Vol. 12, 2025.