หุ่นยนต์เคลื่อนที่ควบคุมด้วยระบบ GPS

GPS Mobile Robot

นายปวเรศ ปิ่นแก้ว

6010110680

โครงงานวิศวกรรมคอมพิวเตอร์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญา

วิศวกรรมศาสตร์บัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์

2565

หุ่นยนต์เคลื่อนที่ควบคุมด้วยระบบ GPS

GPS Mobile Robot

นายปวเรศ ปิ่นแก้ว

6010110680

โครงงานวิศวกรรมคอมพิวเตอร์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญา
วิศวกรรมศาสตร์บัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์
ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์
2565

ชื่อโครงงาน	**หุ่นยนต์เคลื่อนที่ควบคุมด้วยระบบ GPS**		
ผู้จัดทำ	**นายปวเรศ ปิ่นแก้ว**	**รหัสนักศึกษา**	**6010110680**
สาขาวิชา	**วิศวกรรมคอมพิวเตอร์**		
ปีการศึกษา	**2565**		

อาจารย์ที่ปรึกษาโครงงาน

.....

(ผศ.ดร.ธเนศ เคารพาพงศ์)

คณะกรรมการสอบ

.....

(ดร.สมชัย หลิมศิโรรัตน์)

.....

(ดร.อนันต์ ชกสุริวงค์)

.....

(ผศ.ดร.นิคม สุวรรณวร)

.....

(ผศ.ดร.ธเนศ เคารพาพงศ์)

โครงงานนี้เป็นส่วนหนึ่งของรายวิชาโครงงานวิศวกรรมคอมพิวเตอร์ 1 และ 2 ตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยสงขลานครินทร์

.....

(รศ.ดร.พิชญา ตัณฑัยย์)

รักษาการในหัวหน้าสาขาวิชาวิศวกรรมคอมพิวเตอร์

หนังสือรับรองความเป็นเอกลักษณ์

ข้าพเจ้าผู้ลงนามท้ายนี้ ขอรับรองว่ารายงานฉบับนี้เป็นรายงานที่มีความเป็นเอกลักษณ์ โดยที่ข้าพเจ้ามิได้มีการคัดลอกมาจากที่ใด เนื้อหาในรายงานทั้งหมดถูกรวบรวมจากการพัฒนาในขั้นตอนต่าง ๆ ของการจัดทำโครงงาน หากส่วนใดที่จำเป็นต้องนำข้อความจากผลงานของบุคคลอื่นใดที่ไม่ใช่ตัวข้าพเจ้า ข้าพเจ้าได้ทำการอ้างอิงถึงเอกสารเหล่านั้นไว้อย่างเหมาะสม และขอรับรองว่ารายงานฉบับนี้ไม่เคยเสนอต่อสถาบันใดมาก่อน

ผู้จัดทำ

ปวเรศ ปิ่นแก้ว

.....

(นายปวเรศ ปิ่นแก้ว)

ชื่อโครงงาน	**หุ่นยนต์เคลื่อนที่ควบคุมด้วยระบบ GPS**		
ผู้จัดทำ	**นายปวเรศ ปิ่นแก้ว**	**รหัสนักศึกษา**	**6010110680**
สาขาวิชา	**วิศวกรรมคอมพิวเตอร์**		
ปีการศึกษา	**2565**		

บทคัดย่อ

เนื้อความบทคัดย่อ

โครงงานหุ่นยนต์เคลื่อนที่ควบคุมด้วยระบบ GPS เป็นโครงงานที่พัฒนาขึ้นเพื่อนำระบบระบุตำแหน่งบนโลก(GPS) มาประยุกต์ใช้กับหุ่นยนต์ โดยใช้การระบุพิกัดของ GPS ในการกำหนดพิกัดที่ต้องการให้หุ่นยนต์เคลื่อนที่ไป เพื่อนำไปประยุกต์ในการใช้งานต่างๆภายนอกอาคาร เช่น การ ส่งพัสดุ หรือการสำรวจพื้นที่ โดยจะใช้หุ่นยนต์เคลื่อนที่อย่างง่าย ที่มีบอร์ด Arduino เป็นตัวหลักในการควบคุม สั่งการเซ็นเซอร์และการทำงานของหุ่นยนต์เพื่อให้หุ่นยนต์เคลื่อนที่ไปโดยอัตโนมัติ

คำสำคัญ ระบบระบุตำแหน่งบนโลก(GPS) มุมทิศ Haversine formular Azimuth formular

Project	**GPS Mobile Robot**		
Author	**Mr. Pawaret Pinkaew**	**Student ID**	**6010110680**
Major Program	**Computer Engineering**		
Academic Year	**2022**		

Abstract

Description

GPS Mobile Robot is a project developed to apply global positioning system (GPS) to robots, using GPS coordinates to determine the target location that you want the robot to move. To be used in various applications outside the building such as parcel delivery or surveying the area, using a simple mobile robot that has an Arduino board as the main control and command the robot's sensors and actions to make the robot move automatically.

Keyword Global Positioning System (GPS) Azimuth angle Haversine formular Azimuth formular

กิตติกรรมประกาศ

โครงงานนี้สำเร็จลุล่วงได้ด้วยความกรุณาจาก ผศ.ดร.ธเนศ เคารพาพงศ์ อาจารย์ที่ปรึกษาโครงงานที่ได้ให้คำเสนอแนะ แนวคิด ตลอดจนการแก้ไขข้อบกพร่องต่าง ๆ มาโดยตลอดจนโครงงานเล่มนี้เสร็จสมบูรณ์ ผู้จัดทำจึงขอกราบขอบพระคุณเป็นอย่างสูง

ขอบพระคุณ ผศ.ดร.นิคม สุวรรณวร ดร.อนันท์ ชกสุริวงค์ และ ดร.สมชัย หลิมศิโรรัตน์ ที่ให้คำแนะนำและข้อคิดต่าง ๆ ในการจัดทำโครงงานฉบับนี้

ขอบพระคุณ คุณอนันต์ นิลโกสีย์ ที่ให้คำแนะนำในการใช้งานเครื่องมือช่าง การเลือกวัสดุและอุปกรณ์ที่ใช้งานในโครงงานและออำนวยความสะดวกในการใช้งานเครื่องมือ

ขอบพระคุณคุณวิมล คำจันทร์ คุณบงกช พฤกษพงษ์ ที่อำนวยความสะดวกและคำปรึกษาในการลงทะเบียนเรียนรายและการสอบวิชาโครงงาน

ขอบพระคุณผู้ปกครอง ที่ให้การสนับสนุน และขอขอบคุณพี่ๆรวมทั้งเพื่อน ๆ ที่ให้ความช่วยเหลือและให้คำปรึกษาแนวทำงในกำรทำโครงงานนี้

ปวเรศ ปิ่นแก้ว

สารบัญ

สารบัญรูปภาพ

สารบัญตาราง

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

ปัจจุบันหุ่นยนต์เริ่มเข้ามามีบทบาทในกิจกรรมต่างๆของมนุษย์มากขึ้น จึงมีการพัฒนาอย่างรวดเร็วและ กว้างขวาง เพื่อช่วยแบ่งเบาภาระการทำงานหรือเพิ่มประสิทธิภาพในการทำงาน เช่น หุ่นยนต์บริการในร้านอาหาร BellaBot [1] ทำหน้าที่ให้บริการต้อนรับลูกค้า รับออร์เดอร์ และเสิร์ฟอาหาร ซึ่งหุ่นยนต์ตัวนี้มีการใช้ AI ในการ ควบคุมเซ็นเซอร์หลายๆตัวเช่น การตรวจสอบการเคลื่อนไหว การระบุตำแหน่งภายในร้าน หากกล่าวถึงการระบุตำแหน่งแล้ว เทคโนโลยีที่มีความสามารถในการระบุตำแหน่งได้อย่างแม่นยำในปัจจุบันก็ คือระบบ Global Positioning System (GPS) ซึ่งสามารถนำมาประยุกต์ใช้เพื่อให้เกิดประโยชน์มากมายไม่ว่าจะ เป็น การทำแผนที่ ระบบนำร่อง หรือการติดตามยานพาหนะ อีกทั้งยังเป็นระบบที่ทุกคนสามารถเข้าถึงและใช้งาน ได้ง่าย ผู้จัดทำจึงมีความสนใจที่จะนำระบบ GPS มาประยุกต์ใช้กับหุ่นยนต์ โดยใช้การระบุพิกัดของระบบ GPS มา ใช้ในการควบคุมทิศทางการเคลื่อนที่ของหุ่นยนต์เพื่อนำไปประยุกต์ในการใช้งานต่างๆภายนอกอาคาร เช่น การส่งพัสดุหรือการสำรวจพื้นที่ โดยจะใช้หุ่นยนต์เคลื่อนที่อย่างง่าย ที่มีบอร์ด Arduino เป็นตัวหลักในการควบคุม และสั่งการเซ็นเซอร์ต่างๆของหุ่นยนต์

1.2 วัตถุประสงค์ของโครงงาน

1. เพื่อสร้างหุ่นยนต์ที่สามารถเคลื่อนที่ได้อัตโนมัติโดยใช้การระบุพิกัดด้วย GPS เป็นตัวควบคุม
2. เพื่อสร้างโปรแกรมในการกำหนดตำแหน่งเพื่อให้หุ่นยนต์เคลื่อนที่

1.3 ประโยชน์ที่คาดว่าจะได้รับ

1. สามารถสร้างหุ่นยนต์ที่สามารถเคลื่อนที่ได้อัตโนมัติตามตำแหน่งที่ระบุด้วยพิกัดจาก GPS ได้
2. สามารถสร้างโปรแกรมสำหรับการระบุตำแหน่งที่ต้องการให้หุ่นยนต์เคลื่อนที่ไปได้
3. สามารถนำการระบุพิกัด GPS ไปใช้ในการควบคุมทิศทางการเคลื่อนที่ของหุ่นยนต์ได้

1.4 ขอบเขตของโครงงาน

1. หุ่นยนต์สามารถเคลื่อนที่ได้ตามตำแหน่งที่ระบุบน GPS
2. หุ่นยนต์สามารถหลบสิ่งกีดขวางขนาดเล็กที่หยุดนิ่งได้
3. สามารถคำนวณระยะทางและความเร็วในการเคลื่อนที่ศึกของหุ่นได้

1.5แผนการดำเนินงาน

ขั้นตอนการดำเนินการ	Project Preparation (2/63)					Project I (1/64)					Project II (1/65)				
	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.
1.ศึกษาหัวข้อโครงงาน															
2.ศึกษางานวิจัยและเอกสารที่เกี่ยวข้องกับโครงงาน															
3.ศึกษาระบบ GPS และระบบควบคุมที่ใช้ GPS															
4.ศึกษาการทำงานของอุปกรณ์และเซ็นเซอร์ต่างๆ ที่ใช้ในโครงงาน															
5.ศึกษาการเขียนโปรแกรมสำหรับควบคุม microcontroller															
6.ออกแบบหุ่นยนต์และระบบการทำงาน															
7.ออกแบบโปรแกรมและเขียนโปรแกรมสำหรับการทำงาน ของ Microcontroller															
8.พัฒนาหุ่นยนต์และเขียนโปรแกรมสำหรับควบคุม Microcontroller															
9.ทดลองและวัดผลการทดลอง															
10.สรุปผลการทดลอง															
11.จัดทำเล่มโครงงาน															

ตารางที่ 1 แผนการดำเนินงาน

บทที่ 2
ทฤษฎีและความรู้พื้นฐาน

2.1 GPS

GPS ย่อมาจาก Global Positioning System หรือระบบระบุตำแหน่งบนพื้นโลก โดยอาศัยดาวเทียม ซึ่งโคจรอยู่รอบโลก ดาวเทียมดวงหนึ่งโคจรรอบโลก 1 รอบ ใช้เวลา 12 ชั่วโมง ระบบ GPS ทำงานโดยการรับสัญญาณจากดาวเทียม ซึ่งประกอบไปด้วยประกอบไปด้วยข้อมูลที่ระบุตำแหน่งและเวลาขณะส่งสัญญาณ เครื่องรับสัญญาณดาวเทียมจะคำนวนระยะทางระหว่างดาวเทียมและเครื่องรับสัญญาณ เพื่อความแม่นยำในการคำนวณตำแหน่ง ต้องใช้ดาวเทียมอย่างน้อย 4 ดวง นอกจากนี้ความแม่นยำของการระบุตำแหน่งนั้นขึ้นอยู่กับตำแหน่งของดาวเทียมแต่ละดวง ความแปรปรวนของชั้นบรรยากาศ และสิ่งแวดล้อมในบริเวณรับสัญญาณ [2][3]

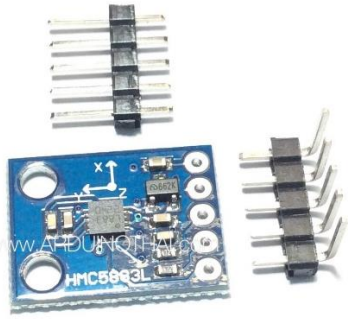
2.2 ส่วนการควบคุมการนำทางของหุ่นยนต์

2.2.1 เครื่องรับสัญญาณ GPS GY-NEO6MV2

โมดูล GPS GY-NEO6MV2 เป็นโมดูล U-blox รุ่น NEO-6M สามารถเชื่อมต่อได้กับไมโครคอนโทรลเลอร์หลายหลายประเภทไม่ว่าจะเป็น Arduino, NodeMCU, Raspberry Pi, ESP32 ผ่านทาง Serial UART ความเร็วที่ 9600 (สามารถเพิ่มได้) และตำแหน่งอัพเดทตลอดทุกๆ 1 วินาที สามารถตั้งค่าให้เร็วกว่านี้ได้ การทำงานเมื่อตัวโมดูลจับสัญญาณได้จะขึ้นไฟสีเขียวกระพริบ ตัวโมดูลมีแบตเตอรี่เก็บตำแหน่งล่าสุดและคอนฟิกต่างๆ โดยจะมีความแม่นยำของพิกัดอยู่ที่ 2.5 CEP หรือก็คือภายในรัศมี 2.5 เมตร [4]

รูปที่ 1 เครื่องรับสัญญาณ GPS GY-NEO6MV2 และเสาอากาศ [5]

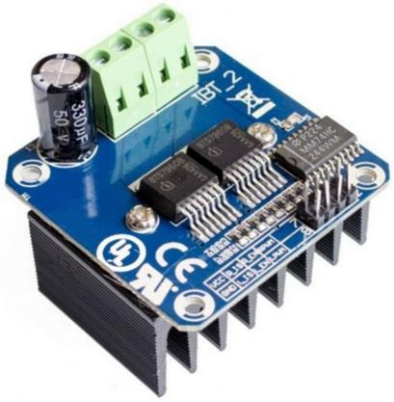
2.2.2 GY-273 3-axis Compass Module (HMC5883L)



รูปที่ 2 GY-273 3-axis Compass Module (HMC5883L) [6]

เป็นโมดูลเข็มทิศ ใช้ชิป HMC5883L วัดสนามแม่เหล็ก 3 แกน สามารถประยุกต์ใช้งานเป็นเข็มทิศได้ [7] มีความแม่นยำของเข็มทิศที่ 1-2 องศา [8]

2.2.3 BTS7960 H-Bridge DC Motor Drive (6-27V 47A Max) Module



รูปที่ 3 Motor Drive Module BTS7960 [9]

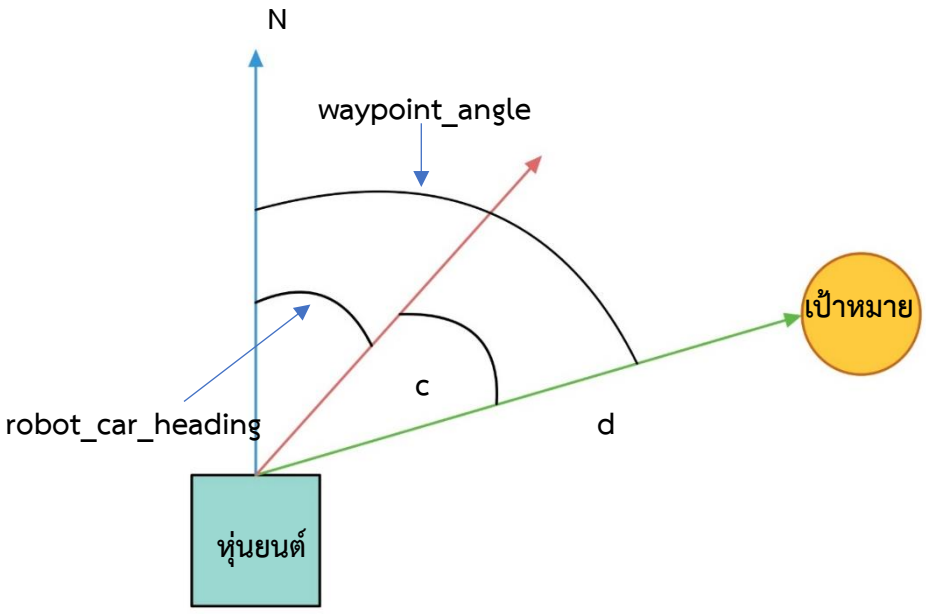
ใช้สำหรับขับดีซีมอเตอร์ที่ต้องการกระแสสูงๆ ใช้สัญญาณ PWM ในการควบคุมความเร็ว รองรับความเร็ว ของ PWM ได้ถึง 25 KHz สามารถควบคุมมอเตอร์ได้ 1 ตัว และควบคุมหมุนซ้าย ขวา (กลับทาง)ได้ แรงดันไฟเลี้ยงมอเตอร์ : 6-27Vdc กระแสเอาต์พุตสูงสูง: 47A Max (กำหนดจากสเปกของ BTS7960) ในทางปฏิบัติควรใช้กระแสไม่เกิน 20A เพื่อความปลอดภัย [10]

2.2.4 Shinano Kenshi DCG-5216-038

รูปที่ 4 Shinano Kenshi DCG-5216-038

เป็นมอเตอร์กระแสตรง รองรับแรงดันสูงสุดที่ 24 โวลต์ รองรับกระแสที่ 0.34 แอมป์ สูงสุดที่ 4 แอมป์โดยที่ไม่มีโหลด ความเร็วรอบอยู่ที่ 670 rpm และกำลัง 50 วัตต์

2.2.5 การคำนวณทิศทางและมุมสำหรับการนำทาง



รูปที่ 5 การคำนวณทิศทางและมุมสำหรับการนำทาง

ในคำนวณระยะทางระหว่างพิกัดที่หุ่นอยู่ ณ ปัจจุบันกับพิกัดจุดหมาย ใช้พิกัดละติจูด ลองจิจูด ของตำแหน่ง 2 ตำแหน่งในการคำนวณ ใช้ Haversine formular เป็นสมการในการค้นหา จากสมการ

$$\Delta\phi = \phi_2 - \phi_1$$

$$\Delta\lambda = \lambda_2 - \lambda_1$$

$$a = sin^2\left(\frac{\Delta\phi}{2}\right) + \cos(\phi_1) \cdot \cos(\phi_2) \cdot sin^2\left(\frac{\Delta\lambda}{2}\right)$$

$$d = R \cdot 2 \cdot arctan2\left(\sqrt{a}, \sqrt{1-a}\right)$$

เมื่อ $\emptyset_1$ คือละติจูดของหุ่นยนต์

$\emptyset_2$ คือละติจูดของเป้าหมาย

λ_1 คือลองจิจูดของหุ่นยนต์

λ_2 คือลองจิจูดของเป้าหมาย

R คือรัศมีของโลก (6317 km.)

d คือระยะทางระหว่าง GPS 2 จุด

สำหรับการหามุมที่รถต้องหมุนไป(มุม c) เพื่อให้หันไปตรงกับพิกัดเป้าหมายหาได้จากสมการ

c = waypoint_angle – robot_car_heading

robot_car_heading เป็นมุมตามเข็มนาฬิกาที่หุ่นยนต์ทำกับทิศเหนือหาได้จากการใช้โมดูลเข็มทิศ

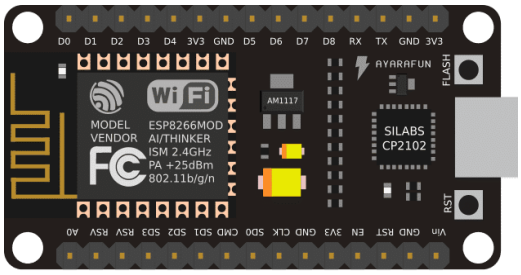
waypoint_angle เป็นมุมตามเข็มนาฬิกาที่เป้าทำกับทิศเหนือหาได้จาก Azimuth formular จากสมการ

$$c = \text{arctan2}(\sin(\Delta\lambda)\cos(\phi_2), \cos(\phi_1)\sin(\phi_2) - \sin(\phi_1)\cos(\phi_2)\cos(\Delta\lambda))$$

ค่าที่ได้จากสมการมีทั้งบวกและลบขึ้นอยู่กับทิศของเป้าหมาย หากอยู่ฝั่งทิศเหนือจะมีค่าเป็นบวก หากอยู่ฝั่งทิศใต้จะมีค่าเป็นลบ จึงต้องนำมาบวกด้วย 360 เพื่อให้ได้มุมตามเข็มนาฬิกาที่เป้าทำกับทิศเหนือ

2.3 ส่วนการกำหนดพิกัดเป้าหมาย

2.3.1 Node MCU ESP8266



รูปที่ 6 NodeMCU ESP8266 [11]

เป็นชื่อเรียกของชิฟของโมดูล ESP8266 สำหรับติดต่อสื่อสารบนมาตรฐาน Wi-Fi ทำงานที่แรงดันไฟฟ้า 3.0-3.6V ทำงานใช้กระแสโดยเฉลี่ย 80mA ภายในมี Low power MCU 32bit ทำให้เราเขียนโปรแกรมสั่งงานโดยใช้ Arduino IDE ได้ มีวงจร analog digital converter ทำให้สามารถอ่านค่าจาก analog ได้ความละเอียด 10bit [12]

2.3.2 การสื่อสารแบบอนุกรม

การสื่อสารแบบอนุกรม (Serial communication) คือ การใช้สาย 1 เส้นรับ-ส่งข้อมูลแบบต่อเนื่อง โดยอาศัยเทคนิคต่าง ๆ ในการสื่อสาร เช่นการใช้สัญญาณทริกเพื่อรับข้อมูลเข้า การใช้บิตเริ่มต้นกำหนดการรับข้อมูล โดยอาจจะอาศัยและไม่อาศัยเวลาในการทำงาน ทั้งนี้การสื่อสารแบบอนุกรมสามารถแบ่งได้เป็น 2 รูปแบบคือ

1. แบบซิงโคนัส (Synchronous) – เป็นการสื่อสารที่ใช้สายสัญญาณข้อมูลอย่างน้อย 1 เส้น และมีสายอีก 1 เส้นกำหนดจังหวะการรับข้อมูล ข้อดีของการสื่อสารแบบนี้คือการรับส่งข้อมูลมีความผิดพลาดน้อยหรือไม่มีความผิดพลาดเลย แต่ข้อเสียคือต้องใช้สายสัญญาณอย่างน้อย 2 เส้นในการสื่อสาร โดยโปรโตคอลที่ทำงานแบบซิงโคนัสได้แก่ I2C I2S และ SPI
2. แบบอะซิงโคนัส (Asynchronous) – เป็นการสื่อสารที่ใช้สายสัญญาณข้อมูลเพียงเส้นเดียวในการทำงาน โดยอาศัยสัญญาณจากบิตเริ่มต้น และบิตสิ้นสุดในการบอกจังหวะการรับส่งข้อมูล การสื่อสารแบบนี้จำเป็นต้องอาศัยเวลามาเป็นตัวกำหนดการรับสัญญาณเข้ามาซึ่งหากมีการตั้งค่าที่ผิดจะทำให้อ่านข้อมูลที่ส่งมาได้ผิดพลาด ข้อดีของการสื่อสารแบบนี้คือใช้สายสัญญาณเพียง 1 เส้นก็สามารถรับ-ส่งข้อมูลได้แล้ว แต่ข้อเสียคือมีความผิดพลาดในการสื่อสารได้ง่าย โดยโปรโตคอลที่ทำงานแบบอะซิงโคนัสคือ UART [13]

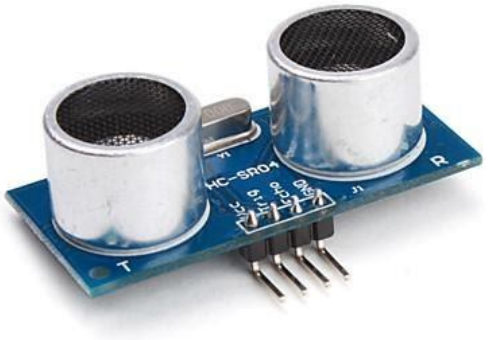
2.3.3 Blynk

รูปที่ 7 Blynk Application [14]

Blynk เป็นแพลตฟอร์มแอฟพลิเคชั่นที่ช่วยให้สามารถสร้างอินเตอร์เฟซสำหรับควบคุมและตรวจสอบฮาร์ดแวร์โปรเจคได้จากอุปกรณ์ทั้ง IOS และ Android ได้อย่างรวดเร็ว [15] โดยผู้ใช้สามารถลากและวางเครื่องมือ (widgets) ที่มีให้เลือกใช้อย่างมากมาย Blynk นั้นถูกออกแบบมาสำหรับ Internet Of Things ทำให้สามารถควบคุมฮาร์ดแวร์ได้จากระยะไกล สามารถแสดงข้อมูลเซ็นเซอร์ จัดเก็บข้อมูล แสดงภาพ และทำสิ่งอื่นๆ ได้มากมาย รองรับอุปกรณ์ฮาร์ดแวร์ที่หลากหลาย ไม่ว่าจะเป็น Arduino NodeMCU หรือ Raspberry pi สามารถใช้งานได้ฟรี แต่จะจำกัดเครื่องมือ หากต้องการใช้เครื่องมือนอกเหนือจากที่ให้บริการฟรี จะมีค่าใช้จ่ายเพิ่มเติม

2.4 ส่วนการควบคุมการหลบสิ่งกีดขวาง

2.4.1 Ultrasonic sensor HC-SR04



รูปที่ 8 Ultrasonic Sensor HC-SR04 [16]

เป็นโมดูลที่ใช้หาระยะห่างระหว่างวัตถุกับเซ็นเซอร์ โดยส่งสัญญาณอัลตร้าโซนิค ความถี่ 40 kHz ไปที่วัตถุที่ต้องการวัดและรับสัญญาณที่สะท้อนกลับมาพร้อมทั้งจับเวลาเพื่อนำมาใช้ในการคำนวณระยะทาง [17]

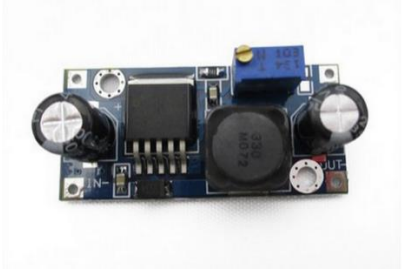
2.5 อุปกรณ์และเทคโนโลยีอื่นๆที่ใช้

2.5.1 Arduino Mega 2560

รูปที่ 9 Arduino Mega 2560 [18]

Arduino MEGA คือบอร์ดรุ่นใหญ่ในกลุ่มบอร์ด Arduino โดยใช้ Atmega2560 เป็นไมโครคอนโทรลเลอร์หลัก โดย Arduino MEGA มี Digital Pins ขา input/output digital จำนวน 54 ขา (เป็น PWM ได้ 15 ขา) มี Analog Input 16 ขา Serial UART 4 ชุด I2C 1 ชุด SPI 1 ชุด และขาแหล่งจ่ายไฟ 5V จำนวน 3 ขา สามารถเขียนโปรแกรมบน Arduino IDE และโปรแกรมผ่าน USB [19]

2.5.2 Step down



รูปที่ 10 Step down [20]

DC-to-DC Step Down LM2596 Module เป็นโมดูลแปลงไฟ DC Step Down (แปลงไฟลง) สามารถปรับค่าได้ตั้งแต่ 1.25-35V โดยปรับค่าที่ Potentiometer โมดูลสามารถจ่ากระแสได้สูงสุด 3A [21]

2.5.3 Arduino IDE

Arduino IDE หรือชื่อเต็มคือ Arduino Integrated Development Environment เป็นชุดเครื่องมือสำหรับการเขียนโปรแกรมควบคุม Microcontroller เช่น Text editor สำหรับการเขียน Code ตัว Compiler และอัปโหลดโปรแกรมลงบอร์ด Arduino [22]

แนวคิดการใช้งานโปรแกรม Arduino IDE

1. เขียนโปรแกรมด้วยภาษา C/C++
2. คอมไพล์โปรแกรมภาษา C/C++ เป็นภาษาสำหรับ Microcontroller และบันทึกเป็น Hex file
3. อัปโหลด Intel Hex File ลงบน Microcontroller ผ่านสาย USB

2.5.4 Shapr3D

รูปที่ 11 Shapr3D [23]

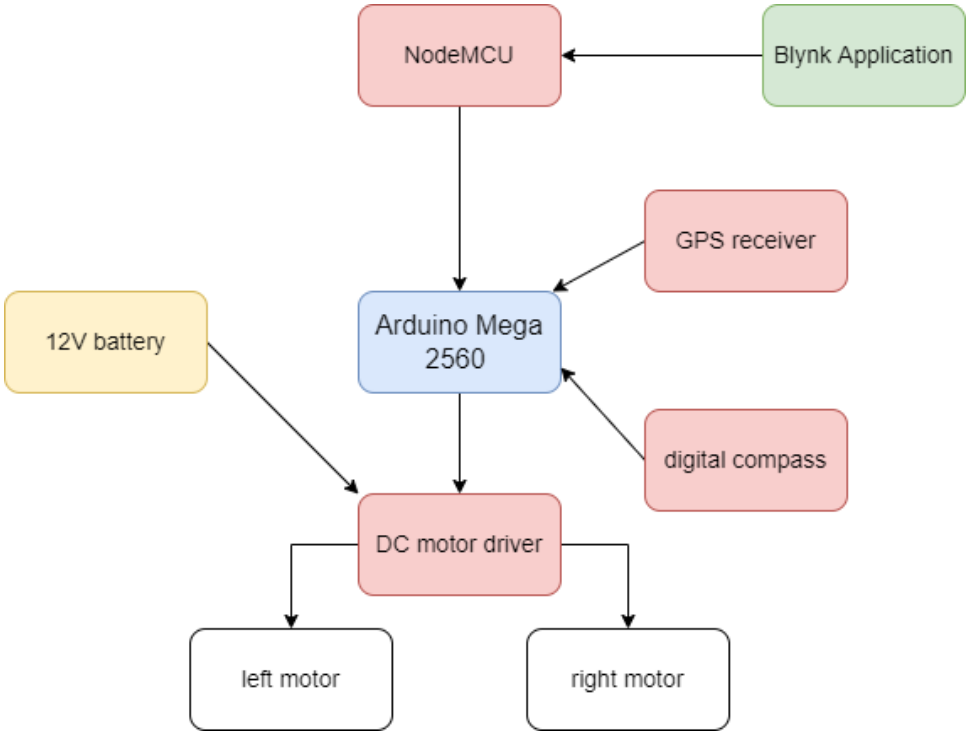
Shapr3D เป็นเครื่องมือการสร้างแบบจำลอง 3 มิติสำหรับการออกแบบทางกลไก สามารถใช้งานได้ทั้ง IOS Android และ Windows บนอุปกรณ์ที่รองรับ

บทที่ 3
วิธีดำเนินงาน

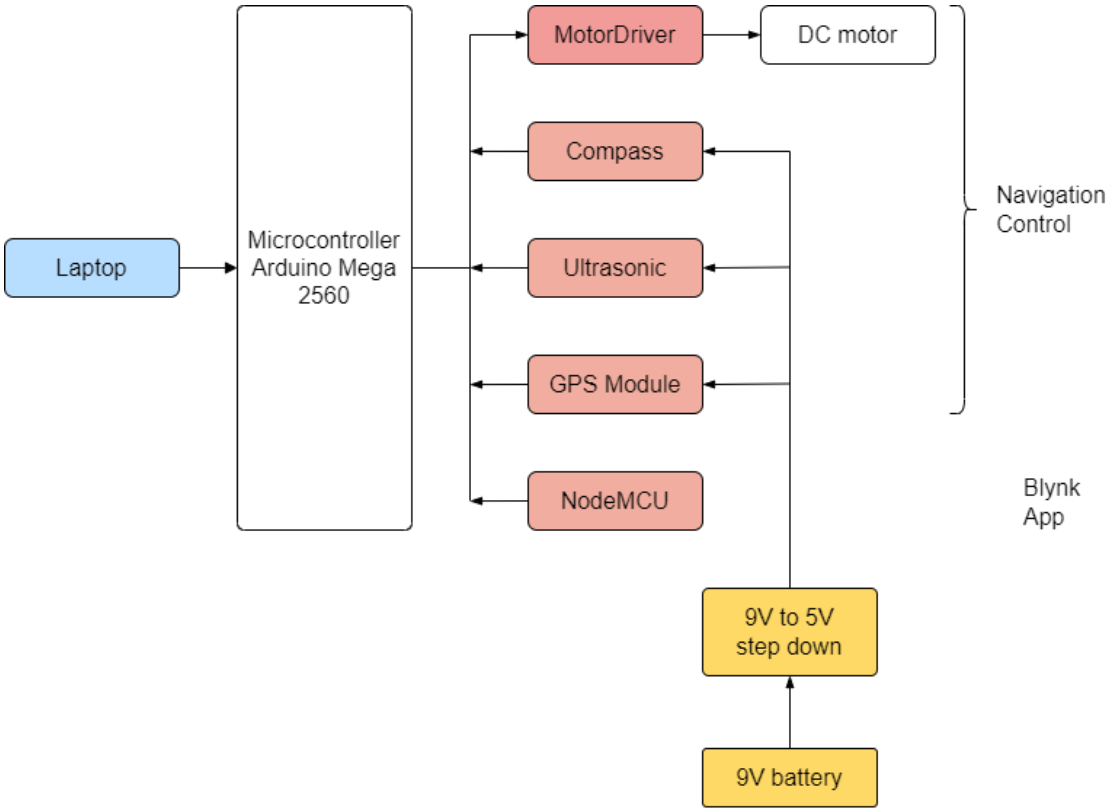
3.1 แนวคิดในการออกแบบ

โครงงานนี้มีความต้องการที่จะสร้างหุ่นยนต์เคลื่อนที่อัตโนมัติ ด้วยนำระบบการระบุพิกัดบนพื้นโลกมาใช้ในการกำหนดตำแหน่งที่ต้อการให้แก่หุ่นยนต์ โดยในโครงงานจะใช้หุ่นยนต์รถขับเคลื่อน 2 ล้อ และมีล้อหมุนอิสระ 1 ล้อ

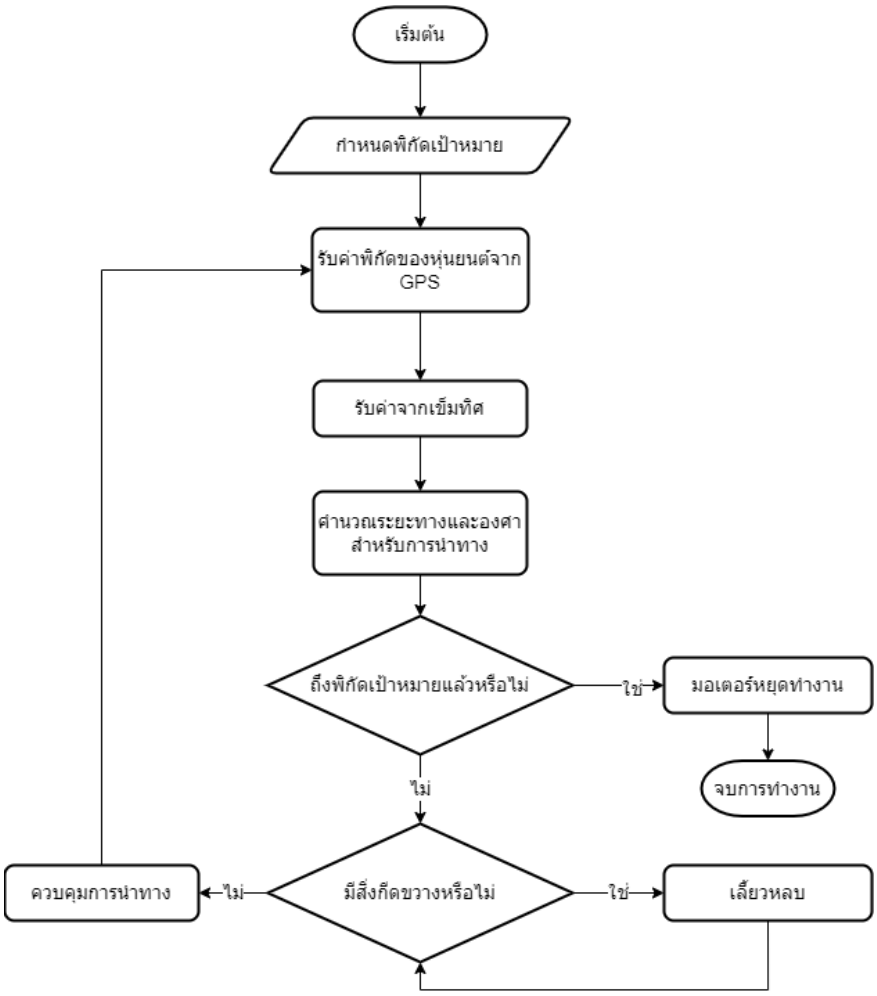
3.2 ภาพรวมของโครงงาน



รูปที่ 12 ภาพรวมของโครงงาน



รูปที่ 13 ส่วนประกอบและการเชื่อมต่อของหุ่น



รูปที่ 14 การทำงานของระบบ

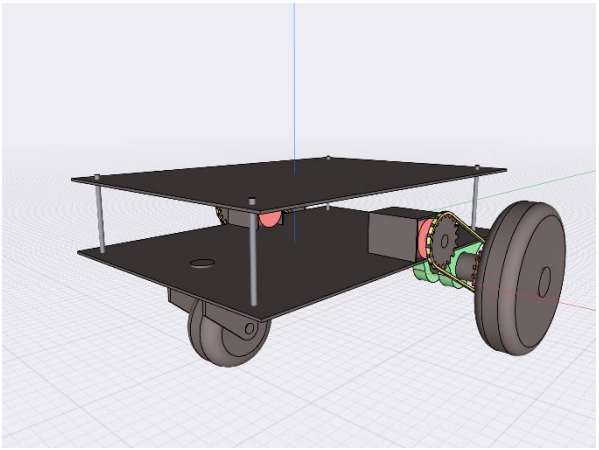
3.3 ส่วนประกอบของโครงงาน

ส่วนประกอบของโครงงานสามารถแบ่งออกเป็น 3 ส่วน ได้แก่

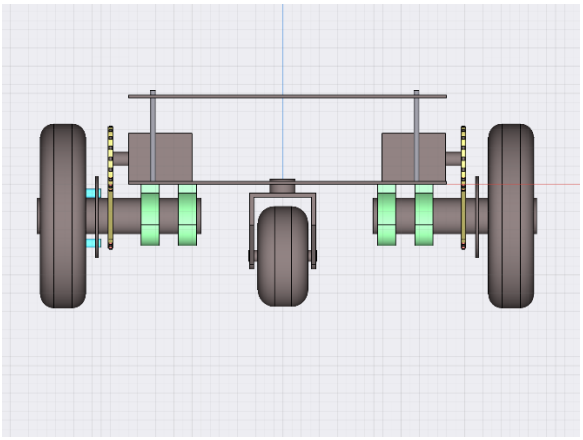
1. ส่วนการกำหนดพิกัด
2. ส่วนควบคุมการนำทาง
3. ส่วนควบคุมการหลบสิ่งกีดขวาง

3.4 ออกแบบหุ่นยนต์

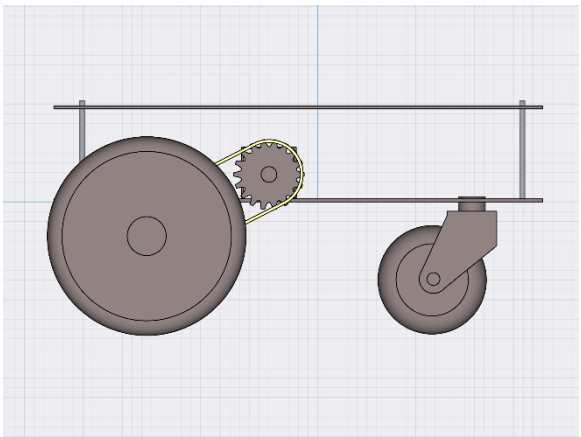
ออกแบบหุ่นยนต์เคลื่อนให้มีความเหมาะสมสำหรับการใช้งานภายนอกอาคาร โดยตัวหุ่นจะมีขนาด 35 x 50 เซนติเมตร ใช้ล้อยางตันที่มีขนาดเส้นผ่านศูนย์กลาง 8 นิ้ว ใช้มอเตอร์ SHINANO KENSHI DCG-5216-038 ที่มีสามารถขับแรงดันได้สูงสุด 24V ในการขับเคลื่อนล้อ มีล้อหมุนอิสระข้างหน้า 1 ล้อเพื่อค้ำโครงสร้างของตัวหุ่น



รูปที่ 15 โครงสร้างของหุ่นยนต์ที่ออกแบบ



รูปที่ 16 มุมมองด้านหน้าของหุ่นยนต์



รูปที่ 17 มุมมองด้านข้างของหุ่นยนต์ที่ออกแบบไว้

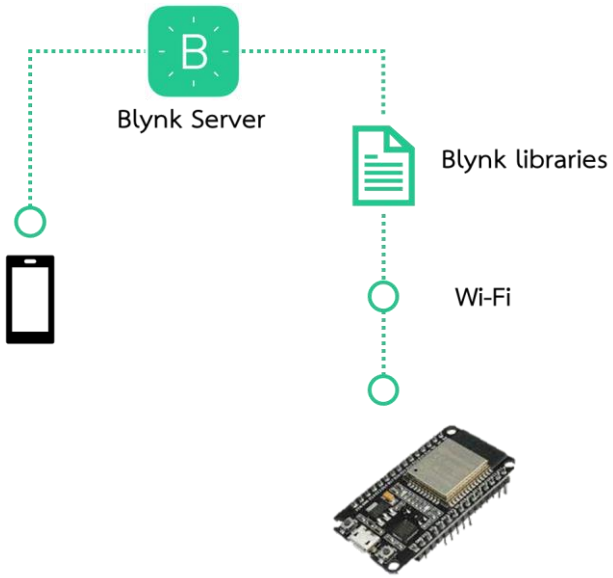
รูปที่ 18 มุมมองด้านข้างของหุ่นยนต์

รูปที่ 19 มุมมองด้านบนของหุ่นยนต์

3.5 ส่วนการกำหนดพิกัด

3.5.1 แอพพลิเคชันที่ใช้

ในส่วนของการกำหนดพิกัดแก่หุ่นยนต์ จะกำหนดผ่านการใช้แอพพลิเคชัน Blynk เพื่อส่งข้อมูลผ่านเครือข่ายเน็ตเวิร์คเข้าสู่บอร์ด Node MCU จากนั้นส่งข้อมูลไปยังบอร์ด Arduino Mega 2560 เพื่อนำไปใช้ในการคำนวณหาเส้นทางในการเคลื่อนที่ของหุ่นยนต์



รูปที่ 20 การเชื่อมต่ออุปกรณ์กับ Blynk

การกำหนดพิกัดผ่าน Blynk

รูปที่ 21 การกำหนดพิกัดเป้าหมายแก่หุ่นยนต์ผ่าน Blynk

latitude (text input): สำหรับป้อนค่าละติจูดของพิกัด ใช้ทศนิยม 6 ตำแหน่ง

longitude (text input): สำหรับป้อนค่าละจิจูดของพิกัด ใช้ทศนิยม 6 ตำแหน่ง

โดยหลังจากป้อนค่าพิกัดแล้วจะต้องกด Enter บนแป้นพิมพ์เพื่อนเป็นการส่งค่าไปยังบอร์ด Node MCU

3.5.2 การส่งข้อมูลระหว่าง Arduino และ Node MCU

เชื่อมต่อ Arduino mega 2560 พิน 14(TX) และ 15(RX) เข้ากับ Node mcu ขาD2(RX) และ D1(TX) ตามลำดับ การรับส่งข้อมูลจะใช้การสื่อสารแบบอนุกรม(Serial) โดยที่ Node mcu จะเป็นตัวส่งข้อมูลพิกัดที่ป้อนผ่าน Blynk application มายัง Arduino

เนื่องจากข้อจำกัดของตัวแปรประเภท float ที่ใช้ในการเก็บค่าพิกัดละติจูดและลองจิจูด ในเรื่องของขนาดของข้อมูลและความแม่นยำของตำแหน่งทศนิยม ก่อนการส่งข้อมูลจึงคูณค่าพิกัดด้วย 1000000 เพื่อให้ตำแหน่งทศนิยมที่ใช้มาเป็นจำนวณเต็ม จากนั้นเมื่อ Arduino รับข้อมูลมา จะนำข้อมูลนั้นมาหารด้วย 100000 เพื่อให้เหลือทศนิยมตำแหน่งที่ต้องการ รูปด้านล่างแสดง Serial Monitor ของบอร์ด Arduino ที่แสดงข้อมูลพิกัดที่ได้รับมาจากบอร์ด Node MCU

```
latitude : 7.006596     longitude : 14.602849
latitude : 7.006596     longitude : 14.602849
latitude : 7.006596     longitude : 14.602849
latitude : 7.006596     longitude : 14.602849
latitude : 7.006596     longitude : 14.602849
latitude : 7.006596     longitude : 14.602849
latitude : 7.006596     longitude : 14.602849
latitude : 7.006596     longitude : 14.602849
latitude : 7.006596     longitude : 14.602849
```

รูปที่ 22 ข้อมูลที่ยังไม่ได้คูณด้วย 1000000 ก่อนการส่ง

```
· 13.00 : 502189.68       latitude : 7.000013     longitude : 100.502189
· 6595.13 : 502189.68     latitude : 7.006595     longitude : 100.502189
· 6595.13 : 502189.68     latitude : 7.006595     longitude : 100.502189
· 13.00 : 502189.68       latitude : 7.000013     longitude : 100.502189
· 6595.13 : 502189.68     latitude : 7.006595     longitude : 100.502189
· 502.00 : 9.63   latitude : 7.000502     longitude : 100.000007
· 6595.13 : 502189.68     latitude : 7.006595     longitude : 100.502189
· 6595.13 : 502189.68     latitude : 7.006595     longitude : 100.502189
· 502189.68 : 502189.68   latitude : 7.502190     longitude : 100.502189
· 6595.13 : 502189.68     latitude : 7.006595     longitude : 100.502189
· 6595.13 : 2189.63       latitude : 7.006595     longitude : 100.002189
· 6595.13 : 502189.68     latitude : 7.006595     longitude : 100.502189
· 6595.13 : 502189.68     latitude : 7.006595     longitude : 100.502189
```

รูปที่ 23 ข้อมูลที่คูณด้วย 1000000 ก่อนการส่งแล้ว

3.6 ส่วนการควบคุมการนำทาง

3.6.1 Navigation vector

เพื่อให้หุ่นยนต์สามารถเคลื่อนที่ไปยังพิกัดที่กำหนดได้ จะใช้ระยะทาง และมุมที่ต้องเคลื่อนที่ในการนำทาง และเพื่อให้รู้ค่าดังกล่าว จะใช้ค่าพารามิเตอร์ 3 ตัวคือ พิกัดเป้าหมาย พิกัดของหุ่นยนต์ และทิศทางของหัวหุ่นยนต์ที่สัมพันธ์กับเหนือ(robot_car_heading) ซึ่งได้จากโมดูลเข็มทิศและโมดูล GPS

3.6.2 Navigation control

ในการควบคุมสำหรับการนำทางนั้น ก่อนอื่นต้องลดความคลาดเคลื่อนของทิศทางการมุ่งหน้าไปยังพิกัดเป้าหมายโดยหันหุ่นยนต์ไปทางขวาหากค่าความคลาดเคลื่อนเป็นบวก และหันไปทางซ้ายหากค่าความคลาดเคลื่อนเป็นลบ ผู้จัดทำโครงงานได้กำหนดช่วงของความคลาดเคลื่อนไว้ที่ ±10 องศา

เนื่องจากเป็นเรื่องยากที่จะลดค่าความคลาดเคลื่อนให้เหลือศูนย์ ในขณะเดียวกันก็ต้องลดความคลาดเคลื่อนของระยะทางโดยการให้หุ่นยนต์เคลื่อนที่ไปข้างหน้าในทิศทางที่ใกล้เคียงกับพิกัดเป้าหมาย แต่การเคลื่อนที่ไปข้างหน้านั้นจะทำให้ค่าความคลาดเคลื่อนของทิศทางการเคลื่อนที่มีค่าเพิ่มขึ้นทั้งในทางบวกและทางลบ จึงต้องมีการทำซ้ำไปเรื่อยๆในการคำนวณ navigation vector และการควบคุมการนำทางเพื่อลดค่าความคลาดเคลื่อนทั้งสองให้เหลือน้อยที่สุดจนกว่าหุ่นยนต์จะเคลื่อนที่ไปยังพิกัดเป้าหมายได้ โดยเมื่อหุ่นยนต์เคลื่อนที่ไปอยู่ในรัศมีประมาณ 1-3 เมตรของพิกัดเป้าหมายจะถือว่าสามารถเคลื่อนที่ไปยังพิกัดเป้าหมายได้สำเร็จ

สำหรับเส้นทางในการเคลื่อนที่ของหุ่นยนต์นั้น จะเป็นเส้นทางตรงไปยังพิกัดเป้าหมาย คือใช้ตำแหน่ง ณ ปัจจุบันของตัวหุ่นยนต์เป็นจุดอ้างอิงในการหาเส้นทางในการเคลื่อนที่ ดังนั้นเมื่อหุ่นยนต์มีการเลี้ยวหรือเคลื่อนที่ออกนอกทิศทางของการเคลื่อนที่ เส้นทางการเคลื่อนที่จะเปลี่ยนไปเสมอ แต่ทุกเส้นทางจะเป็นระยะกระจัดจากหุ่นยนต์ถึงพิกัดเป้าหมาย

3.6.3 การทดสอบส่วนควบคุมการนำทาง

ทดสอบโดยการให้หุ่นยนต์เคลื่อนที่ไปยังพิกัดที่กำหนด โดยใช้พิกัดเดิมทุกครั้งในการเคลื่อนที่ บันทึกระยะห่างระหว่างพิกัดเป้าหมายกับพิกัดที่หุ่นยนต์เคลื่อนที่ไปถึง หากอยู่ในระยะ 3 เมตรถือว่าถึงพิกัดเป้าหมายแล้ว (ค่าความแม่นยำของ GPS อยู่ในระยะ 2.5 เมตร)

รูปที่ 24 พื้นที่สำหรับการทดสอบส่วนควบคุมการนำทาง

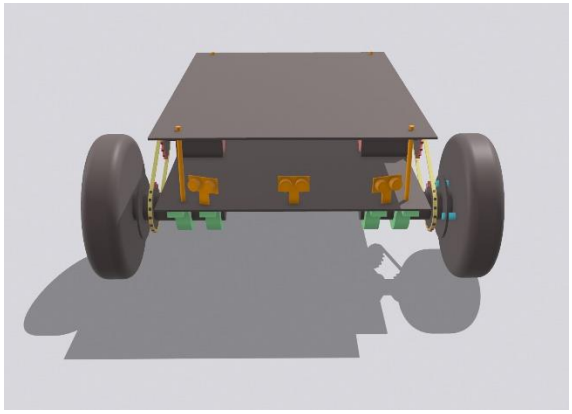
รูปที่ 25 พื้นที่ที่หุ่นยนต์ต้องเคลื่อนที่ไป

3.7 ส่วนการควบคุมการหลบสิ่งกีดขวาง

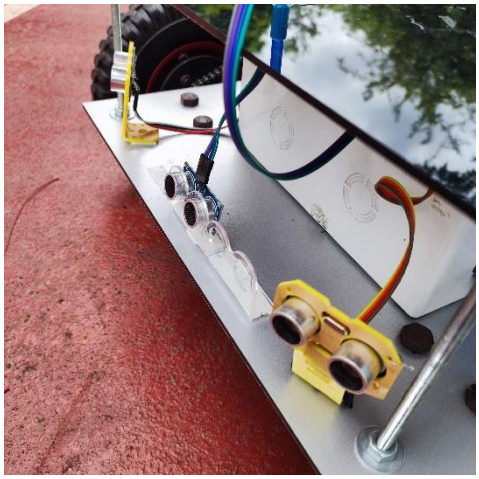
ใช้ ultrasonic sensor 3 ตัวติดตั้งไว้มุมซ้าย ขวา และตรงกลางบริเวณด้านหน้าของหุ่นยนต์ เพื่อใช้ในการตรวจจับวัตถุ หากเซ็นเซอร์ตัวใดตัวหนึ่งตรวจจับเจอวัตถุในระยะ 50 เซนติเมตร จะควบคุมให้หุ่นยนต์เลี้ยวหลบสิ่งกีดขวาง ทิศทางของการเลี้ยวนั้นจะขึ้นอยู่กับว่าเซ็นเซอร์ตัวใดตรวจจับเจอวัตถุ หากเซ็นเซอร์ทางขวาตรวจเจอก็จะแล้วซ้าย หากเซ็นเซอร์ตรงกลางหรือทางซ้ายตรวจเจอก็จะเลี้ยวขวา

ใช้ interrupt แทนการใช้ if else เช็คเงื่อนไขการทำงานของ ultrasonic sensor ในการตรวจจับวัตถุ หากเซ็นเซอร์มีการเปลี่ยนแปลง จะขัดจังหวะการทำงานของโปรแกรม ไปทำตามคำสั่ง

ในฟังก์ชันของ interrupt service routine แทน และหากมีวัตถุอยู่ในระยะที่กำหนด จะควบคุมให้หุ่นยนต์เลี้ยวหลบสิ่งกีดขวาง



รูปที่ 26 ภาพจำลองการติดตั้ง ultrasonic sensor



รูปที่ 27 การติดตั้ง ultrasonic sensor

3.7.1 แนวคิดในการเขียนโปรแกรมการใช้งาน Interrupt กับ Ultrasonic

การใช้งาน interrupt กับ ultrasonic นั้นคล้ายกับการใช้งาน ultrasonic ทั่วไป แต่จะต้องมีขาสำหรับรับสัญญาณเมื่อมีการเกิด interrupt โดยจะกำหนดรูปแบบการเกิด interrupt คือ CHANGE โดย library ที่ใช้ในการสร้าง interrupt คือ TimerOne.h [ ]และมีฟังก์ชันดังนี้

trigger_pulse(); จะถูกเรียกทุกๆ 50 µs เพื่อใช้ในการสร้าง trigger pulse ทุกๆ 4000 pulse

echo_interrupt(); ถูกเรียกทุกครั้งที่สัญญาณเข้าขา echo เปลี่ยนสถานะ โดยใช้สัญญาณเข้าจากขา echo พิจารณาจุดเริ่มต้นและสิ้นสุดของเสียงสะท้อน และเวลาเริ่มต้นและสิ้นสุดของสัญญาณ ถ้า input เปลี่ยนจากสูงไปต่ำแสดงว่าสัญญาณ pulse สิ้นสุด และคำนวณระยะทาง

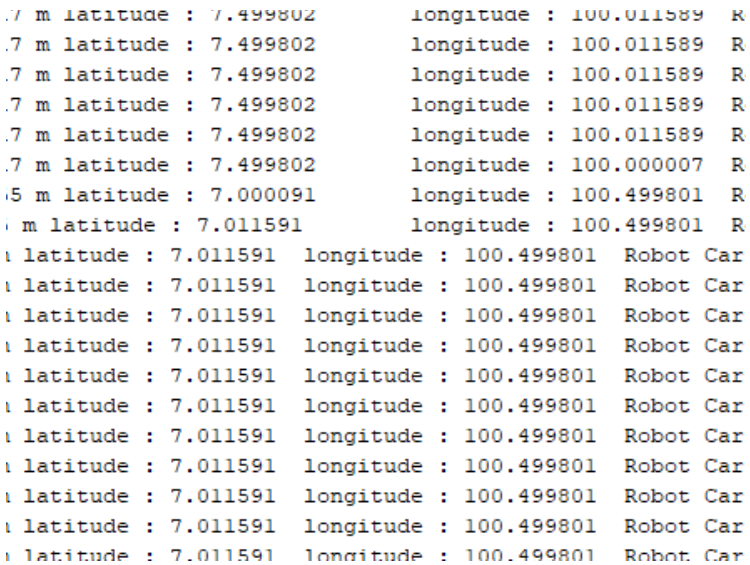
บทที่ 4
ผลและวิเคราะห์ผล

จากวิธีการดำเนินงานที่กล่าวไปแล้วในบทที่ 3 จะแบ่งการทดสอบระบบบอกเป็น 3 ส่วนคือ ส่วนการกำหนดพิกัด ส่วนการควบคุมการนำทาง และส่วนการควบคุมการหลบสิ่งกีดขวาง

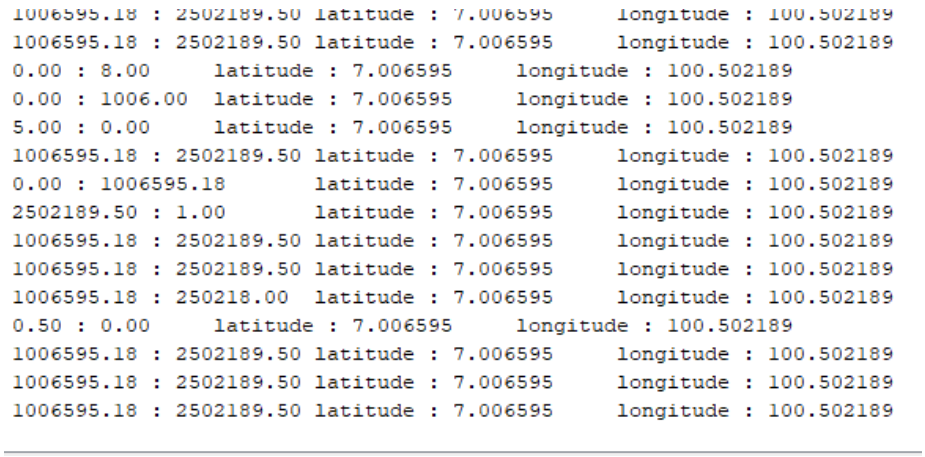
4.1 ส่วนการกำหนดพิกัด

จากการทดสอบการกำหนดพิกัดเป้าหมายแก่หุ่นยนต์นั้น สามารถกำหนดพิกัดละติจูดและลองจิจูดผ่าน Blynk ไปยังบอร์ด Node MCU ได้ และสามารถส่งข้อมูลที่ได้รับมาจากบอร์ด Node MCU ไปยังบอร์ด Arduino ได้

ในการรับส่งข้อมูลนั้น เมื่อรับส่งข้อมูลไปได้ระยะเวลาหนึ่ง พิกัดละติจูดและลองติจูดที่ได้รับจะสลับตำแหน่งกันและค่าพิกัดคลาดเคลื่อนไปเยอะมาก จึงต้องใส่ prefix ก่อนการส่งข้อมูล นั่นคือก่อนการส่งข้อมูลจะบวก 1000000 สำหรับค่าละติจูด และ บวก 2000000 สำหนับค่าลองจิจูด เพื่อใช้ในการตรวจสอบว่า ข้อมูลที่ได้รับมานั้นเป็นพิกัดละติจูดหรือลองจิจูด



รูปที่ 28 การรับส่งค่าพิกัดก่อนใส่ prefix



รูปที่ 29 การรับส่งพิกัดหลังการใส่ prefix

หลังจากใส่ prefix ในการส่งข้อมูลแล้ว เมื่อตรวจสอบแล้วว่าข้อมูลที่รับมาถูกต้อง จะอัพเดทข้อมูลพิกัด หากข้อมูลที่รับมาไม่ถูกต้อง ก็จะไม่มีการเก็บข้อมูลในตัวแปรนั้น

4.2 ส่วนควบคุมการนำทาง

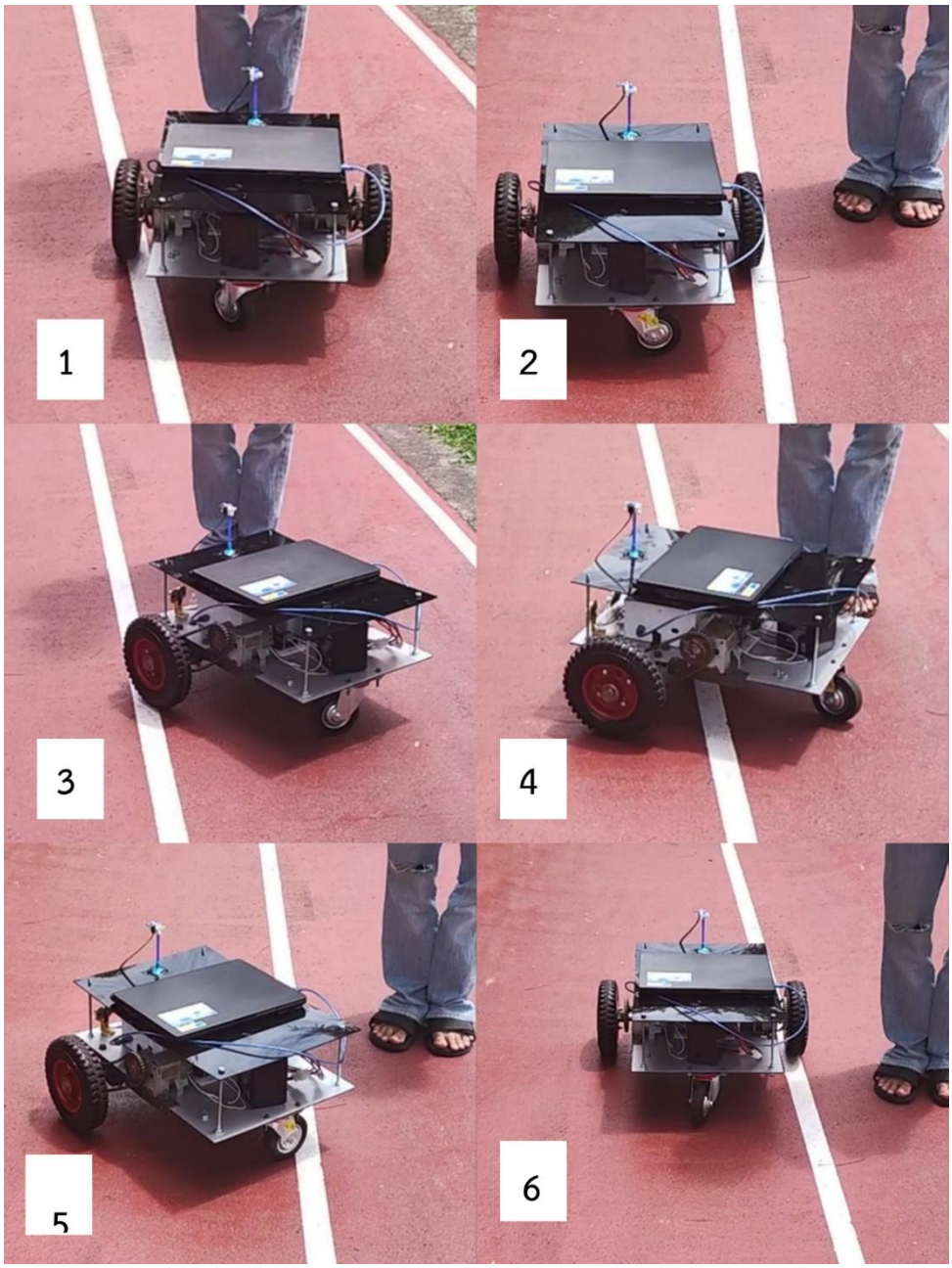
จากการทดสอบส่วนควบคุมการนำทางของหุ่นยนต์ที่ได้ออกแบบไว้ หุ่นยนต์สามารถเคลื่อนที่ไปยังพิกัดที่กำหนดไว้ได้ โดยระยะห่างเฉลี่ยระหว่างพิกัดเป้าหมายกับพิกัดที่หุ่นยนต์เคลื่อนที่ไปถึงทั้ง 10 ครั้ง อยู่ที่ 2.390 เมตร

RobotCarLatitude= 7.011293 RobotCarLongitude= 100.505187 heading: 185	distance : 2.99 m
RobotCarLatitude= 7.011292 RobotCarLongitude= 100.505187 heading: 183	distance : 2.90 m
RobotCarLatitude= 7.011291 RobotCarLongitude= 100.505187 heading: 183	distance : 2.81 m
RobotCarLatitude= 7.011284 RobotCarLongitude= 100.505187 heading: 169	distance : 2.04 m
RobotCarLatitude= 7.011284 RobotCarLongitude= 100.505187 heading: 169	distance : 2.04 m
RobotCarLatitude= 7.011285 RobotCarLongitude= 100.505187 heading: 169	distance : 2.09 m
RobotCarLatitude= 7.011293 RobotCarLongitude= 100.505187 heading: 178	distance : 2.95 m
RobotCarLatitude= 7.011292 RobotCarLongitude= 100.505187 heading: 171	distance : 2.85 m
RobotCarLatitude= 7.011291 RobotCarLongitude= 100.505187 heading: 171	distance : 2.81 m
RobotCarLatitude= 7.011287 RobotCarLongitude= 100.505187 heading: 178	distance : 2.35 m
RobotCarLatitude= 7.011288 RobotCarLongitude= 100.505187 heading: 179	distance : 2.45 m
RobotCarLatitude= 7.011289 RobotCarLongitude= 100.505187 heading: 178	distance : 2.54 m
RobotCarLatitude= 7.011280 RobotCarLongitude= 100.505180 heading: 131	distance : 2.07 m
RobotCarLatitude= 7.011279 RobotCarLongitude= 100.505187 heading: 132	distance : 1.53 m
RobotCarLatitude= 7.011279 RobotCarLongitude= 100.505187 heading: 132	distance : 1.45 m
RobotCarLatitude= 7.011288 RobotCarLongitude= 100.505180 heading: 101	distance : 2.77 m
RobotCarLatitude= 7.011288 RobotCarLongitude= 100.505180 heading: 112	distance : 2.77 m
RobotCarLatitude= 7.011289 RobotCarLongitude= 100.505180 heading: 107	distance : 2.81 m
RobotCarLatitude= 7.011287 RobotCarLongitude= 100.505180 heading: 113	distance : 2.61 m
RobotCarLatitude= 7.011287 RobotCarLongitude= 100.505180 heading: 146	distance : 2.61 m
RobotCarLatitude= 7.011287 RobotCarLongitude= 100.505180 heading: 145	distance : 2.69 m
RobotCarLatitude= 7.011280 RobotCarLongitude= 100.505180 heading: 102	distance : 2.07 m
RobotCarLatitude= 7.011276 RobotCarLongitude= 100.505180 heading: 102	distance : 1.78 m
RobotCarLatitude= 7.011276 RobotCarLongitude= 100.505180 heading: 102	distance : 1.78 m
RobotCarLatitude= 7.011289 RobotCarLongitude= 100.505187 heading: 136	distance : 2.49 m
RobotCarLatitude= 7.011285 RobotCarLongitude= 100.505187 heading: 123	distance : 2.13 m
RobotCarLatitude= 7.011285 RobotCarLongitude= 100.505187 heading: 121	distance : 2.13 m
RobotCarLatitude= 7.011284 RobotCarLongitude= 100.505180 heading: 173	distance : 2.42 m
RobotCarLatitude= 7.011284 RobotCarLongitude= 100.505180 heading: 174	distance : 2.39 m
RobotCarLatitude= 7.011283 RobotCarLongitude= 100.505180 heading: 175	distance : 2.28 m

รูปที่ 30 ผลจากการทดสอบการควบคุมการนำทาง 10 ครั้ง

4.3 ส่วนควบคุมการหลบสิ่งกีดขวาง

จากการทดสอบส่วนควบคุมการหลบสิ่งกีดขวาง หุ่นยนต์สามารถหลบสิ่งกีดขวางซึ่งหยุดนิ่งอยู่หน้าหน้าตัวหุ่นยนต์ได้ โดยเมื่อเซนเซอร์ตรวจพบวัตถุในระยะ 50 เซนติเมตร โปรแกรมในลูปของหุ่นยนต์จะถูกขัดจังหวะ แล้วไปทำงานในส่วนของ interrupt service routine แทน เพื่อควบคุมให้หุ่นยนต์หลบสิ่งกีดขวางนั้นๆตามที่ได้โปรแกรมไว้



รูปที่ 31การหลบสิ่งกีดขวางของหุ่นยนต์ แสดงตามลำดับตัวเลข

4.4 สรุปผล

จากการทำงานของระบบในภาพรวม ส่วนของการกำหนดพิกัด เริ่มเมื่อบอร์ด Node MCU และอุปกรณ์ที่ติดตั้ง Blynk มีการเชื่อมต่อเครือข่าย Wi-Fi เลือกพิกัดเป้าหมายที่ต้องการให้หุ่นยนต์เคลื่อนที่ไป (สามารถดูได้จาก google map) จากนั้นป้อนค่าพิกัดละติจูลและลองจิจูดผ่านทาง Blynk สามารถกำหนดพิกัดเป้าหมายให้หุ่นยนต์ได้

จากการทำงานของระบบในภาพรวม ส่วนควบคุมการนำทาง เมื่อบอร์ด Arduino รับข้อมูลมาจากบอร์ด Node MCU แล้ว จะนำค่าที่ได้มาคำนวณหามุมและระยะทางในการเคลื่อนที่ร่วมกับค่าที่ได้จากโมดูลเข็มทิศและโมดูล GPS จากนั้นนำค่าที่ได้มาเช็คว่าหุ่นยนต์อยู่ในรัศมีของพิกัดเป้าหมายหรือไม่ หากไม่ได้อยู่ในรัศมีของพิกัดเป้าหมายที่กำหนดไว้ ก็จะควบคุมมอเตอร์ให้หมุนเพื่อให้หุ่นยนต์เคลื่อนที่ไปยังพิกัดนั้น

จากการทำงานของระบบในภาพรวม ส่วนควบคุมการหลบสิ่งกีดขวาง เมื่อเซนเซอร์ตรวจพบวัตถุในระยะ 50 เซนติเมตร มอเตอร์จะหยุดทำงาน จากนั้นจะทำการควบคุมให้หุ่นยนต์หลบสิ่งกีดขวางนั้นๆตามที่ได้อธิบายไว้แล้วในบทที่ 3

4.5 วิเคราะห์ผล

จากผลการทดลองที่กล่าวมาแล้วในข้างต้นสรุปได้ว่า สามารถกำหนดพิกัดเป้าหมายแก่หุ่นยนต์ได้ แต่เนื่องจากข้อจำกัดของตัวแปรประเภท float ที่ใช้ในการเก็บค่าในเรื่องของขนาดข้อมูลและความแม่นยำของตำแหน่งทศนิยม ทำให้ในบางครั้งพิกัดที่กำหนดให้แก่หุ่นยนต์มีความคลาดเคลื่อนเล็กน้อย ดังนั้นระยะทางของเส้นทางที่หุ่นยนต์ต้องเคลื่อนที่จึงคลาดเคลื่อนไปน้อยกว่า 0.3 เมตรจากการคำนวณ แต่หุ่นยนต์ยังสามารถเคลื่อนที่ไปยังพิกัดที่กำหนดได้

บทที่ 5
สรุปและข้อเสนอแนะ

5.1 สรุป

จากการออกแบบและขั้นตอนในการพัฒนาระบบ ทำให้นำไปสู่ผลงานที่สามารถทำได้ตามเป้าหมายขอบเขตที่วางไว้ โดยมีอุปสรรคที่พบได้แก่ การรับส่งข้อมูลที่มีความคลาดเคลื่อนอยู่เล็กน้อย และข้อจำกัดของ library ที่นำมาใช้งาน แต่อย่างไรก็ตาม ในภาพรวมถือว่าสามารถบรรลุเป้าหมายที่วางไว้ โดยจากการพัฒนาระบบที่ผ่านมาสามารถสรุปการทำงานโดยรวมของระบบได้ดังนี้

1. สามารถกำหนดพิกัดเป้าหมายให้แก่หุ่นยนต์ได้ผ่านทาง Blynk application
2. สามารถส่งข้อมูลระหว่าง Arduino และ Node MCU โดยใช้การสื่อสารแบบอนุกรมได้
3. รับค่าพิกัดจากโมดูล GPS และเข็มทิศเพื่อนำมาใช้ในการคำนวณระยะทางและมุม
4. คำนวณเวกเตอร์สำหรับการนำทาง คือระยะทางและมุมเพื่อนำไปใช้ในการควบคุมการนำทางของหุ่นยนต์ จากพิกัดเป้าหมาย พิกัดของหุ่นยนต์ และทิศทางของหัวหุ่นยนต์ที่สัมพันธ์กับเหนือ
5. หุ่นยนต์สามารถเคลื่อนที่ไปยังพิกัดเป้าหมายได้โดยใช้เวกเตอร์สำหรับการนำทางมาใช้ในการหาเส้นทางการเคลื่อนที่
6. ควบคุมการหมุนของมอเตอร์ เพื่อควบคุมการเคลื่อนที่ของหุ่นยนต์ตามเวกเตอร์สำหรับการนำทางที่ได้จากการคำนวณ
7. หุ่นยนต์สามารถหลบสิ่งกีดขวางซึ่งหยุดนิ่งในระยะ 50 เซนติเมตรจากตัวหุ่นยนต์ได้

5.2 ข้อเสนอแนะ

1. ก่อนการใช้งาน ผู้ใช้จะต้องเชื่อมต่อเครือข่ายอินเตอร์เน็ตกับบอร์ด Node MCU และอุปกรณ์ที่ติดตั้งแอพพลิเคชั้น Blynk แต่เนื่องจาก Wi-Fi ของมหาวิทยาลัยที่เป็บแบบ 802.1x ทำให้ค่อนข้างลำบากต่อการเชื่อมต่อกับบอร์ด Node MC จึงใช้การเชื่อมต่อกับ Hotspot จากมือถือ
2. ในระหว่างการเคลื่อนที่ของหุ่นยนต์นั้น ในบางครั้งล้อหมุนอิสระที่ใช้ในการค้ำโครงสร้างของหุ่นยนต์ไม่สามารถหมุนได้ หรือแรงของมอเตอร์ไม่เพียงพอ จึงต้องใช้มือในการดันหุ่นนต์เพื่อให้ล้อสามารถหมุนได้
3. การติดตั้งโมดูลเข็มทิศ จำเป็นต้องติดตั้งให้สูงขึ้นมาและให้ห่างจากอุปกรณ์อื่นๆ เพื่อป้องกันสนามแม่เหล็กรบกวนซึ่งอาจจะทำให้ทิศที่ได้มีความคลาดเคลื่อน
4. ในระหว่างการใช้งานหุ่นยนต์ ต้องเชื่อมต่ออุปกรณ์แล็ปท็อปเพื่อแสดงผลผ่านทาง serial monitor เนื่องจากใช้งานและทดสอบช่วงเวลากลางวัน จึงไม่ควรใช้งานเป็นเวลานานๆ เพราะทำให้เกิดความร้อนแก่อุปกรณ์ได้

บรรณานุกรม

[1] Sunrobotics. “BellaBot,” [Online]. https://www.sunrobotics.co.th/th/articles/202731-bellabot-หุ่นยนต์บริการ-(-service-robot-)-หุ่นยนต์แมวน้อยเสิร์ฟอาหาร-ถูกออกแบบมาให้เหมือนแมวน้อยน่ารัก-cute-สุดๆ . [Accessed Jan. 9, 2021].

[2] global5thailand. “ความรู้ทั่วไปเกี่ยวกับ GPS,” [Online]. https://global5thailand.com/thai/gps.htm [Accessed Jan. 9, 2021].

[3] Garmin ประเทศไทย. “GPS คืออะไร,” [Online]. https://www.garmin.com/th-TH/aboutgps/. [Accessed Jan. 9, 2021].

[4] IoT code mobiles. “GY-NEO6MV2 GPS module,” [Online]. http://www.iot.codemobiles.com/product/296/gy-neo6mv2-gps-module-neo6mv2-with-gps-antenna. [Accessed Fab. 15, 2021].

[5] IoT code mobiles. “GY-NEO6MV2 GPS module,” [Online]. http://www.iot.codemobiles.com/product/296/gy-neo6mv2-gps-module-neo6mv2-with-gps-antenna. [Accessed Fab. 15, 2021].

[6] Analog Read. “GY-273 3-axis Compass Module,” [Online]. https://www.analogread.com/product/446/gy-273-3-axis-compass-module-hmc5883l. [Accessed Fab. 28, 2021].

[7] Analog Read. “GY-273 3-axis Compass Module,” [Online]. https://www.analogread.com/product/446/gy-273-3-axis-compass-module-hmc5883l. [Accessed Feb. 28, 2021].

[8] Honeywell. “3-Axis Digital Compass IC HMC5883L,” [Online]. https://cdn-shop.adafruit.com/datasheets/HMC5883L_3-Axis_Digital_Compass_IC.pdf. [Accessed Aug. 27, 2022].

[9] Arduino4. “BTN7960 BTS7960 43A Current Limiting High-Power H-Bridge DC Motor Drive Module,” [Online]. https://www.arduino4.com/product/844/btn7960-bts7960-43a-current-limiting-high-power-h-bridge-dc-motor-drive-module-โมดูขับ-motor-dc. [Accessed Feb. 28, 2021].

[10] Arduino4. “BTN7960 BTS7960 43A Current Limiting High-Power H-Bridge DC Motor Drive Module,” [Online]. https://www.arduino4.com/product/844/btn7960-bts7960-43a-current-limiting-

high-power-h-bridge-dc-motor-drive-module-โมดูขับ-motor-dc. [Accessed Feb. 28, 2021].

[11] Allnewstep. “NodeMCU ESP8266,” [Online]. https://www.allnewstep.com/article/30/nodemcu-esp8266-esp8285-arduino-1-esp8266-คือ . [Accessed Feb. 15, 2021].

[12] Allnewstep. “NodeMCU ESP8266,” [Online]. https://www.allnewstep.com/article/30/nodemcu-esp8266-esp8285-arduino-1-esp8266-คือ . [Accessed Feb. 15, 2021].

[13] Artronshop. “ESP32 เบื้องต้น :: บทที่ 7 การสื่อสารแบบอนุกรม,” [Online]. https://www.artronshop.co.th/article/57/esp32-เบื้องต้น-บทที่-7-การสื่อสารแบบอนุกรม. [Accessed Seb. 17, 2022].

[14] Blynk. “Blynk,” [Online]. https://blynk.io/ . [Accessed Seb. 17, 2022].

[15] Ermi Media's, Syufrijal and Muhammad Rif'an. “Internet of Things (IoT): BLYNK Framework for Smart Home,” [Online]. https://knepublishing.com/index.php/Kne-Social/article/view/4128/8495#info. [Accessed Seb. 17, 2022].

[16] Apiset. “Ultrasonic Sensor Module (HC-SR04) 5V,” [Online]. https://www.arduitronics.com/product/20/ultrasonic-sensor-module-hc-sr04-5v . [Accessed Feb. 14, 2021].

[17] Apiset. “Ultrasonic Sensor Module (HC-SR04) 5V,” [Online]. https://www.arduitronics.com/product/20/ultrasonic-sensor-module-hc-sr04-5v . [Accessed Feb. 14, 2021].

[18] LungMaker. “การใช้งานบอร์ด Arduino MEGA 2560 เบื้องต้น,” [Online]. http://www.lungmaker.com/arduino-mega-2560-การใช้งาน/. [Accessed Feb. 15, 2021].

[19] LungMaker. “การใช้งานบอร์ด Arduino MEGA 2560 เบื้องต้น,” [Online]. http://www.lungmaker.com/arduino-mega-2560-การใช้งาน/. [Accessed Feb. 15, 2021].

[20] Analog Read. “Step Down/ โมดูลลดแรงดัน,” [Online]. https://www.analogread.com/product/254/dc-to-dc-step-down-lm2596-module-3a-โมดูลลดแรงดันไฟฟ้า-จ่ายกระแสได้สูงสุด-3a-สต็อกไทยส่งไว. [Accessed Seb. 24, 2022].

[21] Analog Read. “Step Down/ โมดูลลดแรงดัน,” [Online]. https://www.analogread.com/product/254/dc-to-dc-step-down-lm2596-module-3a-โมดูลลดแรงดันไฟฟ้า-จ่ายกระแสได้สูงสุด-3a-สต็อกไทยส่งไว. [Accessed Seb. 24, 2022].

[22] Myarduino. “บทความ Arduino เบื้องต้นแบบละเอียด,” [Online]. https://www.myarduino.net/article. [Accessed Jan. 9, 2021].

[23] Shapr3D. “Shapr3D,” [Online]. https://www.shapr3d.com/. [Accessed Seb. 17, 2022].

ประวัติผู้เขียน

ชื่อ นายปวเรศ ปิ่นแก้ว

รหัสนักศึกษา 6010110680

การศึกษา

คุณวุฒิ	สถาบัน	ปีที่สำเร็จการศึกษา
มัธยมศึกษาตอนต้น	โรงเรียนวรนารีเฉลิม สงขลา	2556
มัธยมศึกษาตอนปลาย	โรงเรียนวรนารีเฉลิม สงขลา	2559

กิจกรรม

- 2562 ผู้จัดกิจกรมมค่ายยุวชนคอมพิวเตอร์ ComCamp 29 ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์
- 2565 เข้าร่วมอบรมโครงการ Blazor Programming และ Dev Guidelines
- 2565 เข้าร่วมอบรมโครงการ UX/UI Design with Figma