

สารบัญ

เรื่อง	หน้า
สารบัญ	1
บทที่ 1 บทนำ	2-3
บทที่ 2 โจทย์และแนวคิดในการแก้ปัญหา	4-5
บทที่ 3 ขอบเขตของโครงการ	6
บทที่ 4 ทฤษฎีที่เกี่ยวข้อง	7-19
บทที่ 5 วิธีการดำเนินงาน	20-34
บทที่ 6 แผนการดำเนินงาน	35
บทที่ 7 งบประมาณ	36
บทที่ 8 สรุปผล	37
บทที่ 9 เอกสารอ้างอิง	38

บทที่ 1

1. บทนำ

หุ่นยนต์ (robot) คือ เครื่องจักรกลหรือหุ่นที่มีเครื่องกลไกอยู่ภายใน สามารถทำงานได้หลายอย่าง ร่วมกับมนุษย์หรือแทนมนุษย์ และสามารถตั้งลำดับแผนการทำงานก่อนหลังได้ โดยสถาบันหุ่นยนต์แห่งสหรัฐอเมริกา (RIA : The Robotics Institute of America) สามารถจำแนกเป็น 2 ประเภทใหญ่ๆ ตามลักษณะการใช้งาน คือ 1.หุ่นยนต์ชนิดติดตั้งอยู่กับที่ (fixed robot) หุ่นยนต์ประเภทนี้ มีลักษณะเป็นแขนกลสามารถขยับและเคลื่อนไหวได้เฉพาะข้อต่อ นิยมใช้ในโรงงานอุตสาหกรรม และ 2.หุ่นยนต์ชนิดเคลื่อนที่ได้ (mobile robot) หุ่นยนต์ประเภทนี้ สามารถขยับเคลื่อนที่ไปได้ด้วยตัวเอง โดยการใช้ล้อ ขา หรือการขับเคลื่อนในรูปแบบอื่นๆ[1]

หุ่นยนต์ (Robot) และระบบอัตโนมัติมีความคล้ายกันในแง่ของการเป็นเครื่องจักรอัตโนมัติ (Automation Machine) โดยหุ่นยนต์สามารถเรียกได้ว่าเป็นส่วนหนึ่งในระบบอัตโนมัติได้เนื่องจากมีองค์ประกอบและการทำงานที่คล้ายกัน แต่หุ่นยนต์จะสามารถทำงานจากโปรแกรมการตัดสินใจและสามารถปรับเปลี่ยนโปรแกรม การทำงานให้ทำงานหลากหลายหน้าที่ได้ ซึ่งระบบอัตโนมัติไม่สามารถทำได้ สำหรับองค์ประกอบที่สามารถ แสดงให้เห็นอย่างชัดเจนสำหรับหุ่นยนต์ คือ องค์ประกอบของระบบในการควบคุม หุ่นยนต์ซึ่งประกอบด้วย องค์ประกอบหลักซึ่งจะมีอยู่ด้วยกัน 3 ส่วนที่มีความสัมพันธ์กัน ได้แก่ Programming Pendant หรือ อุปกรณ์ ที่ทำหน้าที่ในการป้อนคำสั่งโดยผู้ควบคุมหรือผู้ใช้งาน, Controller หรือ ส่วนที่ทำหน้าที่ในการรับคำสั่งจาก ผู้ใช้งานผ่านอุปกรณ์ที่ทำหน้าที่ในการป้อนคำสั่งและนำมาประมวลผล เพื่อทำการควบคุมหรือสั่งการทำงาน ของหุ่นยนต์ต่อไป และสุดท้าย Manipulator หรือเรียกง่ายๆ ว่า “ตัวหุ่นยนต์” ที่ จะทำงานตามคำสั่งที่ผ่าน การประมวลผลจากส่วนที่ทำหน้าที่ในการรับคำสั่งจากผู้ใช้งาน

หุ่นยนต์มีวิวัฒนาการและความก้าวหน้าอย่างรวดเร็วต่อเนื่องมาตลอดหลายปีที่ผ่านมา โดยได้เข้ามามีบทบาทมากขึ้นในชีวิตของมนุษย์ ทั้งในด้านการช่วยเพิ่มผลผลิตในกระบวนการผลิตสินค้า ช่วยดูแลในเรื่องคุณภาพชีวิต ไปจนถึงการสร้างความสะดวกสบายต่างๆ สำหรับการใช้งานหุ่นยนต์และระบบอัตโนมัติในประเทศไทยมีแนวโน้มเพิ่มขึ้นอย่างต่อเนื่อง แต่อย่างไรก็ตาม ปัญหาที่สำคัญของอุตสาหกรรมหุ่นยนต์ไทย คือ การขาดตลาดภายในประเทศ เพราะผู้ใช้งานมักไม่ได้ให้ความสำคัญกับหุ่นยนต์ที่พัฒนาในประเทศเท่าที่ควร จากการที่ผู้ผลิตรายใหญ่ๆ ที่มีการปรับปรุงกระบวนการผลิตและการบริการเป็นระบบอัตโนมัติต่างๆ ล้วนนำเข้าหุ่นยนต์จากต่างประเทศ นอกจากนี้ ยังขาดการส่งเสริมผู้ประกอบการหุ่นยนต์รุ่นใหม่ที่มีศักยภาพในการคิดค้นนวัตกรรมให้เติบโตจนกลายเป็นวิสาหกิจเริ่มต้นทางด้านเทคโนโลยี ที่เป็นฐานเศรษฐกิจใหม่ของประเทศ ในอนาคต ทำให้ไทยยังต้องอาศัยการนำเข้าหุ่นยนต์และระบบอัตโนมัติที่มีมูลค่าสูงจากต่างประเทศเป็นหลัก ดังนั้น เพื่อเป็นการสร้างความแข็งแกร่งให้กับประเทศ จึงมีความจำเป็นอย่างยิ่งที่จะต้องเตรียมความพร้อมในด้านต่างๆ เพื่อส่งเสริมให้เกิดการวิจัย พัฒนา ตลอดจนส่งเสริมอุตสาหกรรมต่างๆ ที่เกี่ยวข้องทั้งทางตรงและทางอ้อมต่อไป[2]

คณะผู้จัดทำได้เล็งเห็นถึงความสำคัญในการนำหุ่นยนต์มาใช้ในการเรียนการสอนรายวิชา PRE-PROJECT เพื่อเป็นการต่อยอดและเพิ่มศักยภาพในการพัฒนาหุ่นยนต์ให้นักศึกษา จึงนำไปสู่การประดิษฐ์

หุ่นยนต์ขึ้นมาเพื่อใช้ในการแข่งขันภายในห้องเรียนโดยมีชื่อว่า “SUPERCAR ROBOT” ซึ่งการแข่งขันหุ่นยนต์มีลักษณะคล้ายกับการเล่น บอลลูกด่าน หรือ เล่นเตย โดยแบ่งเป็นทีมรุกและทีมรับสลับกันในการแข่งแต่ละรอบ โดยทีมหนึ่งจะประกอบด้วยหุ่นยนต์ 7 ตัว ฝ่ายทีมรุกจะต้องวิ่งไปหาฝั่งตรงข้าม จนผ่านเส้นแดง แล้วกลับมาอย่างปลอดภัย(ผ่านเส้นสีเหลือง) โดยที่ไม่ถูกทีมรับจับได้ก็จะเป็นฝ่ายชนะในการแข่งขันรอบนั้น หุ่นยนต์ที่ถูกจับ ได้จะถูกตัดออกจากการแข่งขันในรอบนั้น ส่วนทีมรับจะสามารถวิ่งสกัดกั้นฝ่ายตรงข้ามในพื้นที่ป้องกันเท่านั้น ถ้าวิ่งออก นอกพื้นที่ก็จะถูกตัดออกจากการแข่งขันในรอบนั้นเช่นกัน ถ้าไม่มีหุ่นยนต์ตัวไหนสามารถผ่านด่านได้ ทีมรับจะเป็นฝ่ายชนะ การแข่งขันของแต่ละรอบจะยุติเมื่อทีมรุกสามารถผ่านด่านได้สำเร็จ หรือเมื่อทีมใดทีมหนึ่งไม่เหลือผู้เล่น

บทที่ 2

2. โจทย์และปัญหาที่ต้องการแก้ไข

โจทย์การแข่งขัน

การแข่งขันหุ่นยนต์มีลักษณะคล้ายกับการเล่น บอลลุนดำน หรือ เล่นเตย โดยแบ่งเป็นทีมรุกและทีมรับสลับกันในการแข่งแต่ละรอบ โดยทีมหนึ่งจะประกอบด้วยหุ่นยนต์ 7 ตัว ฝ่ายทีมรุกจะต้องวิ่งไปหาฝั่งตรงข้าม จนผ่านเส้นแดง แล้วกลับมาอย่างปลอดภัย(ผ่านเส้นสีเหลือง) โดยที่ไม่ถูกทีมรับจับได้ก็จะเป็นฝ่ายชนะในการแข่งขันรอบนั้น หุ่นยนต์ที่ถูกจับ ได้จะถูกตัดออกจากการแข่งขันในรอบนั้น ส่วนทีมรับจะสามารถวิ่งสกัดกั้นฝ่ายตรงข้ามในพื้นที่ป้องกันเท่านั้น ถ้าวิ่งออก นอกพื้นที่ก็จะถูกตัดออกจากการแข่งขันในรอบนั้นเช่นกัน ถ้าไม่มีหุ่นยนต์ตัวไหนสามารถผ่านด่านได้ ทีมรับจะเป็นฝ่ายชนะ การแข่งขันของแต่ละรอบจะยุติเมื่อทีมรุกสามารถผ่านด่านได้สำเร็จ หรือเมื่อทีมใดทีมหนึ่งไม่เหลือผู้เล่น

การคิดคะแนน

- คะแนนจะแบ่งเป็นคะแนนกลุ่ม และคะแนนทีม
- หุ่นยนต์ตัวใดถูกตัดออกจากการแข่งขัน กลุ่มนั้นก็จะไม่ได้คะแนนในรอบนั้น ส่วนกลุ่มใดสามารถผ่านด่านได้สำเร็จ(ทีมรุก) หรือสามารถสกัดกั้นฝ่ายตรงข้ามได้สำเร็จ ก็จะได้ 1 คะแนน
- ทีมที่ชนะ จะได้คะแนนทีมละ 5 คะแนน
- คะแนนรวมของแต่ละกลุ่มจะคิดจากคะแนนกลุ่มรวมกับคะแนนทีม โดยจะแข่งกันทั้งหมด 12 รอบ (รุก 3 รับ 3 และสลับฝั่ง)
- นอกจากนั้นจะมีการโหวตคะแนนให้กับแต่ละกลุ่มจากผู้ชม รวมทั้งอาจารย์ที่เข้าร่วมกิจกรรมด้วย โดยแบ่งเป็นคะแนนเทคนิคยอดเยี่ยม คะแนนการออกแบบยอดเยี่ยม คะแนนขวัญใจผู้ชม

แนวคิดในการแก้ปัญหา

1.กลยุทธเกมรุก

1.1 ใช้ IR Infrared Obstacle Avoidance Sensor ตรวจจับสิ่งกีดขวางข้างหน้าด้านซ้ายและด้านขวา

- เซนเซอร์ด้านขวาตรวจจับวัตถุได้ให้รถวิ่งไปทางซ้ายแล้วหยุด 0.04 วินาที
- เซนเซอร์ด้านซ้ายตรวจจับวัตถุได้ให้รถวิ่งไปทางขวาแล้วหยุด 0.04 วินาที
- เซนเซอร์ตรวจเจอวัตถุระยะทั้งซ้ายและขวาให้รถถอยหลังเป็นเวลา 3 วินาที
- เซนเซอร์ตรวจไม่เจอสิ่งกีดขวางทั้งสองข้างให้รถเดินหน้าเต็มกำลัง

1.2 ใช้ Ultrasonic Sensor ตรวจจับวัตถุและเซ็นระยะตรวจจับได้

- เซนเซอร์ตรวจจับวัตถุได้ระยะน้อยกว่าหรือเท่ากับ 10 เซนติเมตรให้รถหยุด 0.04 วินาที
- เซนเซอร์ตรวจจับวัตถุได้ระยะมากกว่า 10 เซนติเมตรให้รถตรวจสอบสิ่งกีดขวางข้างหน้าด้านซ้ายและด้านขวา

1.3 ใช้ TCRT5000 Infrared IR sensor detection ตรวจสอบความเข้มแสงของเส้นสีดำและสีแดง

- เซนเซอร์ตรวจจับเส้นสีแดงได้ให้หันรถกลับ

2.กลยุทธ์เกมรับ

2.1 ใช้ IR Infrared Obstacle Avoidance Sensor ตรวจสอบสิ่งกีดขวางข้างหน้าด้านซ้ายและด้านขวา

- เซนเซอร์ด้านขวาตรวจจับวัตถุได้ให้หันรถวิ่งไปทางขวาแล้วหยุด 0.04 วินาที
- เซนเซอร์ด้านซ้ายตรวจจับวัตถุได้ให้หันรถวิ่งไปทางซ้ายแล้วหยุด 0.04 วินาที
- เซนเซอร์ตรวจเจอวัตถุระยะทั้งซ้ายและขวาให้รถหยุดนิ่ง

2.2 ใช้ TCRT5000 Infrared IR sensor detection ตรวจสอบความเข้มแสงของเส้นสีดำและสีแดง

- เซนเซอร์ตรวจจับเส้นสีดำได้ให้หันรถกลับ

บทที่ 3

3. ขอบเขตของโครงการ

ขนาด น้ำหนัก ความเร็ว

ขนาด : 10 x 10 เซนติเมตร ,ความสูงไม่เกิน 13 เซนติเมตร, ล้อที่ใช้ในการขับเคลื่อน 2 ล้อ

เครื่องมือ อุปกรณ์ที่จำเป็นในการทำโครงการ

1. Arduino ESP32 Wemos D1
2. US-025 ultrasonic sensor module
3. IR Infrared Obstacle Avoidance Sensor Module
4. TCRT5000 Infrared IR sensor detection
5. DC/DC Step-up Converter รุ่น MT3608
6. H-Bridge Driver Mini L298N 2-Way
7. DC N20 Mini Micro Gear Motor
8. ตัวต้านทาน 220 โอห์ม 5%
9. ไมโครสวิตช์กดติดปลดปล่อยดับแบบ 4 ขา
10. Male to Male Jumper Wires Cables
11. Battery Li-ion 18650 3.7 V
12. ล้อยางเหมาะสำหรับ N20
13. Photo board 8.5CM X 5.5CM
14. ล้อรถเซ็น Smart Car 15 mm
15. โครงรถ
16. น็อต M3 ตัวผู้และตัวเมีย

บทที่ 4

4. ทฤษฎีที่เกี่ยวข้อง

4.1 หลักการเขียนโปรแกรม Arduino C++[3]

ภาษาซีของ Arduino จะจัดรูปแบบโครงสร้างของการเขียนโปรแกรมออกเป็นส่วนย่อยๆ หลายๆ ส่วน โดยเรียกแต่ละส่วนว่า “ฟังก์ชัน” และเมื่อนำฟังก์ชันมารวมเข้าด้วยกัน ก็จะเรียกว่าโปรแกรม โดยโครงสร้างการเขียนโปรแกรมของ Arduino นั้น ทุกๆ โปรแกรมจะต้องประกอบไปด้วยฟังก์ชันจำนวนเท่าใดก็ได้ แต่อย่างน้อยที่สุดต้องมีฟังก์ชันจำนวน 2 ฟังก์ชัน คือ setup() และ loop()

4.1.1 ฟังก์ชัน

ฟังก์ชัน setup() คือฟังก์ชันใช้ในการประกาศค่าเริ่มต้น ตำแหน่งพอร์ตที่ใช้งานรวมถึงฟังก์ชันที่อยู่โลบารีที่ใช้งาน เป็นฟังก์ชันที่ทำงานเพียงครั้งเดียวจะทำงานทุกครั้งที่มีการรีเซต หรือรีบูตเครื่องใหม่เท่านั้น เช่น

```
int buttonPin = 3; // การตั้งค่าตัวแปร buttonPin เท่ากับ 3
void setup()
{
    Serial.begin(9600); // ประกาศการใช้งานการสื่อสารรับส่งข้อมูลผ่าน พอร์ตRS232
    pinMode(buttonPin, INPUT); // การตั้งค่าโหมด ของตัวแปรแบบคงที่ buttonPin เป็นโหมด
    อินพุต
}
```

ฟังก์ชัน Loop () คือฟังก์ชันใช้ในการเขียนโค้ดโปรแกรมการทำงานของ Arduino เป็นฟังก์ชันการวนลูปไปเรื่อยๆ เช่น

```
const int buttonPin = 3; // การตั้งค่าตัวแปร buttonPin เท่ากับ 3
void loop ()
{
    if (digitalRead(buttonPin) == HIGH) // ตรวจสอบค่าอินพุตที่รับมา เป็น HIGH ใช่หรือไม่
        Serial.write('H'); // ใช่ ส่งค่าอักษร H ผ่าน พอร์ตRS232
    else
        Serial.write('L'); // ไม่ ส่งค่าอักษร L ผ่าน พอร์ตRS232
    delay(1000); // หน่วงเวลา 1วินาที
}
```

4.1.2 ข้อกำหนดของไวยากรณ์[4]

เป็นกฎเกณฑ์ในการสร้างประโยคขึ้นมาอธิบายความหมายของโปรแกรม มีดังต่อไปนี้

- เครื่องหมาย ; (เซมิโคลอน) เป็นการจบคำสั่งในบรรทัดนั้น ๆ ตัวอย่าง `int ledPin = 13;`
ข้อควรระวัง ถ้าไม่ได้พิมพ์เครื่องหมาย ; ปิดท้ายคำสั่งในบรรทัดใดบรรทัดหนึ่งส่งผลให้โปรแกรม นั้นคอมไพล์โปรแกรมไม่ผ่าน
- เครื่องหมาย {} (วงเล็บปีกกา) เป็นการกำหนดบล็อกของคำสั่ง ใช้กับคำสั่ง `if`, `else`, `while` หรือ `for`
ข้อควรระวัง การใส่วงเล็บปีกกาต้องใส่ทั้งวงเล็บปีกกา { และ } ให้ครบคู่ กรณีที่ใส่ไม่ครบคู่ส่งผลให้โปรแกรมนั้นคอมไพล์โปรแกรมไม่ผ่าน
- เครื่องหมาย // (หมายเหตุบรรทัดเดียว) เป็นส่วนของผู้เขียนโปรแกรมอธิบายเพิ่มเติม ในคำสั่งต่าง ๆ ว่าโปรแกรมทำงานอย่างไรในแต่ละบรรทัด เมื่อทำการคอมไพล์โปรแกรมประโยคที่อยู่ หลังเครื่องหมาย // ไม่ได้ถูกนำไปคอมไพล์ด้วย การใส่เครื่องหมาย // (หมายเหตุ) มีประโยชน์สำหรับ การตรวจสอบโปรแกรมภายหลัง หรือให้ผู้พัฒนาโปรแกรมท่านอื่นสามารถเข้าใจการเขียนโปรแกรม นั้น ๆ ได้
- เครื่องหมาย /* */ (หมายเหตุหลายบรรทัด) เป็นส่วนของผู้เขียนโปรแกรมอธิบาย เพิ่มเติมในคำสั่งต่าง ๆ ว่าโปรแกรมทำงานอย่างไรสามารถอธิบายได้หลายบรรทัด
- เครื่องหมาย `#define` เป็นคำสั่งในการกำหนดค่าคงที่ให้กับโปรแกรม มีรูปแบบคำสั่งดังนี้
`#define ชื่อตัวแปรค่าคงที่ ค่าคงที่`
- เครื่องหมาย `#include` เป็นคำสั่งให้นำไฟล์อื่นเข้ามาพร้อมกับไฟล์โปรแกรมหลักมีรูปแบบคำสั่งดังนี้ `#include <ชื่อไฟล์>`

4.1.3 ชุดคำสั่งในการควบคุม

- คำสั่ง `if` เป็นคำสั่งในการตรวจสอบเงื่อนไขการทำงานของโปรแกรมถ้าเงื่อนไขเป็นจริง ให้ทำงานตามคำสั่งที่กำหนดนั้น
- คำสั่ง `if...else` เป็นคำสั่งกำหนดเงื่อนไขการทำงานของโปรแกรม โดยมี 2 เงื่อนไข ถ้าเงื่อนไขเป็นจริงทำงานตามคำสั่งที่กำหนดแบบหนึ่ง ถ้าเงื่อนไขเป็นเท็จทำงานตามคำสั่งที่กำหนดอีกแบบหนึ่ง
- คำสั่ง `for` เป็นคำสั่งให้โปรแกรมทำงานซ้ำตามจำนวนรอบที่ต้องการมีรูปแบบคำสั่งคือ `for` (ค่าเริ่มต้น, เงื่อนไขการทำซ้ำ; การเพิ่มหรือลดค่าตัวแปรในแต่ละรอบ)
- คำสั่ง `Switch case` เป็นคำสั่งเพื่อกำหนดการทำงานของโปรแกรมหลาย ๆ เงื่อนไข ถ้าตัวแปรที่กำหนดตรงกับเงื่อนไขนั้น ๆ ทำให้โปรแกรมทำงานตามที่กำหนดไว้แต่ละเงื่อนไข
- คำสั่ง `while` เป็นคำสั่งทำซ้ำแบบวนรอบ ถ้าเงื่อนไขเป็นจริงโปรแกรมทำงานตามคำสั่งที่เขียนไว้ในวงเล็บปีกกา แต่ถ้าเงื่อนไขเป็นเท็จโปรแกรมจบการทำงานในคำสั่ง `while`
- คำสั่ง `do..while` เป็นคำสั่งทำซ้ำแบบวนรอบ โดยมีการทำงานตรงกันข้ามกับคำสั่ง

while คือทำงานตามคำสั่งที่เขียนไว้ในวงเล็บปีกกา แล้วจึงมาตรวจสอบเงื่อนไข แต่ถ้าเงื่อนไขเป็นเท็จ โปรแกรมจบการทำงานในคำสั่ง do

- คำสั่ง break เป็นคำสั่งใช้ร่วมกับคำสั่งการทำงานแบบวนรอบ ได้แก่ คำสั่ง do, for while หรือ Switch เพื่อให้โปรแกรมหยุดการทำงานจากการวนรอบโดยไม่มีเงื่อนไข
- คำสั่ง continue เป็นคำสั่งใช้สำหรับข้ามการทำงานของคำสั่งถัดไป คำสั่งนี้เขียนอยู่ใน คำสั่งการทำงานแบบวนรอบ ได้แก่ คำสั่ง do, for หรือ while
- คำสั่ง return เป็นคำสั่งจบการทำงานในโปรแกรมน้อย
- คำสั่ง goto เป็นคำสั่งกระโดดโดยไม่มีเงื่อนไขไปยังตำแหน่งที่กำหนด โดยอ้างถึงตำแหน่ง Label ที่กระโดดไป

4.1.4 การดำเนินการทางคณิตศาสตร์

เครื่องหมายทางคณิตศาสตร์ในการเขียนโปรแกรม สามารถกระทำกับข้อมูลได้หลายรูปแบบ มีเครื่องหมายดังต่อไปนี้

- เครื่องหมาย + เป็นการบวกของตัวถูกกระทำสองตัว
- เครื่องหมาย - เป็นการลบของตัวถูกกระทำสองตัว
- เครื่องหมาย * เป็นการคูณของตัวถูกกระทำสองตัว
- เครื่องหมาย / เป็นการหารของตัวถูกกระทำสองตัว
- เครื่องหมาย % เป็นการหารเอาเศษ ใช้หาค่าเศษที่ได้จากการหาร

4.1.5 การดำเนินการเปรียบเทียบ

เป็นเครื่องหมายที่ใช้ในการเปรียบเทียบทางคณิตศาสตร์ มีเครื่องหมายดังต่อไปนี้

- เครื่องหมาย == เป็นการเปรียบเทียบเท่ากับ
- เครื่องหมาย != เป็นการเปรียบเทียบไม่เท่ากับ
- เครื่องหมาย < เป็นการเปรียบเทียบน้อยกว่า
- เครื่องหมาย > เป็นการเปรียบเทียบมากกว่า
- เครื่องหมาย <= เป็นการเปรียบเทียบน้อยกว่าหรือเท่ากับ
- เครื่องหมาย >= เป็นการเปรียบเทียบมากกว่าหรือเท่ากับ

4.1.6 ตัวแปร

ตัวแปร เป็นชื่อเรียกแทนพื้นที่เก็บข้อมูลในหน่วยความจำของไมโครคอนโทรลเลอร์ โดยมีการตั้งชื่อเรียกหน่วยความจำในตำแหน่งนั้น เพื่อความสะดวกในการเรียกใช้ข้อมูล ซึ่งมีชนิดของข้อมูลหรือแบบของตัวแปรต่าง ๆ ดังนี้

1.ค่าคงที่

เป็นคำสั่งข้อความที่กำหนดไว้ในโปรแกรม Arduino มีคำสั่งดังต่อไปนี้

- คำสั่ง HIGH/LOW แทนสถานะลอจิก “1” กับลอจิก “0”
- คำสั่ง INPUT/OUTPUT ใช้สำหรับกำหนดค่าอินพุตกับเอาต์พุต

- คำสั่ง true/false เป็นค่าคงที่แบบบูลีน โดย true แทนสถานะค่าใด ๆ ที่ไม่ใช่ 0 ถือว่า เป็นจริง ส่วน false มีค่าเป็น 0 หรือเป็นเท็จ
- คำสั่ง integer Constants เป็นค่าคงที่ของเลขจำนวนเต็ม
- คำสั่ง floating point Constants เป็นค่าคงที่ของเลขทศนิยม

2. ชนิดของข้อมูล

สามารถแบ่งชนิดของข้อมูลได้ดังนี้

- Void ใช้เฉพาะในการประกาศฟังก์ชัน
- boolean มีค่าจริงหรือเท็จ
- char มีค่าตั้งแต่ 127 ถึง 127 ใช้สำหรับเก็บข้อมูลที่เป็นตัวอักษร
- unsigned char มีค่าตั้งแต่ 0 ถึง 255
- byte มีค่าตั้งแต่ 0 ถึง 255
- unsigned int มีค่าตั้งแต่ 0 ถึง 65,555
- unsigned long มีค่าตั้งแต่ 0 ถึง 4,294,967,295
- float มีค่าตั้งแต่ 3.4028235E+38 ถึง 3.4028235E+38
- double มีค่าตั้งแต่ 3,4028235E+38 ถึง 3.4028235E+38
- string ตัวแปรสำหรับเก็บข้อความ
- array ตัวแปรหลายตัวที่ถูกเก็บรวมไว้ในตัวแปรชื่อเดียวกัน

4.1.6 ชุดคำสั่ง ชุดคำสั่งเป็นชุดคำสั่งในการเขียนโปรแกรมเพื่อให้ไมโครคอนโทรลเลอร์ทำงานตามโปรแกรมที่ออกแบบไว้ โดยมีคำสั่งต่าง ๆ ดังนี้

1. คำสั่งดิจิตอล อินพุต/เอาต์พุต มีคำสั่งดังต่อไปนี้

- คำสั่ง pinMode() เป็นการกำหนดพอร์ตเป็นอินพุตหรือเอาต์พุต
- คำสั่ง digitalWrite() เป็นการเขียนข้อมูลออกพอร์ตที่กำหนด
- คำสั่ง digitalRead() เป็นการอ่านข้อมูลเข้าพอร์ตที่กำหนด

2. คำสั่งอนาล็อก อินพุต/เอาต์พุต มีคำสั่งดังต่อไปนี้

- คำสั่ง analogReference() เป็นการกำหนดค่าแรงดันอ้างอิงที่ใช้สำหรับอนาล็อกอินพุต
- คำสั่ง analogRead() เป็นการอ่านแรงดันไฟฟ้าแบบอนาล็อกและแปลงเป็นจำนวนเต็ม มีค่าระหว่าง 0 ถึง 1023
- คำสั่ง analogWrite() เป็นการใช้ PWM เขียนค่าออกทางพอร์ตที่กำหนด

3. คำสั่งเวลา มีคำสั่งดังต่อไปนี้

- คำสั่ง millis() เป็นการหนดเวลา มีหน่วยเป็นมิลลิวินาทีของ Arduino ทันทีที่มีไฟเลี้ยงเข้า Arduino
- คำสั่ง delay() เป็นการหนดเวลาตามค่าที่กำหนด มีหน่วยเป็นมิลลิวินาที
- คำสั่ง delayMicroseconds() เป็นการหนดเวลาตามค่าที่กำหนด มีหน่วยเป็นไมโครวินาที

4. คำสั่งคณิตศาสตร์ มีคำสั่งดังต่อไปนี้

- คำสั่ง min() เป็นการหาค่าต่ำสุด
- คำสั่ง max() เป็นการหาค่ามากที่สุด

5. คำสั่งบิตและไบต์ มีคำสั่งดังต่อไปนี้

- คำสั่ง LowByte() เป็นตัวแปรของไบต์ต่ำสุด
- คำสั่ง highByte() เป็นตัวแปรของไบต์สูงสุด
- คำสั่ง bitRead() เป็นการอ่านบิตของตัวแปร
- คำสั่ง bitWrite() เป็นการเขียนบิตของตัวแปร
- คำสั่ง bitSet() เป็นการตั้งบิตของตัวแปรเท่ากับ 1
- คำสั่ง bitClear() เป็นการตั้งบิตของตัวแปรเท่ากับ 0
- คำสั่ง bit() เป็นการตั้งค่าบิตตามค่าที่กำหนด

6. คำสั่งการติดต่อสื่อสาร มีคำสั่งดังต่อไปนี้

- คำสั่ง Serial.begin() เป็นการกำหนดอัตราการส่งข้อมูล
- คำสั่ง Serial.end() เป็นการปิดใช้งานการสื่อสารแบบอนุกรม
- คำสั่ง Serial.available() เป็นการตรวจสอบการรับข้อมูลจากการสื่อสารแบบอนุกรม
- คำสั่ง Serial.read() เป็นการอ่านข้อมูลจากการสื่อสารแบบอนุกรมที่เข้ามา
- คำสั่ง Serial.peek() เป็นการส่งกลับไบต์ต่อไปของข้อมูลการสื่อสารแบบอนุกรม
- คำสั่ง Serial.flush() เป็นการลบข้อมูลทั้งหมดในบัฟเฟอร์
- คำสั่ง Serial.print() เป็นการพิมพ์ข้อมูลไปยังพอร์ตอนุกรม
- คำสั่ง Serial.println() เป็นการพิมพ์ข้อมูลไปยังพอร์ตอนุกรม และขึ้นบรรทัดใหม่
- คำสั่ง Serial.write() เป็นการส่งข้อมูลไบต์ไปยังพอร์ตอนุกรม

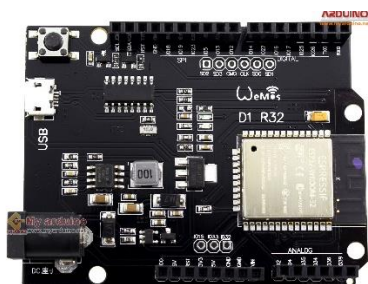
4.2 ไมโครคอนโทรลเลอร์ (Microcontroller)

ไมโครคอนโทรลเลอร์ (อังกฤษ: Microcontroller มักย่อว่า μ C, uC หรือ MCU) คืออุปกรณ์ควบคุมขนาดเล็ก ซึ่งบรรจุความสามารถที่คล้ายคลึงกับระบบคอมพิวเตอร์ โดยในไมโครคอนโทรลเลอร์ได้รวมเอาซีพียู หน่วยความจำ และพอร์ตซึ่งเป็นส่วนประกอบหลักสำคัญของระบบคอมพิวเตอร์เข้าไว้ด้วยกัน โดยทำการบรรจุเข้าไว้ในตัวถังเดียวกัน ถ้าแปลความหมายไมโครคอนโทรลเลอร์แบบตรงตัวก็คือ ระบบคอนโทรลขนาดเล็ก เรียกอีกอย่างหนึ่งคือเป็นระบบคอมพิวเตอร์ขนาดเล็ก ที่สามารถนำมาประยุกต์ใช้งานได้หลากหลาย โดยผ่านการออกแบบ วงจรให้เหมาะกับงานต่างๆ และยังสามารถโปรแกรมคำสั่งเพื่อควบคุมขา Input / Output เพื่อสั่งงานให้ไป ควบคุมอุปกรณ์ต่างๆ ได้อีกด้วย ซึ่งก็นับว่าเป็นระบบที่สามารถนำมาประยุกต์ใช้งานได้หลากหลาย ทั้งทางด้าน Digital และ Analog ยกตัวอย่างเช่น ระบบสัญญาณตอบรับอัตโนมัติ, ระบบบัตรคิว, ระบบตอกบัตร พนักงาน และอื่นๆ ยิ่งระบบไมโครคอนโทรลเลอร์ ในยุคปัจจุบันนี้สามารถทำการเชื่อมต่อกับระบบ

Network ของคอมพิวเตอร์ทั่วไปได้อีกด้วย ดังนั้นการส่งงานจึงไม่ใช่แค่หน้าแผงวงจร แต่อาจจะเป็นการส่งงานอยู่คนละ ซีกโลกผ่านเครือข่ายอินเทอร์เน็ตก็ได้[5]

ไมโครคอนโทรลเลอร์ Arduino (ออกเสียงเป็นภาษาอิตาลีว่า อา-ดู-อี-โน หรือ อาดูยโน) ที่พัฒนาขึ้นในแบบโอเพนซอร์ส (Open Source) คือเปิดเผยข้อมูลทั้งทางด้านฮาร์ดแวร์และด้านซอฟต์แวร์ที่นักพัฒนาด้านไมโครคอนโทรลเลอร์สามารถนำไปพัฒนาต่อยอดได้ในด้านธุรกิจและการศึกษาโดยไม่เสียค่าลิขสิทธิ์ แผงวงจรหรือบอร์ด Arduino ใช้ชิปไมโครคอนโทรลเลอร์ตระกูล AVR ของบริษัท Atmel ที่ออกแบบมาให้ใช้งานได้ง่าย และซอฟต์แวร์ Arduino IDE ที่ใช้ภาษาซี (C/C++) สำหรับการพัฒนาโปรแกรม รวมถึงได้พัฒนาโปรแกรมควบคุมการทำงานพื้นฐานที่เรียกว่า “บูทโหลดเดอร์” (Boot loader Firmware) ที่ทำหน้าที่ควบคุมการทำงานของรีจิสเตอร์ภายในและเป็นเครื่องโปรแกรม (Programmer) ทำให้การพัฒนาโปรแกรมควบคุมไมโครคอนโทรลเลอร์ทำได้ง่ายและสะดวกขึ้น จึงได้รับความนิยมในการนำมาพัฒนาและประยุกต์ใช้งานอย่างมากในปัจจุบัน

แผงวงจรของไมโครคอนโทรลเลอร์ Arduino มีหลายรุ่น แต่ละรุ่นมีคุณสมบัติ, จำนวนขาพอร์ต (I/O Port หรือเรียกว่า Pin) และหน่วยความจำแตกต่างกันตามความต้องการของผู้ใช้งานในหน่วยนี้จะศึกษาและปฏิบัติเกี่ยวกับการใช้งานไมโครคอนโทรลเลอร์ Arduino รุ่น UNO, การใช้ซอฟต์แวร์ Arduino IDE ภาษาซีสำหรับไมโครคอนโทรลเลอร์ Arduino และการเชื่อมต่อกับอุปกรณ์อินพุต-เอาต์พุตพื้นฐาน และการเขียนโปรแกรมควบคุมเบื้องต้นโดยใช้บอร์ด i-Duino UNO ที่มีคุณสมบัติเหมือนกันกับบอร์ด Arduino UNO ต้นแบบ[6]



รูปที่ 1 Arduino WEMOS-D1-R32

Arduino WEMOS-D1-R32[7]

รูปแบบบอร์ดและขาที่ต่อ เช่นเดียวกับบอร์ด Arduino UNO โดยเปลี่ยนจาก MCU ในตระกูล AVR เป็น MCU ในตระกูล ESPRESSIF รุ่น ESP32 มีวงจร WIFI, BLUETOOTH, ขยายหน่วยความจำที่ใช้เขียนให้มากขึ้น, พร้อมความเร็วในการทำงานที่สูงขึ้น และที่สำคัญผู้ใช้สามารถเขียนโปรแกรมด้วยภาษา C ของ Arduino ได้อีกด้วย, นำไปใช้ทำอุปกรณ์ IOT ได้โดยง่าย

- ใช้ MCU ประจำบอร์ดเป็น MODULE ESP-WROOM-32 ของบริษัท ESPRESSIF
 - MCU สถาปัตยกรรม TENSILICA LX6 แบบ 2 DUAL CORE, RUN 240 MHz (600 DMIPS)
 - มี WIFI (802.11 b/g/n/e/i) และ BLUETOOTH 4.2 อยู่ใน MODULE
 - หน่วยความจำ FLASH โปรแกรม 4 MBYTE, RAM 520 KBYTE

- 32 GPIO โดยบางขาสามารถทำงานได้มากกว่าหนึ่งหน้าที่
- A TO D ขนาด 12 BIT ได้ถึง 18 ช่อง, D TO A ขนาด 8 BIT 2 ช่อง, ต่อ CAPACITIVE TOUCH ได้ 10 ช่อง, SPI ได้ 3 ช่อง UART ได้ 3 ช่อง, I2C ได้ 2 ช่อง, PWM ได้ 16 ช่อง
- ใช้ CHIP ในการติดต่อทาง USB PORT ในการเขียนโปรแกรมเข้าบอร์ดทาง PORT USB เบอร์ CH340, ขั้วต่อ USB แบบ MICRO USB
- ระดับสัญญาณลอจิกของสัญญาณ INPUT/OUTPUT 3.3V ท ระดับสัญญาณในส่วน ANALOG A/D 12 BIT 3.2V
- ขั้วต่อ CONNECTOR จะเหมือนกับ Arduino UNO สามารถใช้ร่วมกับ SHIELD ต่างๆ ได้
- ใช้ไฟเลี้ยงวงจรจาก
 - USB PORT แบบ MICRO 5VDC (มากกว่า 500 mA.)
 - ขั้ว DC JACK 2.0 mm. ลบนอก, ในบวก 7-12 VDC (มากกว่า 500 mA.)

4.3 โมดูลตรวจจับวัตถุระยะทาง (US-025 ultrasonic sensor module)[8]

เป็นเซนเซอร์ตรวจจับวัตถุระยะทางแบบ Ultrasonic รุ่น US-025 สามารถตรวจวัดระยะทางจากวัตถุโดยอาศัยเสียงสะท้อนกลับแบบ Ultrasonic ตรวจจับได้ระยะ 2-600cm มุมตรวจจับ 15 องศา จ่ายไฟเลี้ยงที่ 3.3-5V ใช้กระแสเพียง 5.3mA

สำหรับการต่อขา (Pins)

- VCC ---> 3.3V or 5V
- Trig ---> Digital Output (0/1)
- Echo ---> Digital input (0/1)
- GND ---> Ground



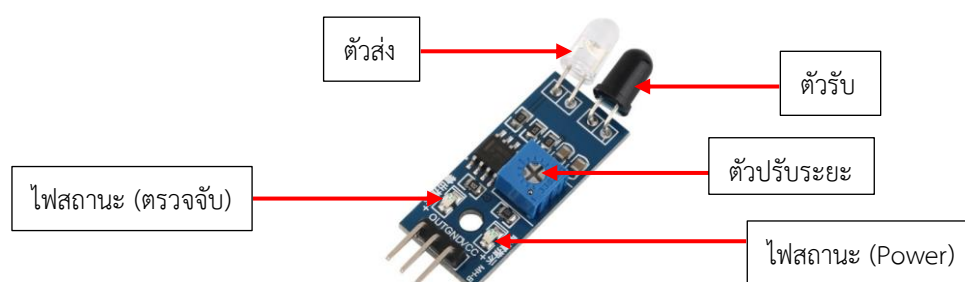
รูปที่ 2 โมดูลตรวจจับวัตถุระยะทาง US-025

Electrical parameters	US-025 ultrasonic sensor module
Operating Voltage	DC 3V-5.5V
Working current	5.3mA
Operating temperature	-40 °C-85°C
Output method	GPIO
Induction angle	Less than 15 degrees
Detection distance	2cm-600cm
Detection accuracy	0.1cm+1%

ตาราง 1 คุณสมบัติของ US-025

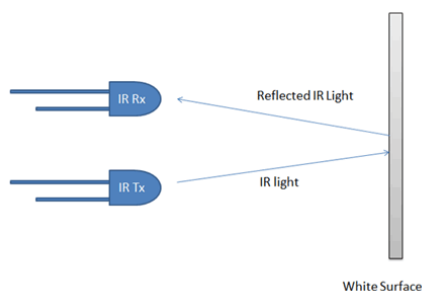
4.4 โมดูลเซ็นเซอร์แสงสำหรับตรวจจับวัตถุทึดขวาง (IR Infrared Obstacle Avoidance Sensor)

โมดูลเซ็นเซอร์แสงสำหรับตรวจจับวัตถุทึดขวาง จะมีตัวรับและตัวส่ง infrared ในตัว ตัวสัญญาณ(สีขาว) infrared จะส่งสัญญาณออกมา และเมื่อมีวัตถุมาบังคลื่นสัญญาณ infrared ที่ถูกส่งออกมาจะสะท้อนกลับเข้าตัวรับสัญญาณ (สีดำ) สามารถนำมาใช้ตรวจจับวัตถุที่อยู่ตรงหน้าได้ และสามารถปรับความไวระยะการตรวจจับใกล้หรือไกลได้ ภายตัวเซ็นเซอร์แบบนี้จะมีตัวส่ง Emitter และ ตัวรับ Receiver ติดตั้งภายในตัวเดียวกัน ทำให้ไม่จำเป็นต้องเดินสายไฟทั้งสองฝั่ง เหมือนแบบ Opposed Mode ทำให้การติดตั้งใช้งานได้ง่ายกว่า แต่อย่างไรก็ตามจำเป็นต้องติดตั้งตัวแผ่นสะท้อนหรือ Reflector ไว้ตรงข้ามกับตัวเซ็นเซอร์เอง โดยโฟโต้เซ็นเซอร์แบบที่ใช้แผ่นสะท้อนแบบนี้จะเหมาะสำหรับชิ้นงานที่มีลักษณะทึบแสงไม่เป็นมันวาว เนื่องจากอาจทำให้ตัวเซ็นเซอร์เข้าใจผิดว่าเป็นตัวแผ่นสะท้อน และ ทำให้ทำงานผิดพลาดได้



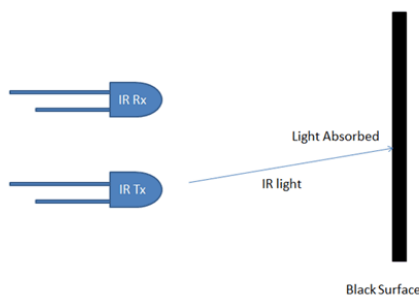
รูปที่ 3 IR Infrared Obstacle Avoidance Sensor Module

เซ็นเซอร์แบบนี้จะมีช่วงในการทำงานหรือระยะในการตรวจจับจะได้ไกลกว่าแบบ Opposed mode ซึ่งในสภาวะการทำงานปกติตัวรับ Receiver จะสามารถรับสัญญาณแสงจากตัวส่ง Emitter ได้ตลอดเวลา เนื่องจากลำแสงจะสะท้อนกับแผ่นสะท้อน Reflector อยู่ตลอดเวลาจะแสดงค่าเป็น 0



รูปที่ 4 หลักการทำงานของเซนเซอร์เมื่อไม่มีวัตถุขวาง

หน้าที่หลักของเซนเซอร์ชนิดนี้ จะคอยตรวจจับวัตถุที่เคลื่อนที่ตัดผ่านหน้าเซนเซอร์ เมื่อวัตถุ หรือ ชิ้นงานผ่านเข้ามาที่หน้าเซนเซอร์ แล้วจะการขวางลำแสงที่ส่งจากตัวส่ง Emitter ที่ส่งไปยังแผ่นสะท้อน จึงทำให้ตัวรับ Receiver ไม่สามารถรับลำแสงที่จะสะท้อนกลับไปได้ จะแสดงค่า เป็น 1 ซึ่งจะทำให้วงจรภายในรับรู้ได้ว่า มีวัตถุหรือชิ้นงานขวางอยู่ ทำให้สถานะของเอาต์พุตของตัวรับเปลี่ยนแปลงไป โดยเราเรียกลักษณะการทำงานแบบนี้ว่า Dark On หรือ Dark Operate



รูปที่ 4 หลักการทำงานของเซนเซอร์เมื่อมีวัตถุขวาง

ไฟเลี้ยง VCC	3.3V-5V
ดิจิตอลเอาต์พุต	0 หรือ 1
ระยะตรวจจับ	2-30 cm
มุมในการตรวจจับ	35 องศา
ขนาดบอร์ด	3.1 x 1.5 cm

ตาราง 2 คุณสมบัติของ IR Infrared Obstacle Avoidance Sensor Module

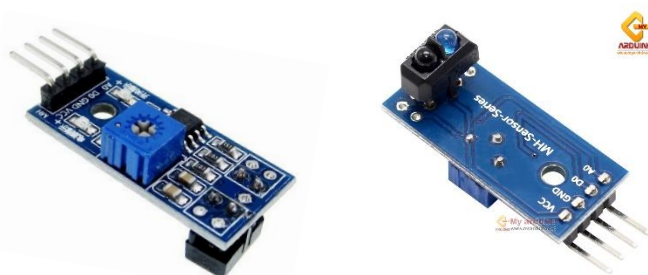
4.5 เซนเซอร์เช็คสิ่งกีดขวาง เส้นขาวดำ นับจำนวน [11]

โมดูลอ่านค่าสะท้อนกลับของแสง ใช้ไฟ 3.3-5V เหมาะสำหรับใช้กับ Arduino ให้เอาต์พุตออกมา 2 แบบคือแบบดิจิตอลสามารถปรับค่าที่ต้องการได้ เมื่อค่าที่อ่านได้ถึงระดับที่ต้องการก็จะส่งค่า 1 ออกมา ถ้ายังไม่ถึงระดับก็จะส่งค่า 0 ออกมา และอีกแบบคือเอาต์พุตแบบอะนาล็อกอ่านค่าได้เป็นตัวเลข 0-1023 หรือ สัญญาณไฟในช่วง 0-5V

สำหรับการต่อขา (Pins)

- VCC ---> 3.3V or 5V
- GND ---> Ground

- D0 ---> Digital Output (0/1)
- A0 ---> Analog Output

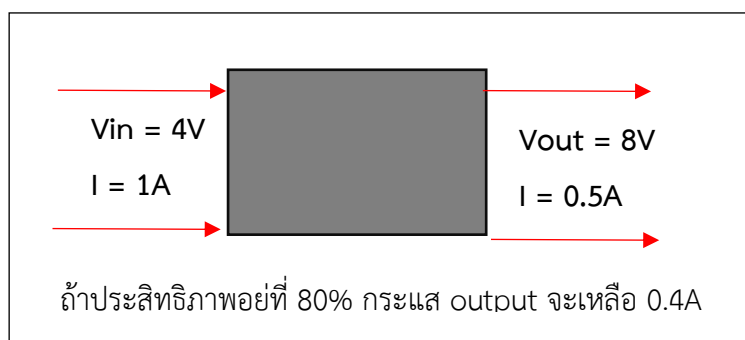


รูปที่ 5 TCRT5000 Infrared IR sensor detection

เซนเซอร์ TCRT5000 เป็นเซนเซอร์ที่ใช้ตรวจจับวัตถุโดยใช้แสงอินฟราเรด โดยจะมี led แบบอินฟราเรดยิงแสงอินฟราเรดออกไป และมีตัวรับแสงอินฟราเรดรับค่าแสงที่สะท้อนกลับมา เมื่อวัตถุอยู่ใกล้จะมีแสงสะท้อนกลับมามากกว่าวัตถุที่อยู่ไกลจึงสามารถนำมาใช้วัดวัตถุผ่าน หรือใช้ตรวจจับเส้นสีขาว/ดำได้ โดยเส้นขาวจะให้แสงสะท้อนกลับมากกว่าสีดำ สามารถนำไปประยุกต์กับงานได้หลายแบบ เช่น ใช้เป็นตัวตรวจจับเส้นสีขาวกับสีดำสำหรับรถ smart car ,ใช้เป็นสวิทช์แบบไร้สัมผัส หรือใช้เป็นเซนเซอร์หลีกเลี่ยงการชน

4.6 วงจรแรงดันปรับค่าได้

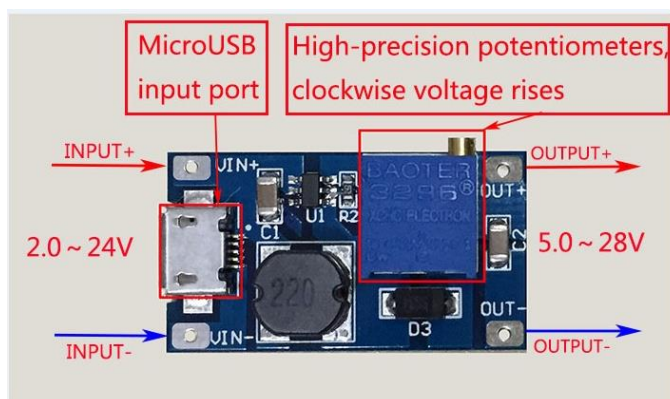
วงจรเพิ่มแรงดันแบบปรับค่าได้ทุกตัวกระแสที่ระบุเป็นกระแส Input ที่รับได้สูงสุด หลักการทำงานของวงจรคือ ใช้กระแส Input ไปแลกเป็นแรงดัน Output ดังนั้นยังเพิ่มแรงดันให้สูงขึ้นมากเท่าไร กระแสก็จะลดลงผกผันกัน[12]



รูปที่ 6 ตัวอย่างความสัมพันธ์ระหว่างแรงดันกับกระแสไฟฟ้า

วิธีคิดกระแส Output โดยประมาณคิดได้จาก

แรงดัน input คูณ กระแส input สูงสุด และหาร ด้วยแรงดัน output จากนั้นคูณด้วย 0.8 เช่น วงจรระบุ 6A แรงดันไฟเข้าที่ต่อใช้งาน 12V คูณกันได้ 72(W) หากปรับแรงดันขาออกไว้ที่ 24V กระแสสูงสุดที่จะใช้ได้สามารถหาได้จาก $(72W / 24V) \times 0.8 = 2.4A$



รูปที่ 7 DC/DC Step-up Converter รุ่น MT3608

กระแสไฟฟ้าขาออกมากที่สุด (I max)	2A
แรงดันไฟฟ้าขาเข้า (Vin)	2V ~ 24V
แรงดันไฟฟ้าขาออกสูงสุด (Vout max)	5V~ 28V
ประสิทธิภาพ (%Effective)	> 93%
ขนาด	17mm*30mm * 14mm

ตาราง 3 รายละเอียดของ DC/DC Step-up Converter รุ่น MT3608 [13]

4.7 บอร์ดขับเคลื่อนมอเตอร์ 2 ช่องขนาดเล็ก [14]

มอเตอร์กระแสตรงเป็นอุปกรณ์จักรกลไฟฟ้าที่แปลงไฟฟ้ากระแสตรงให้เป็นพลังงานกลโดยการหมุนของเพลลา เป็นการทำงานบนหลักการของแรงลอเรนซ์ (Lorentz force) โดยที่ตัวนำกระแสไฟฟ้าในสนามแม่เหล็กสัมผัสกับแรงจึงทำให้ตัวนำเคลื่อนที่ไปในทิศทางของแรงที่เรียกว่าแรงลอเรนซ์(Lorentz force) โดยทั่วไปแล้วมอเตอร์ DC ประกอบด้วยแม่เหล็กไฟฟ้าหรือแม่เหล็กถาวรและขดลวดที่ได้รับการเคลือบด้วยน้ำยาฉนวน เมื่อจ่ายไฟฟ้ากระแสตรงเข้าไปยังขดลวดจะเกิดการเหนี่ยวนำของสนามแม่เหล็กไฟฟ้า เนื่องจากการเหนี่ยวนำแม่เหล็กไฟฟ้าทำให้ armature เคลื่อนที่ไปตามทิศทางของแรง มอเตอร์กระแสตรงมีการใช้กันอย่างแพร่หลายในระบบอัตโนมัติทางอุตสาหกรรมของเล่นและหุ่นยนต์ ความเร็วของมอเตอร์ DC สามารถควบคุมได้ทั้งโดยการควบคุมกระแสไฟฟ้าไปยังกระดองหรือการใช้แหล่งจ่ายไฟแบบแปรผัน

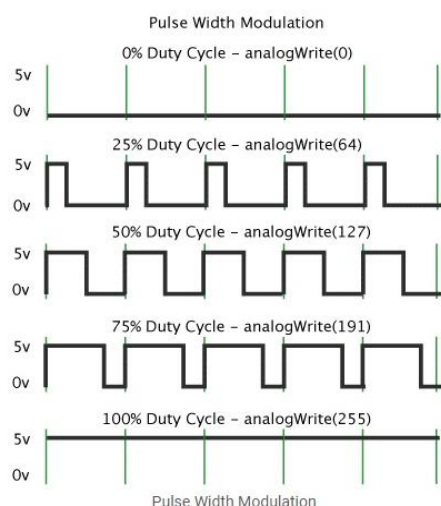
H-Bridge Driver Mini L298N 2-Way หลักการทำงานวงจรจะขับเคลื่อนกระแสเข้ามอเตอร์ตามข้อที่กำหนดด้วยลอจิกเพื่อควบคุมทิศทาง ส่วนความเร็วของมอเตอร์นั้นจะถูกควบคุมด้วยสัญญาณ PWM (Pulse Width Modulation) เป็นวิธีการควบคุมการจ่ายกำลังโดยการปรับความกว้างของสัญญาณ Pulse ด้วย

ความถี่สูงเพื่อให้ได้กำลังเฉลี่ยเป็นไปตามส่วนที่ต้องการ ซึ่งต้องมีการปรับความถี่ให้เหมาะสมกับเป็น พารามิเตอร์ที่ใช้กำหนดสัดส่วนการทำงาน (ON) ของ Load (มอเตอร์)

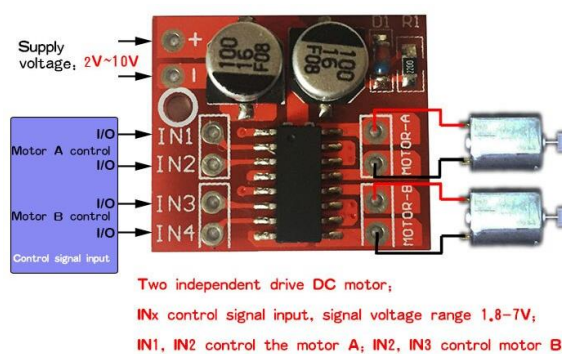
การใช้ Pulse Width Modulation (PWM) ควบคุมความเร็วมอเตอร์

โดยปกติจะทำปรับเพิ่มและลดแรงดันที่ส่งออกไปยังมอเตอร์ แต่การปรับเพิ่มหรือลดแรงดันนั้นเป็น แนวทางที่ต้องใช้วงจรที่ซับซ้อนมีความยุ่งยากค่อนข้างมาก ดังนั้นโดยทั่วไปจึงนิยมใช้เทคนิคที่เรียกว่า Pulse Width Modulation (PWM) ซึ่งไม่ได้ปรับเพิ่มหรือลดแรงดันโดยตรง หากแต่ใช้หลักการเปิด/ปิดมอเตอร์ด้วยความเร็วสูงๆ จนผลค่าเฉลี่ยของแรงดันที่ได้ออกมาเทียบเท่ากับการเปลี่ยนแรงดันโดยตรง เทคนิคนี้ทำให้ไม่ต้องใช้วงจรซับซ้อนแต่การเขียนโปรแกรมจะยุ่งยากขึ้นบ้าง

PWM คือการปรับเปลี่ยนความกว้างของลูกคลื่นในแต่ละคาบ โดยถ้าลูกคลื่นสั้นก็จะทำให้แรงดันเฉลี่ย ที่ออกมาค่าน้อย และถ้าลูกคลื่นยาวแรงดันเฉลี่ยก็จะมีค่ามากขึ้น จากรูปด้านล่าง V เฉลี่ยจะสูงหรือต่ำนั้น ขึ้นอยู่กับความกว้างของลูกคลื่น ซึ่งความกว้างของลูกคลื่นนี้เรียกว่า pulse width หรือ Duty Cycle Pulse width จะต้องน้อยกว่าค่าความยาวคาบเสมอ Duty Cycle จะมีหน่วยเป็น % ของความยาวคาบ เช่น ถ้าคาบ = 10ms และ Duty Cycle = 40% นั้นหมายความว่า Pulse width = $10\text{ms} \times 0.4 = 4\text{ms}$ เป็นต้น



รูปที่ 8 คาบของสัญญาณแสดงความกว้างของ Duty Cycle



รูปที่ 9 H-Bridge Driver Mini L298N 2-Way

Supply Voltage	2-10V
Signal Input voltage	1.8-7V
Max Input current	1.5A*2
Control signal	PWM

ตาราง 4 คุณสมบัติของ H-Bridge Driver Mini L298N 2-Way

in1	in2	Motor Operation
0	1	Forward
1	0	Reward
0	0	Stop
1	1	Break

ตาราง 5 การป้อนคำสั่งให้มอเตอร์ทำงานและผลลัพธ์

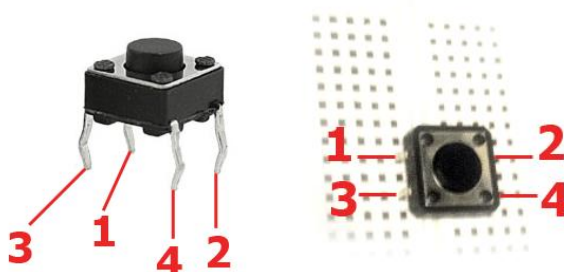
4.8 ไมโครสวิตช์กดติดปล่อยดับแบบ 4 ขา [15]

Push button คือ สวิตช์วงจรอิเล็กทรอนิกส์ชนิดหนึ่ง มีหน้าที่ควบคุมการเปิด และ ปิด ของวงจรส่วนนั้นๆ โดยทั่วไปอาจมี 2 ขา หรือ 4 ขา โดยปุ่มกดติดปล่อยดับนั้น เมื่อทำการกดจะเป็นการปิดวงจรทำให้กระแสไฟฟ้าสามารถไหลผ่านวงจรได้ เมื่อไม่ได้กดจะทำให้วงจรเปิดกระแสไฟฟ้าจะไม่สามารถไหลผ่านวงจรได้

4 Pins Push Button (ปุ่มกดติดปล่อยดับ 4 ขา) ปุ่มกดติดปล่อยดับ 4 ขา เป็นที่นิยมกันอย่างมาก การต่อวงจรอาจดูเหมือนยุ่งยาก แต่อันที่จริงแล้วสามารถต่อได้ง่ายดายมากถ้าเข้าใจหลักการทำงานของมัน

หลักการทำงานของ 4 Pins Push Button

การทำงานของมันก็แยกเป็น 2 ส่วน คือ ตอนที่ยังไม่กด (Not Pressed) และ ตอนที่กด (Pressed) ซึ่งเมื่อเราลองจัดวางปุ่ม Button นี้ ดังรูป โดยให้ด้านหน้าและหลังของปุ่ม ไม่มีขา ส่วนด้านซ้ายเป็นขา 1 กับ 3 และด้านขวาเป็นขา 2 กับ 4



รูปที่ 10 แสดงตำแหน่งขาของ 4 Pins Push Button

เมื่อวงจรเปิด (Not Pressed) : ขาที่ 1 จะเชื่อมอยู่กับขาที่ 2 / ขาที่ 3 จะเชื่อมอยู่กับขาที่ 4






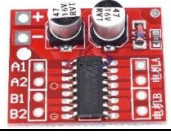


เมื่อวงจรปิด (Pressed) : ขาที่ 1 จะเชื่อมอยู่กับขาที่ 3 / ขาที่ 2 จะเชื่อมอยู่กับขาที่ 4





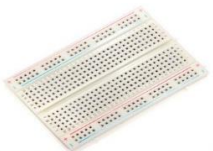



พูดง่ายๆคือ เมื่อเรายังไม่ได้กดปุ่ม ขาที่1 จะเชื่อมอยู่กับขาที่2 และ ขาที่3 จะเชื่อมอยู่กับขาที่4 แต่เมื่อเรากดปุ่มแล้ว จะเกิดการสลับคู่ของขา เพื่อเปลี่ยนแปลงเส้นทางการจ่ายกระแสไฟฟ้าในวงจรนั่นเอง เราสามารถใช้หลักการนี้มาควบคุมการเปิด และ ปิดของวงจรอิเล็กทรอนิกส์ได้ง่ายๆ

บทที่ 5

5.วิธีการดำเนินงาน

5.1 ส่วนประกอบหุ่นยนต์

อุปกรณ์	จำนวน (Unit)
1 Arduino ESP32 Wemos D1 	1
2 US-025 ultrasonic sensor module 	1
3 IR Infrared Obstacle Avoidance Sensor Module 	2
4 TCRT5000 Infrared IR sensor detection 	1
5 DC/DC Step-up Converter รุ่น MT3608 	1
6 H-Bridge Driver Mini L298N 2-Way 	1
7 DC N20 Mini Micro Gear Motor 	2
8 ตัวต้านทาน 220 โอห์ม 5% 	1

<p>9 ไมโครสวิตช์กดติดปลายดัดแบบ 4 ขา</p> 	1
<p>10 Male to Male Jumper Wires Cables</p> 	40
<p>11 Battery Li-ion 18650 3.7 V</p> 	1
<p>12 ล้อยางเหมาะสำหรับ N20</p> 	2
<p>13 Photo board 8.5CM X 5.5CM</p> 	1
<p>14 ล้อรถเข็น Smart Car 15 mm</p> 	1
<p>15 โครงรถ</p> 	1
<p>16 น็อต M3 (ตัวผู้และตัวเมีย)</p> 	12

ตาราง 6 อุปกรณ์สำหรับการสร้างรถหุ่นยนต์

5.2 ส่วนประกอบแบบจำลองหุ่นยนต์

5.2.1 อุปกรณ์อิเล็กทรอนิกส์

1. Arduino ESP32 Wemos D1 ติดกับฐานรอง Arduino ESP เป็นแนวตั้งฉากกับฐานรถยนต์
2. IR Infrared Obstacle Avoidance Sensor Module ติดตั้งฝั่งซ้ายและขวาของ Photo Board
3. US-025 Ultrasonic Sensor Module ติดตั้งบน Photo Board ระยะตรงกลางระหว่าง IR Infrared Obstacle Avoidance Sensor Module ทั้ง 2 ชิ้น
4. TCRT5000 Infrared Reflective Sensor ติดตั้งด้านหน้าของใต้ฐานรถยนต์
5. DC/DC Step-up Converter รุ่น MT3608 ติดตั้งด้านข้างซ้ายและขวาของใต้ฐานรถยนต์เพื่อเชื่อมต่อกับล้อรถยนต์ทั้งสองข้าง
6. H-Bridge Driver Mini Micro Gear Motor ติดตั้งติดกับฐานรถยนต์
7. Micro Switch ติดตั้งติดกับ Arduino ESP32 Wemos D1
8. Battery Li-ion 18650 3.7V ติดตั้งติดกับฐานรอง Battery
9. Photo Board 8.5 cm x 5.5 cm ติดตั้งกับฐานรอง Photo Board และตัวบอร์ดติดตั้ง IR Infrared Obstacle Avoidance Sensor Module และ US-025 Ultrasonic Sensor Module

5.2.2 อุปกรณ์เสริมและขับเคลื่อนของแบบจำลองหุ่นยนต์

1. ล้อที่เหมาะสมสำหรับ N20 ล้อขนาด 4.2 cm จำนวน 2 ล้อติดกับ DC/DC Step-up Converter รุ่น MT3608
2. ล้อรถเข็น Smart Car ขนาด 15 mm ติดตั้งคร่อม TCRT5000 Infrared Reflective Sensor และใช้แกนล้อรถเข็น Smart Car 4 ชิ้น ติดล็อกด้วยน็อตทั้งหมด 8 ตัว ให้เชื่อมต่อกับฐานรถยนต์
3. Battery Case ติดตั้งพร้อม Battery ที่ฐานรอง Battery

5.2.3 อุปกรณ์จาก 3D Printer

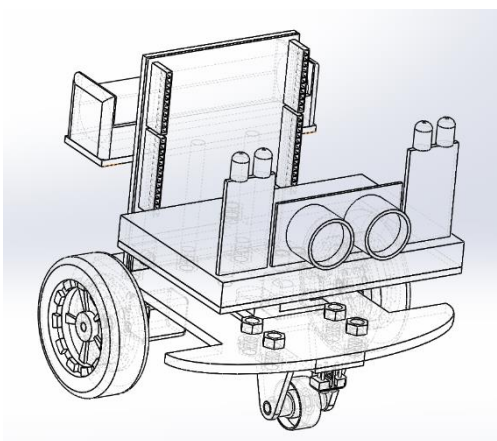
1. ฐานรถยนต์ ขนาด 10x10 cm สูง 2 cm เจาะรูตรงกลางสำหรับสายไฟจาก DC/DC Step-up Converter รุ่น MT3608 และเจาะรู 3 รูเพื่อเชื่อมต่อแกนฐานรอง Battery เจาะรู 1 รูเพื่อเชื่อมต่อกับแกนฐานรอง Photo Board เจาะรู 4 รูเพื่อเชื่อมต่อกับแกนล้อรถเข็น Smart Car
2. ฐานรอง Photo Board ขนาด 5.5x8.5 cm สูง 2 cm เจาะรู 1 รูตรงท้ายฐาน ใช้แกนฐานรอง Photo Board 1 ชิ้นติดล็อกด้วยน็อต 1 ตัว ให้เชื่อมต่อกับฐานรถยนต์
3. ฐานรอง Arduino ESP ขนาด 5.3x6.8 cm สูง 2 cm ติดแนวตั้งฉากกับฐานรถยนต์ ด้านหน้าติดเข้ากับ Arduino ESP32 Wemos D1 ด้านหลังติดกับฐานรอง Battery และ Battery Case ในแนวตั้งฉากเช่นกัน
4. ฐานรอง Battery ขนาด 2x6.5 cm สูง 2 cm เจาะรู 3 รูเพื่อเชื่อมต่อกับแกนฐานรอง Battery 3 ชิ้นและล็อกด้วยน็อต 3 ตัวเพื่อเชื่อมต่อเข้ากับฐานรถยนต์
5. น็อต ขนาดเส้นผ่าศูนย์กลาง 0.4 cm สูง 0.3 cm ใช้ล็อกกับแกนประเภทต่างๆให้เชื่อมต่อกับฐานรองแต่ละประเภท
6. แกน

- 6.1 แกนฐานรอง Battery ขนาดเส้นผ่าศูนย์กลาง 0.4 cm สูง 4 cm ใช้เชื่อมต่อฐานรอง Battery กับฐานรถยนต์

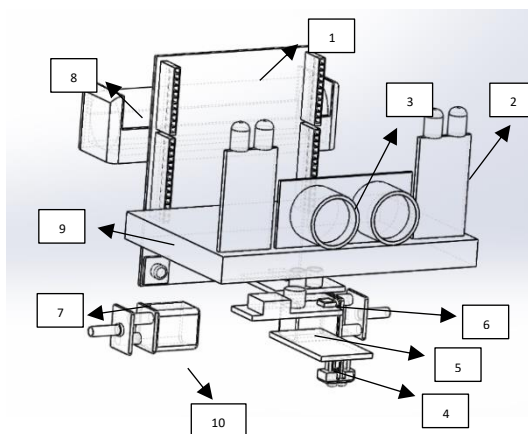
6.2 แกนฐานรอง Photo Board ขนาดเส้นผ่าศูนย์กลาง 0.4 cm สูง 2.5 cm ใช้เชื่อมต่อฐานรอง Photo Board กับฐานรถยนต์

6.3 แกนล้อรถเซ็น Smart Car ขนาดเส้นผ่าศูนย์กลาง 0.3 cm สูง 2.1 cm ใช้เชื่อมต่อฐานรอง Smart Car กับฐานรถยนต์

5.2.4 แบบจำลองหุ่นยนต์



รูปที่ 11 แบบจำลองหุ่นยนต์

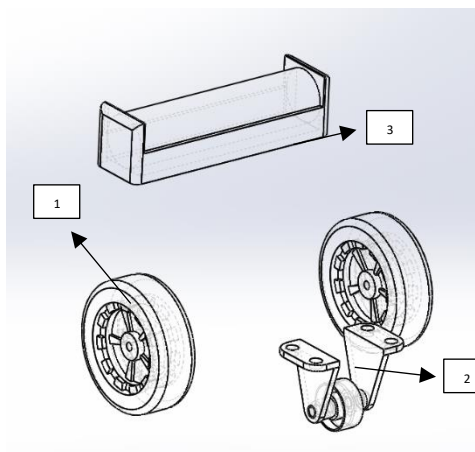


รูปที่ 12 แสดงหมายเลขอุปกรณ์อิเล็กทรอนิกส์ของแบบจำลอง

อุปกรณ์อิเล็กทรอนิกส์	จำนวน	หมายเลข
1. Arduino ESP32 Wemos D1	1	1
2. IR Infrared Obstacle Avoidance Sensor Module	2	2
3. US-025 Ultrasonic Sensor Module	1	3
4. TCRT5000 Infrared Reflective Sensor	1	4
5. DC/DC Step-up Converter รุ่น MT3608	1	5
6. H-Bridge Driver Mini L298N 2-Way	1	6

7. Micro Switch	1	7
8. Battery Li-ion 18650 3.7V	1	8
9. Photo Board 8.5 cm x 5.5 cm	1	9
10. DC N20 Mini Micro Gear Motor	2	10

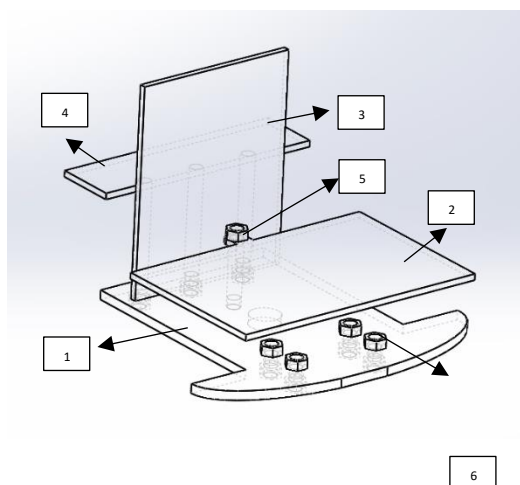
ตาราง 7 แสดงหมายเลขอุปกรณ์อิเล็กทรอนิกส์ของแบบจำลองหุ่นยนต์



รูปที่ 13 แสดงหมายเลขอุปกรณ์เสริมและขับเคลื่อนของแบบจำลองหุ่นยนต์

อุปกรณ์	จำนวน	หมายเลข
1. ล้อยางเหมาะสำหรับ N20	2	1
2. ล้อรถเข็น Smart Car 15 mm	1	2
3. Battery Case	1	3

ตาราง 8 แสดงหมายเลขอุปกรณ์เสริมและขับเคลื่อนของ

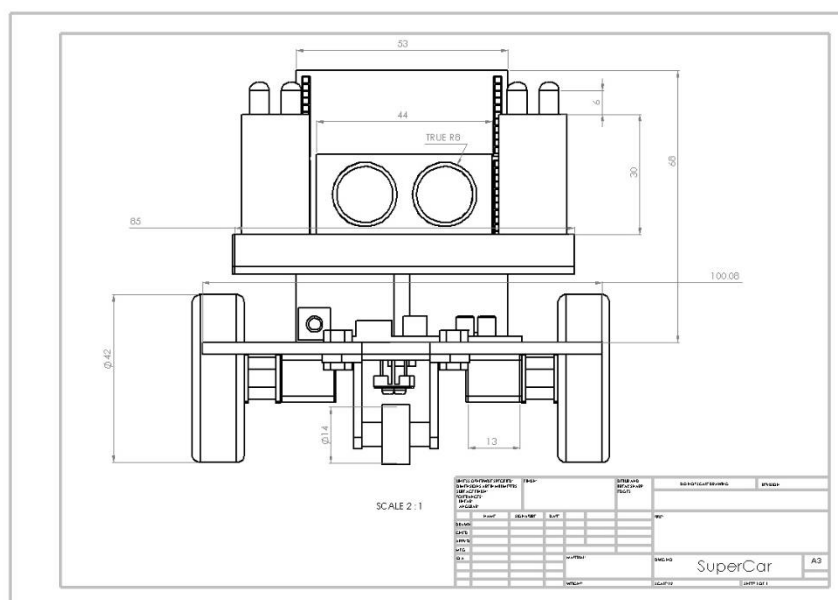


รูปที่ 14 แสดงหมายเลขอุปกรณ์จาก 3D Printer

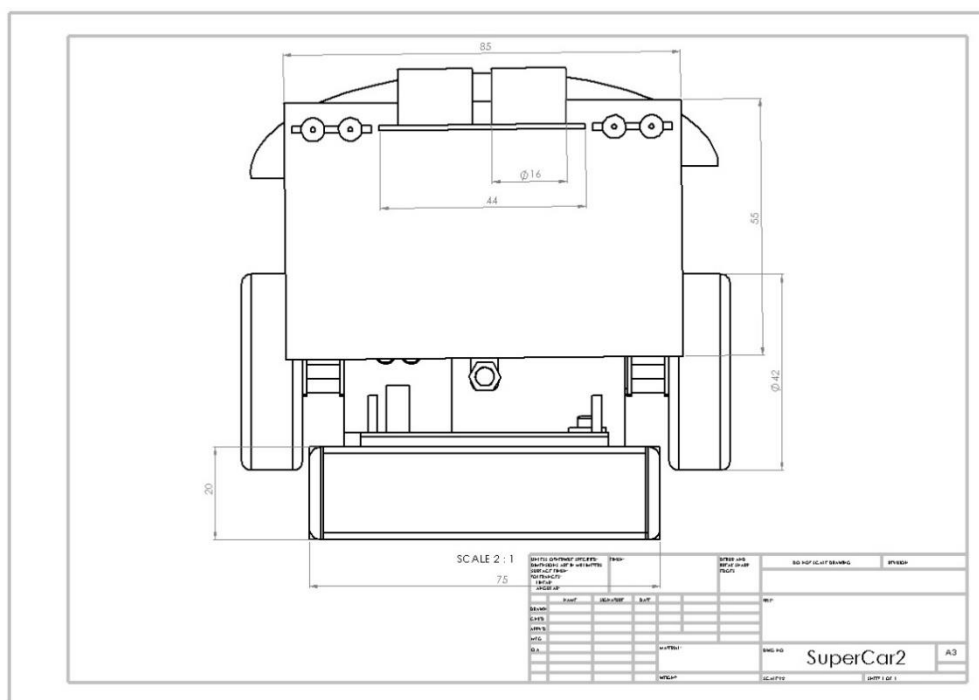
อุปกรณ์จาก 3D Printer	จำนวน	หมายเลข
1. ฐานรถยนต์	1	1

2. ฐานรอง Photo Board	1	2
3. ฐานรอง Arduino ESP	1	3
4. ฐานรอง Battery	1	4
5. น็อต M3	12	5
6. แกน		6
6.2 แกนฐานรอง Battery	3	
6.2 แกนฐานรอง Photo Board	1	
6.3 แกนล้อรถเข็น Smart Car	4	

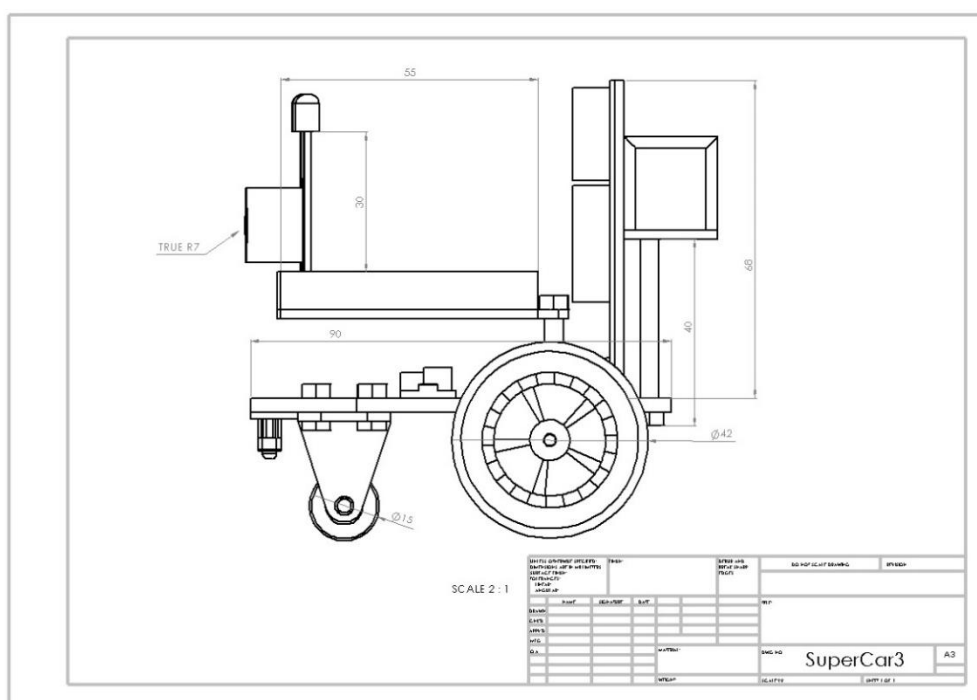
5.2.5 Drawing sheet แบบจำลองหุ่นยนต์



รูปที่ 15 รูปด้าน FRONT



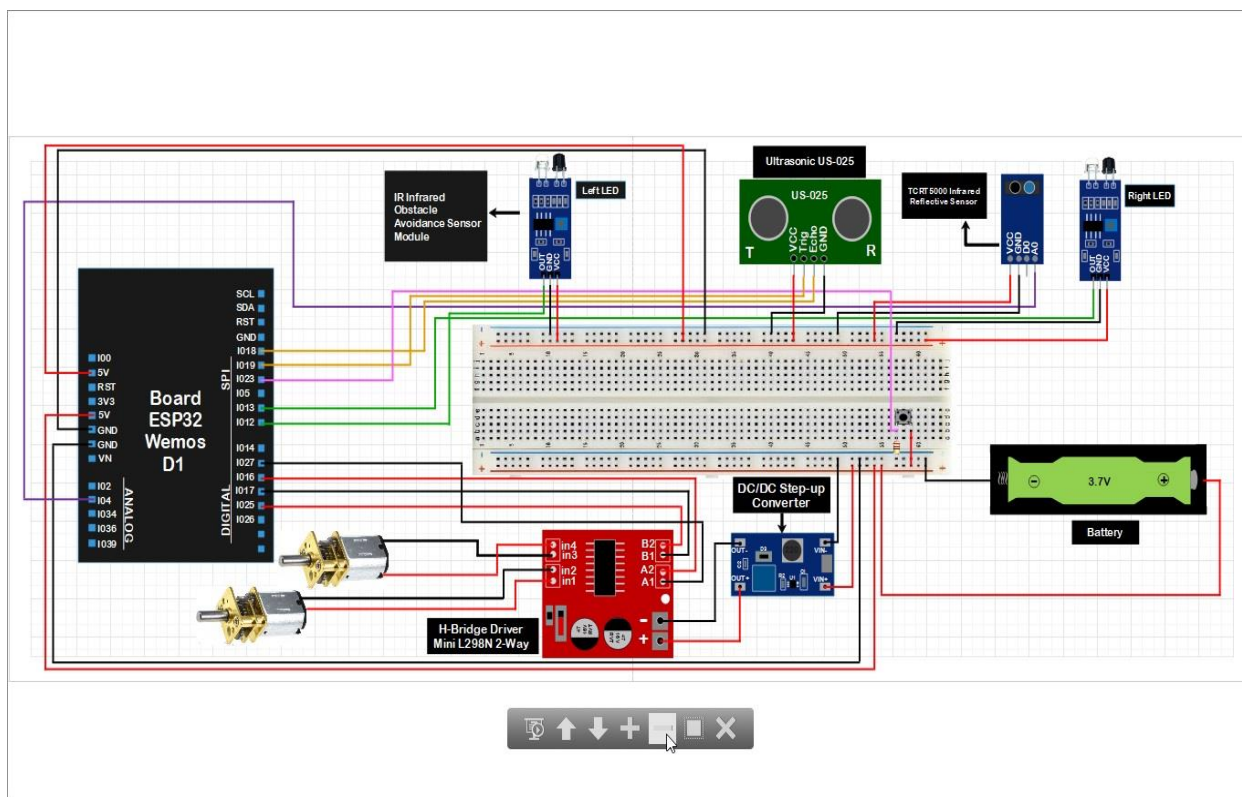
รูปที่ 16 รูปด้าน TOP



รูปที่ 17 รูปด้าน Side

5.3 ส่วนของการต่อวงจร

การต่อวงจรในที่นี้จะเป็นการแสดงการต่อวงจรของโมดูลเท่านั้น โดยใช้โปรแกรม E-DrawMax ในการวาดวงจร



รูปที่ 11 การต่อวงจรภายในหุ่นยนต์

5.4 ส่วนของการเขียนโปรแกรม

ส่วนของการเขียนโปรแกรมนี้นี้เป็นการเขียนโปรแกรม **Arduino C++** ในการออกคำสั่งควบคุมรถหุ่นยนต์ให้เป็นไปตามกลยุทธ์ของผู้เขียนโปรแกรม ทั้งในกลยุทธ์เกมรุกและกลยุทธ์เกมรับ โดยใช้โปรแกรม **Arduino IDE** ในการเขียนโปรแกรม ดังนี้

```
#include <analogWrite.h> // library สำหรับคำสั่ง analogWrite
#include <HCSR04.h>       // library สำหรับ Ultrasonic sensor
#define maxSpd 255        // กำหนดค่าความเร็วมอเตอร์
#define ia1 25            // กำหนด Pin สำหรับมอเตอร์
#define ia2 27            // กำหนด Pin สำหรับมอเตอร์
#define ib1 16            // กำหนด Pin สำหรับมอเตอร์
#define ib2 17            // กำหนด Pin สำหรับมอเตอร์
HCSR04 hc(19,18);         // กำหนด Pin สำหรับ trig และ echo ของ Ultrasonic sensor
const int buttonPin = 23; // กำหนด Pin สำหรับ Pushbutton
```

```

int SensorL=13;          // กำหนด Pin สำหรับ Sensor ตัวซ้าย
int SensorR=12;          // กำหนด Pin สำหรับ Sensor ตัวขวา
int buttonState = 0;      // กำหนดค่าของตัวแปร buttonState
int TurningDely = 5000;  // กำหนดค่าของตัวแปร TurningDely

void setup()
{
  Serial.begin(9600);
  Serial.begin(38400);
  pinMode(SensorL,INPUT); // ประกาศให้ SensorL เป็น INPUT
  pinMode(SensorR,INPUT); // ประกาศให้ SensorR เป็น INPUT
  pinMode(ia1, OUTPUT);   // ประกาศให้ ia1 เป็น OUTPUT
  pinMode(ia2, OUTPUT);   // ประกาศให้ ia2 เป็น OUTPUT
  pinMode(ib1, OUTPUT);   // ประกาศให้ ib1 เป็น OUTPUT
  pinMode(ib2, OUTPUT);   // ประกาศให้ ib2 เป็น OUTPUT
  Serial.begin(115200);
}

void loop()
{
  int sensorValue = analogRead(4); // read the input on analog infrared pin 4
  buttonState = digitalRead(buttonPin); // read the state of the pushbutton value
  if (buttonState == HIGH) // ในคำสั่ง if นี้เป็น code แนวรุกเมื่อ buttonState มีค่าเท่ากับ HIGH

  {
    int maxspeed = maxSpd; // กำหนดให้ตัวแปร maxspeed มีค่าเท่ากับ maxSpd
    if(digitalRead(SensorL)==LOW) // ถ้า Sensor ด้านซ้ายตรวจจับวัตถุได้ให้หันรถวิ่งไป
    ทางขวาแล้วหยุดสักพัก
    {
      turnRight(maxspeed);
      delay(TurningDely);
      moveStop();
      delay(40);
    }
  }
}

```

```

    }
else if(digitalRead(SensorR)==LOW) // ถ้า Sensor ด้านขวาตรวจจับวัตถุได้ให้หันรถวิ่ง
ไปทางซ้ายแล้วหยุดสักพัก
{
    turnLeft(maxspeed);
    delay(TurningDely);
    moveStop();
    delay(40);
}
else if(hc.dist()<=10) // detectFront20cm ถ้า Ultrasonic sensor ด้านหน้าตรวจจับ
วัตถุได้ในระยะน้อยกว่าเท่ากับ 10 cm ให้รถหยุดสักพัก
{
    moveStop();
    delay(40);
}
else if((digitalRead(SensorR)==LOW)&&(digitalRead(SensorR)==LOW))
// ถ้า Sensor ด้านซ้ายและขวาตรวจจับวัตถุได้ให้หันรถถอยหลังเป็นเวลา 3 วินาที
{
    moveBackwardTime(3000); //backward 3 seconds
}
else if(sensorValue >= 2560 && sensorValue <= 2700) // red detect(put in red
number ถ้า Infrared Sensor ตรวจจับเส้นสีแดงได้ให้หันรถกลับ

{
    turnLeft(maxspeed); // U-Tern
    delay(TurningDely);
    moveStop();
    delay(40);
    turnLeft(maxspeed);
    delay(TurningDely);
    moveStop();
    delay(40);
}

```

```

else // ถ้าไม่มีเงื่อนไขใน if และ else if ด้านบนให้รถเดินหน้าไปเรื่อยๆ
{
    moveForward(maxspeed);
}
}
else // ในคำสั่ง else นี้เป็น code แนวรับเมื่อ buttonState มีค่าเท่ากับอย่างอื่นนอกเหนือจาก
HIGH
{
    int maxspeed = maxSpd; // กำหนดให้ตัวแปร maxspeed มีค่าเท่ากับ maxSpd
    if(digitalRead(SensorL)==LOW) // ถ้า sensor ด้านซ้ายตรวจพบวัตถุให้หันรถวิ่งไป
    ทางซ้าย
    {
        turnLeft(maxspeed);
        delay(TurningDely);
        moveStop();
        delay(40);
    }
    else if(digitalRead(SensorR)==LOW) // ถ้า sensor ด้านขวาตรวจพบวัตถุให้หันรถวิ่ง
    ไปทางขวา
    {
        turnRight(maxspeed);
        delay(TurningDely);
        moveStop();
        delay(40);
    }
    else if(sensorValue >= 3000 && sensorValue <= 3200) // ถ้า Infrared Sensor
    ตรวจจับเส้นสีดำได้ให้หันรถกลับ
    {
        turnLeft(maxspeed); // U-Turn
        delay(TurningDely);
        moveStop();
        delay(40);
        turnLeft(maxspeed);
    }
}

```

```

        delay(TurningDely);
        moveStop();
        delay(40);
    }
    else // ถ้าไม่มีเงื่อนไขใน if และ else if ด้านบนให้รถหยุดอยู่นิ่งๆ
    {
        moveStop();
    }
}

void moveStop() // กำหนดให้ motor a และ b ไม่ทำงานเมื่อมีคำสั่ง moveStop()
{
    digitalWrite(ia1, LOW);
    digitalWrite(ia2, LOW);
    digitalWrite(ib1, LOW);
    digitalWrite(ib2, LOW);
}

void moveForward(int speed) // กำหนดให้ motor ia1 และ ib1 ไม่ทำงานและ motor ia2 และ ib2
ทำงานเมื่อมีคำสั่ง moveForward()
{
    digitalWrite(ia1, LOW);
    digitalWrite(ia2, HIGH);
    digitalWrite(ib1, LOW);
    digitalWrite(ib2, HIGH);
}

void moveBackward(int speed) // กำหนดให้ motor ia1 และ ib1 ทำงานและ motor ia2 และ ib2 ไม่
ทำงานเมื่อมีคำสั่ง moveBackward()
{
    digitalWrite(ia1, HIGH);
    digitalWrite(ia2, LOW);
    digitalWrite(ib1, HIGH);
    digitalWrite(ib2, LOW);
}

```

void turnRight(int speed) // กำหนดให้ motor ia1 และ ia2 ทำงานและ motor ib1 ทำงาน และ ib2 ไม่ทำงานเมื่อมีคำสั่ง turnRight() ด้วยความเร็วเท่ากับค่า int speed ยกเว้น ib1 วิ่งด้วยความเร็ว 30 เปอร์เซ็นต์ของ maxSpd

```
{
    digitalWrite(ia1, HIGH);
    digitalWrite(ia2, HIGH);
    analogWrite(ib1, 0.3*maxSpd);
    //digitalWrite(ib1, LOW);
    digitalWrite(ib2, LOW);
}
```

void turnLeft(int speed) // กำหนดให้ motor ia1 ทำงานและ ia2 ไม่ทำงานและ motor ib2 และ ib1 ทำงานเมื่อมีคำสั่ง turnLeft() ด้วยความเร็วเท่ากับค่า int speed ยกเว้น ia1 วิ่งด้วยความเร็ว 30 เปอร์เซ็นต์ของ maxSpd

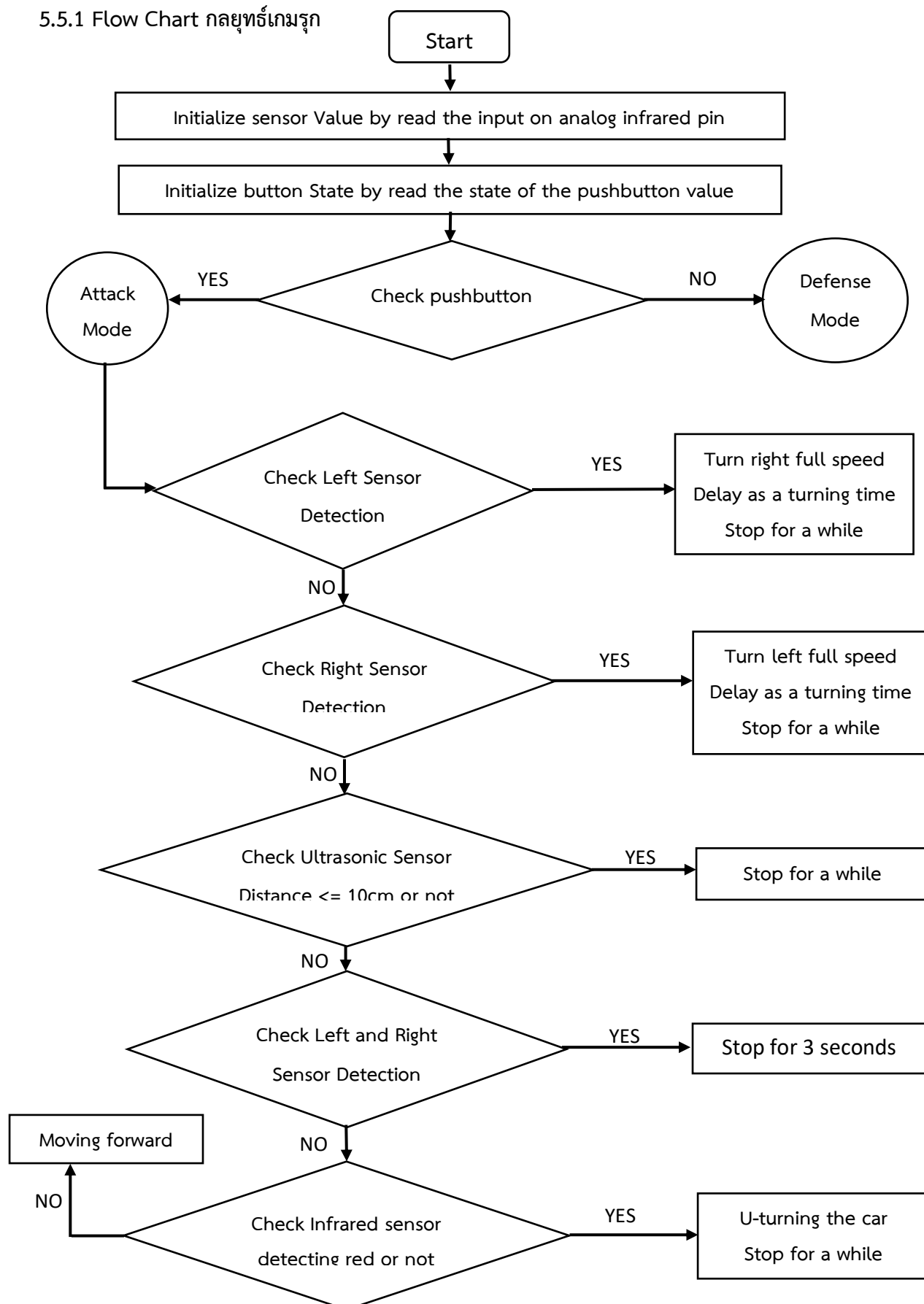
```
{
    analogWrite(ia1, 0.3*maxSpd);
    //digitalWrite(ia1, LOW);
    digitalWrite(ia2, LOW);
    digitalWrite(ib1, HIGH);
    digitalWrite(ib2, HIGH);
}
```

void moveBackwardTime(int time) // กำหนดให้ motor ia1 ไม่ทำงานและ ia2 ทำงานด้วยความเร็วเท่ากับ maxSpd และ motor ib2 ทำงานด้วยความเร็วเท่ากับ maxSpd และ ib1 ไม่ทำงานเมื่อมีคำสั่ง moveBackwardTime() ด้วยระยะทางเท่ากับค่า int time

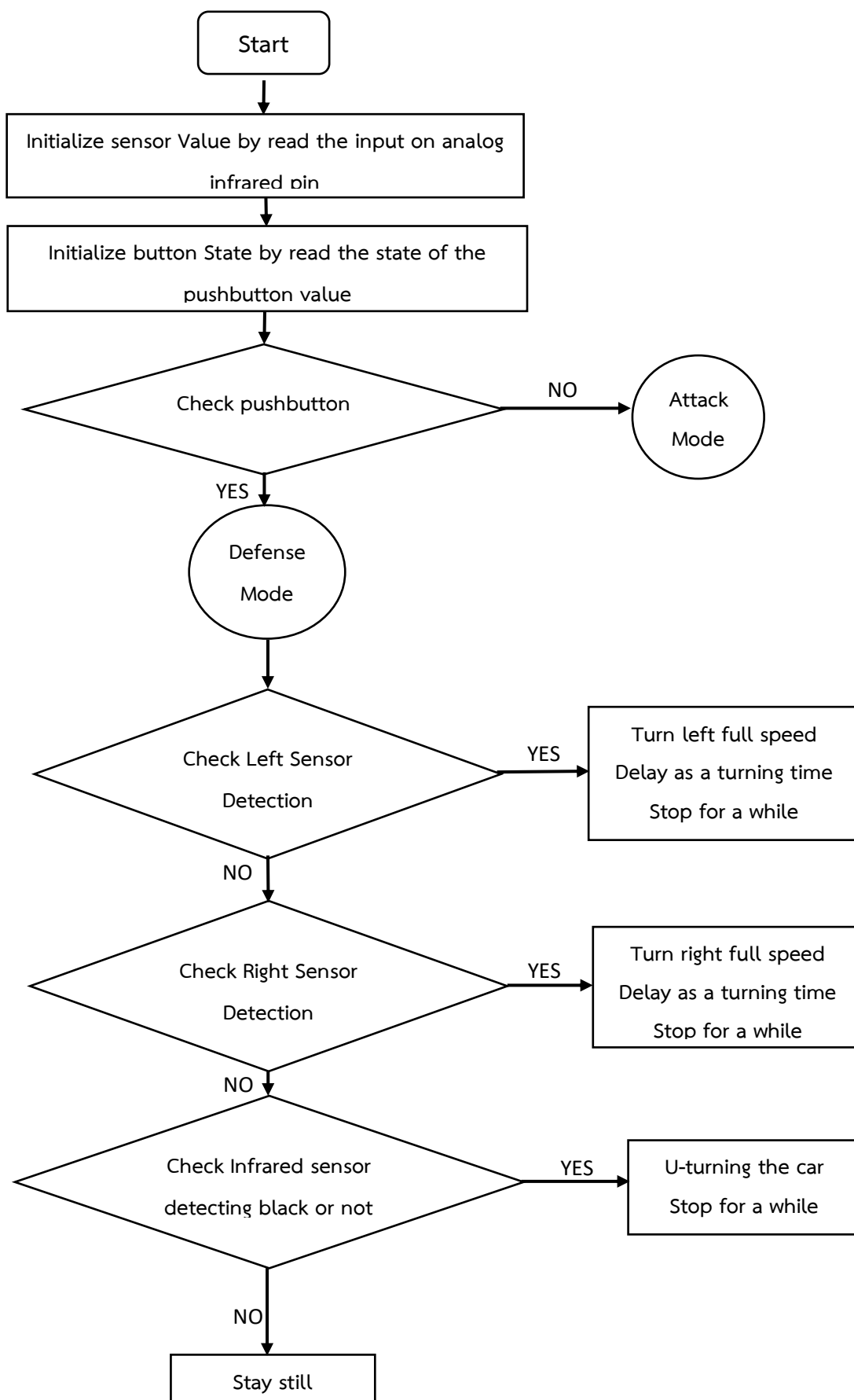
```
{
    digitalWrite(ia1, LOW);
    analogWrite(ia2, maxSpd);
    delay(time);
    digitalWrite(ib1, LOW);
    analogWrite(ib2, maxSpd);
    delay(time);
}
```


5.5 ส่วนของการทำงานของโปรแกรม (Flow Chart)

5.5.1 Flow Chart กลยุทธ์เกมรุก



5.5.2 Flow Chart กลยุทธ์เกมรับ



บทที่ 6

6. แผนการดำเนินงาน

หน้าที่ความรับผิดชอบของสมาชิกในทีม

1. นายปฐมจิตร รุ่งโรจน์วัฒนศิริ รหัสนักศึกษา 60010567
ตำแหน่ง เขียนโปรแกรม
2. นางสาวพรรณกาญจน์ กาญจนประดิษฐ์ รหัสนักศึกษา 60010662
ตำแหน่ง ออกแบบรถหุ่นยนต์
3. นางสาวอริญญา ทองนำ รหัสนักศึกษา 60011167
ตำแหน่ง ต่อบังคับ

รายการ	W1-2	W3-4	W5-6	W7-8	W9-10	W11-12	W13-14	W15
เรียนรู้เนื้อหา	↔							
<ul style="list-style-type: none"> • เรียนรู้การสร้างหุ่นยนต์ • 3D Printing • การขับเคลื่อน • Arduino • Program Technique • Sensor interface 		↔	↔	↔	↔	↔	↔	
ทดสอบ		↔	↔	↔	↔	↔	↔	↔
แข่งขัน								↔

หมายเหตุ : เนื่องจากสถานการณ์ Covid-19 ทำให้แผนดำเนินการเกิดความคลาดเคลื่อน

บทที่ 7

7.งบประมาณ

1. Arduino ESP32 Wemos D1	0 บาทต่อชิ้น
2. US-025 ultrasonic sensor module	0 บาทต่อชิ้น
3. IR Infrared Obstacle Avoidance Sensor Module	0 บาทต่อชิ้น
4. TCRT5000 Infrared IR sensor detection	0 บาทต่อชิ้น
5. DC/DC Step-up Converter รุ่น MT3608	0 บาทต่อชิ้น
6. H-Bridge Driver Mini L298N 2-Way	0 บาทต่อชิ้น
7. DC N20 Mini Micro Gear Motor	0 บาทต่อชิ้น
8. ตัวต้านทาน 220 โอห์ม 5%	0 บาทต่อชิ้น
9. ไมโครสวิตช์กดติดปล่อยดับแบบ 4 ขา	0 บาทต่อชิ้น
10. Male to Male Jumper Wires Cables	0 บาทต่อชิ้น
11. Battery Li-ion 18650 3.7 V	0 บาทต่อชิ้น
12. ล้อยางเหมาะสำหรับ N20	0 บาทต่อชิ้น
13. Photo board 8.5CM X 5.5CM	0 บาทต่อชิ้น
14. ล้อรถเข็น Smart Car 15 mm	50 บาทต่อชิ้น
15. โครงรถ	160 บาทต่อชิ้น
16. น็อต M3 ตัวผู้และตัวเมีย	192 บาทต่อชิ้น
รวมค่าใช้จ่ายทั้งหมด	402 บาท

หมายเหตุ : 0 บาทต่อชิ้น คืออุปกรณ์ที่ได้รับจากอาจารย์ กรณีนี้สมาชิกในกลุ่มจึงไม่เสียค่าใช้จ่าย

บทที่ 8

8.สรุปผล

การสร้างรถหุ่นยนต์นี้ทำให้นักศึกษาได้พัฒนาทักษะความรู้ด้านเทคโนโลยีหุ่นยนต์ ได้เรียนรู้ถึงกระบวนการในการได้มาซึ่งรถหุ่นยนต์ เช่น การวางแผนการทำงาน การระดมความคิดในการออกแบบหุ่นยนต์ การเขียนโปรแกรม Arduino C++ วิธีการต่อวงจรไฟฟ้า นำไปสู่การสร้างรถหุ่นยนต์ SUPERCAR ROBOT เพื่อใช้ในการแข่งขันบอลลูนด้านภายในห้องเรียน โดยผู้จัดทำได้บอณาสั่งให้รถหุ่นยนต์โดยใช้โปรแกรม Arduino IDE เขียนโปรแกรมคำสั่งให้รถหุ่นยนต์นี้ โดยมี 2 กลยุทธ์ คือกลยุทธ์เกมรุก และกลยุทธ์เกมรับ ในกลยุทธ์เกมรุกมีการใช้ IR Infrared Obstacle Avoidance Sensor, Ultrasonic Sensor และ TCRT5000 Infrared IR sensor detection แต่ในกลยุทธ์เกมรับมีการใช้ IR Infrared Obstacle Avoidance Sensor และ TCRT5000 Infrared IR sensor detection เท่านั้น การออกแบบรถหุ่นยนต์ใช้โปรแกรม Solid Work ในการออกแบบโครงสร้าง และในการเขียนแผนวงจรของโมดูลใช้โปรแกรม E-DrawMax ในการวาดวงจร ซึ่งเมื่อเสร็จสิ้นกระบวนการสร้างรถหุ่นยนต์เหล่านี้แล้วจะมีการทดสอบ และแข่งขันรถหุ่นยนต์ ซึ่งในกรณีนี้เราไม่สามารถระบุได้เนื่องจากประสบกับปัญหาสถานการณ์ Covid-19 ทำให้ขั้นตอนนี้ผู้จัดทำไม่สามารถระบุได้

เอกสารอ้างอิง

- [1] คู่มือเรื่อง ROBOT, <https://www.applicadthai.com/articles/article-creative-design/%E0%B8%84%>, สืบค้นเมื่อวันที่ 26 มีนาคม 2563.
- [2] เรื่องอุตสาหกรรมหุ่นยนต์ของประเทศไทย, สำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ (สวทช.), สืบค้นเมื่อวันที่ 26 มีนาคม 2563.
- [3] Arduino ภาษา C/C++, <https://arduinothing.blogspot.com/2016/04/arduino-cc.html>, สืบค้นเมื่อวันที่ 29 มีนาคม 2563.
- [4] ภาษาซีสำหรับไมโครคอนโทรลเลอร์, <https://sites.google.com/site/mikhorkhxnthorllexr1/phasa-si-sahrab-mikhor-khxnthorllexr-arduino>, สืบค้นเมื่อวันที่ 29 มีนาคม 2563.
- [5] เอกสารประกอบการสอนวิชาไมโครคอนโทรลเลอร์เบื้องต้น, เรียบเรียงโดยครูทันพงษ์ ภูริรักษ์, หน้า(1-27), สืบค้นเมื่อวันที่ 30 มีนาคม 2563.
- [6] ไมโครคอนโทรลเลอร์ Arduino, เอกสารประกอบการเรียน วิชาไมโครคอนโทรลเลอร์ (2105-2105), เรียบเรียงโดยครูบุญเกิด สนธิพันธ์, หน้า(335-340),สืบค้นเมื่อวันที่ 30 มีนาคม 2563.
- [7] Arduino ESP32 Wemos D1, <https://www.ett.co.th/prodESP/WEMOS-D1-R32/WEMOS-D1-R32.html>, สืบค้นเมื่อวันที่ 30 มีนาคม 2563.
- [8] สอนใช้งาน Arduino วัดระยะทางด้วยเซ็นเซอร์วัดระยะทาง Ultrasonic Module HC-SR04, <https://www.myarduino.net/article/110/%E0%B8%AA%E0%B8%AD%E0%B8%99%E0%B9%83%87-ultrasonic-module-hc-sr04>, สืบค้นเมื่อวันที่ 30 มีนาคม 2563.
- [9] โมดูลวัดระยะทาง Ultrasonic US-025, <https://www.arduinoall.com/product/>, สืบค้นเมื่อวันที่ 30 มีนาคม 2563.
- [10] การใช้งาน IR Infrared Obstacle, <https://robotsiam.blogspot.comance-sensor.html>, สืบค้นเมื่อวันที่ 30 มีนาคม 2563.
- [11] TCRT5000 Infrared Reflectance Obstacle Avoidance Line Tracking sensor <https://www.arduitronics.com/product/>, สืบค้นเมื่อวันที่ 30 มีนาคม 2563.
- [12] DC to DC Converter: Step-Up, <https://www.thaicconverter.com/category/3/dc-step-up>, สืบค้นเมื่อวันที่ 30 มีนาคม 2563.
- [13] 2A booster step-up board, <https://www.arduinoall.com/product/838/2a-booster-stepup-%E0>, สืบค้นเมื่อวันที่ 30 มีนาคม 2563.
- [14] DC มอเตอร์, https://www.tenergyinnovation.co.th/arduino_learning, สืบค้นเมื่อวันที่ 2 เมษายน 2563.
- [15] 4 pins Push Button, <https://commandronestore.com/products/bg001.php>, สืบค้นเมื่อวันที่ 2 เมษายน 2563.