



# Robot Defense

รายงานวิชา Pre-Project รหัสวิชา 01216747

จัดทำโดย

- |                                    |              |          |
|------------------------------------|--------------|----------|
| 1. นางสาวฐิติรัตน์ เกษมวงศ์        | รหัสนักศึกษา | 60010268 |
| 2. นางสาวณัฐปัทมชญา จิรทีปต์กุลเมธ | รหัสนักศึกษา | 60010314 |
| 3. นางสาวมนัสนันท์ อินนุพัฒน์      | รหัสนักศึกษา | 60010825 |

เสนอ

ผศ.ดร.อุดม จันทร์จรัสสุข

ภาคเรียนที่ 2 ปีการศึกษา 2562

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

## คำนำ

รายงานฉบับนี้จัดทำขึ้นเพื่อเป็นส่วนหนึ่งของวิชา Pre Project รหัสวิชา 01216747 ชั้นปีที่ 3 เพื่อให้ได้ศึกษาหาความรู้ในเรื่องของการทำหุ่นยนต์ แข่งขันหุ่นยนต์ และได้ศึกษาอย่างเข้าใจเพื่อเป็นประโยชน์กับการเรียน

คณะผู้จัดทำหวังว่า รายงานฉบับนี้จะเป็นประโยชน์กับผู้อ่าน หรือนักศึกษาที่กำลังหาข้อมูลเกี่ยวกับเรื่องนี้อยู่หากมีข้อแนะนำหรือเกิดข้อผิดพลาดประการใด คณะผู้จัดทำขอน้อมรับไว้และขออภัยมา ณ ที่นี้ด้วย

คณะผู้จัดทำ

วันที่ 27 มีนาคม 2563

# สารบัญ

บทนำ	1
ขอบเขตโครงการ	2
ทฤษฎีที่เกี่ยวข้อง	3 - 10
การออกแบบวงจร	11 - 14
การออกแบบหุ่นยนต์แข่งขัน	15 - 16
Flow chart	16
Ardunio code	17 - 26
บรรณานุกรม	27 - 28

## บทนำ

หุ่นยนต์ คือ เครื่องจักรชนิดหนึ่งที่ใช้งานแทนมนุษย์ที่มีลักษณะ โครงสร้างที่แตกต่างกันขึ้นอยู่กับประเภท และหน้าที่ของการทำงาน ตามการควบคุมของมนุษย์โดยตรง โดยทั่วไปหุ่นยนต์มักถูกสร้างขึ้นสำหรับการทำงานที่มีความซับซ้อน ในปัจจุบันหุ่นยนต์ได้เริ่มเข้ามามีบทบาทในชีวิตประจำวันของมนุษย์ในด้านต่างๆมากขึ้น ไม่ว่าจะเป็นเรื่องของการหุ่นยนต์ในการช่วยผลิต หุ่นยนต์ในโรงพยาบาล หรือหุ่นยนต์ในการสำรวจ ซึ่งในอดีตหุ่นยนต์มักถูกนำไปใช้ในโรงงานอุตสาหกรรมเป็นส่วนใหญ่ ดังนั้น เทคโนโลยีของหุ่นยนต์จึงเจริญก้าวหน้าอย่างรวดเร็วในปัจจุบัน

หุ่นยนต์เริ่มมีบทบาทกับชีวิตประจำวันของมนุษย์มากขึ้นเรื่อยๆ และได้รับการพัฒนาอย่างต่อเนื่องในปัจจุบัน ทำให้ความสามารถของหุ่นยนต์พัฒนาขึ้นอย่างรวดเร็ว สามารถทำงานต่าง ๆ ที่มนุษย์ไม่สามารถทำได้จำนวนมาก ซึ่งการนำหุ่นยนต์เข้าใช้งานแทนมนุษย์นั้น ทำให้ลดบทบาทมนุษย์ให้น้อยลง สามารถแบ่งประเภทตามความสามารถของหุ่นยนต์ได้

หุ่นยนต์อาจถูกแบ่งออกเป็น 2 ประเภทตามลักษณะการใช้งาน คือ

1. หุ่นยนต์ชนิดติดตั้งอยู่กับที่ (fixed robot) เป็นหุ่นยนต์ที่ไม่สามารถเคลื่อนที่ด้วยตัวเอง มี ลักษณะเป็นแขนกล สามารถเคลื่อนไหวได้เฉพาะแต่ละข้อต่อเท่านั้น มักถูกนำไปใช้ในลักษณะที่ต้องการแรงงานคนเป็นอย่างมาก หุ่นยนต์ชนิดนี้จึงนิยมใช้มากในโรงงานอุตสาหกรรม เช่น ประกอบรถยนต์ เพื่อลดต้นทุนในการจ้างแรงงาน และเพื่อเพิ่มประสิทธิภาพในการผลิต

2. หุ่นยนต์ชนิดที่เคลื่อนที่ได้ (mobile robot) เป็นหุ่นยนต์ที่สามารถเคลื่อนที่ด้วยตัวเอง โดยการใช้ล้อหรือขา ในปัจจุบันหุ่นยนต์ชนิดนี้กำลังถูกพัฒนาและเจริญเติบโตอย่างรวดเร็ว และมีความสามารถในการประกอบงานที่ หลากหลายมากขึ้น ปัจจุบันมีหุ่นยนต์ที่ใช้ในการแข่งขันต่าง ๆ เช่น ด้านกีฬาอาจเป็นหุ่นยนต์ฟุตบอล โดรน ด้านดนตรี เช่น หุ่นยนต์นักเต้น เป็นต้น [1]

ทางคณะผู้จัดทำได้ทำการรวบรวมข้อมูลที่เกี่ยวข้องกับการสร้างหุ่นยนต์เพื่อการศึกษา และนำไปสู่การสร้างหุ่นยนต์แข่งขันที่มีชื่อว่า “FBT Robot” การแข่งขันหุ่นยนต์มีลักษณะคล้ายกับการเล่นบอลลูกด้าน หรือเล่นเตยโดยแบ่งเป็นทีมรุกและทีมรับสลับกันในการแข่งแต่ละรอบ โดยทีมหนึ่งจะประกอบด้วยหุ่นยนต์ 7 ตัว ฝ่ายทีมรุกจะต้องวิ่งไปหาฝั่งตรงข้าม จนผ่านเส้นแดงแล้วกลับมาโดยที่ไม่ถูกทีมรับจับได้ก็จะเป็นฝ่ายชนะในการแข่งขันรอบนั้น หุ่นยนต์ที่ถูกจับได้จะถูกตัดออกจากการแข่งขันในรอบนั้น ส่วนทีมรับจะสามารถวิ่งสกัดกั้นฝ่ายตรงข้ามในพื้นที่ป้องกันเท่านั้นถ้าวิ่งออกนอกพื้นที่ก็จะถูกตัดออกจากการแข่งขันในรอบนั้นเช่นกัน ถ้าไม่มีหุ่นยนต์ตัวไหนสามารถผ่านด่านได้ทีมรับจะเป็นฝ่ายชนะ การแข่งขันของแต่ละรอบจะยุติเมื่อทีมรุกสามารถผ่านด่านได้สำเร็จ หรือเมื่อทีมใดทีมหนึ่งไม่เหลือผู้เล่น

## ขอบเขตโครงการ

### ปัญหา

การแข่งขันหุ่นยนต์มีลักษณะคล้ายกับการเล่นบอลลูกด้าน หรือเล่นเตยโดยแบ่งเป็นทีมรุกและทีมรับ สลับกันในการแข่งแต่ละรอบ โดยทีมหนึ่งจะประกอบด้วยหุ่นยนต์ 7 ตัว ฝ่ายทีมรุกจะต้องวิ่งไปหาฝั่งตรงข้าม จนผ่านเส้นแดงแล้วกลับมาอย่างปลอดภัย(ผ่านเส้นสีเหลือง) โดยที่ไม่ถูกทีมรับจับได้ก็จะเป็นฝ่ายชนะในการแข่งขัน รอบนั้น หุ่นยนต์ที่ถูกจับได้จะถูกตัดออกจากการแข่งขันในรอบนั้น ส่วนทีมรับจะสามารถวิ่งสกัดกั้นฝ่ายตรงข้ามในพื้นที่ป้องกันเท่านั้นถ้าวิ่งออกนอกพื้นที่ก็จะถูกตัดออกจากการแข่งขันในรอบนั้นเช่นกัน ถ้าไม่มีหุ่นยนต์ตัวไหนสามารถผ่านด่านได้ทีมรับจะเป็นฝ่ายชนะ การแข่งขันของแต่ละรอบจะยุติเมื่อทีมรุกสามารถผ่านด่านได้สำเร็จ หรือเมื่อทีมใดทีมหนึ่งไม่เหลือผู้เล่น

### แนวคิดในการแก้ปัญหา

#### 1.กลยุทธ์เกมรุก

การเดินรถตัวแรกจะมีเซ็นเซอร์ตัวจับรถด้านหน้า ด้านข้างซ้ายขวา และหลังรถ เมื่อไรที่เจอรถฝ่ายตรงข้าม จะทำการหยุด ถ้าเจอด้านข้างจะทำการเร่งเครื่องเพื่อหลบหลีก แต่ถ้าเจอรถทั้งด้านหน้าและด้านข้างจะทำการถอยรถโดยเมื่อเซ็นเซอร์ตัวหลังตรวจจับรถด้านหลังจะทำการหยุดเพื่อไม่ให้ชน

นอกจากนี้ตัวรถจะมีเซ็นเซอร์ที่ได้ท้องรถเพื่อคอยจับเส้นแดงว่ารถได้ผ่านโซนป้องกันมาแล้ว หลังจากนั้น จะทำการกลับรถและเดินรถโดยใช้วิธีการเดินรถแบบเดียวกับตอนแรก แล้วเมื่อเซ็นเซอร์จับเส้นเหลืองจะทำการหยุดรถเพราะผ่านเส้นชัยแล้ว

#### 2.กลยุทธ์เกมรับ

จะทำการรับฝ่ายตรงข้ามด้วยการวิ่งเดินหน้าถอยหลัง โดยในแถวเดียวกันจะมีเพื่อนอยู่ด้วย 1 คัน เพื่อช่วยป้องกันโดยจะทำการแบ่งคนละครึ่งสนาม โดยใช้เซ็นเซอร์ได้ท้องรถตรวจจับเส้นขอบสนาม เมื่อพบเส้นจะทำการถอยหลังเพื่อไม่ให้รถออกนอกสนาม และจะใช้เวลาในการวิ่งเดินหน้าถอยหลังในเวลาสั้นๆเพื่อให้มีช่องโหว่ในการรุกซ้ำของฝ่ายรุกร่น้อยลง

## ทฤษฎีที่เกี่ยวข้อง

1.ภาษาของการเขียนโปรแกรมใช้งาน Arduino Board ได้แก่ ภาษา C/C++

ฟังก์ชันหลัก(Structure) เป็นฟังก์ชันหลักในการเขียนโปรแกรม

- Setup() คือ ฟังก์ชันใช้ในการประกาศค่าเริ่มต้น ตำแหน่งพอร์ตที่ใช้งาน รวมถึงฟังก์ชันที่อยู่ไลบรารีที่ใช้ งาน เป็นฟังก์ชันที่ทำงานเพียงครั้งเดียว จะทำงานทุกครั้ง ที่มีการรีเซต หรือรีบูตเครื่องใหม่ เท่านั้น
- Loop () คือ ฟังก์ชันใช้ในการเขียนโค้ดโปรแกรมการทำงานของArduinoเป็นฟังก์ชันการวนลูปไปเรื่อย ๆ

ชุดคำสั่งในการควบคุม (Control Structures) เป็นชุดคำสั่งในการใช้ในการตัดสินใจหาทางออก เพื่อใช้ในการทำงาน

- If คือ คำสั่งในการตัดสินใจ แบบตัวเลือกเดียว โดยใช้งานร่วมกับ and, or not, ==, !=, <, >เพื่อใช้ในการตัดสินใจในการหาคำตอบ
- If...else คือ คำสั่งในการตัดสินใจ แบบหลายตัวเลือก โดยใช้งานร่วมกับ And, Or Not, ==, !=, <, > เพื่อใช้ในการตัดสินใจในการหา
- For เป็นคำสั่งกำหนดเงื่อนไขเป็นจำนวนครั้งที่จะทำตามชุดคำสั่งต่าง ๆ ภายใน loop เหมาะที่จะใช้กับงานประเภทที่ไม่มีการ
- Switch case ใช้ในการจัดการเงื่อนไขหลายเงื่อนไขโดยเฉพาะการใช้งานโครงสร้าง การจำแนกเงื่อนไขไม่จำเป็นต้องอาศัยเฉพาะตัวแปรที่เก็บค่าจำนวนเต็มเท่านั้น ข้อมูลแบบอื่นก็ใช้ได้
- While คือเงื่อนไขที่จะทำการตรวจสอบว่าเป็นจริงหรือเท็จ ชุดคำสั่งก็คือ ส่วนที่ทำงานซ้ำๆ โดยจะต้องมีคำสั่งที่จะทำให้ เงื่อนไข เป็นเท็จด้วย
- Do... while เป็นคำสั่งที่กำหนดให้มีการทำงานวนรอบ คล้าย ๆ คำสั่ง While
- Break เป็นคำสั่งที่ให้โปรแกรมออกจาก loop ทันที โดยไม่ทำคำสั่งที่เหลือต่อ
- Continue ใช้สำหรับสั่งให้กลับไปเริ่มต้นที่จุดเริ่มต้นใหม่ ใช้ร่วมกับคำสั่งการวนลูปต่างๆจะต่างกับคำสั่ง คำสั่ง break นั้นจะเป็นคำสั่งเพื่อออกจาก loop ส่วนคำสั่ง continue นั้นจะเป็นคำสั่งเพื่อไปยังต้น
- Return คือ คำสั่งที่ส่งค่าอะไรก็ได้กลับออกไปจากฟังก์ชัน
- Goto เป็นคำสั่งที่ทำให้ กระโดดไปทำบรรทัดนั้น
- #define คือ คำสั่งกำหนดค่านิพจน์ต่าง ๆ ให้กับชื่อของตัวคงที่
- #include การกำหนดชื่อไฟล์ตามหลัง include จะใช้เครื่องหมาย <> ซึ่งจะเป็นการอ่านไฟล์จากโพลเดอร์ที่กำหนดไว้

### การแปลงค่า (Conversion)

- Char () แปลงค่าข้อมูลให้เป็น char
- Byte () แปลงค่าข้อมูลให้เป็น byte
- Int () แปลงค่าข้อมูลให้เป็น int
- Word () แปลงค่าข้อมูลให้เป็น word
- Long () แปลงค่าข้อมูลให้เป็น long
- Float () แปลงค่าข้อมูลให้เป็น float

### Time

- delay () คือ คำสั่งหยุดการทำงานโปรแกรมสำหรับจำนวนของเวลา (ใน milliseconds) Milliseconds = จำนวน มิลลิวินาทีในการหยุดการทำงานชั่วคราว

### Functions

- pinMode () ใช้ในกลุ่ม void setup () เพื่อกำหนดหน้าที่ขาของไมโครคอนโทรลเลอร์ให้เป็นขารับสัญญาณ INPUT หรือขาส่งสัญญาณ OUTPUT
- digitalWrite () คือ การส่งค่าลอจิก HIGH หรือ LOW (เปิด หรือปิด) ไปยังขา digital ที่กำหนดหมายเลขขาไอซีอาจ กำหนดเป็นตัวแปรหรือค่าคงที่ (0-13)
- Digital Read () อ่านค่าจาก ขาไอซีที่ถูกกำหนดให้เป็น digital pin ซึ่งจะได้ผลลัพธ์เป็น HIGH หรือ LOW หมายเลขขาไอซีอาจกำหนดเป็นตัวแปรหรือค่าคงที่ (0-13)

### Analog I/O

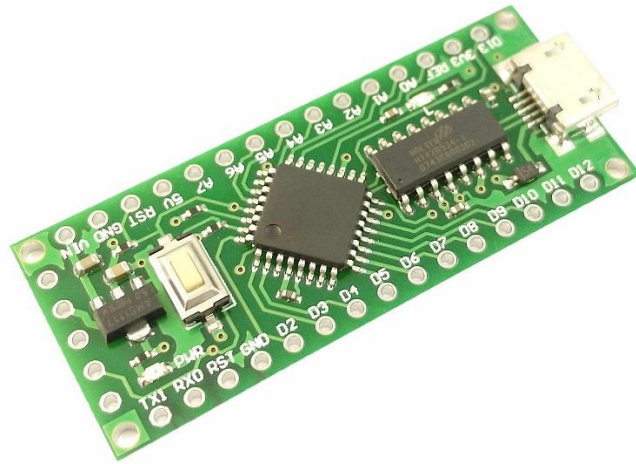
- Analog Read () คำสั่งนี้อ่านค่าจากขา Analog จะได้ค่า 10 bit คำสั่งนี้จะทำงานกับขา analog input (0-5) เท่านั้น และได้ผลลัพธ์เป็นเลขจำนวนเต็มค่า 0 – 1023
- AnalogWrite() เป็นคำสั่งเขียนค่า analog เทียมโดยใช้ hardware enabled pulse width modulation(PWM) ไปยังขา outputที่สามารถทำ PWM ได้ ใน Arduino รุ่นใหม่ที่ใช้ชิพ Atmega168 คำสั่งนี้จะทำงานกับขา 3, 5, 6, 9, 10, และ 11 ส่วน Arduino รุ่นเก่าที่ใช้ Atmega8 จะรองรับเพียงขา 9, 10 และ 11 ค่าที่เขียนสามารถใช้เป็นตัวแปรหรือค่าคงที่จาก 0 – 255
- Delay () หยุดการทำงานโปรแกรมสำหรับจำนวนของเวลา (ใน milliseconds) ระบุเป็นพารามิเตอร์ (มี 1,000 มิลลิวินาทีในทั้งสองเป็น.) Milliseconds = จำนวนมิลลิวินาทีในการหยุดการทำงานชั่วคราว [2]

3. ไมโครคอนโทรลเลอร์ (microcontroller มักย่อว่า  $\mu C$ ,  $uC$  หรือ  $MCU$ ) คือ อุปกรณ์ควบคุมขนาดเล็ก ซึ่งบรรจุความสามารถที่คล้ายคลึงกับระบบคอมพิวเตอร์ โดยในไมโครคอนโทรลเลอร์ได้รวมเอาซีพียู, หน่วยความจำ และพอร์ต ซึ่งเป็นส่วนประกอบหลักสำคัญของระบบคอมพิวเตอร์เข้าไว้ด้วยกัน โดยทำการบรรจุเข้าไว้ในตัวถังเดียวกัน แบ่งออกได้ 5 ส่วน ดังนี้

- หน่วยประมวลผลกลางหรือซีพียู (CPU : Central Processing Unit)
- หน่วยความจำ (Memory) สามารถแบ่งออกเป็น 2 ส่วน คือ
  - หน่วยความจำที่มีไว้สำหรับเก็บโปรแกรมหลัก (Program Memory) เปรียบเสมือนฮาร์ดดิสก์ของเครื่องคอมพิวเตอร์ตั้งโต๊ะ คือข้อมูลใดๆ ที่ถูกเก็บไว้ในนี้จะไม่สูญหายไปแม้ไม่มีไฟเลี้ยง อีกส่วนหนึ่งคือ-
  - หน่วยความจำข้อมูล (Data Memory) ใช้เป็นเหมือนกับกระดานทดในการคำนวณของซีพียู และเป็นที่พักข้อมูลชั่วคราวขณะทำงาน แต่หากไม่มีไฟเลี้ยง ข้อมูลก็จะหายไปคล้ายกับหน่วยความจำแรม (RAM) ในเครื่องคอมพิวเตอร์ต่างๆ ไป แต่สำหรับไมโครคอนโทรลเลอร์สมัยใหม่ หน่วยความจำข้อมูลจะมีทั้งที่เป็นหน่วยความจำแรม ซึ่งข้อมูลจะหายไปเมื่อไม่มีไฟเลี้ยง และเป็นอีอีพรอม (EEPROM : Electrically Erasable Programmable Read-Only Memory) ซึ่งสามารถเก็บข้อมูลได้แม้ไม่มีไฟเลี้ยงก็ตาม
- ส่วนติดต่อกับอุปกรณ์ภายนอก หรือพอร์ต (Port) มี 2 ลักษณะคือ พอร์ตอินพุต (Input Port) และพอร์ตส่งสัญญาณหรือพอร์ตเอาต์พุต (Output Port) ส่วนนี้จะใช้ในการเชื่อมต่อกับอุปกรณ์ภายนอก ถือว่าเป็นส่วนที่สำคัญมาก ใช้ร่วมกันระหว่างพอร์ตอินพุต เพื่อรับสัญญาณ อาจจะด้วยการกดสวิตช์ เพื่อนำไปประมวลผลและส่งไปพอร์ตเอาต์พุต เพื่อแสดงผลเช่น การติดสว่างของหลอดไฟ เป็นต้น
- ช่องทางเดินของสัญญาณ หรือบัส (BUS) คือเส้นทางการแลกเปลี่ยนสัญญาณข้อมูลระหว่าง ซีพียู หน่วยความจำและพอร์ต เป็นลักษณะของสายสัญญาณ จำนวนมากอยู่ภายในตัวไมโครคอนโทรลเลอร์ โดยแบ่งเป็นบัสข้อมูล (Data Bus) , บัสแอดเดรส (Address Bus) และบัสควบคุม (Control Bus)
- วงจรกำเนิดสัญญาณนาฬิกา เป็นองค์ประกอบที่สำคัญมากอีกส่วนหนึ่ง เนื่องจากการทำงานที่เกิดขึ้นในตัวไมโครคอนโทรลเลอร์ จะขึ้นอยู่กับกำหนัดจังหวะ หากสัญญาณนาฬิกามีความถี่สูง จังหวะการทำงานก็จะสามารถทำได้ถี่ขึ้นส่งผลให้ไมโครคอนโทรลเลอร์ตัวนั้น มีความเร็วในการประมวลผลสูงตามไปด้วย [3]



ในการสร้างหุ่นยนต์แข่งขันครั้งนี้ทางคณะผู้จัดทำได้เลือก Arduino LGT8F328P มาใช้เพราะมีความกะทัดรัดในการใช้งานและมีความเร็วในการประมวลผล แสดงดังรูปที่ 3.1



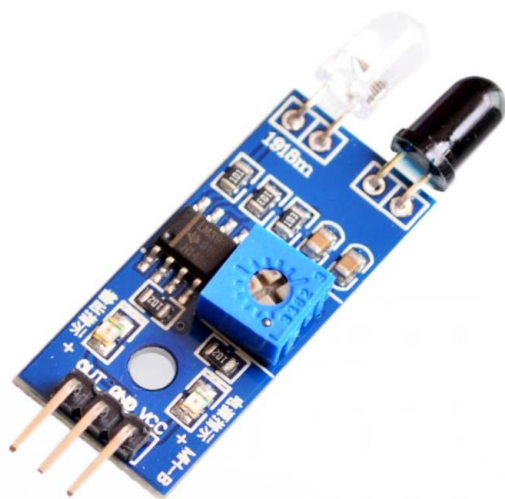
รูปที่ 1 Arduino LGT8F328P [4]

ตารางที่ 1 รายละเอียด Arduino LGT8F328P

MCU	LGT8F328P
FLASH	32Kbytes
SRAM	2Kbytes
E2PROM	0K/1K/2K/4K/8K(FLASH Share)
PWM	8
Frequency	16MHz(Maximum 32MHz)
ADC	6 passageway12 position
DAC	1passageway8 position
UART	1
SPI	YES
TWI(I2C)	YES
GUID	YES
Internal benchmark	1.024V/2.048V/4.096V $\pm 0.5\%$
System logic level	Factory 3V3 (switch from pad to 5V)

#### 4. โมดูลเซ็นเซอร์แสงสำหรับตรวจจับวัตถุกีดขวาง (IR Infrared Obstacle Avoidance Sensor Module)

เซ็นเซอร์ใช้ตรวจจับวัตถุโดยใช้หลักการสะท้อนของแสงเมื่อไปชนวัตถุ (Reflective) สามารถปรับความไวในการตรวจจับได้ ใช้แสงอินฟราเรดในการตรวจจับโดยจะมีตัวรับและตัวส่ง infrared ในตัว ตัวส่งสัญญาณ(สีขาว) infrared จะส่งสัญญาณออกมา และเมื่อมีวัตถุมาบัง คลื่นสัญญาณ infrared ที่ถูกส่งออกมาจะสะท้อนกลับไปเข้าตัวรับสัญญาณ (สีดำ) สามารถนำมาใช้ตรวจจับวัตถุที่อยู่ตรงหน้าได้ และสามารถปรับความไว ระยะการตรวจจับ ใกล้หรือไกลได้ เซ็นเซอร์แบบนี้จะมีช่องในการทำงาน หรือ ระยะในการตรวจจับจะได้ใกล้กว่าแบบ Opposed mode ซึ่งในสภาวะการทำงานปกติตัวรับ Receiver จะสามารถรับสัญญาณแสงจากตัวส่ง Emitter ได้ตลอดเวลา เนื่องจากลำแสงจะสะท้อนกับแผ่นสะท้อน Reflector อยู่ตลอดเวลา จะแสดงค่า เป็น 0 หน้าหลักของเซ็นเซอร์ชนิดนี้ จะคอยตรวจจับวัตถุที่เคลื่อนที่ตัดผ่านหน้าเซ็นเซอร์ เมื่อวัตถุ หรือ ชิ้นงานผ่านเข้ามาที่หน้าเซ็นเซอร์ แล้วจะการขวางลำแสงที่ส่งจากตัวส่ง Emitter ที่ส่งไปยังแผ่นสะท้อน จึงทำให้ตัวรับ Receiver ไม่สามารถรับลำแสงที่จะสะท้อนกลับมาได้ จะแสดงค่า เป็น 1 แสดงดังรูปที่ 4.2



รูปที่ 2 โมดูลเซ็นเซอร์แสงสำหรับตรวจจับวัตถุกีดขวาง

ตารางที่ 2 แสดงรายละเอียด IR Infrared Obstacle Avoidance Sensor Module [5]

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

## 5. เซ็นเซอร์วัดระยะทาง (Ultrasonic Module)

สามารถใช้งานได้ทั้งแรงดัน 3.3V และ 5V สำหรับเซ็นเซอร์วัดระยะทางที่มีลักษณะใกล้เคียงกันมักจะใช้ได้เฉพาะแรงดันไฟฟ้า 5V เซ็นเซอร์วัดระยะทาง ใช้คลื่นเสียงในย่านอัลตราโซนิกในการทำงาน โดยหลักการคือ ตัวส่งเมื่อส่งเสียงออกไปแล้วเสียงไปกระทบกับวัตถุแล้วจะทำให้คลื่นนั้นสะท้อนกลับมาแล้วตัวรับทำหน้าที่รับเข้ามา ค่าเวลาที่วัดได้หลังส่งออกไปแล้วรับกลับมาจะถูกนำไปคำนวณโดยเทียบกับความเร็วเสียงทำให้ได้ระยะทางออกมา นอกจากนี้ความเร็วของเสียงยังขึ้นอยู่กับอุณหภูมิ ณ ขณะนั้นด้วย แสดงดังรูปที่ 5.3



รูปที่ 3 เซ็นเซอร์วัดระยะทาง

ตารางที่ 3 แสดงรายละเอียด Ultrasonic Module[6]

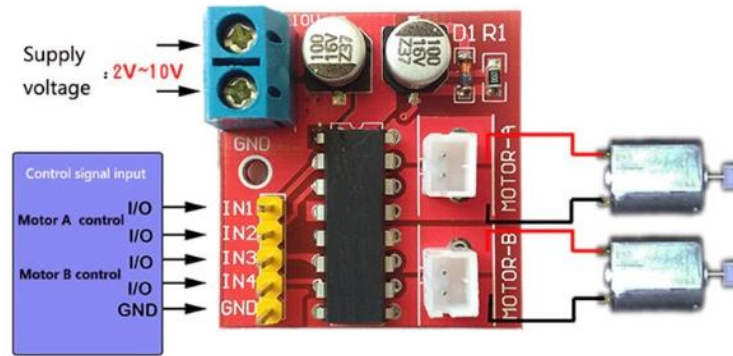
ไฟเลี้ยง VCC	3.3 – 5 V
ดิจิตอลอินพุต	0 หรือ 1
ดิจิตอลเอาต์พุต	0 หรือ 1
HC-SR04	ระยะตรวจจับ 2 – 400 cm
US-025	ระยะตรวจจับ 2 – 600 cm

## 6. โมดูลขับมอเตอร์ (DC Motor Speed Control)

บอร์ดขับมอเตอร์ ควบคุมความเร็วได้ สำหรับควบคุมมอเตอร์ 2 ตัว กำลังขับต่อเนื่อง 1.5A สูงสุดที่ 2.5A มี วงป้องกันโหลดและความร้อนเกิน ประหยัดพลังงาน ความร้อนขณะทำงานต่ำ

โดยคณะผู้จัดทำได้เลือกใช้ คือ Mini-298N 2-Channel PWM Motor Driver ประกอบด้วย H-Bridge เป็นโมดูลที่ใช้ในการควบคุมความเร็วและทิศทางของมอเตอร์ และยังสามารถนำไปประยุกต์ใช้กับ Project

อื่นได้อีกด้วย เป็นวงจรที่สามารถใช้ควบคุมกระแสได้ทั้งชั่ววอกและลบด้วยการควบคุม pulse width modulation (PWM) เป็นการควบคุมแบบ digital ที่มีการนำมาใช้กันมากโดยส่วนมากเพื่อเป็นการประหยัดพลังงานและ สามารถควบคุม Output ได้ โดยมีการกระตุ้นอย่างต่อเนื่อง ซึ่งทำให้เกิดการสูญเสียพลังงานน้อยมาก [7] แสดงดังรูปที่ 6.4



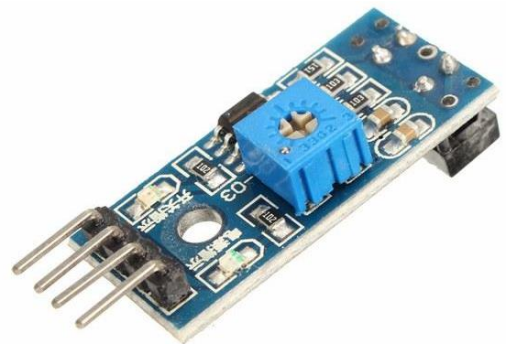
รูปที่ 4 โมดูลขับมอเตอร์ [8]

ตารางที่ 4 แสดงรายละเอียด Mini-298N 2-Channel PWM Motor Driver

Supply Voltage	2 – 10 V
Signal Input	1.8 – 7 V
Max Output current	3 A (1.5*2 A)
Control signal	PWM

## 7. เซ็นเซอร์แทรกตามเส้น (TCRT5000 Infrared Reflective sensor)

เซนเซอร์ใช้คลื่นกีดขวาง เส้นขาวดำ นับจำนวน เป็นโมดูลอ่านค่าสะท้อนกลับของแสง ใช้ไฟ 3.3-5V เหมาะสำหรับใช้กับ Arduino ให้เอาต์พุตออกมา 2 แบบ คือ แบบ digital สามารถปรับค่าที่ต้องการได้ เมื่อค่าที่อ่านได้ถึงระดับที่ต้องการก็จะส่งค่า 1 ออกมา ถ้ายังไม่ถึงระดับก็จะส่งค่า 0 ออกมา และอีกแบบคือเอาต์พุตแบบanalog อ่านค่าได้เป็นตัวเลข 0-1023 หรือสัญญาณไฟในช่วง 0-5V สามารถนำไปประยุกต์กับงานได้หลายแบบ เช่น ใช้เป็นเซนเซอร์หลักเลี้ยงการชน แสดงดังรูปที่ 7.5



รูปที่ 5 เซ็นเซอร์แทรกตามเส้น

ตารางที่ 5 แสดงรายละเอียด TCRT5000 Infrared Reflective sensor [9]

ไฟเลี้ยง VCC	3.3 – 5 V
ดิจิตอลเอาต์พุต	0 หรือ 1
อนาล็อกเอาต์พุต	0 - 1023

## 8. วงจร DC/DC Step-up

เป็นแผงวงจรที่เพิ่มแรงดันแบบปรับค่าได้ แสดงดังรูปที่ 8.6



รูปที่ 6 วงจร DC/DC Step-up รุ่น MT3608

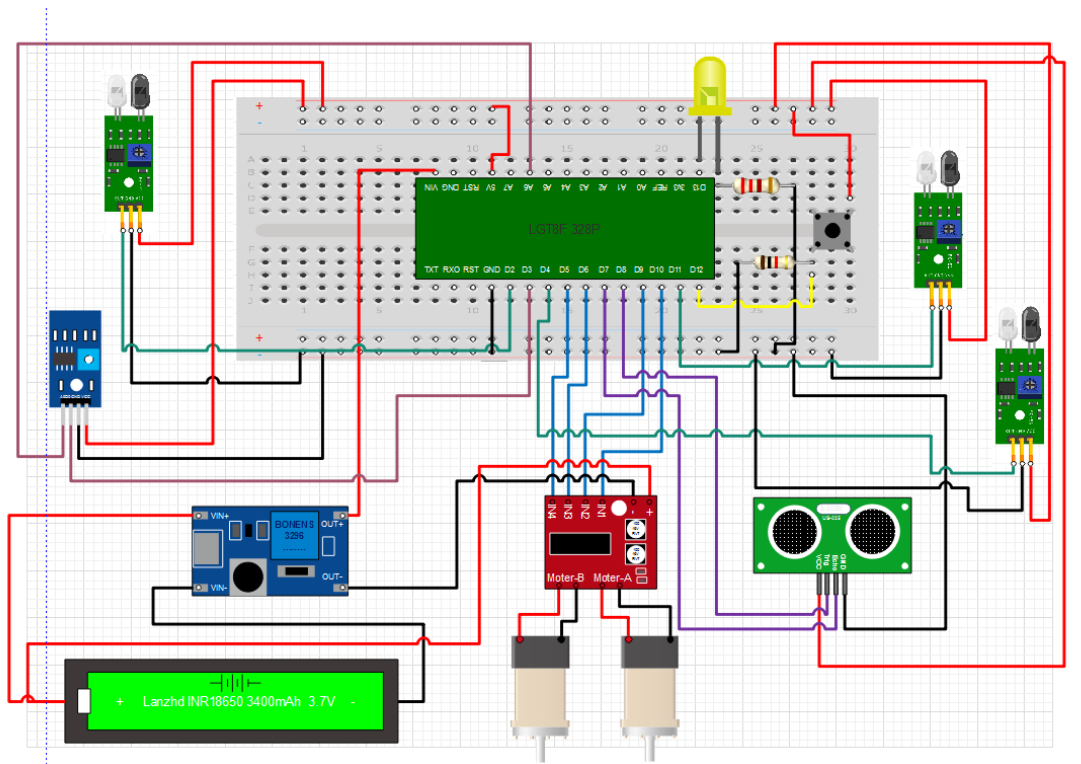
ตารางที่ 6 แสดงรายละเอียด วงจร DC/DC Step-up รุ่น MT3608 [10]

กระแสไฟขาออกสูงสุด	2 A
แรงดันไฟเข้า	2 – 24 V
แรงดันขาออกสูงสุด	28 V
% efficiency	> 93 %

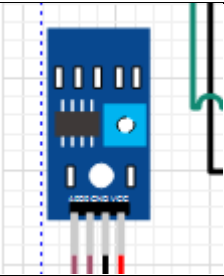
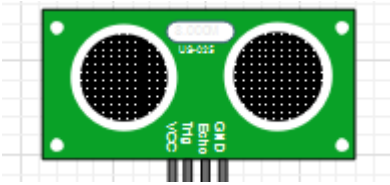
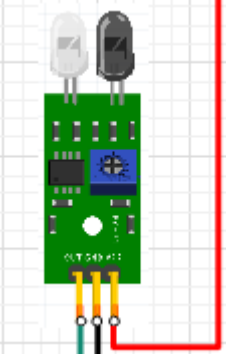
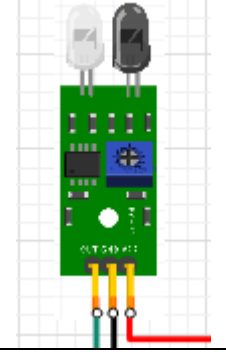
## เครื่องมือ หรืออุปกรณ์ที่จำเป็น

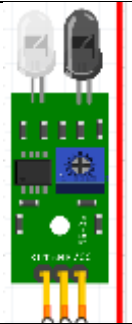
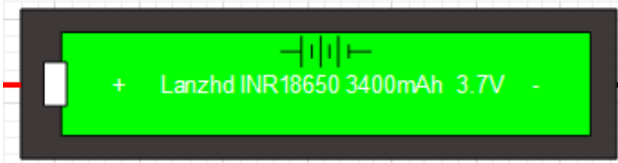
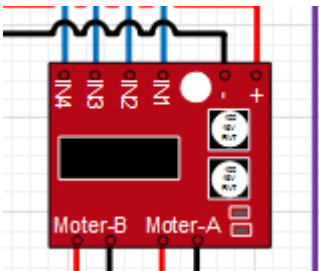
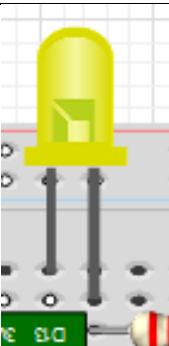
1. Arduino board (LGT8F328P) จำนวน 1 อัน
2. ถ่าน Lanzhd INR18650 3400mAh 3.7V จำนวน 1 ก้อน
3. Motors จำนวน 2 ตัว
4. H-bridge Driver จำนวน 1 อัน
5. Breadboard จำนวน 1 อัน
6. DC/DC Step-up Converter จำนวน 1 อัน
7. TCRT5000 Infrared Reflective sensor จำนวน 1 อัน
8. IR Infrared Obstacle Avoidance Sensor จำนวน 2 อัน
9. Ultrasonic Sensor จำนวน 1 อัน
10. LED จำนวน 1 อัน
11. ตัวต้านทาน จำนวน 2 อัน
12. Switch จำนวน 1 อัน
13. สายไฟ

## แผนภาพวงจร


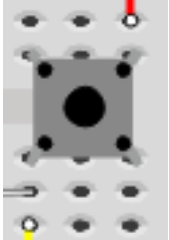

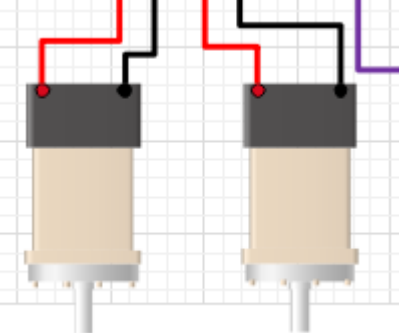
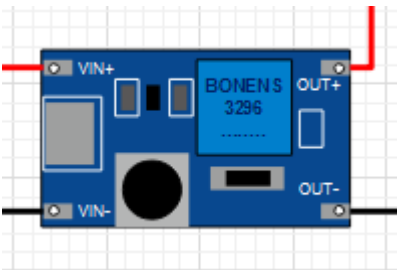


โดยมีรายละเอียดการต่อวงจรดังนี้

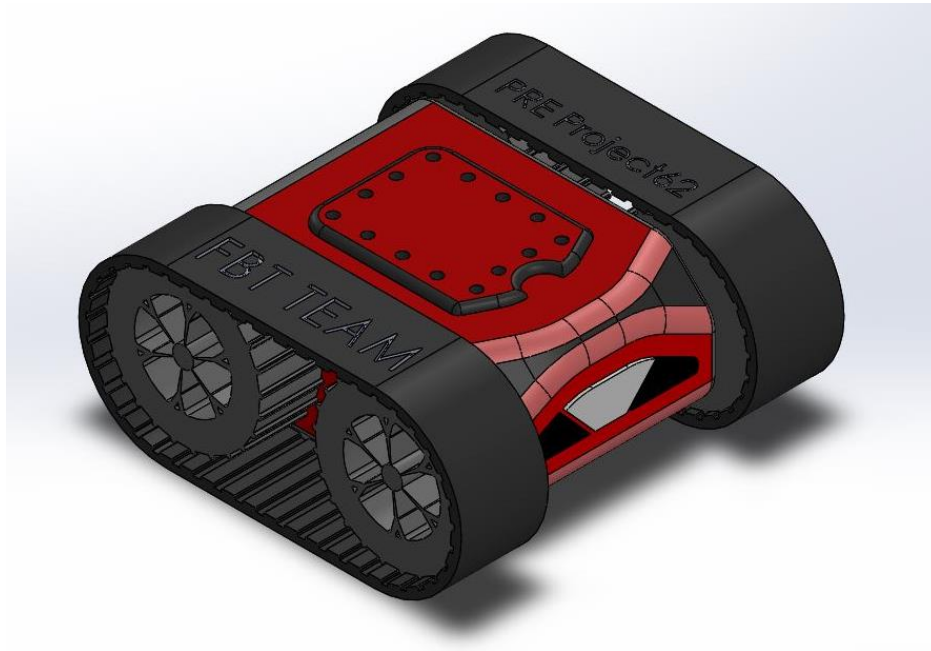
อุปกรณ์	รูปภาพ	การต่อ
TCRT5000 Infrared Reflective sensor		VCC: 5V GND: Ground D0: Digital output (0/1): D3 A0: Analog output: A6
Ultrasonic Sensor		VCC: 5V Trig: Digital output (0/1): D8 Echo: Digital Input (0/1): D7 GND: Ground
IR Infrared Obstacle Avoidance Sensor (ซ้าย)		OUT: Digital Input (0/1): D2 GND: Ground VCC: 5V
IR Infrared Obstacle Avoidance Sensor (ขวา)		OUT: Digital Input (0/1): D11 GND: Ground VCC: 5V

อุปกรณ์	รูปภาพ	การต่อ
IR Infrared Obstacle Avoidance Sensor (หลัง)		OUT: Digital Input (0/1): D4 GND: Ground VCC: 5V
Lanzhd INR18650 3400mAh 3.7V		ขั้วบวก: VIN+ ของ DC/DC Step-up Converter และขั้วบวกของ H-bridge Driver ขั้วลบ: VIN- ของ DC/DC Step-up Converter
H-bridge Driver		ขั้วบวก: ขั้วบวกของ battery ลบ: Ground IN1: D10 IN2: D9 IN3: D6 IN4: D5 Motor A : Motor Motor B : Motor
LED		ขาลบ : D13 ขาบวก : ตัวต้านทาน



อุปกรณ์	รูปภาพ	การต่อ
ตัวต้านทาน		ขาซ้าย : LED ขาขวา : Ground
Switch		ข้างขวา : เข้า 5V ข้างซ้าย : เข้าตัวต้านทานและ D12
ตัวต้านทาน		ข้างขวา : Switch ข้างซ้าย : Ground
DC Geared Motors		ต่อกับ H-bridge Driver
DC/DC Step-up Converter		Vin- : battery (-) Vin+ : battery (+) Vout- : เข้าลบของ H-bridge Driver Vout+ : VIN (Arduino)

## 11.การออกแบบหุ่นยนต์แข่งขัน



รูปที่ 7 การออกแบบตัวรถ

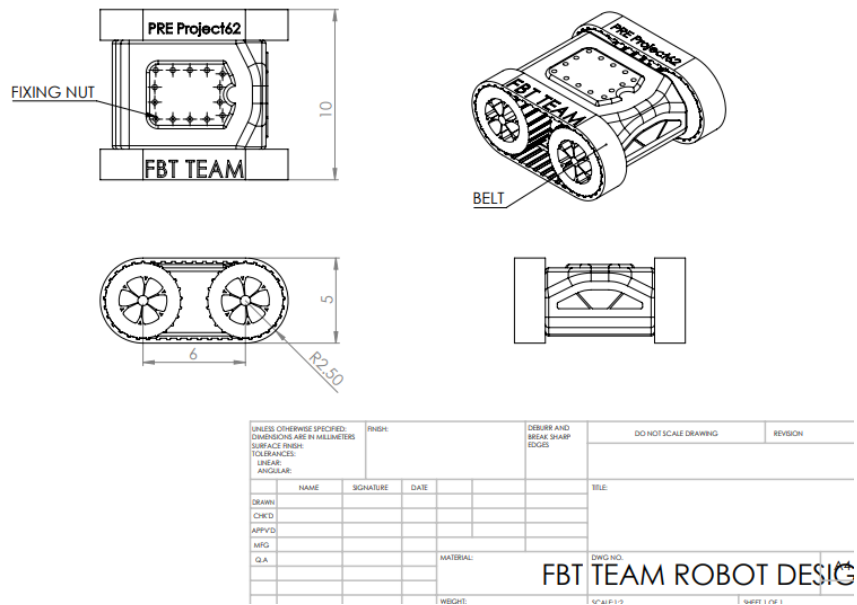
โปรแกรมที่ใช้ออกแบบ : Solid work

ขนาด : 10\*10 เซนติเมตร

การขับเคลื่อน : ใช้Motor สองตัว แล้วเดินรถด้วยระบบสายพาน

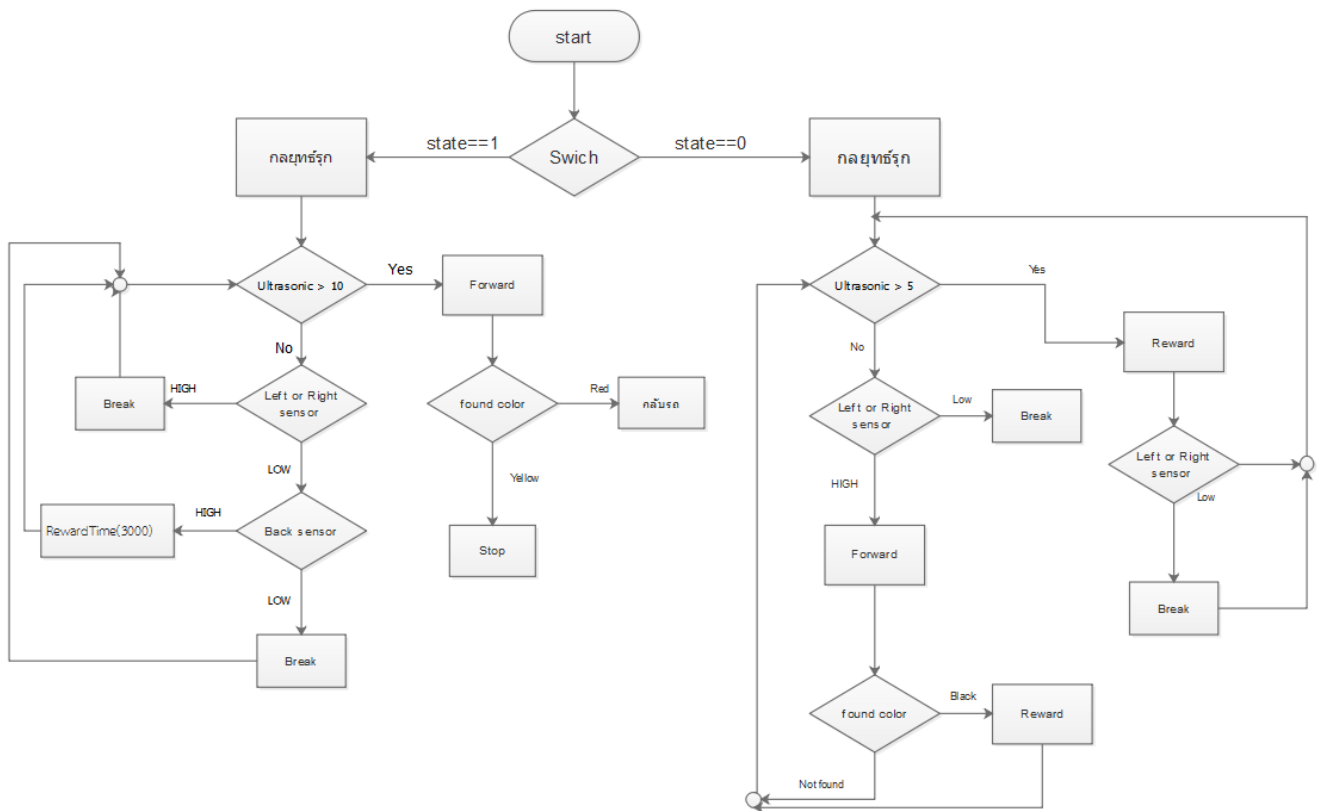
การติดตั้งsensor :

1. หน้ารถมี Ultrasonic Sensor 1 ตัว ไว้ตรวจจับรถด้านหน้า
2. ด้านข้างซ้ายขวามี IR Infrared Obstacle Avoidance Sensor 1 ตัว/ข้าง ไว้ตรวจจับรถด้านข้าง
3. ด้านหลังมี IR Infrared Obstacle Avoidance Sensor 1 ตัว ไว้ตรวจจับรถด้านหลัง
4. ใต้ท้องรถมี TCRT5000 Infrared Reflective sensor 1 ตัวไว้ตรวจจับเส้น



รูปที่ 8 ภาพฉายตัวรถ

### Flow chart



## Arduino Codes ที่ใช้ในการขับเคลื่อน

```
#define AIN1 D10 //motor A

#define AIN2 D9 //motor A

#define BIN3 D6 //motor B

#define BIN4 D5 //motor B

#define lefts D2 //sensorซ้าย

#define rights D11 //sensorขวา

#define backs D4 //sensorหลัง

#define RED 1

#define YELLOW 2

#define BLACK 3

#define NOCOLOR 0

#define maxSpd 255 // motor max speed

#include <HCSR04.h>

HCSR04 hc(D8,D7); //initialisation class HCSR04 (trig,echo);

int analogPin = A6; //ประกาศตัวแปร ให้ analogPin แทนขา analog ขาที่6

int led1 = D13;

int buttonPin = D12;

int val=0;

int old_val=0;
```

```

int state=0;

int color=0;

int getColor() {

    int NO_color = analogRead(analogPin);    //อ่านค่าสัญญาณ analog ขา6 ที่ต่อกับ TCRT5000

    if ((NO_color>1900)&&(NO_color<2200))    //สีเหลือง

        return YELLOW;

    else if ((NO_color>2200)&&(NO_color<2500))    //สีแดง

        return RED;

    else if ((NO_color>3600)&&(NO_color<3900))    //สีดำ

        return BLACK;

    else    //ไม่พบสี

        return NOCOLOR;

}

void setup() {

    pinMode(lefts, INPUT);

    pinMode(rights, INPUT);

    pinMode(backs, INPUT);

    pinMode(buttonPin, INPUT);

    pinMode(led1, OUTPUT);

    pinMode(AIN1, OUTPUT);

```

```

pinMode(AIN2, OUTPUT);

pinMode(BIN3, OUTPUT);

pinMode(BIN4, OUTPUT);

Serial.begin(115200);

}

void loop() {

    color = getColor();

    val = digitalRead(buttonPin);

    if( (val==HIGH) && (old_val==LOW))

    {

        state=!state;

    }

    old_val=val;

    if (state==1) //เมื่อกดสวิตช์ 1 ครั้ง ใช้กลยุทธรูก

    {

        digitalWrite(led1, HIGH);

        if((hc.dist())>10)&&((digitalRead(lefts)==HIGH)||((digitalRead(rights)==HIGH))&&(color == NOCOLOR))
        // เดินหน้าเมื่อ sensor ด้านหน้า ด้านข้าง และด้านล่างไม่ทำงาน

        {

            AForward(maxSpd);

```

```

BForward(maxSpd);

}

if((hc.dist()<10)&&(digitalRead(backs)==LOW)&&((digitalRead(lefts)==HIGH)||(digitalRead(rights)==HIGH))&&(color == NOCOLOR))

// break เมื่อ sensor ด้านหน้า ด้านหลังทำงาน แต่ด้านข้างและด้านล่างไม่ทำงาน

{

ABreakTime(1000);

BBreakTime(1000);

}

if((hc.dist()<10)&&(digitalRead(backs)==HIGH)&&((digitalRead(lefts)==LOW)||(digitalRead(rights)==LOW))&&(color == NOCOLOR))

// เดินถอยหลัง 3 วิ เมื่อ sensor ด้านหน้าและข้างซ้ายหรือขวาทำงาน แต่ด้านหลังไม่ทำงาน

{

ARewardTime(3000);

BRewardTime(3000);

}

if(color == RED) // เดินกลับรถ เมื่อ sensor ด้านล่างตรวจจับเส้นสีแดงได้

{

AForwardTime(5000);

BRewardTime(5000);

}

```

```

if(color == YELLOW) // รถหยุด เมื่อ sensor ด้านล่างตรวจจับเส้นสีเหลืองได้

{

  AStop();

  BStop();

}

}

else

{

  digitalWrite (led1,LOW);

  if((hc.dist())>5)&&((digitalRead(lefts)==HIGH)||(digitalRead(rights)==HIGH))&&(color == NOCOLOR))
  // ถอยหลังเมื่อ sensor ด้านหน้า ด้านข้าง และด้านล่าง ไม่ทำงาน

  {

    AReward(maxSpd);

    BReward(maxSpd);

  }

  if((hc.dist())<5)&&((digitalRead(lefts)==HIGH)||(digitalRead(rights)==HIGH))&&(color == NOCOLOR))
  // เดินหน้าเมื่อ sensor ด้านหน้าทำงาน แต่ด้านข้างและล่างไม่ทำงาน

  {

    AForwardTime(3000);

    BForwardTime(3000);

  }

```



```

if((digitalRead(lefts)==LOW)||((digitalRead(rights)==LOW)&&(color == NOCOLOR))

// เบรก เมื่อ sensor ด้านข้างซ้ายหรือขวาทำงาน แต่ล่างไม่ทำงาน

{

ABreakTime(1000);

BBreakTime(1000);

}

if(color == BLACK) // กลับรถ เมื่อ sensor ด้านล่างตรวจจับเส้นสีดำได้

{

AForwardTime(5000);

BRewardTime(5000);

}

}

delay(20);

}

void AStop()

{

digitalWrite(AIN1, LOW); // motor stop

digitalWrite(AIN2, LOW);

}

```

```
void ABreak()

{

    digitalWrite(AIN1, HIGH); // motor break

    digitalWrite(AIN2, HIGH);

}

void BStop()

{

    digitalWrite(BIN3, LOW); // motor stop

    digitalWrite(BIN4, LOW);

}

void BBreak()

{

    digitalWrite(BIN3, HIGH); // motor break

    digitalWrite(BIN4, HIGH);

}

void AForward(int speed)

{

    digitalWrite(AIN2, LOW);

    analogWrite(AIN1, speed);

}
```

```
void BForward(int speed)
{
    digitalWrite(BIN4, LOW);
    analogWrite(BIN3, speed);
}
```

```
void AReward(int speed)
{
    digitalWrite(AIN1, LOW);
    analogWrite(AIN2, speed);
}
```

```
void BReward(int speed)
{
    digitalWrite(BIN3, LOW);
    analogWrite(BIN4, speed);
}
```

```
void ARewardTime(int time)
{
    digitalWrite(AIN1, LOW);
    analogWrite(AIN2, maxSpd);
    delay (time);  }
```

```
void BRewardTime(int time)
{
    digitalWrite(BIN3, LOW);

    analogWrite(BIN4, maxSpd);

    delay (time);
}

void AForwardTime(int time)
{
    digitalWrite(AIN2, LOW);

    analogWrite(AIN1, maxSpd);

    delay (time);
}

void BForwardTime(int time)
{
    digitalWrite(BIN4, LOW);

    analogWrite(BIN3, maxSpd);

    delay (time);
}
```

```
void ABreakTime(int time)
```

```
{
```

```
    digitalWrite(AIN1, HIGH);
```

```
    digitalWrite(AIN2, HIGH);
```

```
    delay (time);
```

```
}
```

```
void BBreakTime(int time)
```

```
{
```

```
    digitalWrite(BIN3, HIGH);
```

```
    digitalWrite(BIN4, HIGH);
```

```
    delay (time);
```

```
}
```

## เอกสารอ้างอิง

- [1] <http://krunisit.rwb.ac.th/robot.html>
- [2] <https://arduinothing.blogspot.com/2016/04/arduino-cc.html>
- [3] <https://th.wikipedia.org/wiki/%E0%B9%84%E0%B8%A1%E0%B9%82%E0%B8%84%E0%B8%A3%E0%B8%84%E0%B8%AD%E0%B8%99%E0%B9%82%E0%B8%97%E0%B8%A3%E0%B8%A5%E0%B9%80%E0%B8%A5%E0%B8%AD%E0%B8%A3%E0%B9%8C>
- [4] <https://th.aliexpress.com/item/33001783898.html>
- [5] <https://robotsiam.blogspot.com/2016/10/ir-infrared-obstacle-avoidance-sensor.html>
- [6] <https://www.ioxhop.com/article/69/esp32-%E0%B9%80%E0%B8%9A%E0%B8%B7%E0%B9%89%E0%B8%AD%E0%B8%87%E0%B8%95%E0%B9%89%E0%B8%99-%E0%B8%9A%E0%B8%97%E0%B8%97%E0%B8%B5%E0%B9%88-8-%E0%B8%81%E0%B8%B2%E0%B8%A3%E0%B9%83%E0%B8%8A%E0%B9%89%E0%B8%87%E0%B8%B2%E0%B8%99%E0%B9%80%E0%B8%8B%E0%B9%87%E0%B8%99%E0%B9%80%E0%B8%8B%E0%B8%AD%E0%B8%A3%E0%B9%8C%E0%B8%95%E0%B9%88%E0%B8%B2%E0%B8%87-%E0%B9%86>
- [7] <http://www.robotsiam.com/article/7/%E0%B8%81%E0%B8%B2%E0%B8%A3%E0%B9%83%E0%B8%8A%E0%B9%89-arduino-uno-r3-%E0%B8%81%E0%B8%B1%E0%B8%9A-l298n-%E0%B8%84%E0%B8%A7%E0%B8%9A%E0%B8%84%E0%B8%B8%E0%B8%A1%E0%B8%A1%E0%B8%AD%E0%B9%80%E0%B8%95%E0%B8%AD%E0%B8%A3%E0%B9%8C>
- [8] <https://www.ebay.co.uk/itm/Kit-2-5A-Dual-Channel-DC-Motor-Driver-Mini-Module-Beyond-L298N-PWM-Speed-Control-/322465517099>

[9] <https://www.arduitronics.com/product/1295/ir-reflective-obstacle-avoidance-line-tracking-sensor-tcrt5000>

[10] <http://www.ett.co.th/prod2018/DC-DC%20STEP%20UP%20A/DC-DC%20STEP%20UP%20A.html>