การศึกษาวิธีการปรับปรุงตำแหน่งเซนเซอร์ตรวจจับเส้นสำหรับหุ่นยนต์เดินตามเส้น

(A Study ofTracking Sensor Improvement for Follow Line Robot)

รหัสโครงงาน**CE-01/2556**

ณฐภัทร ลวดเงิน 51113018-9

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีไทย – ญี่ปุ่น

ปีการศึกษา 2556

หัวข้อปริญญานิพนธ์	การศึกษาวิธีการปรับปรุงตำแหน่งเซนเซอร์ตรวจจับเส้นสำหรับหุ่นยนต์เดินตามเส้น A Study ofTracking Sensor Improvement for Follow Line Robot รหัสโครงงาน CE-01/2556
โดย	นางสาวณฐภัทร ลวดเงิน51113018-9
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์	อาจารย์ประเวศน์ เอื้อตรงจิตต์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีไทย-ญี่ปุ่น อนุมัติให้นับปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์
(ผู้ช่วยศาสตราจารย์ ดร.เลอเกียรติ์ วงศ์สารพิกูล)

วัน.....เดือน.....พ.ศ.....

คณะกรรมการสอบปริญญานิพนธ์

..... ประธานกรรมการสอบ
(อาจารย์ดร. กรกฎ เหมสถาปัตย์)

..... กรรมการ
(อาจารย์วรวุฒิ จิตขจรวานิช)

..... กรรมการ
(อาจารย์อัดนา เซ็งโต๊ะ)

อาจารย์ที่ปรึกษาปริญญานิพนธ์

.....
(อาจารย์ประเวศน์ เอื้อตรงจิตต์)

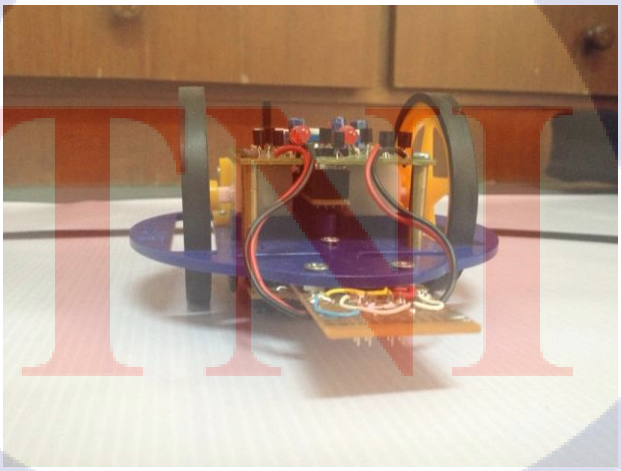
การศึกษาวิธีการปรับปรุงตำแหน่งเซนเซอร์ตรวจจับเส้นสำหรับหุ่นยนต์เดินตามเส้น

ณฐภัทร ลวดเงิน51113018-9
อาจารย์ที่ปรึกษา : อาจารย์ประเวศน์ เอื้อตรงจิตต์

ปัจจุบันมีการศึกษาค้นคว้าเกี่ยวกับหุ่นยนต์ที่ช่วยหรือทำงานแทนมนุษย์มากมาย มีทั้งปฏิบัติงานบนพื้นที่จำกัดไม่จำเป็นต้องเคลื่อนที่หรือสามารถเคลื่อนที่ไปปฏิบัติงานบนพื้นต่าง ๆ ซึ่งการเคลื่อนที่ของหุ่นยนต์ไปปฏิบัติงานบนพื้นที่เป้าหมายสามารถทำได้หลากหลายวิธี แต่วิธีที่ง่ายสุดคือการเคลื่อนที่ตามเส้นทางที่กำหนดไว้บนพื้น แต่การศึกษาหุ่นยนต์เดินตามเส้นที่ผ่านมาเน้นการพัฒนาโปรแกรมควบคุมการขับเคลื่อน ซึ่งการทำงานพื้นฐานถูกมองข้ามไป เช่น ตำแหน่งเซนเซอร์ตรวจจับเส้นทางต้องสัมพันธ์กับความเร็วในการเคลื่อนที่

โครงงานนี้เป็นการศึกษาตำแหน่งการติดตั้งเซนเซอร์ของหุ่นยนต์เคลื่อนที่ตามเส้นทาง เพื่อหาตำแหน่งที่เหมาะสมกับความเร็วในการเคลื่อนที่ไปตามเส้นทางที่กำหนดไว้บนพื้นโดยอาศัยหุ่นยนต์จำลองขนาดเล็กที่สามารถขับเคลื่อนไปตามเส้นทางจำลองได้ แบบไม่มีการควบคุมด้วยโปรแกรม และทดลองตามวิธีการ DOE (Design of Experiment) เพื่อทำให้การออกแบบหุ่นยนต์เดินตามเส้นมีประสิทธิภาพสูงสุด

ผลการทดลองพบว่าระยะรัศมีเซนเซอร์ตำแหน่ง 5.7cm. (ระยะกลาง) สามารถทำความได้เร็วดีสุด แต่หากพิจารณาความแปรปวนที่เกิดขึ้นจะพบว่า ระยะรัศมีเซนเซอร์ตำแหน่ง 8cm. (ระยะไกล) มีความแปรปวนน้อยกว่า ซึ่งหากพิจารณาประสิทธิภาพจะสรุปได้ว่า ความเร็วในการเคลื่อนที่ยิ่งสูง ควรจะมีระยะรัศมีเซนเซอร์ที่ไกลจากจุดกึ่งกลางของหุ่นยนต์



ภาพที่ 1ด้านหน้าของหุ่นยนต์

A Study ofTracking Sensor Improvement for Follow Line Robot

Nathaphat Laudngoren51113018-9

Academic Advisor:Prawet Euatrongchit

In present time, there have been numerous studies of human-assisted robots; involving stand-alone robots that are fixed at a station in orderto operate on a limited area or those of robots that are movable. Several studies have particularly been proposed for a control of movable robots, especially the studies of techniques to control robots to move under pre-determined routes. Despite the fact that robot control algorithms are extensively reported,one fundamental practice of well-correlated between the line detection sensors and moving speed has nevertheless studied.

This project therefore studies the correlation between the line detection sensors and moving speed of the movable line detection robot. This project alternatively employs a small movable robot as a prototype of an automated robot for the Design of Experiment (DOE) in order to achieve the maximum robot function efficiency.

The result reveals that the sensors operating at the distance of 5.7 cm. (middle distance) can provides the maximum speed. Considerations of data variance also expose that the sensors operating at 8 cm. (long distance) has smaller values of variances. In conclusion, a self-automated movable robot that is preferred to operate at higher speed should have a sensor distance that is far from the robot centroid.

กิตติกรรมประกาศ

งานวิจัยนี้เกิดจากความสนใจที่จะศึกษาวิทยาการด้านหุ่นยนต์และระบบเซนเซอร์ ข้าพเจ้าต้องขอขอบคุณอาจารย์ที่ปรึกษา อาจารย์ประเวศน์ เอื้อตรงจิตต์ เป็นอย่างสูงที่ได้แนะนำ และให้แนวทางในการทำงานวิจัยนี้ของข้าพเจ้าด้วยความตั้งใจจริง แม้ว่างานอาจจะล่าช้าไปบ้าง แต่ก็เข้าใจในตัวข้าพเข้า รวมทั้งทำงานหนักเพื่อให้งานวิจัยลุล่วงไปด้วยดี นอกจากนี้ยังมีอีกท่านที่ได้ช่วยดูงานวิจัยมาตั้งแต่ต้น อาจารย์วิมล แสนอุ้ม ที่ได้เสียสละเวลามาช่วยเหลือดูแลแม้ว่าท่านจะมีภารงานที่มากก็ตาม แม้ว่าจะไม่ได้เป็นอาจารย์ที่ปรึกษาจนจบโครงงาน แต่ก็ได้ให้แนวทางการทำวิจัยไว้เป็นโอกาสและประสบการณ์ที่ดีของข้าพเจ้า รวมทั้งให้คำแนะนำปรึกษาในด้านภาษาอังกฤษในส่วนบทคัดย่อภาษาอังกฤษอีกด้วย จึงขอขอบคุณท่านมา ณ ที่นี้

โครงงานฉบับนี้จะสำเร็จลุล่วงไปไม่ได้ถ้าปราศจากการสนับสนุนกำลังใจและการช่วยเหลือจากครอบครอบครัว น.ส.ปริญญ์ธนัช ภิญโญเทพประทาน น้องสาวของข้าพเจ้า คุณพ่อและคุณแม่ อีกทั้งอาจารย์กรกฎ เหมสถาปัตย์และคณะกรรมการ อาจารย์อัดนา เซ็งโต๊ะ อาจารย์วรวุฒิ จิตขรจรวนิช ที่ได้ตรวจรูปเล่ม ตลอดจน ข้อซักถามต่างๆ ที่ช่วยแนะนำให้ความเห็นและให้ความรู้ ขอกราบขอบพระคุณทุกท่านมา ณ ที่นี้เป็นอย่างสูง

ณฐภัทร ลวดเงิน

สารบัญ

สารบัญ(ต่อ)

หน้า

สารบัญตาราง

สารบัญภาพประกอบ

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ปัจจุบันมีการศึกษาค้นคว้าเกี่ยวกับหุ่นยนต์ที่ช่วยหรือทำงานแทนมนุษย์มากมาย มีทั้งปฏิบัติงานบนพื้นที่จำกัดไม่จำเป็นต้องเคลื่อนที่หรือสามารถเคลื่อนที่ไปปฏิบัติงานบนพื้นต่าง ๆ ซึ่งการเคลื่อนที่ของหุ่นยนต์ไปปฏิบัติงานบนพื้นที่เป้าหมายสามารถทำได้หลากหลายวิธี แต่วิธีที่ง่ายสุดคือการเคลื่อนที่ตามเส้นทางที่กำหนดไว้บนพื้นที่เรียกว่า หุ่นยนต์เดินตามเส้น ซึ่งมีการนำใช้ประกอบการเรียนรู้เกี่ยวหุ่นยนต์ในระดับพื้นฐานโดยการศึกษาหุ่นยนต์เดินตามเส้นที่ผ่านมาเน้นการพัฒนาด้านโปรแกรมควบคุมการขับเคลื่อน ซึ่งการทำงานพื้นฐานถูกมองข้ามไป เช่น ตำแหน่งเซนเซอร์ตรวจจับเส้นทางที่ต้องสัมพันธ์กับความเร็วในการเคลื่อนที่ปเป็นต้น จึงเป็นที่มาของแนวความคิดที่จะศึกษาตำแหน่งการติดตั้งเซนเซอร์ของหุ่นยนต์เคลื่อนที่ตามเส้น เพื่อช่วยให้การออกแบบหุ่นยนต์ประเภทนี้มีแนวทางที่ชัดเจน ซึ่งจะทำให้การพัฒนาหุ่นยนต์ที่มีประสิทธิภาพประสบผลสำเร็จมากยิ่งขึ้น

โครงงานนี้ เป็นการศึกษาตำแหน่งการติดตั้งเซนเซอร์ของหุ่นยนต์เคลื่อนที่ตามเส้น เพื่อหาตำแหน่งที่เหมาะสมกับความเร็วในการเคลื่อนที่ไปตามเส้นทางที่กำหนดไว้บนพื้น โดยอาศัยหุ่นยนต์จำลองขนาดเล็กที่สามารถขับเคลื่อนไปตามเส้นทางจำลอง แบบไม่มีการควบคุมที่ซับซ้อน และทำการทดลองตามวิธี DOE (Design of Experiment) เพื่อทำให้การออกแบบหุ่นยนต์เดินตามเส้นมีประสิทธิภาพสูงสุด

1.2 ความมุ่งหมายของการวิจัย

ในการวิจัยครั้งนี้ผู้วิจัยได้ตั้งความมุ่งหมายไว้ดังนี้

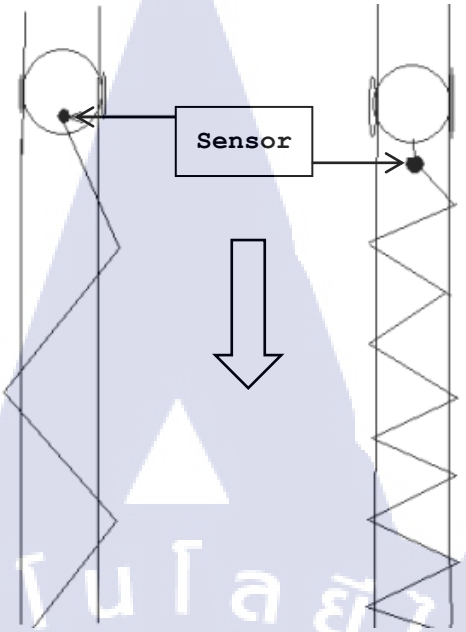
1.2.1 ศึกษาหุ่นยนต์เดินตามเส้นที่มีจำหน่ายในประเทศไทย

1.2.2 ทดสอบตำแหน่งเซนเซอร์เปรียบเทียบกับความเร็วของหุ่นยนต์

1.2.3 หาความเหมาะสมในการติดตั้งเซนเซอร์

1.3 สมมุติฐานของการวิจัย

กรณีที่เซนเซอร์อยู่ใกล้จุดหมุนจะทำให้ การตรวจจับเส้นทาง การเปลี่ยนแปลงช้ากว่าการติดเซนเซอร์ที่ห่างจากจุดหมุนมากกว่า ซึ่งจะทำให้เกิดลักษณะการเคลื่อนที่ดังภาพที่ 1.1



ภาพที่ 1.1 สมมุติฐานของการวิจัย

1.4 ขอบเขตการวิจัย

1.4.1 ศึกษาความสัมพันธ์ระหว่างตำแหน่งเซนเซอร์เปรียบเทียบกับความเร็วของหุ่นยนต์

1.4.2 เซนเซอร์ตรวจจับขอบนอกของเส้นทางเส้นทางจำนวน 2 จุด

1.4.3 ทดสอบตำแน่งเซนเซอร์ 3 ระยะ และความเร็ว 3 ระดับ

1.4.4 ลักษณะเส้นทางทดสอบเป็นสีดำและพื้นสนามเป็นสีขาว

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1.5.1 แนวทางการออกแบบหุ่นยนต์เดินตามเส้น

1.5.2 หลักพื้นฐานการวิจัยอย่างเป็นระบบ

1.6 สมมุติฐานในการวิจัย

1.6.1 วิเคราะห์การทำงานของหุ่นยนต์เดินตามเส้นที่มีจำหน่ายในประเทศไทย

1.6.2 ออกแบบการทดสอบตามหลักการ DOE

1.6.3 จัดเตรียมอุปกรณ์ทดสอบ

1.6.4 ทดสอบและสรุปผล

1.6.5 วิเคราะห์ความสัมพันธ์ระหว่างตำแหน่งเซนเซอร์กับความเร็วของหุ่นยนต์

1.7 แผนงานและระยะเวลาการดำเนินงาน

ตารางที่ 1.1 แผนงานและระยะเวลาการดำเนินงาน

ขั้นตอนการดำเนินงาน	พ.ศ.2556							พ.ศ.2557	
	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.
Plan.									
1. การเก็บข้อมูลผู้พัฒนาหุ่นยนต์เดินตามเส้น	■	■							
2. จัดกลุ่มหุ่นยนต์ และคัดเลือกกลุ่ม เพื่อการทดสอบ		■	■						
3. ค้นคว้าหาข้อมูล และทฤษฎีที่เกี่ยวข้อง			■	■					
4. ออกแบบวิธีการทดสอบ				■	■				
Do.									
5. สร้างหุ่นยนต์ เพื่อการทดสอบ					■	■	■		
6. ดำเนินการทดสอบ							■	■	
Check.									
7. ตรวจสอบความถูกต้อง และประเมินผล								■	
Act.									
8. วิเคราะห์ผลการทดสอบ และสรุป								■	■
9. จัดทำเอกสาร และนำเสนอผลงาน									■

บทที่ 2
หลักการพื้นฐาน

2.1 ทฤษฎีพื้นฐาน

การติดตั้งอุปกรณ์ตรวจวัดหรือตรวจจับแบบต่าง ๆ ขึ้นอยู่กับลักษณะการใช้งานเช่น การสำรวจพื้นผิวดาวเคราะห์ของหุ่นยนต์สำรวจที่ติดตั้งเซ็นเซอร์มากกว่า 100 รูปแบบ ซึ่งเผยให้เห็นข้อมูลทางวิทยาศาตร์ที่สามารถนำไปวิเคราะห์ข้อมูลเพื่อประกอบการตัดสินใจ หรือใช้ควบคุมการทำงานของหุ่นยนต์ เช่น การเคลื่อนที่ด้วยระบบขับเคลื่อนตามการวิเคราะห์เส้นทาง เป็นต้น

โครงงานนี้ต้องการอุปกรณ์ตรวจจับเส้นทางที่เป็นสีดำบนพื้นสีขาว จึงเลือกใช้เซนเซอร์ที่เกี่ยวกับแสงแบบสะท้อนกลับ(Reflector) โดยมีอุปกรณ์กำเนิดแสงและอุปกรณ์รับแสงที่ต้องทำงานร่วมกันเช่น แอลอีดี (LED : Light Emitting Diode) เป็นไดโอดเปร่งแสงที่สามารถมองเห็นได้ด้วยตาเปล่าทำงานร่วมกับแอลดีอาร์ (LDR: Light Dependent Resistor) ทำหน้าที่เปลี่ยนระดับความเข้มแสงให้กลายเป็นค่าความต้านทานทางไฟฟ้าเมื่อมีแสงตกกระทบปริมาณความเข้มแสงมากทำให้ค่าความต้านทานน้อยดังภาพที่ 2.1

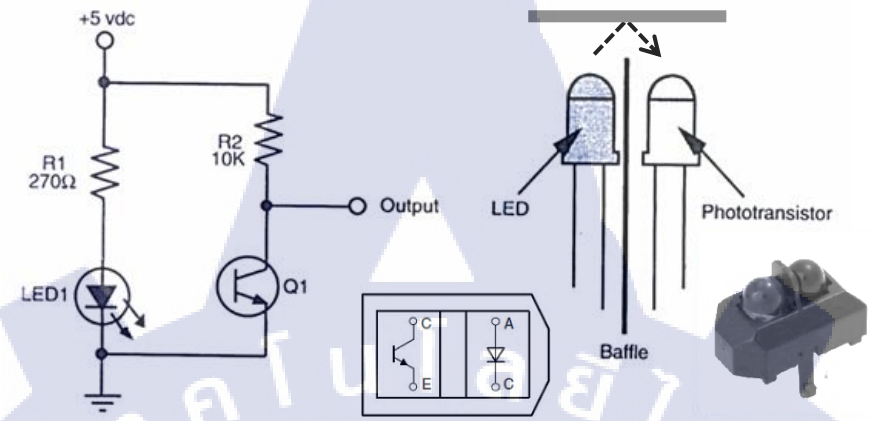
ภาพที่ 2.1 Reflector แบบ LED + LDR

2.2 เซนเซอร์ตรวจจับวัตถุด้วยแสงอินฟาร์เรด

อินฟราเรดเป็นแสงที่มีความยาวคลื่นอยู่ในช่วงประมาณ 780 - 3000 นาโนเมตร ซึ่งเป็นแสงที่ไม่สามารถมองเห็นได้ด้วยตาเปล่าเป็นที่นิยมนำมาใช้ในการสื่อสารหรือตรวจจับวัตถุต่าง ๆ เพราะปัญหาการรบกวนของสัญญาณจากแสงอื่นๆมีน้อยและการออกแบบวงจรระบบอินฟราเรดมีความซับซ้อนไม่มากนัก และความน่าเชื่อถือของสัญญาณสูงเหมาะกับการนำไปใช้งาน

ระบบอินฟราเรดจะต้องมีอุปกรณ์ส่งสัญญาณและอุปกรณ์รับสัญญาณ ซึ่งอุปกรณ์ส่งสัญญาณอินฟาร์เรดเป็น IR LED ที่เปร่งแสงออกมาในช่วงความถี่ที่สูงกว่าความถี่แสงธรรมดาคือมากกว่า 20 kHzส่วนอุปกรณ์รับสัญญาณอินฟาร์เรดเป็นโฟโต้ไดโอด หรือโฟโต้ทรานซิสเตอร์ ที่มี

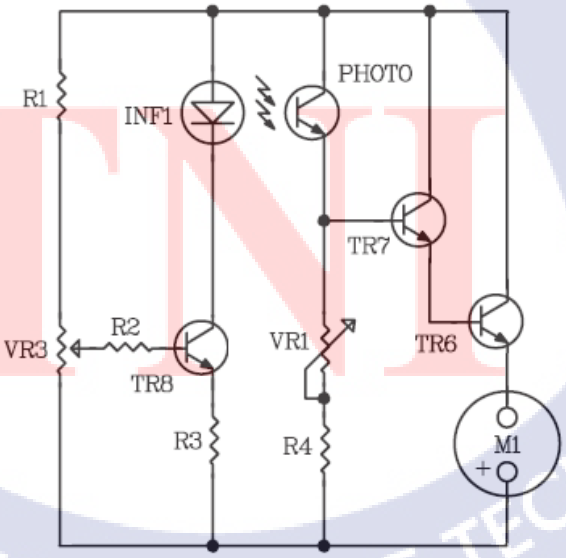
ความถี่เท่ากับอุปกรณ์ส่งสัญญาณ สำหรับโครงงานนี้จะใช้คุณสมบัติของตัวโฟโต้ทรานซิสเตอร์ดังภาพที่ 2.2 เมื่อมีแสงมาตกกระทบจะทำให้ปริมาณของกระแสที่วิ่งผ่านมีค่าเพิ่มขึ้นตามปริมาณความเข้มของแสงทำให้ Voltage ที่ตกคร่อมโฟโต้ทรานซิสเตอร์ มีค่าน้อยลงตามไปด้วย



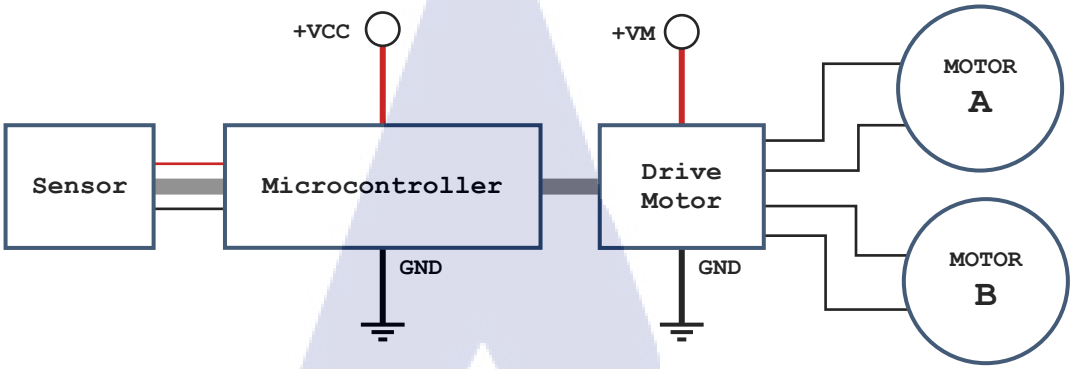
ภาพที่ 2.2 Reflector แบบ IR LED + Phototransistor

2.3 การวิเคราะห์และประมวลผลเซนเซอร์

จากการศึกษาหุ่นยนต์เดินตามเส้นที่มีจำหน่ายในประเทศไทยพบว่า การประมวลผลเซนเซอร์ เพื่อการควบคุมการเลื่อนที่หุ่นยนต์ให้เดินตามเส้นเส้นทางมี 2 ประเภทคือ การควบคุมด้วยวงจรอิเล็กทรอนิกส์อย่างง่าย และการควบคุมด้วยไมโครโปรเซสเซอร์ ดังรูปที่ 2.3 และ 2.4



ภาพที่ 2.3วงจรอิเล็กทรอนิกส์อย่างง่ายสำหรับควบคุมหุ่นยนต์เดินตามเส้น



ภาพที่ 2.4วงจรควบคุมหุ่นยนต์เดินตามเส้นด้วยไมโครคอนโทรลเลอร์

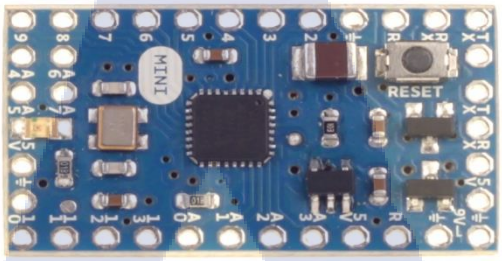
ไมโครคอนโทรลเลอร์มีให้เลือกใช้มากมายหลายค่ายและหลายรุ่นเช่น Microchip PIC, Philips ARM-Cortex, Atmel AVR, STMicroelectronicsSTM เป็นต้น ซึ่งไมโครคอนโทรลเลอร์เปรียบเหมือนกับสมองของมนุษย์คือมีหน้าที่คิดคำนวณทางคณิตศาสตร์ คำนวณทางลอจิก และสั่งการ โดยมีหน่วยความจำ เพื่อใช้เก็บข้อมูลในการคำนวณ หรือประมวลผลต่างๆและทำงานร่วมกับอุปกรณ์ประกอบควบคู่ (Accessories) อื่น ๆ เช่น เซนเซอร์ มอเตอร์และระบบแสดงผล เป็นต้น การใช้งานไมโครคอนโทรลเลอร์อาจเป็นเรื่องยุ่งยากในการออกแบบวงจรที่จำเป็นต้องมีส่วนเพิ่มเติมเช่น วงจรรีเซต วงจรควบคุมแรงดัน วงจรกำเนิดสัญญาณนาฬิกา จึงสามารถใช้งานไมโครคอนโทรลเลอร์ได้ แต่ปัจจุบันมีการพัฒนาชุดไมโครคอนโทรลเลอร์ที่พร้อมใช้งาน ราคาถูก และที่สำคัญมีซอฟต์แวร์สำหรับพัฒนาโปรแกรมเป็นแบบฟรี (Open Source) ที่เรียกว่า Arduino

2.3.1 Arduino

"อาดูยโน่"เป็นระบบที่ใช้ในการพัฒนาอุปกรณ์อิเล็กทรอนิกส์ต้นแบบ ซึ่งออกแบบมาให้ใช้งานง่ายทั้งฮาร์ดแวร์และซอฟต์แวร์ โดยมีกลุ่มเป้าหมายได้แก่นักประดิษฐ์นักออกแบบ ศิลปิน หรือใครๆก็ตามที่สนใจเพื่อใช้ในการประดิษฐ์นวัตกรรม และงานสร้างสรรค์

ภาพที่ 2.5สัญญลักษณ์ Arduino

Arduino ถูกพัฒนาโดยชาวอิตาเลียนที่ชื่อว่า Massimo Banzi และ David Cuartiellesใน ปี 2005 ที่เมือง Ivrea ทางตะวันตกเฉียงเหนือของประเทศอิตาลี และผลิตบอร์ด Arduinoเวอร์ชั่นแรก ออกจำหน่ายในปี 2006 เป็นรุ่นArduino Mini มีขนาดเล็กและราคาถูกเพียงไม่กี่ร้อยบาท



ภาพที่ 2.6บอร์ดArduinoMini

ปัจจุบัน Arduino มีบอร์ดหลายแบบให้เลือกใช้งานตามความถนัดและความเหมาะสม มากกว่า 20 รุ่น แต่ละรุ่นก็มีขนาดหน่วยความจำ ความเร็ว และจำนวนขาพอร์ตอินพุตเอาท์พุต แตกต่างกันออกไปซึ่งมีตั้งแต่ราคาหลักร้อยบาทไปจนถึงหลักพันบาท นอกจากนี้ยังมีอุปกรณ์ต่อ พ่วง (Shield) ที่หลากหลายให้เลือก โดยมีราคาก็เป็นไปตาม Concept เดิมคือ สมเหตุสมผลจึงทำ ให้เป็นที่นิยมมากขึ้นเรื่อยๆ และทำให้เกิดชุมชนคนใช้ Arduinoบนโลกไซเบอร์มากมายเช่น Arduino.cc, Makezine.com, instructables.com ซึ่งเป็นแหล่งความรู้ขนาดใหญ่ที่แจกแบบร่าง และไฟล์ติดตั้ง (Sketch) ให้ฟรี

2.3.2 การพัฒนาโปรแกรมบน Arduino

การโปรแกรมบอร์ดArduino มีส่วนประกอบอย่างง่ายๆแบ่งเป็นโปรแกรมย่อยหรือฟังชั่น ออกเป็นสองส่วนหลัก ๆ คือ void setup() และvoidloop()

```
void setup()
{
    statements;
}

void loop()
{
    statements;
}
```

โปรแกรมย่อยในส่วน setup() คือส่วนการเตรียมการเริ่มต้นใช้งาน ส่วน loop() เป็นส่วนการทำงานหลัก ซึ่งทั้งสองส่วนมีความสำคัญในการทำงานของบอร์ด โดยฟังชั่นในส่วน setup() ควรจะมีส่วนที่กำหนดค่าตัวแปรที่ส่วนหัวของโปรแกรม ซึ่งมันจะทำงานเพียงครั้งเดียวและทำงานก่อนโปรแกรมอื่นๆ ส่วนมากโปรแกรมในส่วนนี้จะใช้สำหรับการกำหนดหน้าที่ขาของตัว AVR ไมโครคอนโทรลเลอร์ หรือตั้งค่าการติดต่อแบบ serial กับอุปกรณ์อื่น ส่วนโปรแกรม loop จะทำงานต่อจากโปรแกรม setup และมันจะทำงานอย่างต่อเนื่องต่อไปเช่นอ่านค่า input ส่งสัญญาณ triggering outputs เป็นต้น โปรแกรมนี้เป็นโปรแกรมทำงานหลักของ Arduino และทำงานส่วนใหญ่ของระบบ

นอกจากฟังชั่น void setup() และ void loop() แล้วยังมีฟังชั่นอื่นๆที่พร้อมใช้งาน(build-in) ซึ่งฟังชั่นที่ทำงานเฉพาะสามารถเขียนขึ้นสำหรับใช้ทำงานที่ซ้ำกันเพื่อลดความยาวของคำสั่ง การสร้างฟังชั่นเริ่มต้นด้วยการกำหนดชนิดตัวแปรที่ฟังชั่นจะคืนกลับให้โปรแกรมที่เรียกใช้เช่น int สำหรับค่า integer ถ้าฟังชั่นไม่มีการคืนค่าจะใช้คำว่า void ต่อจากชนิดค่าที่ฟังชั่นคืนกลับจะเป็นชื่อฟังชั่นและตามด้วยวงเล็บซึ่งจะเป็นค่าพารามิเตอร์ที่จะส่งค่าให้กับฟังชั่น

```
type functionName(parameters)
{
      statements;
}
```

ตัวอย่างเช่น ค่า integer type function delayVal() ที่ใช้ตั้งค่าหน่วงเวลาในโปรแกรมโดยอ่านค่าของ potentiometer การทำงานของฟังชั่นจะกำหนดตัวแปร v และนำค่าของ potentiometer ซึ่งมีค่า 0-1023 มาเก็บไว้ใน v จากนั้นหารค่านั้นด้วย 4 เพื่อให้ได้ค่า 0-255 สุดท้ายจึงส่งค่ากลับไปยัง main program

```
int delayVal()
{
      int v;                  // create temporary variable 'v'
      v = analogRead(pot);    // read potentiometer value
      v /= 6;                 // converts 2"-"3245 to 2"-"477
      return v;
}
```

วงเล็บปีกกา{ }ใช้เป็นเครื่องหมายเพื่อแสดงจุดเริ่มต้นและจุดสิ้นสุดของกลุ่มคำสั่งของฟังชั่น และกลุ่มคำสั่งเช่น void loop() function และกลุ่มคำสั่ง for และ if statementวงเล็บปีกกาเปิด { จะต้องตามด้วยวงเล็บปีกกาปิด } เสมอ ในโปรแกรมอาจมีวงเล็บเปิดและปิดหลายๆอันแต่

จะต้องมีจำนวนเท่ากัน ถ้าวงเล็บเปิดปิดไม่เท่ากันจะทำให้เกิด compiler error และบางครั้งทำให้หาข้อผิดพลาดได้ยากในโปรแกรมที่มีความซับซ้อน

สำหรับ โปรแกรม Arduino environment จะมีความสามารถในการตรวจความสมดุลย์ของวงเล็บปีกกา โดย คลิกที่ วงเล็บปีกกา หรือคลิกด้านข้างขวาของวงเล็บปีกกาที่ต้องการตรวจสอบ โปรแกรมจะแสดงวงเล็บปีกกาที่เป็นคู่ของมันทันที

เครื่องหมายเซมิโคลอน ; (SEMICOLON)จะต้องใช้ที่ส่วนท้ายของคำสั่งและใช้แยกส่วนของโปรแกรมออกจากกัน นอกจากนี้เซมิโคลอนยังใช้แยกส่วนของ for loop ออกเป็นส่วนๆด้วย ถ้าลืมใส่เซมิโคลอนที่ท้ายบรรทัดจะทำให้ compiler error ซึ่งการ error อาจมาจากหลายสาเหตุแต่การลืมใส่เซมิโคลอนก็เป็นสาเหตุหนึ่ง ให้ตรวจสอบบรรทัดที่ไกล้จุดที่ compiler error ว่าใส่เซมิโคลอนหรือไม่

การเขียนคำอธิบายโปรแกรม สามารถใส่ไว้ในเครื่องหมาย /*...คำอธิบาย...*/ (BLOCK COMMENTS)คำอธิบายที่อยู่ในเครื่องหมาย/*...*/ นี้ตัว compiler จะไม่นำมาประมวลผล ซึ่งจะมีประโยชน์ในการอธิบายความมุ่งหมายของคำสั่ง เพื่อให้เข้าใจการทำงานแต่ละส่วนของโปรแกรม ซึ่งจะเริ่มด้วยเครื่องหมาย /* และจบด้วยเครื่องหมาย */ และสามารถเขียน comment ได้หลายๆ บรรทัดถ้าต้องการเนื่องจากการเขียน คำอธิบายไม่ได้เอาไปใช้ในโปรแกรมทำให้ไม่สิ้นเปลืองหน่วยความจำโปรแกรม ดังนั้นควรเขียนคำอธิบายให้ชัดเจนและทำให้เป็นนิสัย อีกอย่างที่เครื่องหมาย comment มีประโยชน์คือใช้ตัดส่วนของโปรแกรมที่ไม่ต้องการออกไปสำหรับการตรวจสอบความผิดพลาดของโปรแกรม

การเขียนคำอธิบายโปรแกรมบรรทัดเดียวใช้เครื่องหมาย // เป็นเครื่องหมายเริ่มต้นและสิ้นสุดด้วยการขึ้นบรรทัดใหม่ ทำให้ไม่เปลืองเนื้อที่โปรแกรมใช้งาน เนื่องจาก compiler จะไม่นำไปใช้ในโปรแกรมเช่นเดียวกับ block comment

ตัวแปรหรือ variable เป็นชื่อและที่เก็บค่าของตัวเลขสำหรับใช้งานในโปรแกรม ตัวแปรมีค่าที่เปลี่ยนแปลงได้ซึ่งตรงกันข้ามกับ constant หรือค่าคงที่ ที่ไม่สามารถเปลี่ยนแปลงค่าได้ การใช้งานตัวแปรจะต้องมีการกำหนดชนิดและอาจกำหนดค่าเริ่มต้นให้กับตัวแปรนั้น การกำหนดชื่อตัวแปรควรจะสื่อถึงหน้าที่ใช้งานเพื่อให้โปรแกรมสามารถอ่านได้ง่ายเช่น tiltSensor หรือ pushButton จะช่วยให้โปรแกรมเมอร์หรือคนอื่นที่อ่านโปรแกรมสามารถเข้าใจได้ว่าตัวแปรนั้นเอาไว้ทำอะไร คุณสามารถตั้งชื่ออะไรก็ได้ที่ไม่ตรงกับชื่อเฉพาะที่ใช้ใน Arduino language

ตัวแปรที่นำไปใช้งานได้จะต้องถูกตั้งค่าตัวแปรก่อนนำไปใช้ การตั้งค่าตัวแปรหมายถึงการระบุชนิดตัวแปร ให้เป็น int, long, float, เป็นต้น การกำหนดชื่อให้ตัวแปรมีชื่อเรียก และอาจมีการกำหนดค่าเริ่มต้นให้กับตัวแปร ซึ่งการตั้งค่านี้จะทำเพียงครั้งเดียวในโปรแกรม แต่ค่าของตัวแปรจะเปลี่ยนแปลงไปตามการทำงานของโปรแกรม

ตัวแปรสามารถกำหนดค่าตอนเริ่มต้นโปรแกรมก่อนฟังชั่นvoid setup() หรือกำหนดค่าตัวแปรภายในฟังชั่นและบางครั้งก็กำหนดค่าตัวแปรภายในกลุ่มคำสั่ง for loop ซึ่งการกำหนดค่าตัวแปรในแบบต่างๆ มีผลถึงขอบเขตการใช้ตัวแปรเช่น

- ตัวแปรแบบ global เป็นตัวแปรที่โปรแกรมมองเห็นและใช้งานได้จากทุกฟังชั่นและทุกกลุ่มคำสั่งในโปรแกรม ตัวแปรนี้ถูกกำหนดค่าที่ตอนเริ่มต้นโปรแกรมก่อน setup() function
- ตัวแปรแบบ local เป็นตัวแปรที่กำหนดค่าภายในฟังชั่นหรือกลุ่มคำสั่ง for loop ซึ่งสามารถมองเห็นและใช้งานได้เฉพาะภายในฟังชั่นที่กำหนดค่านั้น ๆ โดยชื่อตัวแปรแบบนี้อาจซ้ำกันได้ในฟังชั่นที่แตกต่างกัน แต่ในการทำงานฟังชั่นแต่ละกลุ่มจะใช้ตัวแปรตัวนั้นเฉพาะที่กำหนดค่าภายในเท่านั้น

```
int value;                    // 'value' is visible to any function

void setup()
{
     11"no setup needded
}

void loop()
{
     for (int i=2; i<42;)     // 'i is only visible inside the for - loop
     {
          i++;
     }
     float f;                 // 'f' is only visible inside loop
}
```

ชนิดของตัวแปรสามารถกำหนดได้ทั้ง

- byteมีค่า 0 – 255 (8 bit)
- int มีค่าระหว่าง 32,767 ถึง -32,768 (16 bit)
- longมีค่าระหว่าง 2,147,483,647 ถึง -2,147,483,648 (32 bit)
- float มีค่าระหว่าง 3.4028235E+38 ถึง -3.4028235E+38 (32 bit)

```
byte someVariable = 180;    // declares 'someVariable' as a byte type
int someVariable = 1500;    // declares 'someVariable' as an integer type
long someVariable = 90000; // declares 'someVariable' as a long type
float Variable = 3.14;      // declares 'Variable' as a floatint -point type
```

- arrays ป็นตัวแปรหลายมิติสามารถเข้าถึงได้ด้วยค่าตัวชี้หรือ index ค่าตัวแปรใน array อาจเรียกใช้ด้วยการระบุชื่อ array และระบุตัวชี้ index number โดยตัวแปร array จะมี index เริ่มต้นจาก 0 และต้องกำหนดขนาดก่อนจะนำไปใช้งาน หรืออาจกำหนดค่าเริ่มต้น

```
int myArray[] = { value2, value3, value4000}

int myArray[5];             // declares integer array w/ 6 positions
myArray[ 3]  = 10;          // assigns the 4th index the value 10
x = myArray[ 3];            // x now equals 10
```

ตัวแปร arrays มักใช้บ่อยใน for loops ซึ่งจะมีการเพิ่มค่าตัวนับ และใช้ตัวนับเป็นตัวชี้ตำแหน่งของ arrays แต่ละตัว ดังตัวอย่างต่อไปนี้

```
int ledPin = 32;                    // LED on pin 32

byte flicker[ ] = { 3:2, 52, 477, 422, 32, ;2, 372, 82};
                                    // array of :"different values

void setup()
{
    pinMode(ledPin, OUTPUT);    // sets OUTPUT pin
}

void loop()
{
    for(int i=2; i<9; i++)      // loop equals number of values in array
    {
        analogWrite(ledPin, flicker[ i]); // write index value
        delay(422+;
    }
}
```

จากโปรแกรมเป็นการใช้ array ในการสั่ง LED ติดดับ ด้วยกระบวนการfor loop โดยใช้ตัวนับจะเริ่มจาก 0 เขียนค่าที่ได้จาก array ตำแหน่งที่ 0 ของ array flicker[] ซึ่งในที่นี้คือค่า 180 ไปยัง PWM pin 10 และหยุดเป็นเวลา 200 ms แล้วจึงย้ายไปเอาค่าในตำแหน่งถัดไปของ array

ตัวกระทำการทางคณิตศาสตร์ (arithmetic) หมายรวมถึง การบวกการลบการคูณและการหาร ซึ่งจะให้ค่า ผลรวมผลต่างผลคูณหรือผลหาร ของสองตัวดำเนินการเช่น

```
y = y + 5;          x = x - 9;
i = j * 8;          r = r / 7;
```

การทำงานของโปรแกรมจะเป็นการนำชนิดข้อมูลตัวแปรของตัวดำเนินการมาใช้ดังเช่น กรณี9/4 ผลลัพท์จะเท่ากับ 2 แทนที่จะเท่ากับ 2.25 เนื่องจาก 9 และ 4 เป็นตัวแปรจำนวนเต็ม หรือ integer ซึ่งไม่สามารถเก็บค่าทศนิยมได้ และถ้าผลลัพท์ที่ได้มีค่ามากกว่าที่ตัวแปรจะรับได้จะเกิดการ overflowและถ้ามีการคำนวนกับชนิดตัวแปรที่ต่างกันตัวแปรที่มีขนาดใหญ่กว่าจะถูกนำมาใช้เช่นการบวก ลบเลขจำนวนเต็มกับเลขทศนิยม การบวกลบทศนิยมจะถูกนำมาใช้ จึงควรเลือกตัวแปรที่มีขนาดที่ใหญ่พอที่จะเก็บผลลัพท์การคำนวน ให้ตรวจสอบตัวแปรที่มีค่าย้อนกลับหรือ roll over เช่น 0 - 1 หรือ 0 - 32768 สำหรับการคำนวนที่เป็นเลขเศษส่วน ให้ใช้ตัวแปรที่มีจุดทศนิยม แต่ควรระวังผลข้างเคียงถ้าตัวเลขมากจะทำให้ความเร็วในการคำนวนช้าลง

การเปลี่ยนชนิดตัวแปรจากชนิดหนึ่งเป็นอีกชนิดหนึ่งเรียกว่าการ cast ตัวแปรโดยใช้ (int)myFloat เช่น i = (int)3.6 จะเป็นการแปลงค่าให้ i มีค่าเป็น 3

การเขียนคำสั่งโปรแกรมแบบย่อ (compound assignment) หมายถึงการทำงานกับตัวแปรแล้วนำผลลัพท์ที่ได้ไปเก็บในตัวแปรนั้น จะพบมากใน for loop การเขียนคำสั่งแบบย่อทั่วไป เช่น

```
x ++        // same as x = x + 3, or increments x by +3
x --        // same as x = x - 3, or decrements x by -3
x += y      // same as x = x + y, or increments x by +y
x -= y      // same as x = x - y, or decrements x by -y
x *= y      // same as x = x * y, or multiply x by y
x /= y      // same as x = x / y, or devides x by y
```

การเปรียบเทียบตัวแปรหรือค่าคงที่กับค่าอื่นๆ (comparison operator) จะใช้บ่อยในกลุ่มคำสั่ง if เพื่อตรวจค่าว่ามีสถานะตรงตามที่ต้องการหรือไม่ ตัวอย่างต่อไปนี้เป็นคำสั่งเปรียบเทียบตัวแปรสองตัว

```
x == y      // x is equal to y ??
x!= y       // x is not equal to y ??
x < y       // x is less than y ??
x > y       // x is greater than y ??
x <= y      // x is less than or equal to y ??
x >= y      // x is greater than or equal to y ??
```

การตรวจสอบทาง logic (logical operators) จะใช้เปรียบเทียบตัวดำเนินการสองตัวและให้ผลลัพท์ จริง หรือ เท็จ(TRUE OR FALSE) ขึ้นอยู่กับตัวดำเนินการ ตัวดำเนินการทางลอจิกมี 3 ตัวคือ AND, OR และ NOT ซึ่งใช้บ่อยใน กลุ่มคำสั่ง if ดังตัวอย่างต่อไปนี้

```
if (x > 0 && x < 5)       // true only if both expression are true
                          // เป็นจริงเมื่อทั้งสองส่วนเป็นจริง

if (x > 0 || y < 5)       // true only if either expression are true
                          // เป็นจริงเมื่อตัวใดตัวหนึ่งเป็นจริง
```

ในภาษา Arduino มีค่าที่กำหนดไว้แล้วไม่มาก เราเรียกค่าพวกนี้ว่า constant หรือค่าคงที่ ซึ่งมีไว้เพื่อให้โปรแกรมอ่านง่าย ค่าคงที่เหล่านี้จะแบ่งออกได้หลายกลุ่มเช่น

- true/falseซึ่งค่าเหล่านี้เป็นค่าคงที่ boolean ซึ่งบอกสถานะระดับลอจิก FALSE หมายถึง 0(ศูนย์) ในขณะที่ TRUE จะหมายถึง 1 หรืออะไรก็ได้ที่ไม่ใช่ ศูนย์ ดังนั้นในทางลอจิกแล้ว -1, 2, -200 จะหมายถึง TRUE

```
if (b == TRUE){ doSomething;}
```

- high/lowซึ่งค่าคงที่เหล่านี้แสดงถึงระดับลอจิกที่ขาไอซีว่าเป็น HIGH หรือ LOW และใช้เมื่อมีการอ่านหรือเขียนไปที่ขาไอซี HIGH จะแทนระดับลอจิก 1 หรือ ON หรือ 5 volts ในขณะที่ LOW คือระดับลอจิก 0 หรือ OFF หรือ 0 volts

```
digitalWrite(35, HIGH);
```

- input/output ซึ่งค่าคงที่ในฟังชั่น pinMode() ในการกำหนดหน้าที่ของ digital pin ว่าจะให้เป็น INPUT หรือ OUTPUT

```
pinMode(13, OUTPUT);
```

คำสั่ง if จะใช้ตรวจหาสถานะที่ต้องการว่าเกิดขึ้นหรือยังเช่นค่า analog ที่อ่านได้มีค่าเกินจำนวนที่กำหนดไว้หรือยัง และทำงานตามคำสั่งในวงเล็บปีกกาถ้าสถานะเป็นจริง(TRUE) แต่ถ้าสถานะเป็นเท็จ(FALSE)โปรแกรมจะข้ามกลุ่มคำสั่งนั้นไป ดังตัวอย่าง

```
if (someVariable ?? value)
{
     doSomething;
}
```

จากตัวอย่างข้างบนเป็นการเปรียบเทียบ someVariable กับค่าอื่นซึ่งอาจเป็นตัวแปรหรือค่าคงที่ ถ้าผลลัพท์เป็นจริง คำสั่งในวงเล็บปีกกาจะถูกทำงาน แต่ถ้าไม่ โปรแกรมจะข้ามไปทำงานต่อที่หลังวงเล็บปีกกาปิด ควรระมัดระวังการใช้ '=' ในรูปคำสั่ง if (x=10) คำสั่งนี้ถูกต้องตามรูปแบบคือนำค่า 10 ไปใส่ในตัวแปร x และจะทำให้ผลลัพท์เป็นจริงตลอด อาจทำให้เกิดผลที่ไม่คาดคิด ดังนั้นควรใช้ '==' เช่น if (x==10) ซึ่งเพียงแต่ตรวจค่า x ว่ามีค่าเท่ากับ 10 หรือไม่ เท่านั้น

คำสั่ง if... elseเป็นการเพิ่มทางเลือก 'ถ้าไม่ใช่สถานะที่ต้องการ' ให้มีทางเลือกอีกทาง ตัวอย่างเช่น ถ้าคุณต้องการตรวจสอบ digital input และทำงานอย่างหนึ่งถ้าอินพุทเป็น HIGH หรือทำงานอีกอย่างถ้าอินพุทเป็น LOW คุณจะเขียนคำสั่งในลักษณะนี้

```
if (inputPin == HIGH)
{
     doThingA;
}
else
{
     doThingB;
}
```

คำสั่งelse สามารถขยายเงื่อนไขต่อเนื่องออกไปได้อีก จึงทำให้สามารถทดสอบเงื่อนไขได้หลายเงื่อนไขในเวลาเดียวกัน ซึ่งเงื่อนไขทั้งหมดจะมีกลุ่มคำสั่งเดียวที่ทำงานเช่น

```
if (inputPin < 722+
{
     doThingA;
}
else if (inputPin >= 3222+
{
     doThingB;
}
else
{
     doThingC;
}
```

คำสั่ง for จะใช้เพื่อทำกลุ่มคำสั่งในวงเล็บปีกกาหลายๆครั้งเป็นจำนวนที่ต้องการ ปกติจะใช้ตัวนับจำนวนเพิ่มจำนวนครั้งที่ทำงานจนถึงจำนวนที่ต้องการแล้วจึงออกจาก loop เงื่อนไขของการทำงานใน loop จะกำหนดโดยคำสั่งที่ส่วนหัวของ loop ซึ่งมีสามส่วนแยกกันด้วยเซมิโคลอน เช่น

```
for (int i=2; i <42; i++)      // declares i, test if less
                                  than 42, increment i by 3
{
     degitalWrite(35, HIGH);   // turn pin 35 on
     delay(472+;               // pauses for 316 second
     degitalWrite(35, LOW);    //turn pin 35 off
     delay(472+;               // pauses for 316 second
}
```

คำสั่ง while loops จะทำงานอย่างต่อเนื่องตลอดไป จนกระทั่งเงื่อนไขในวงเล็บจะเป็นเท็จ ดังนั้นจะต้องมีเหตุการณ์บางอย่างมาเปลี่ยนแปลงค่าตัวแปรที่ใช้ทดสอบ หรือการทำงานใน while loop อาจทำงานไปตลอดโดยไม่มีการออกจาก loop ขึ้นอยู่กับคำสั่งที่คุณเขียน เช่นเพิ่มค่าตัวแปรหรือทดสอบสถานะภายนอกเช่นตรวจสอบ sensor เป็นต้น

```
while (someVariable ?? value) // test if less than 422
{
     doSomething;         // executes enclosed statements
     someVariable++;      // increments variable by 3
}
```

คำสั่ง do loop จะทำงานและทดสอบสถานะการทำงานที่ด้านท้ายของ loop คล้ายกับการทำงานของ while loop ยกเว้นการตรวจสอบสถานะเพื่อออกจาก loop จะทำที่ส่วนท้ายของ loop ดังนั้น do loop จะทำงานอย่างน้อยหนึ่งครั้งเสมอ

```
do
{
      doSomething;
} while (someVariable ?? value);
```

คำสั่ง pinMode(pin,mode) ใช้ในกลุ่ม void setup() เพื่อกำหนดหน้าที่ขาของไมโครคอนโทรลเลอร์ให้เป็นขารับสัญญาน INPUT หรือขาส่งสัญญาน OUTPUT เช่น

```
pingMode(pin,OUTPUT);       // sets 'pin' to output
```

ปกติขาของ Arduino digital ถูกตั้งให้เป็นขารับอินพุทโดยปริยาย ดังนั้นจึงไม่จำเป็นต้องสั่งให้มันทำหน้าที่เป็นอินพุทด้วยคำสั่ง pinMode() ขาที่ถูกตั้งให้เป็นอินพุทจะมีคุณสมบัติเป็นขาที่มีอิมพีแดนท์สูงนอกจากนี้ยังมีความต้านทานพูลอัพ 20K ที่พร้อมใช้งานในตัว Atmega chip ซึ่งสามารถสั่งให้ทำงานได้ด้วยซอฟท์แวร์ โดยคำสั่งดังนี้

```
pinMode(pin,INPUT);         // set 'pin' to input
digitalWrite(pin,HIGH);     // turn on pullup resistors
```

ความต้านทานพูลอัพปกติจะใช้ต่อกับอุปกรณ์อินพุทเช่นสวิทซ์ ตัวอย่างข้างบนไม่ได้ทำให้ขาเปลี่ยนเป็นขา output แต่เป็นวิธีตั้งให้ความต้านทางพูลอัพทำงานเท่านั้นถ้ากำหนดให้เป็น OUTPUT เรียกได้ว่าเป็นขาที่ทำให้มีอิมพีแดนท์ต่ำและสามารถจ่ายกระแสได้ 40 มิลลิแอมป์ให้อุปกรณ์อื่น ซึ่งเพียงพอที่จะขับ LED ให้ติด(อย่าลืมใส่ความต้านทานจำกัดกระแสด้วย) อย่างไรก็ตามมันไม่สามารถขับรีเลย์โซลีนอยด์หรือมอเตอร์ได้โดยตรง ซึ่งการลัดวงจรบนขาของ Arduino และการจ่ายกระแสมากเกินไปอาจทำความเสียหายให้กับขา output หรืออาจทำให้ตัวไอซีเสียหาย ดังนั้นควรต่อความต้านทานค่า 470 โอห์มหรือ 1k โอห์มอนุกรมกับอุปกรณ์ภายนอก

คำสั่งdigitalRead(pin) ใช้อ่านค่าจากขาไอซีที่ถูกกำหนดเป็น digital pin ซึ่งให้ผลลัพท์เป็น HIGH หรือ LOW หมายเลขขาไอซีอาจกำหนดเป็นตัวแปรหรือค่าคงที่ (0-13)

```
value = digitalRead(pin); // sets 'value' equal to the input pin
```

คำสั่งdigitalWrite(pin, value) ใช้ส่งค่าลอจิก HIGH หรือ LOW (เปิด หรือ ปิด) ไปยังขา digital ที่กำหนดหมายเลขขาไอซีอาจกำหนดเป็นตัวแปรหรือค่าคงที่ (0-13)

```
digitalWrite(pin,HIGH);    // sets 'pin' to high
```

ตัวอย่างการอ่านค่าจากปุ่มกดที่ต่อกับ digital input และเปิด LED ที่ต่อกับ digital output เมื่อปุ่มถูกกด

```
int led = 35;                // connect LED to pin 35
int pin = 9;                 // connect pushbutton to pin 9
int value = 2;               // variable to store the read value

void setup()
{
     pinMode(led,OUTPUT);       // sets pin 35 as output
     pinMode(pin,INPUT);        // sets pin 9 as input
}

void loop()
{
     value = digitalRead(pin);    // sets 'vaue' equal to the input pin
     digitalWrite(led,value);     // sets 'led' to the button's value

}
```

คำสั่งanalogRead(pin) ใช้อ่านค่าจากขา Analog จะได้ค่าที่มีความละเอียด10bit โดยไม่ต้องกำหนดตอนเริ่มต้นว่าเป็น INPUT หรือ OUTPUT ต่างกับขา digital คำสั่งนี้จะทำงานกับขา analog input (0-5) เท่านั้น และได้ผลลัพท์เป็นเลขจำนวนเต็มค่า 0-1023

```
value = analogRead(pin);  // sets 'value' equal to 'pin'
```

คำสั่งanalogWrite(pin, value) ใช้เขียนค่า analog เทียมโดยใช้ hardware enabled pulse width modulation (PWM) ไปยังขา output ที่สามารถทำ PWM ได้ ใน Arduino รุ่นใหม่ที่ใช้ชิพ Atmega168 คำสั่งนี้จะทำงานกับขา 3, 5, 6, 9, 10, และ 11 ส่วน Arduino รุ่นเก่าที่จะรองรับเพียงขา 9, 10 และ 11 ค่าที่เขียนสามารถใช้เป็นตัวแปรหรือค่าคงที่จาก 0-255

```
anllogWrite(pin,value);    // writes 'value' to analog 'pin'
```

ค่า 0 ที่เขียนไปยังขา analog จะทำให้เกิด 0 โวลท์ที่ขานั้น ส่วนค่า 255 จะทำให้เกิด 5 โวลท์ที่ขานั้นเช่นกัน สำหรับค่าระหว่าง 0-255 อยู่ระหว่าง 0 และ 5 โวลท์ ค่ายิ่งมากแรงดันที่

ปรากฏก็ยิ่งเข้าไกล้ HIGH(5 โวลท์) ตัวอย่างเช่น ค่า 64 จะทำให้ได้โวลท์เป็น 0 ประมาณสามในสี่ของทั้งหมด และที่เหลือเป็น 5 โวลท์หรือหนึ่งในสี่ของทั้งหมด คิดเป็น 25% หรือเท่ากับ 1.25 โวลท์โดยประมาณ ถ้าค่าที่เขียนคือ 128 จะทำสัญญาณรูปสี่เหลี่ยมที่มี Duty cycle 50% หรือเท่ากับ 2.5 โวลท์เนื่องจากคำสั่งนี้เป็นคำสั่งให้ hardware ในตัวไอซีทำงาน ขาไอซีจะผลิตคลื่นที่มีคาบเวลาตามค่าที่เขียนอย่างคงที่หลังจากทำคำสั่ง analogWrite ไปเรื่อยๆ จนกระทั่งมีการเรียกใช้คำสั่ง analogWrite ครั้งใหม่ (หรือการเรียกใช้คำสั่ง digitalRead หรือ digitalWrite บนขาไอซีขาเดียวกัน)ตัวอย่างต่อไปนี้เป็นการอ่านค่า analog จากขา analog input pin เปลี่ยนค่าที่อ่านได้ด้วยการหาร 4 และส่งค่าที่ได้เป็นค่า PWM ออกทางขา PWM

```
int led = 32;                   // LED with 442 resistor on pin 32
int pin = 2;                    // potentiometer on analog pin 2
int value;                      // value for reading

void setup(){}                  // no setup needed

void loop()
{
     value = analogRead(pin);   // sets 'value' equal to 'pin'
     value /= 6;                // converts 2"-"3245 to 2"-"477
     analogWrite(led,value);    // outputs PWM signal to led
}
```

คำสั่งdelay(ms) ใช้หยุดการทำงานของโปรแกรมเป็นเวลาตามที่กำหนด มีหน่วยเป็นมิลลิวินาที ซึ่ง 1000 มิลลิวินาทีเท่ากับ 1 วินาที

คำสั่งmillis()จะได้ผลลัพท์ค่าเวลาเป็นมิลลิวินาทีแสดงค่าที่ Arduino board เริ่มต้นทำโปรแกรมจนถึงปัจจุบัน ซึ่งค่าที่ได้เป็นค่า unsigned long ขนาด 32 bitถ้า run โปรแกรมอย่างต่อเนื่องตัวเลขจำนวนนี้จะมีค่าย้อนกลับเป็น0 หลังจากเวลาผ่านไปประมาณ 9 ชั่วโมง

คำสั่งSerial.begin(rate)ใช้เปิดพอร์ท serial และตั้งค่าความเร็วในการรับส่ง โดยปกติการติดต่อกับเครื่องพีซีจะตั้งไว้ที่ความเร็ว 9600 และมันสามารถกำหนดความเร็วได้มากกว่านี้

```
void setup()
{
     Serial.begin(9600);    // opens serial port set data rate to 9600 bps
}
```

เมื่อใช้พอร์ทสำหรับการติดต่อ serial communication ขา digital pin 0 (RX) และ digital pin 1 (TX) จะนำไปใช้งานอย่างอื่นไม่ได้

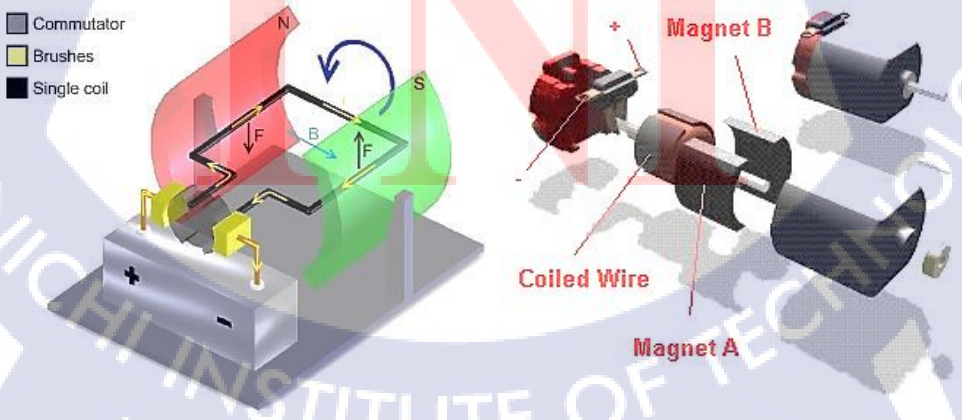
คำสั่งSerial.println(data)จะส่งข้อมูลให้กับ serial port ตามด้วยคำสั่งควบคุมให้ขึ้นบรรทัดใหม่โดยอัตโนมัติ คำสั่งนี้จะมีผลเหมือนกับคำสั่ง Serial.print() ซึ่งสามารถดูข้อมูลได้ที่ Serial monitorตัวอย่างต่อไปนี้เป็นคำสั่งอ่านค่าจากขา analog pin 0 และส่งข้อมูลให้กับเครื่องคอมพิวเตอร์ทุกๆ 1 วินาที

```
viod setup()
{
      Seral.begin(;822+;                  // sets serial to ;822 bps
}

void loop()
{
      Seral.println(analogRead(2++;   // sends analog value
      delay(3222+;                        // pauses for 3 second
}
```

นอกจากนี้ยังมีคำสั่ง max(x,y), min(x,y), random(max), random(min,max), randomSeed(seed)เป็นต้น สามารถหาข้อมูลเพิ่มเติมได้ที่เวบไซด์ Aruduino

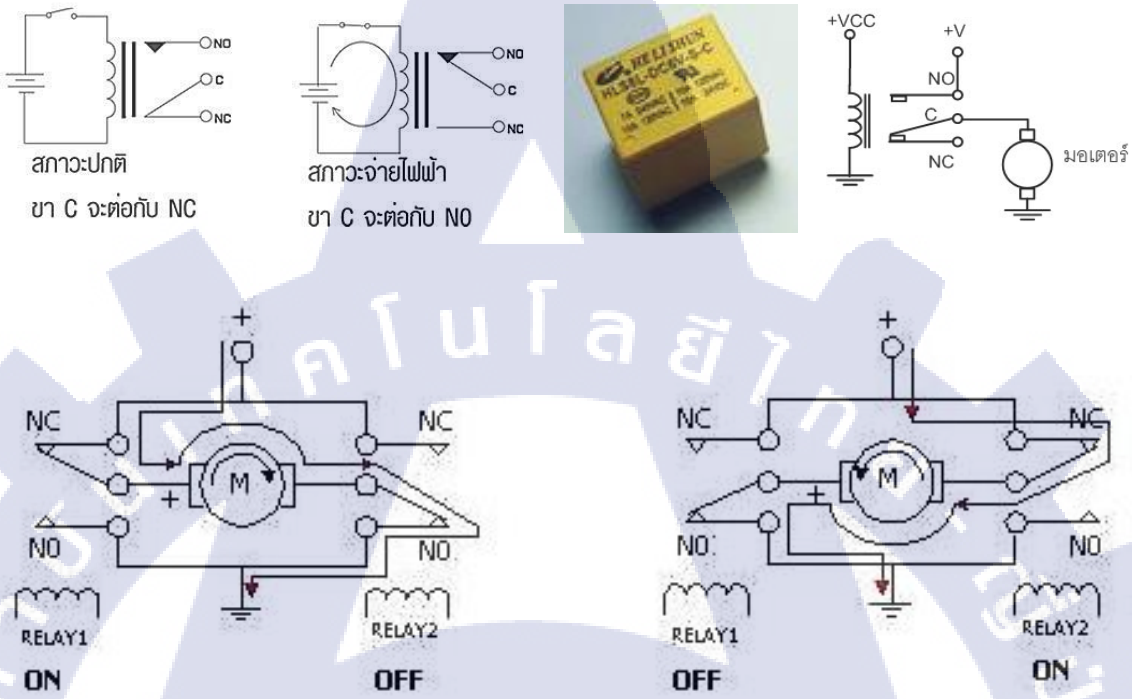
2.4 ระบบขับเคลื่อน

มอเตอร์ไฟฟ้ากระแสตรง(DC Motor) ถูกใช้เป็นอุปกรณ์หลักในระบบขับเคลื่อนของหุ่นยนต์ดังภาพที่ 2.7

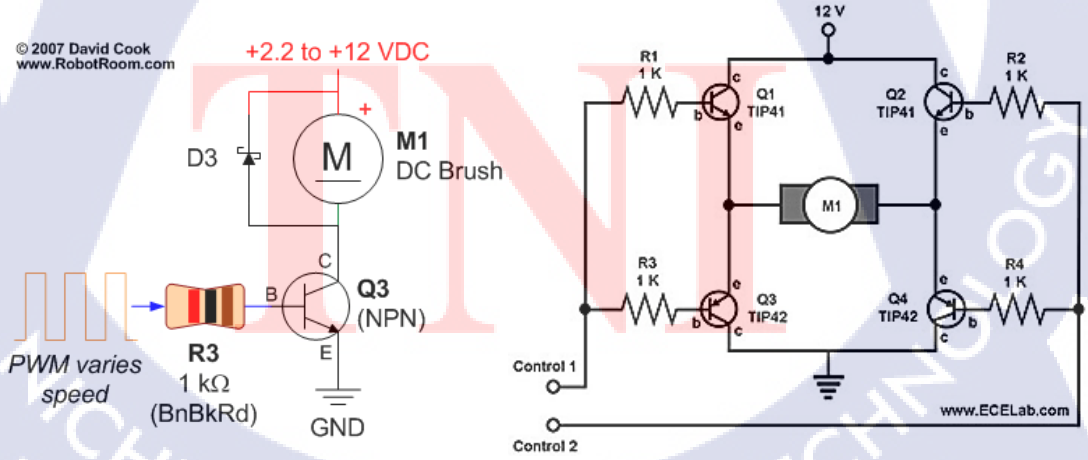


ภาพที่ 2.7มอเตอร์ไฟฟ้ากระแสตรง

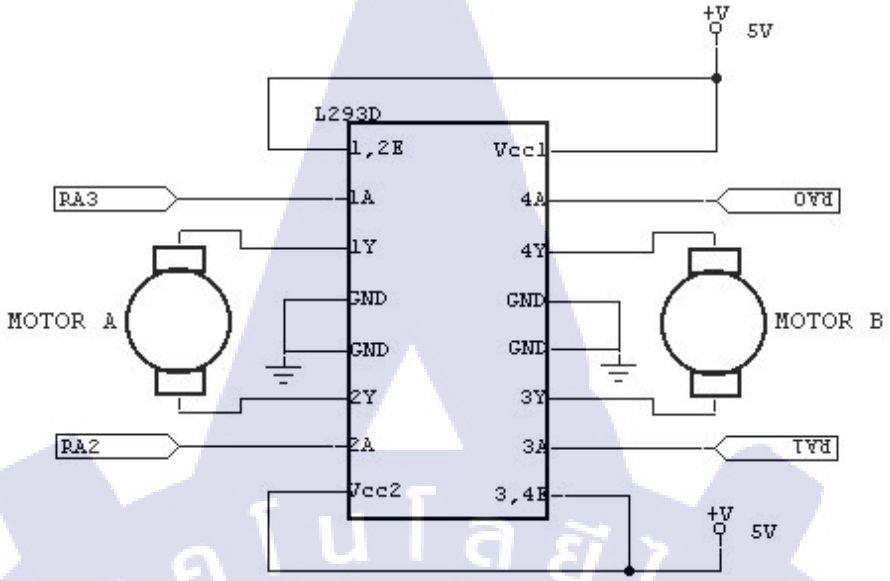
การทำงานของมอเตอร์ไฟฟ้ากระแสตรงสามารถควบคุมได้ด้วยวงจรไฟฟ้าที่ควบคุมการไหลของกระแสไฟฟ้าด้วยอุปกรณ์ต่าง ๆ เช่น รีเลย์ ทรานซิสเตอร์ มอสเฟสหรือไอซี (IC : Intigrated Circuit)



ภาพที่ 2.8วงจรควบคุมมอเตอร์ด้วยรีเลย์



ภาพที่ 2.9วงจรควบคุมมอเตอร์ด้วยทราซิสเตอร์



ภาพที่ 2.10วงจรควบคุมมอเตอร์ด้วย IC เบอร์ L293D

2.5 การทดลองตามหลักการDOE

Design of Experiments(DOE) คือ การออกแบบกระบวนการทดลอง หรือการค้นหาคำตอบจากสิ่งต่าง ๆ ที่ไม่รู้ หรือเป็นการทดสอบสมมุติฐาน หรือเป็นการแสดงในสิ่งที่รู้อยู่แล้วให้เกิดความชัดเจน ซึ่งทั้งหมดจะต้องดำเนินการอย่างเป็นระบบ และ ต้องอาศัยความรู้ในทางสถิติ เช่นฮีสโตแกรม SPC การวิเคราะห์การถดถอย และสหสัมพันธ์ โดยทั่ว ๆ ไป DOE มีปรากฏอยู่หลายรูปแบบ เช่น

- การลองผิดซองถูก (Trial and Error)
- การทดลองทีละปัจจัย (One factor one time)
- การทดลองทุกปัจจัยพร้อมกัน (Full Factors at a Time)
- การแบ่งสัดส่วนแฟคทอเรียล (Fractional Factorial) หรือที่เรียกว่า Taguchi Method

2.5.1 องค์ประกอบของ DOE

1. Factors (ปัจจัย) หมายถึง input ที่ป้อนเข้าสู่กระบวนการ สามารถแบ่งออกได้เป็น 2 ปัจจัยคือ ปัจจัยที่ควบคุมได้ และปัจจัยที่ควบคุมไม่ได้
2. Levels (ระดับ) หมายถึง ปริมาณของแต่ละ Factor หรือปริมาณขององค์ประกอบอื่น ๆ
3. Responses (ผลลัพธ์) หมายถึง ผลที่ได้จากการทดลองที่สัมพันธ์กับ Factors และ Levels

2.5.2 การออกแบบการทดลองแบบแฟคทอเรียล (Factorial Designs)

เป็นการศึกษาอิทธิพลของปัจจัยที่มีต่อกระบวนการ และสิ่งที่เกิดขึ้นพร้อมกัน เมื่อทำการทดลอง ควรปรับเปลี่ยนค่าระดับปัจจัยไปพร้อมกัน เพราะจะทำให้ได้งานที่มีประสิทธิภาพมากกว่าการเปลี่ยนค่าระดับปัจจัยตัวใดตัวหนึ่ง ทั้งในเรื่องการประหยัดเวลา ต้นทุน และยังสามารถวิเคราะห์เรื่องอิทธิพลร่วม (Interaction) ระหว่างปัจจัยได้ด้วย คือผลของการที่มีปัจจัยร่วมกันมีอยู่ในหลาย ๆ กระบวนการ ถ้าไม่ได้ทำการทดลองแบบแฟคทอเรียลอาจจะไม่เห็นผลของอิทธิพลร่วม (Interaction) ได้ชัดเจนนัก

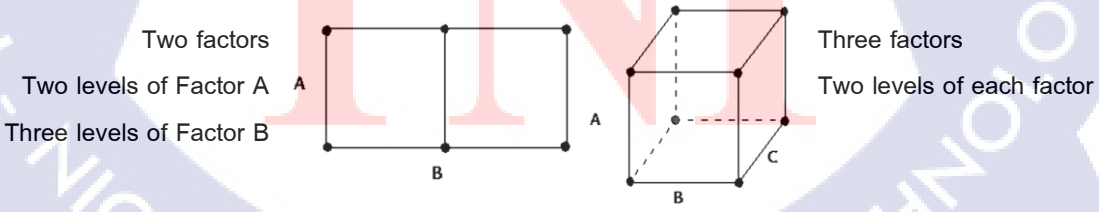
2.5.3 การออกแบบเพื่อการคัดเลือก (Screening Design)

งานพัฒนากระบวนการ และงานการผลิตส่วนมากมีตัวแปรจำนวนมากที่มีแนวโน้มว่าจะมีส่วนในการปรับปรุงการคัดเลือกเป็นการลดจำนวนตัวแปรเหล่านี้ให้มีจำนวนน้อยลง โดยคัดเลือกตัวแปรที่มีความสำคัญอย่างมากต่อคุณภาพของผลิตภัณฑ์ ซึ่งการลดจำนวนตัวแปรนี้ทำให้คุณสามารถพิจารณาเฉพาะตัวแปรที่มีความสำคัญต่อกระบวนการเท่านั้น

การคัดเลือกอาจจะสามารถทำได้ถึงการหาค่าที่เหมาะสม (optimal) ของตัวแปรนั้น ๆ รวมทั้งทำการทดลองเพื่อหาค่าที่ดีที่สุด (optimization) เพื่อบอกว่าค่าตอบสนอง (response) มีสมการความสัมพันธ์ทางคณิตศาสตร์เป็นเส้นตรงหรือเส้นโค้ง

2.5.4 การทดลองแบบ Full Factorial

การทดลองแบบ Full Factorial ค่าตอบสนอง (response) จะถูกวัดค่าที่ทุกๆ เงื่อนไขของทุกค่าระดับปัจจัยที่มี ในการทดลอง โดยเงื่อนไขการทดลอง (combination of factor levels) เป็นเงื่อนไขที่กำหนดให้ทำการทดลองเพื่อวัดค่าตอบสนอง โดยที่เงื่อนไขการทดลองแต่ละอันจะเรียกว่า รัน (run) และมีการทำการทดลองเพื่อวัดค่าตอบสนอง และชุดข้อมูลทั้งหมดในทุกรันจะเรียกว่า แบบการทดลอง (design)



ภาพที่ 2.11ตัวแบบการทดลอง 2 และ 3 ปัจจัย

จากภาพที่ 2.11แสดงตัวแบบของการทดลองแบบ 2 และ 3 ปัจจัย โดยจุดจะแสดงถึงเงื่อนไขการทดลอง (combination) แต่ละอันของการทดลอง ตัวอย่างเช่น ในตัวแบบ 2 ปัจจัย (two-factor design) จุดที่มุมล่างด้านซ้าย ซึ่งเป็นการรันของการทดลองที่มีค่าระดับปัจจัย A ค่าต่ำ (low) และค่าระดับปัจจัย B ค่าต่ำ เช่นกัน

2.5.5 การทดลองแบบ Two-level full factorial

ตัวแบบของ Two-level full factorial ในทุก ๆ การทดลองทุก ๆ ปัจจัยจะมีค่าระดับเพียง 2 ระดับเท่านั้น การทดลองแต่ละการรันจะมีทุก ๆ ค่าระดับของทุก ๆ ปัจจัย ถึงแม้ว่าตัวแบบ Two-level full factorial จะไม่สามารถทำการทดลองค่าปัจจัยที่มีขอบเขต หรือย่าน (range) กว้าง ๆ มากได้ แต่ก็สามารถให้สาระข้อมูลที่มีประโยชน์ได้โดยที่จำนวนรันไม่มากนักต่อหนึ่งปัจจัย และเพราะว่า Two-level full factorial สามารถที่จะแสดงค่าแนวโน้มได้ จึงสามารถนำมาใช้เพื่อนำไปเป็นแนวทางในการสร้างการทดลองต่อไป ตัวอย่างเช่น เมื่อคุณต้องการที่จะทำการทดลองในย่านที่กว้างขึ้นซึ่งคุณมีสมมติฐานเบื้องต้นว่าจะมีค่าที่ดีที่สุดอยู่ คุณอาจใช้ตัวแบบแฟคทอเรียล (factorial) เพิ่มเติมจากจุดนี้ โดยใช้วิธีการออกแบบการทดลองแบบ central composite

2.5.6 การทดลองแบบ General full factorial

ตัวแบบของ General full factorial การทดลองแต่ละครั้งในแต่ละปัจจัยจะมีค่าระดับอยู่หลาย ๆ ค่า ตัวอย่างเช่นปัจจัย A มี 2 ระดับ ปัจจัย B มี 3 ระดับ และ ปัจจัย C มี 5 ระดับ การทดลองในทุกการรันจะทำครบทุกค่าระดับของทุกปัจจัย ตัวแบบ General full factorial อาจจะนำไปใช้ในการทดลองขนาดเล็ก เพื่อทำการคัดเลือกปัจจัย (Screening) หรือเพื่อทำการหาค่าที่ดีที่สุด (Optimization)

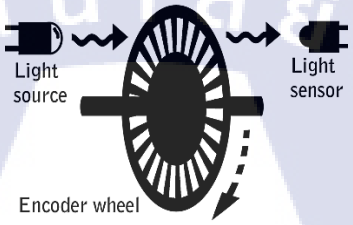
2.5.7 การทดลองแบบ Fractional Factorial

การทดลองแบบ Full factorial ค่าตอบสนองจะถูกวัดค่าในทุก ๆ เงื่อนไขการทดลองซึ่งจะเป็นทุก ๆ ค่าระดับของทุกปัจจัย ซึ่งอาจจะต้องมีการทดลองจำนวนมากครั้ง ตัวอย่างเช่น การทดลองของ two-level full factorial ของ 6 ปัจจัย อย่างน้อยต้องมีการทดลองจำนวน 64 รัน หรือกรณีที่มี 9 ปัจจัย จะมีการทดลองอย่างน้อย 512 รัน เพื่อเป็นการประหยัดเวลาและต้นทุนคุณอาจทำการออกแบบการทดลองให้มีการทำการทดลองเฉพาะบางเงื่อนไข ตัวแบบ Factorial ที่มีการทดลองไม่ครบทุกเงื่อนไขนี้เรียกว่า Fractional factorial designs โปรแกรม Minitab สามารถสร้างตัวแบบ fractional factorial ได้จนถึงจำนวนปัจจัย 15 ตัวต่อหนึ่งการทดลอง Fraction factorial มีความสำคัญอย่างมากในการทดลองเพื่อการคัดเลือกปัจจัย (screening) เพราะว่ามีการลดจำนวนรันลงจนเหลือขนาดการทดลองที่สามารถทำได้จริง รันที่ถูกเลือกมาทำการทดลองเป็นรันที่อยู่ในชุดการทดลองของตัวแบบ full factorial ซึ่งในกรณีที่ไม่ได้ทำการทดลองครบทุกเงื่อนไขของทุกปัจจัยจะทำให้เกิดผลอย่างหนึ่งที่เรียกว่าคอนฟาวด์ (confounded) ซึ่งคอนฟาวด์นี้หมายถึง

อิทธิพลของปัจจัยที่ไม่สามารถทำการประเมินค่าแยกออกมาเดี่ยวๆ และอาจเรียกว่าเป็น Aliased โดย Minitab จะแสดงตารางของ alias ที่อยู่ในรูปแบบของการคอนฟาวด์ เพราะว่าเรื่องคอนฟาวด์ทำให้อิทธิพล (effects) บางตัวไม่สามารถหาค่าได้ทำให้การเลือกการทำ fractional factorial ต้องเลือกส่วนที่จะมาทำให้ถูกต้องเพื่อให้ได้ผลลัพธ์ที่ใช้งานได้ การเลือกส่วนการทดลองที่ดีที่สุด (Best fraction) บางครั้งอาจจะต้องใช้ความรู้เฉพาะเกี่ยวกับกระบวนการและผลิตภัณฑ์เพื่อมาตัดสินใจด้วย

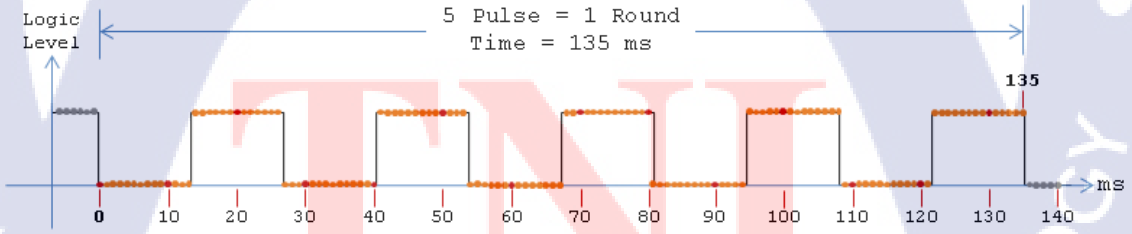
2.6 หลักการคำนวนความเร็วในการเคลื่อนที่

การคำนวนเพื่อหาความเร็วในการเคลื่อนที่ต้องอาศัยเซนเซอร์ในการตรวจนับจำนวนรอบในการหมุนของล้อ เพื่อหาระยะทางเปรียบเทียบกับเวลาที่ใช้ในการเคลื่อนที่



ภาพที่ 2.12 เซนเซอร์ตรวจนับการหมุนของล้อ

การประมวลผลสัญญาณที่ได้จากเซนเซอร์ เพื่อการตรวจนับจำนวนรอบในการหมุนของล้อสามารถใช้หลักการสุ่มสัญญาณตามคาบเวลาที่กำหนดจะได้เวลาต่อรอบ



ภาพที่ 2.13 สัญญาณที่ได้จากเซนเซอร์ต่อ1รอบการหมุน

การคำนวนหาระยะทางเหมือนกับการหาเส้นรอบวงของล้อ พิจารณาขอบนอกสุดของล้อที่สัมผัสกับพื้นสนาม จึงได้สมากรดังนี้

$$\text{Circumference of Wheel(mm)} = 2 \times \text{pi} \times \text{radius(mm)}$$

Distance(mm) ≈ Circumference of Wheel(mm)

เมื่อได้ระยะทางและเวลาที่ใช้ไปกับระยะทาง จึงทำให้สามารถคำนวนหาความเร็วของการเคลื่อนที่ในหน่วย KPH : Kilometer Per Hourได้ด้วยสมการดังนี้

Speed(KPH) = [3.6 x distance(mm) / time(ms)]

ตัวอย่างถ้ารัศมีของล้อเท่ากับ 69mm และใช้เวลาในการหมุน 1 รอบเท่ากับ 135 ms สามารถคำนวนหาความเร็วรอบได้ดังนี้

Circumference of Wheel = 2 x pi x69mm

= 433.54 mm

Speed = 3.6 x 433.54 mm / 135 ms

= 11.56 KPH

บทที่ 3
วิธีดำเนินงาน

ขั้นตอนการดำเนินงาน

1. รวบรวมข้อมูลและศึกษาผลิตภัณฑ์หุ่นยนต์เดินตามเส้นในประเทศไทย
2. วิเคราะห์ข้อมูลเปรียบเทียบผลการศึกษาข้อมูลที่รวบรวมมาได้เพื่อหาแนวทิศทางการพัฒนาหุ่ยนยนต์ และเลือกหุ่นยนต์เดินตามเส้นที่เหมาะกับโครงงานนี้
3. ออกแบบการทดลอง
4. จัดเตรียมหุ่นยนต์สำหรับทดลอง และจัดหาอุปกรณ์ประกอบการทดสอบ
5. ทำการทดลองบันทึกผล และวิเคราะห์ข้อมูลเพื่อหาความสัมพันธ์ที่ได้จากการทดลอง

3.1 การรวบรวมข้อมูล

ข้อมูลที่ต้องการรวบรวมผลิตภัณฑ์หุ่นยนต์เดินตามเส้นในประเทศไทย ดังแสดงในตารางที่ 3.1

ตารางที่ 3.1แบบฟอร์มสำหรับการรวบรวมข้อมูล

ลำดับ	**ชื่อรุ่นหุ่นยนต์** **รูปแบบหุ่นยนต์ขับเคลื่อน 2 ล้อ** **บริษัทผู้ผลิตหรือผู้จัดจำหน่าย** **ราคาที่จำหน่าย (บาท)**	**จำนวนเซนเซอร์** **ลักษณะการควบคุม**
1.		
2.		
3.		
4.		

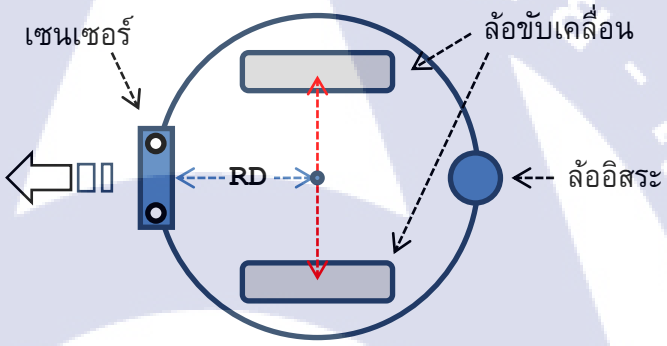
3.2 แนวทางการวิเคราะห์ข้อมูลเปรียบเทียบผลการศึกษาข้อมูล

การพิจารณาเลือกหุ่นยนต์ที่เหมาะสมกับการทดลองมีดังนี้

1. พิจารณาความง่ายต่อการปรับเปลี่ยนหรือดัดแปลงให้เหมาะกับการทดลอง
2. การทำงานไม่ซับซ้อน
3. ราคาไม่แพง

3.3 ออกแบบการทดลอง

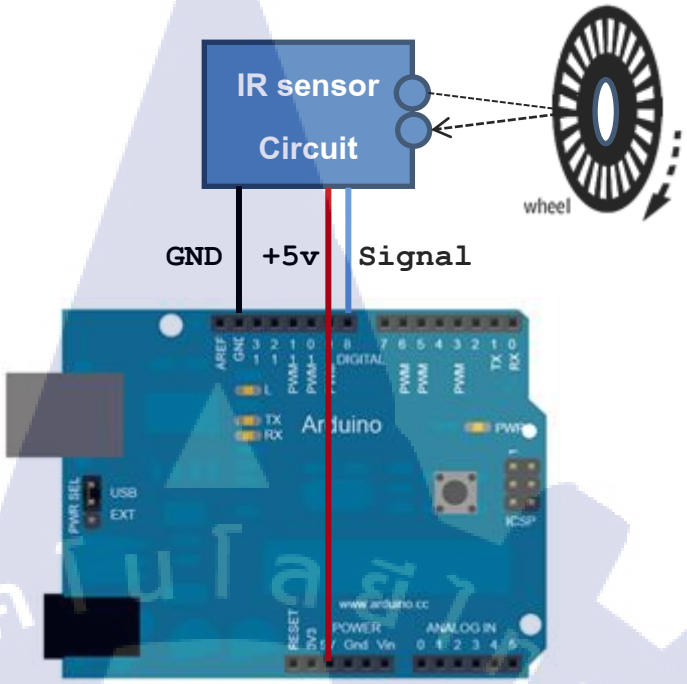
จากแนวความคิดเริ่มแรกที่ว่า เซนเซอร์ตรวจจับเส้นทางจะสัมพันธ์กับความเร็วในการเคลื่อนที่ ซึ่งเป็นตัวกำหนดพฤติกรรมการเคลื่อนที่และส่งผลกับความเร็วโดยรวม ประเด็นที่ต้องการพิจารณาคือระยะรัศมีเซนเซอร์วัดเทียบกับจุดกึ่งกลางหุ่นยนต์ หรือจุดหมุนหุ่นยนต์ หรือจุดกึ่งกลางของเพลาล้อขับเคลื่อน ดังภาพที่ 3.1 ซึ่งสามารถวัดได้ด้วยอุปกรณ์อย่างง่ายเช่น ไม้บรรทัดส่วนความเร็วในการขับเคลื่อนต้องอาศัยอุปกรณ์อิเล็กทรอนิกส์ที่ต้องสร้างขึ้น เพื่อวัดความเร็วในการขับเลื่อน เมื่อได้วิธีการหาข้อมูลเชิงตัวเลขที่ต้องการแล้วจะสามารถนำมาทดลองการหลักการของ DOE แบบ Full Factorialโดยมี 2 ปัจจัยคือ ระยะรัศมีเซนเซอร์และความร็วในการขับเคลื่อน แต่ละปัจจัยจะทำการแบ่งเป็น 3 ระดับคือ ใกล้ กลาง ไกล สำหรับรัศมีเซนเซอร์ และ ช้า ปานกลาง เร็ว สำหรับความเร็วในการขับเคลื่อน



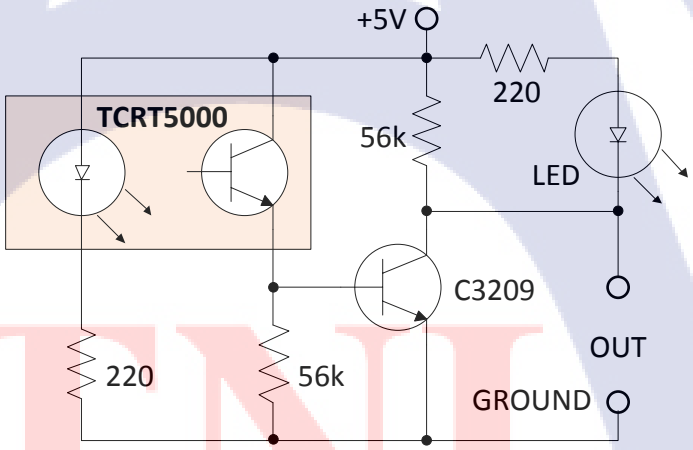
ภาพที่ 3.1 ตำแหน่งเซนเซอร์กับการวัดระยะรัศมี

3.3.1 อุปกรณ์วัดความเร็วในการขับเคลื่อน

เครื่องวัดความเร็วรอบล้อมีองค์ประกอบหลักคือ ไมโครคอนโทรลเลอร์ ทำงานร่วมกับเซนเซอร์ตรวจนับจำนวนรอบการหมุนของล้อ ในโครงงานนี้เลือกใช้ไมโครคอนโทรลเลอร์เป็นบอร์ด Arduino เพราะราคาถูกใช้งานง่าย มีตัวอย่างโปรแกรมให้ศึกษาเพื่อนำมาปรับใช้จำนวนมากส่วนเซนเซอร์ใช้เป็นแบบแสงสะท้อนในการตรวจจับก้านล้อที่มีลักษณะคล้าย Encoder โดยมีภาพรวมของระบบวัดความเร็ว ดังแสดงในภาพที่ 3.2 ส่วนวงจรเซนเซอร์ที่ใช้ตรวจนับ ดังแสดงในภาพที่ 3.3



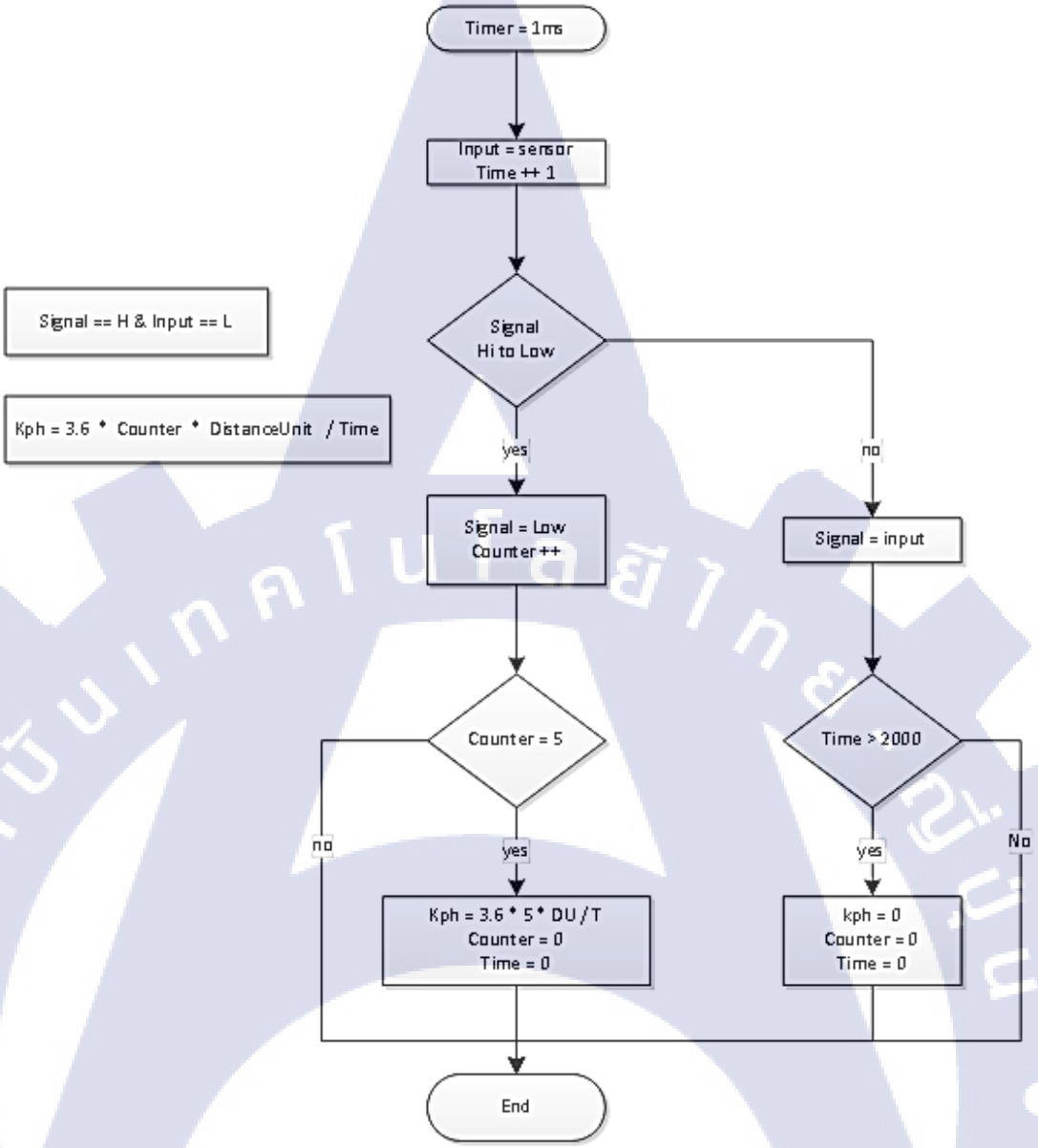
ภาพที่ 3.2ระบบตรวจนับจำนวนรอบการหมุนของล้อ



ภาพที่ 3.3วงจรเซนเซอร์ตรวจนับจำนวนรอบการหมุนของล้อ

3.3.2 ลำดับการทำงานของโปรแกรม

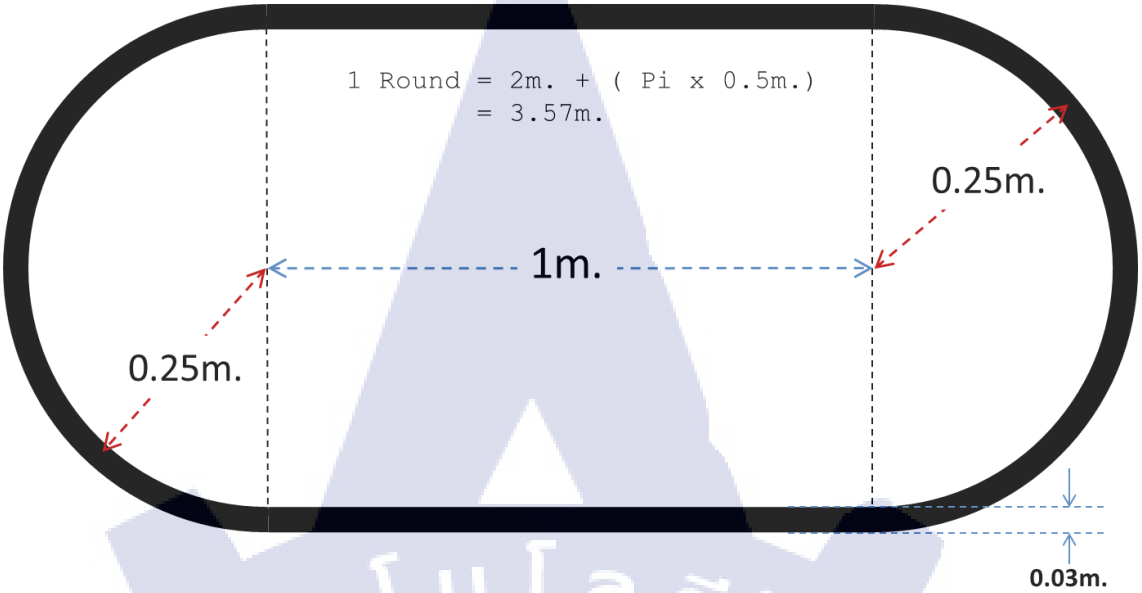
การทำงานหลักจะอาศัย Timer สร้างจังหวะการสุ่มสัญญาณจากเซนเซอร์ทุก ๆ 1 ms เมื่อถึงเวลาโปรแกรมจะตรวจจับการสะท้อนของแสงจากล้อ ถ้านับจนครบรอบให้ค่าเวลาที่บวกเพิ่มขึ้นทีละ 1 ms จะทำให้ได้ระยะทางและเวลาในการหมุนแต่ละรอบ ดังภาพที่ 3.4



ภาพที่ 3.4โฟลว์ชาร์ตการทำงานของโปรแกรมนับจำนวนรอบการหมุนของล้อ

3.3.3 ออกแบบตารางทดลอง

จากปัจจัยการทดลองที่มี 2 ปัจจัย 3 ระดับ จึงเลือกการทดลองแบบ Full Factorial คำนวนหาจำนวนครั้งในการทดสอบจะได้ 3^2เท่ากับ 9 ครั้ง ในแต่ละครั้งต้องอาศัยระยะทางที่ยาวพอประมาณ เพื่อความชัดเจนของข้อมูล ซึ่งด้วยข้อจำกัดด้านสถานที่และงบประมาณ จึงทำให้สนามที่ใช้ทดลองมีเล็ก ดังแสดงในภาพที่ 3.5 ฉะนั้นการทดลองต้องอาศัยจำนวนรอบมากกว่า 1 เพื่อให้ได้ระยะทางที่มากพอสมควร จึงออกแบบตารางทดสอบได้ดังตารางที่ 3.2



ภาพที่ 3.5สนามทดสอบ

ตารางที่ 3.2แบบฟอร์มตารางบันทึกผลการทดลอง

RD. (cm.)	Speed (kph.)	เวลาที่ใช้แต่ละรอบ (ms)										เวลารวม (ms.)	เวลาเฉลี่ย
		1	2	3	4	5	6	7	8	9	10		
ไกล	เร็ว												
	ปกติ												
	ช้า												
กลาง	เร็ว												
	ปกติ												
	ช้า												
ใกล้	เร็ว												
	ปกติ												
	ช้า												

3.4 การวิเคราะห์ข้อมูล

กำหนดแนวทางการวิเคราะห์ข้อมูลไว้ดังนี้

- สามารถวิ่งครบ 10 รอบ ได้หรือไม่ และพิจารณาความเร็วรวมที่ได้จากการวิ่ง
- พิจารณาประสิทธิภาพของการวิ่งในแต่ละรอบ โดยอ้างอิงจากค่า RD เป็นหลัก
- พิจารณาประสิทธิภาพของการวิ่งในแต่ละรอบ โดยอ้างอิงจากค่า Speed เป็นหลัก

บทที่ 4
ผลการดำเนินงานและการวิเคราะห์

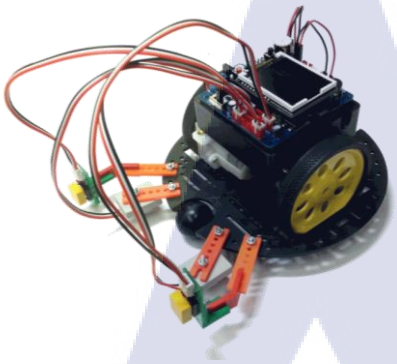
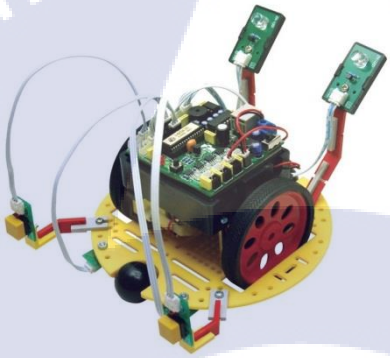
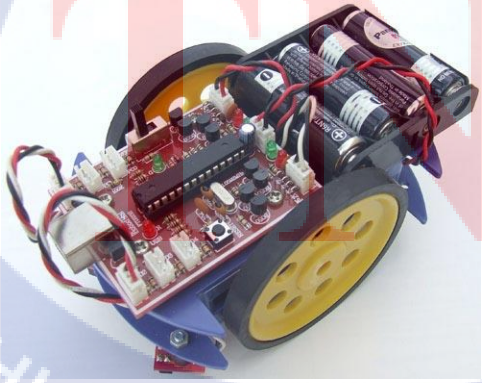
4.1 ผลการดำเนินงาน

4.1.1 ผลการรวบรวมข้อมูลหุ่นยนต์เดินตามเส้นในประเทศไทย

จากการรวบรวมข้อมูลและศึกษาผลิตภัณฑ์หุ่นยนต์เดินตามเส้นในประเทศไทยได้ผลลัพธ์ดังแสดงในตารางที่ 4.1

ตารางที่ 4.1ข้อมูลหุ่นยนต์เดินตามเส้นในประเทศไทย

ลำดับ	รูปแบบหุ่นยนต์ บริษัทผู้ผลิตหรือผู้จัดจำหน่าย ราคาที่จำหน่าย (บาท)	จำนวนเซนเซอร์ ลักษณะการควบคุม
1.	**60101 หุ่นยนต์เดินตามเส้น**  ร้าน ThaiEduRobot เว็บไซต์: http://www.thaiedurobot.com ราคา: 640.-	2 เซนเซอร์ ควบคุมด้วย Basic Circuit
2.	**3pi Robot Line Following**  บริษัท วีนัส ซัพพลาย จำกัด เว็บไซต์: http://www.thaieasyelec.com ราคา: 3,550.-	5 เซนเซอร์ ควบคุมด้วย Microcontroller

3.	**Robot Pop-Bot XT**  บริษัท แอพซอฟต์เทคจำกัด เว็บไซต์: http://www.appsofttech.com ราคา: 4,900.-	2 หรือ 4 เซนเซอร์ ควบคุมด้วย Microcontroller
4.	**Robo-CIRCLE 3S**  บริษัท อินโนเวตีฟ เอ็กเพอริเมนต์ จำกัด เว็บไซต์: http://www.inex.co.th ราคา: 2,500.-	2เซนเซอร์ ควบคุมด้วย Microcontroller
5.	**AP113 หุ่นยนต์ Built-in programmer**  ร้าน บ้านอิเล็กทรอนิกส์ เว็บไซต์: http://www.semi-shop.com ราคา: 2,000.-	2 เซนเซอร์ ควบคุมด้วย Microcontroller

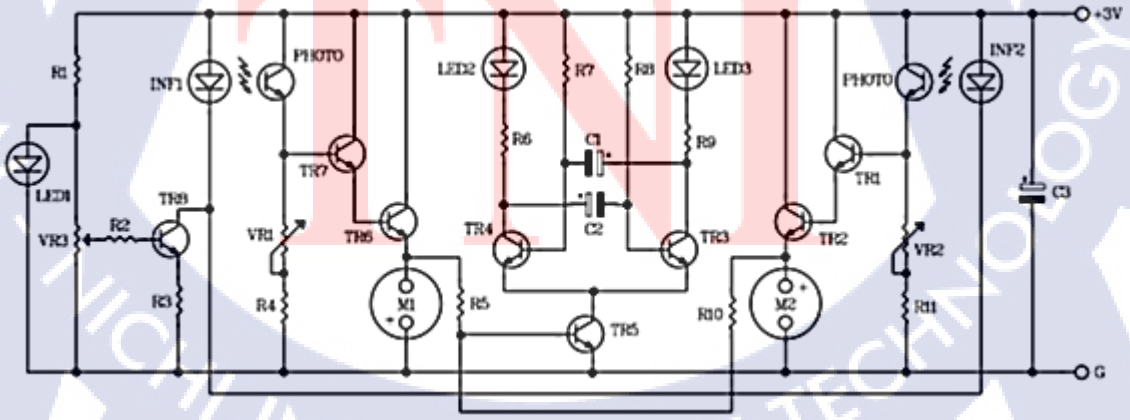
6.	**FK1108 หุ่นยนต์ GEAR TACON**  บริษัท ฟิวเจอร์คิท มาร์เก็ตติ้ง จำกัด เว็บไซต์: http://www.futurekit.com ราคา: 670.-	2 เซนเซอร์ ควบคุมด้วย Basic Circuit

จากข้อมูลที่ได้พบว่า ส่วนใหญ่จะเป็นหุ่นยนต์ที่ควบคุมด้วยไมโครคอนโทรลเลอร์ ซึ่งมีความแตกต่างทั้งชนิดของไมโครคอนโทรลเลอร์ และความซับซ้อนทั้งโปรแกรม ทำให้ยุ่งยากต่อการนำมาดัดแปลง แต่พบว่ามีหุ่นยนต์อยู่ 2 ที่มีความหน่าสนใจคือ

- 60101 หุ่นยนต์เดินตามเส้นของร้าน ThaiEduRobot
- FK1108 หุ่นยนต์ GEAR TACON ของบริษัท ฟิวเจอร์คิท มาร์เก็ตติ้ง จำกัด

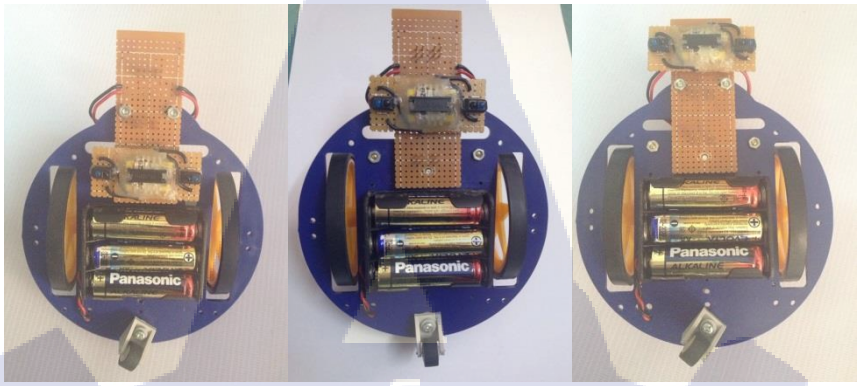
4.1.2 ผลการดัดแปลงหุ่นยนต์และสร้างอุปกรณ์วัดความเร็วรอบล้อ

จากผลสำรวจหุ่นยนต์ที่เหมาะกับการทดลอง เป็นหุ่นยนต์ที่ควบคุมด้วย Basic Circuit ซึ่งไม่ซับซ้อน และราคาไม่แพง สามารถนำมาดัดแปลงให้เหมาะกับการทดลองได้ไม่ยากนัก แต่หุ่นยนต์ TACON มีลักษณะของล้อที่ง่ายต่อการวัดรอบการหมุนของล้อฉะนั้นการทดลองนี้จึงเลือกหุ่นยนต์ TACON เพื่อนำมาศึกษาและดัดแปลงให้เหมาะกับการทดลอง โดยวงจรฉบับย่อส่วนของหุ่นยนต์ TACON ได้แสดงไว้ในภาพที่ 4.1(ฉบับขยายแสดงไว้ที่ภาคผนวก)

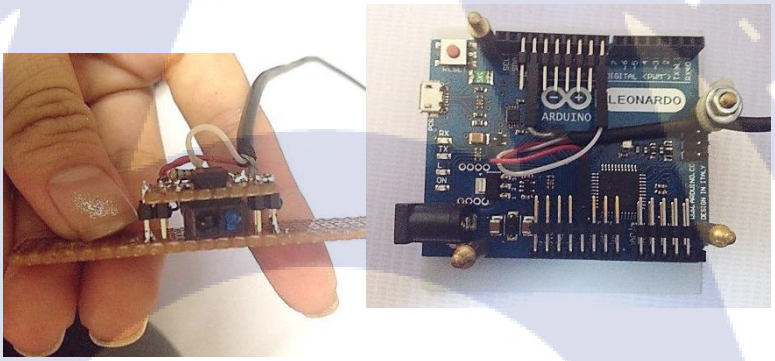


ภาพที่ 4.1วงจรหุ่นยนต์TACON เดินตามเส้น (ย่อส่วน)

หุ่นยนต์ที่ผ่านการดัดแปลงแล้วได้แสดงไว้ในภาพที่ 4.2 และอุปกรณ์ที่ใช้วัดความเร็วรอบล้อได้แสดงไว้ในภาพที่ 4.3 และการทดลองวิ่งบนสนามจริง ดังภาพที่ 4.4



ภาพที่ 4.2การติดตั้งเซนเซอร์ทั้ง 3 ระยะ



ภาพที่ 4.3เซนเซอร์และบอร์ด Arduino สำหรับวัดความเร็วรอบล้อ

ภาพที่ 4.4ทดลองวิ่งบนสนามจริง

4.1.3 ผลทดลองวิ่งบนสนามจริง

ทำการทดลองและจดบันทึกลงตารางที่ 3.2 โดยปรับระยะเซนเซอร์ และปรับความเร็วรอบตามลำดับ หลังจากนั้นนำไปวิ่งบนสนามทดสอบและจับเวลาที่ใช้วิ่งแต่ละรอบต่อเนื่องกันเป็นจำนวน 10 รอบจะได้ระยะทาง 35.7 เมตร ผลลัพธ์ที่ได้แสดงไว้ในตารางที่ 4.2

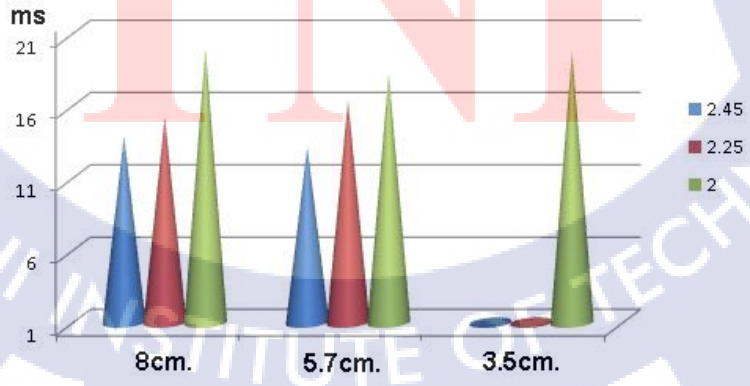
ตารางที่ 4.2ผลการทดลองวิ่งบนสนามจริง

RD. (cm.)	Speed (kph.)	เวลาที่ใช้แต่ละรอบ (ms)										เวลารวม	เวลาเฉลี่ย
		1	2	3	4	5	6	7	8	9	10		
8	2.45	13.93	14.08	13.7	14.05	13.72	13.89	13.74	13.89	14.15	13.89	139.04	13.90
	2.25	15.23	15.36	14.92	15.39	15.33	15.17	15.13	15.06	15.25	15.26	152.10	15.21
	2.00	20.18	19.82	19.86	19.47	19.79	20.04	19.9	20.04	20.24	19.83	199.53	19.95
5.7	2.45	13.33	13.24	13.31	13.5	12.92	13.34	13.29	13.29	12.4	13.01	131.63	13.16
	2.25	13.81	16.33	17.08	16.29	17.71	16.26	15.71	17.3	17.02	16.31	163.82	16.38
	2.00	18.16	18.89	18.25	17.7	17.88	17.85	18.12	18.15	18.32	18.88	182.20	18.22
3.5	2.45	ไม่จบรอบ											
	2.25	ไม่จบรอบ											
	2.00	20.35	19.43	19.16	20.27	19.81	19.88	19.83	19.48	19.53	19.56	197.30	19.73

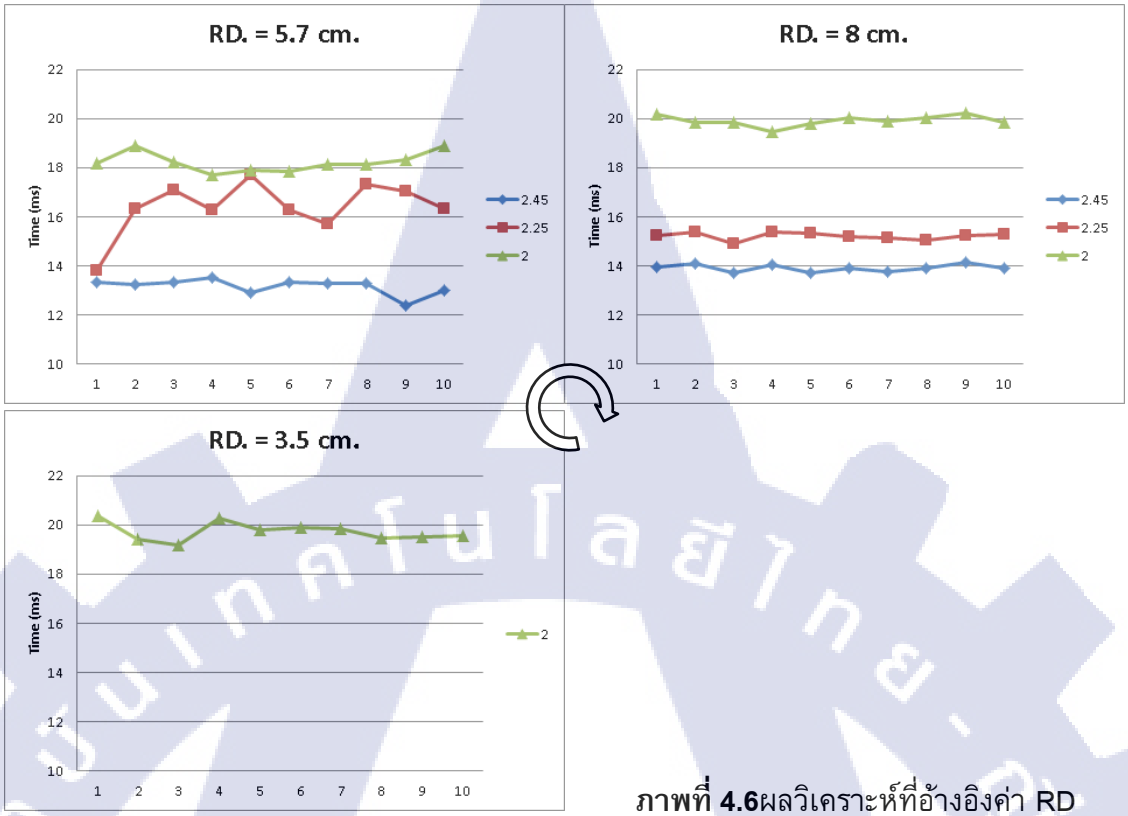
4.2 ผลการวิเคราะห์

จากตารางสรุปผลการทดลองพบว่า ระยะรัศมีเซนเซอร์ตำแหน่ง 5.7 cm. (ระยะกลาง) สามารถทำความได้เร็วดีสุด แต่หากพิจารณาความแปรปวนที่เกิดขึ้นจะพบว่า ระยะรัศมีเซนเซอร์ตำแหน่ง 8 cm. (ระยะไกล) มีความแปรปวนน้อยกว่า ซึ่งหากพิจารณาประสิทธิภาพจะสรุปได้ว่าความเร็วในการเคลื่อนที่ยิ่งสูง ควรจะมีระยะรัศมีเซนเซอร์ที่ไกลจากจุดกึ่งกลางของหุ่นยนต์

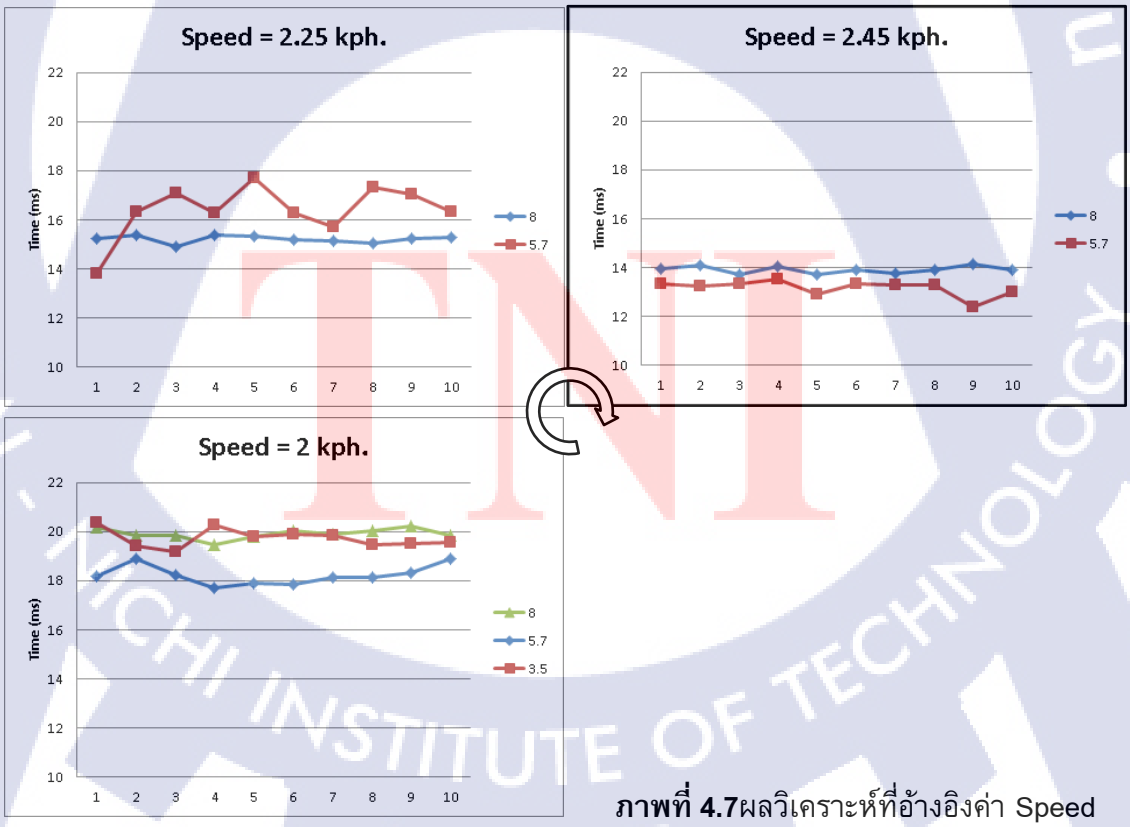
ในทางกลับกัน ระยะรัศมีเซนเซอร์ตำแหน่ง 3.5 cm. (ระยะใกล้) จะยังคงทำงานได้ที่ความเร็วต่ำ ต่างจากระยะกลางและระยะไกลที่ไม่สามารถทำภาระกิจได้สำเร็จ



ภาพที่ 4.5ผลการวิเคราะห์โดยรวม



ภาพที่ 4.6ผลวิเคราะห์ที่อ้างอิงค่า RD



ภาพที่ 4.7ผลวิเคราะห์ที่อ้างอิงค่า Speed

บทที่5
บทสรุปและข้อเสนอแนะ

5.1 สรุปผลการดำเนินงาน

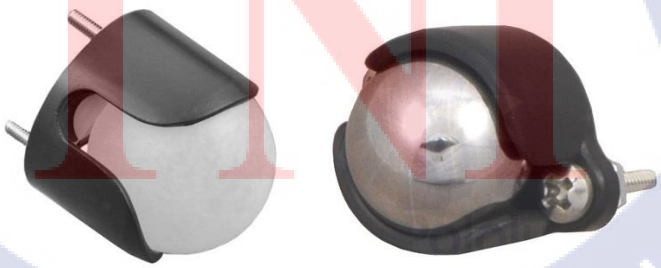
จากการทดสอบพบว่าที่ระยะเซนเซอร์ 5.7 cm. (ระยะกลาง)ใช้เวลาน้อยที่สุดที่ความเร็ว 2.45km/hr (ระดับสูง)แสดงว่าระยะของเซนเซอร์และความเร็วของหุ่นยนต์ มีผลกับการเคลื่อนที่ โดยความเร็วต้องเป็นความเร็วที่เหมาะสมจึงจะทำให้หุ่นยนต์วิ่งได้ดี และใช้เวลาน้อย แต่จากผลวิเคราะห์ที่พิจารณาการวิ่งในแต่ละรอบพบว่ามีความไม่สม่ำเสมอของเวลาที่ใช้ ทำให้คาดการว่าประสิทธภาพไม่แน่นอนต่างจากระยะเซนเซอร์ 8 cm. (ระยะไกล) ที่ให้ความสม่ำเสมอในการวิ่งแต่ละรอบดีกว่า อาจเป็นเพราะที่ตำแหน่งเซนเซอร์ระยะ8 cm. (ระยะไกล) มีความไวต่อการตรวจจับเส้นทางได้ดีกว่า และส่งผลให้โอกาสที่จะหลุดออกจากเส้นทางมีน้อยกว่า

ส่วนที่เซนเซอร์ระยะ5.7 cm. (ระยะกลาง) ทำความเร็วได้ดีกว่า อาจเป็นเพราะมีความไวต่อการตรวจจับเส้นทางได้ช้ากว่าเซนเซอร์ระยะ8 cm. (ระยะไกล) จึงทำให้การวิ่งเป็นแบบแกว่งน้อยกว่าและทำให้ความเร็วในบางรอบอาจจะดีกว่าหรือแย่กว่าได้

ฉนั้นระยะเซนเซอร์ที่เหมาะสมสุดควรจะมีระยะไกลที่สุด แต่ปัจจัยการออกแบบหุ่นยนต์เดินตามเส้นมีอีกหลายอย่างเช่น สภาพสนามที่มีความคดเคี้ยว หรือมีช่องแคบ และสิ่งกีดขวางต่าง ๆ หากเป็นเส้นตรงและมีทางโค้งไม่มากนักให้ใช้เซนเซอร์ระยะปานกลาง จะทำให้ได้ความเร็วสูงที่สุด

5.2 ข้อเสนอแนะ

จากการทดลองพบปัญหาที่ล้ออิสระด้านหลังทำงานได้ไม่ดี ซึ่งอาจส่งผลให้การทดลองนี้คาดเลื่อนไปบ้าง จึงควรให้ความสำคัญ โดยเลือกใช้ล้ออิสระแบบลูกกลิ้ง (ball caster) ดังภาพที่ 5.1 อาจช่วยให้ผลการทดลองมีความเที่ยงตรงกว่านี้ หรือทำให้ประสิทธิภาพของหุ่นยนต์ดีขึ้นได้



ภาพที่ 5.1 Ball Caster

การออกแบบหุ่นยนต์ประเภทขับเคลื่อน 2 ล้อและมีล้อพยุงด้านหน้า หรือด้านหลัง ควรคำนึงถึงจุดศูนย์ถ่วง เพราะล้อพยุงจะลอยทำให้เซนเซอร์มีการขยับขึ้นลงเป็นผลให้การตรวจจับเส้นทางผิดพลาดได้ อาจต้องมีการถ่วงน้ำหนัก เพื่อทำให้ล้อทั้งหมดสัมผัสพื้นสนาม และช่วยทำให้การเคลื่อนที่ดีขึ้น

จากการทดลองที่ผ่านมามีระดับปัจจัยที่หยาบ ทำให้ไม่สามารถนำผลการทดลองไปใช้หาสมการความสัมพันธ์ได้ จึงควรเพิ่มระดับปัจจัยให้มีความละเอียดมากขึ้น

5.3 ประโยชน์ที่ได้จากการทำสารนิพนธ์

สิ่งที่ได้รับจากสารนิพนธ์นี้คือแนวทางการออกแบบหุ่นยนต์ที่ช่วยให้ได้หุ่นยนต์เดินตามเส้นที่มีประสิทธิภาพเหมาะกับเส้นทางที่นำไปใช้ ช่วยลดเวลาในการพัฒนาแบบลองผิดลองถูก และช่วยให้เข้าใจพฤติกรรมการทำงานของเซนเซอร์ตรวจจับเส้นสำหรับหุ่นยนต์เดินตามเส้น

บรรณานุกรม

บรรณานุกรม

ผศ.โอภาส ศิริครรชิตถาวร, วรพจน์ กรแก้ววัฒนกุล, ชัยวัฒน์ ลิ้มพรจิตรวิไล.(2537). **เรียนรู้ระบบควบคุมอย่างง่ายด้วยโปรแกรมภาษาCกับ Arduino และบอร์ดไมโครคอนโทรลเลอร์ POP-168.**กรุงเทพฯ: อินโนเวตีฟ เอ็กเพอริเมนต์

MCU thailand. (2556).**Arduino คืออะไร ตอนที่1.**สืบค้นเมื่อ 1 มีนาคม 2557 จาก http://www.mcu.in.th/article

ชาคริส วราหะ. (2553). **Arduino คืออะไร ?.**สืบค้นเมื่อ 1 มีนาคม 2557 จาก http://gm9.blogspot.com/

Pcman.(2551). **คู่มือการโปรแกรม Arduino board.**สืบค้นเมื่อ 1 มีนาคม 2557 จาก http://www.logicthai.net/node/10

บ. บี เค อาร์ บิสซิเนส จำกัด.(2537). **Design of Experiment – DOE – การออกแบบทดลอง.** สืบค้นเมื่อ 15 สิงหาคม 2556 จาก http://www.topofquality.com/sdoe/indexdoe.html

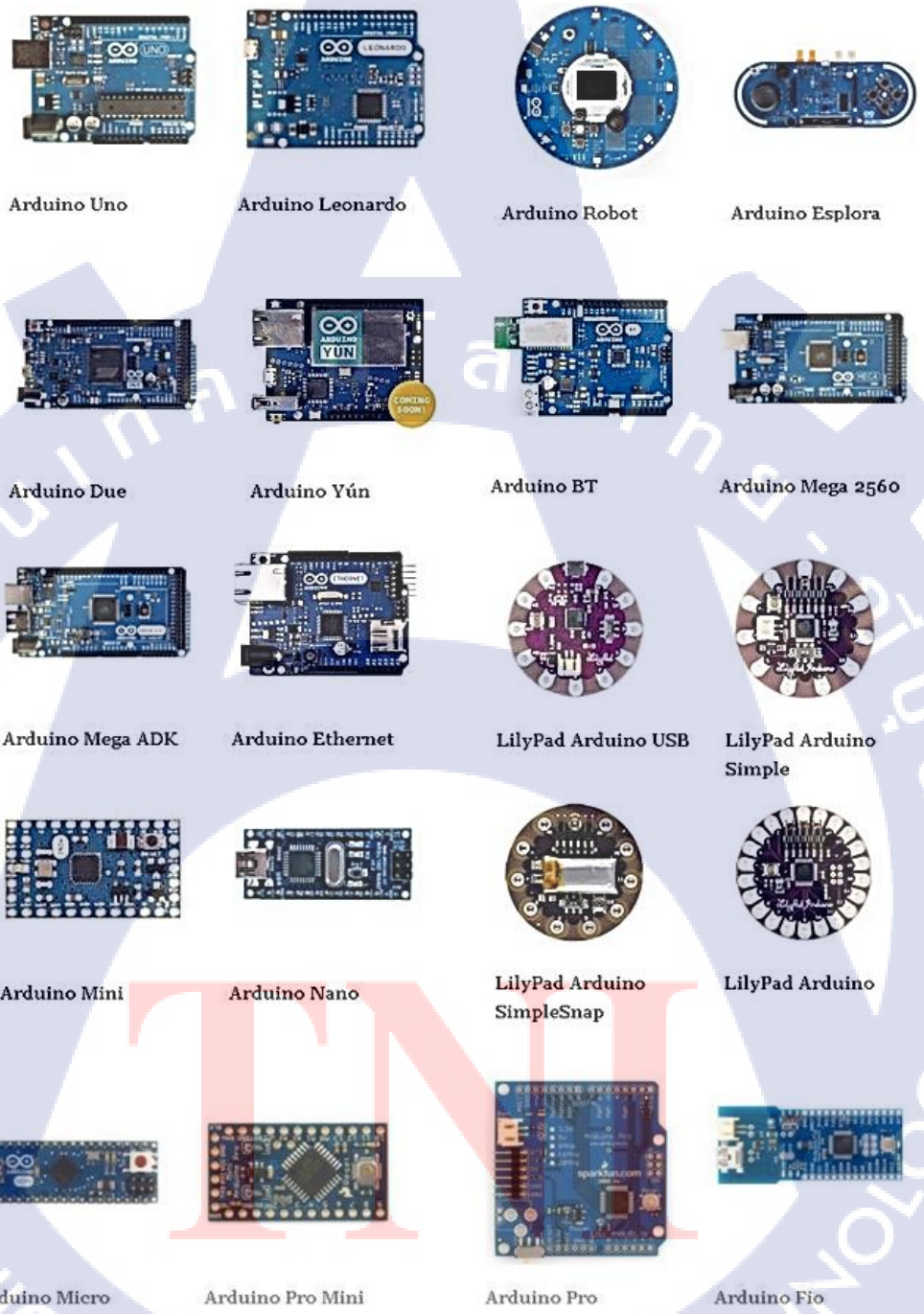
บ. โซลูชั่น เซ็นเตอร์ จำกัด.(2556). **บทที่ 1 Designing Experiments การออกแบบการทดลอง.** สืบค้นเมื่อ 15 สิงหาคม 2556จาก http://www.solutioncenterminitab.com/files/exampledoebook.pdf

ภาคผนวก

ภาคผนวก ก.

ข้อมูลบอร์ด **Arduino**

บอร์ด Arduino เป็น Open Hardware Platform ที่ถูกพัฒนาขึ้นโดยมี Micro-controller ของ Atmel เป็นหัวใจหลัก บอร์ด Arduino ที่ผลิตออกมาจำหน่ายในปัจจุบันมีทั้งหมด 20 รุ่น แต่ละรุ่นมีหน้าตากันดังนี้



(บอร์ดบางรุ่นใช้ชื่อเดียวกันแต่มีรายละเอียดแตกต่างกัน เช่น บอร์ดรุ่น Arduino Pro, Arduino Nano, LilyPad Arduino)

Board	Micro-controller	Clock Speed	Flash Memory	SRAM	EEPROM
Arduino UNO	ATmega328	16 MHz	32 KB	2 KB	1 KB
Arduino Lenonardo	**ATmega32u4**	**16 MHz**	**32 KB**	**2.5 KB**	**1 KB**
Arduino DUE	AT91SAM3X8E	84 MHz	512 KB	96 KB	-
Arduino YUN	ATmega32u4	16 MHz	32 KB	2.5 KB	1 KB
Arduino Mega ADK	ATmega2560	16 MHz	256 KB	8 KB	4 KB
Arduino Ethernet	ATmega328	16 MHz	32 KB	2 KB	1 KB
Arduino Mega 2560	ATmega2560	16 MHz	256 KB	8 KB	4 KB
Arduino BT (Bluetooth)	ATmega328	16 MHz	32 KB	2 KB	1 KB
Arduino Micro	ATmega32u4	16 MHz	32 KB	2.5 KB	1 KB
Arduino Pro Mini	ATmega168	8 MHz	16 KB	1 KB	512 Bytes
Arduino Pro	ATmega168	8 MHz	16 KB	1 KB	512 Bytes
Arduino Pro	ATmega328	16 MHz	32 KB	2 KB	1 KB
Arduino Mini	ATmega328	16 MHz	32 KB	2 KB	1 KB
Arduino Nano	ATmega168	16 MHz	16 KB	1 KB	512 Bytes
Arduino Nano	ATmega328	16 MHz	32 KB	2 KB	1 KB
Arduino Fio	ATmega328P	8 MHz	32 KB	2 KB	1 KB
Arduino Robot	ATmega32u4	16 MHz	32 KB	2.5 KB	1 KB
Arduino Esplora	ATmega32u4	16 MHz	32 KB	2.5 KB	1 KB
LilyPad Arduino USB	ATmega32u4	8 MHz	32 KB	2.5 KB	1 KB
LilyPad Arduino Simple	ATmega328	8 MHz	32 KB	2 KB	1 KB
LilyPad Arduino SimpleSnap	ATmega328	8 MHz	32 KB	2 KB	1 KB
LilyPad Arduino	ATmega168V	8 MHz	16 KB	1 KB	512 Bytes
LilyPad Arduino	ATmega328V	8 MHz	16 KB	1 KB	512 Bytes

ภาคผนวก ข.

เริ่มต้นใช้บอร์ด Arduino และการเพิ่มLibrary

เริ่มต้นใช้งานบอร์ดArduino

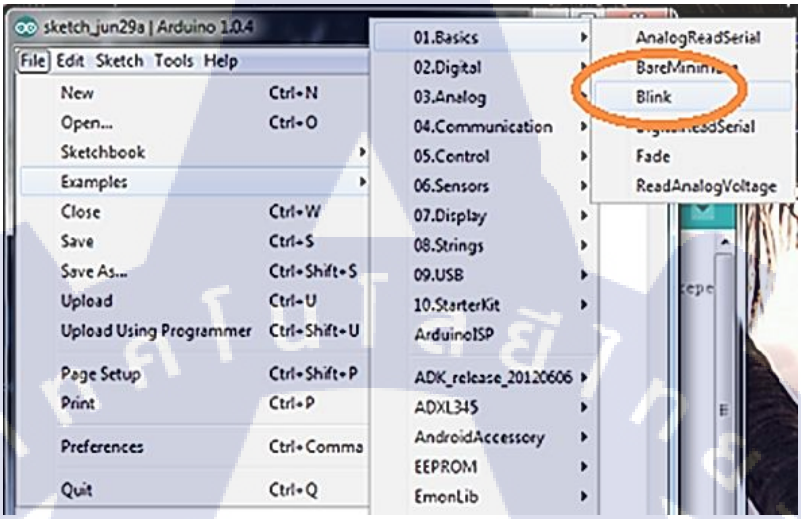
บอร์ด Arduino มีอยู่หลายแบบ แต่ใช้โปรแกรมตัวเดียวกันในการเขียนชุดคำสั่ง โดยโปรแกรมเรียกว่า Arduino IDE (IDE นั้นย่อมาจาก Integrated Development Environment) ซึ่งใช้งานได้ทั้งบนระบบปฏิบัติการ Window (XP, Vista,7, 8) ทั้ง 32 และ 64 บิต, Mac OS X และ Linux ซึ่งเป็นอิสระจากการทำงานของ OS ทุกชนิด ทำให้ไม่ต้องมีการ Install โปรแกรม ให้วุ่นวายเพียงแค่ Download มาและUnzip ไว้ใน Directory ที่ต้องการไม่ต้อง Crack

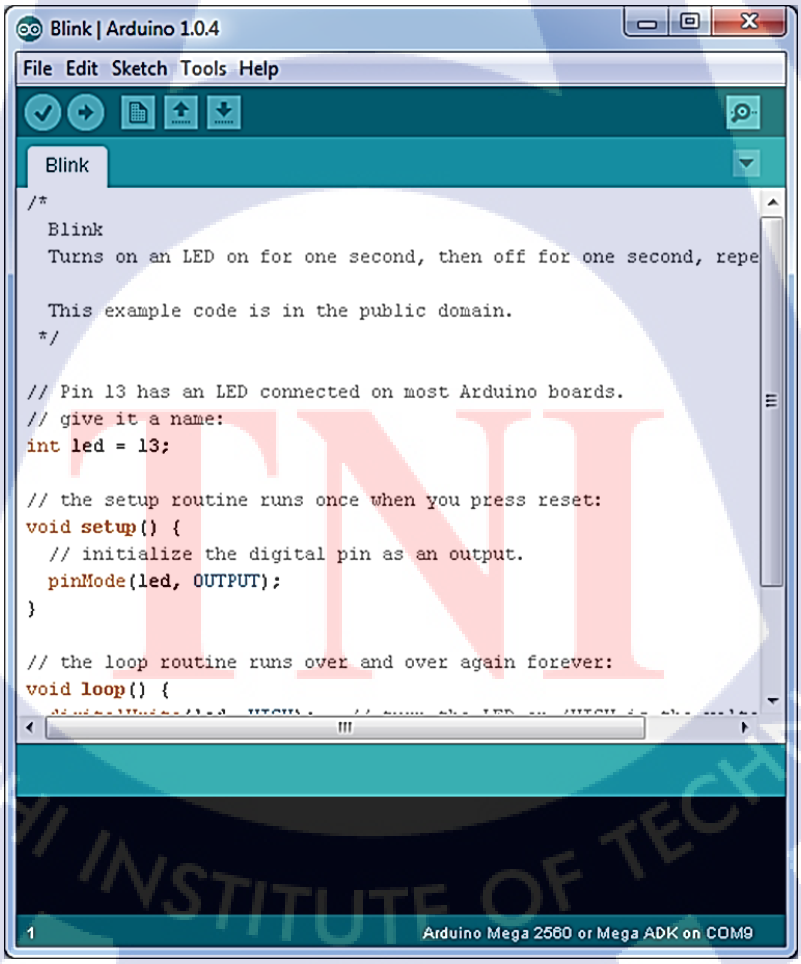
ขั้นตอนแรกในการทดสอบการทำงานของบอร์ด Arduino และโปรแกรม ArduinoIDE

- เลือกบอร์ด Arduino ที่ต้องการทดสอบการทำงาน
- Download และติดตั้งแรแกรม Arduino IDE ที่ URL: http://arduino.ccโดยเลือกให้เหมาะสมกับOS ที่ใช้งาน เช่น Window, Mac, OS X, Linux: 32 bit, 64 bit
- Unzip ไว้ใน Directory ที่ต้องการ

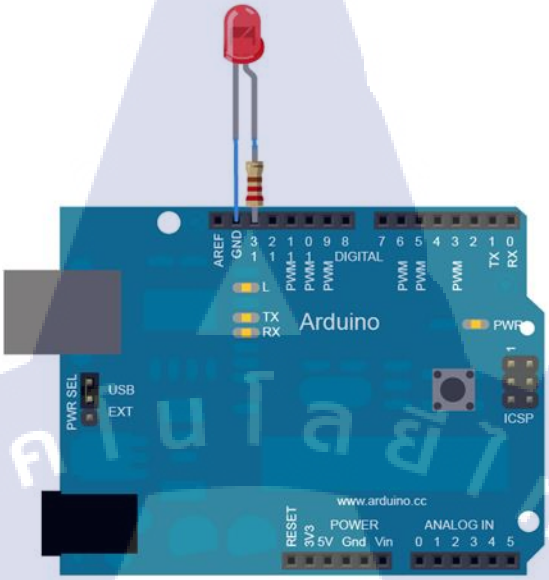
เมื่อได้โปรแกรมพร้อมใช้งานจะพบไฟล์ที่ชื่อ arduino.exeใช้สำหรับ run program

โปรแกรมแรกสำหรับผู้เริ่มต้นใช้งานบอร์ด Arduino คือ "Arduino Blink" ซึ่งเป็น Sketch ตัวอย่างที่ง่ายสุดเหมาะกับการใช้ตรวจสอบการทำงานของบอร์ดที่ Arduino IDE มีให้อยู่ในทุก version และเป็นตัวอย่างที่มีประโยชน์มากหลังจากเปิดโปรแกรมให้เข้าไปเปิดตัวอย่าง "Blink" ที่เมนู File ->Examples -> Basics -> Blink

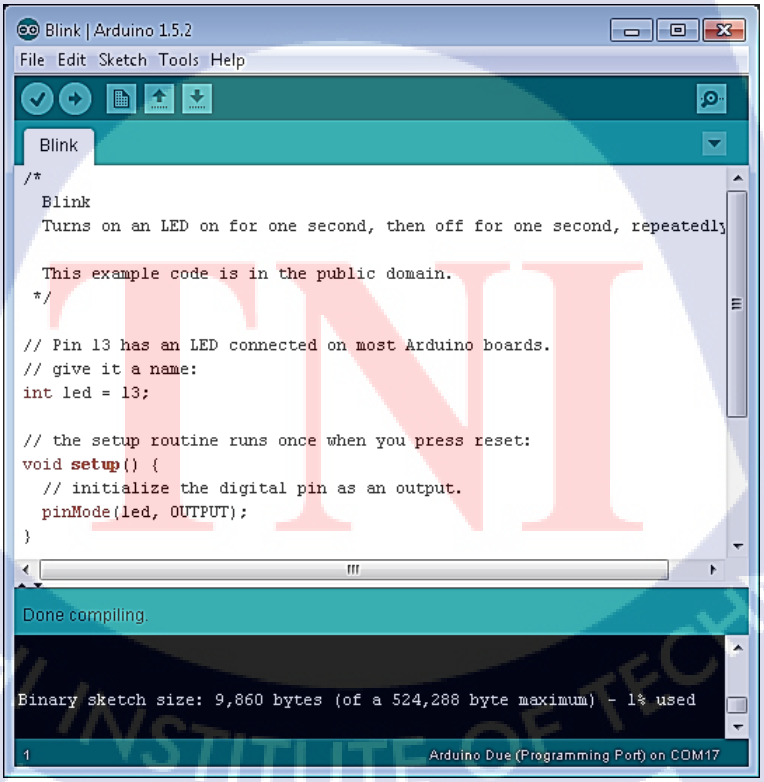




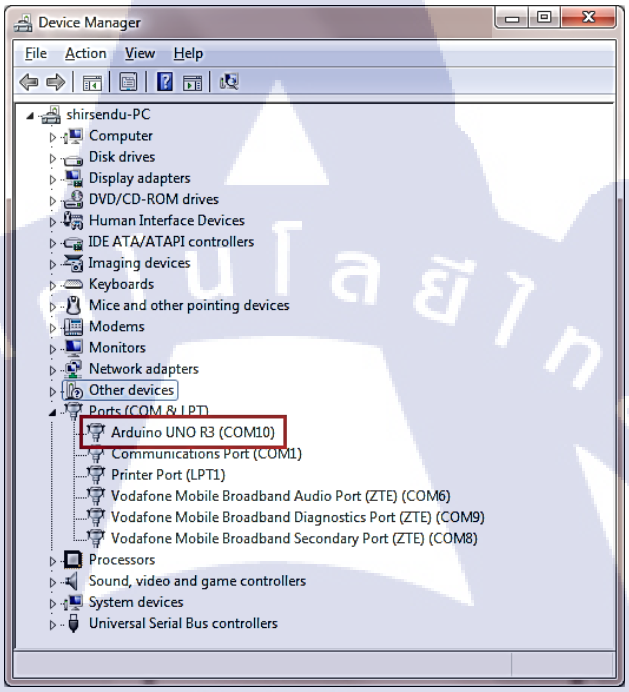
Sketch นี้มีไม่กี่บรรทัด โดยจะสั่งให้บอร์ด Arduinoกระพริบ LED หมายเลข 13 เท่านั้นครับ โดยจกระพริบเปิดปิดสลับกันอย่างละ 1 วินาที



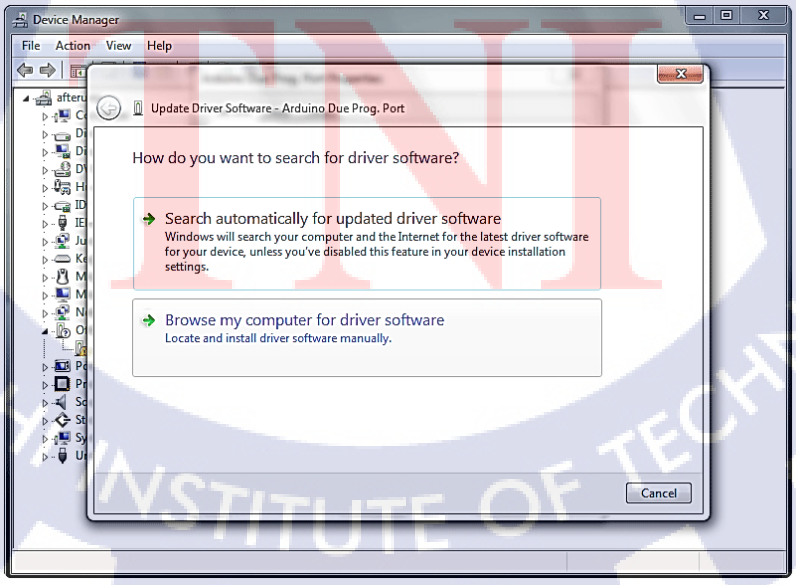
วิธีการ Compile ให้ไปที่ไอคอนเครื่องหมายถูก ถ้า compile ผ่าน จะปรากฏข้อความ Done Compiling ถ้าไม่ผ่านก็จะแจ้งบริเวณบรรทัดที่เกิดความผิดพลาด



ขั้นตอนการทดลองกับบอร์ดทำการเสียบสาย USB ทั้งสองด้านเข้ากับบอร์ด และเครื่องคอมพิวเตอร์ เมื่อเชื่อมต่อแล้วจะพบว่ามีLED สว่างขึ้นแสดงว่ามีแรงดันไฟฟ้าจ่ายไปที่บอร์ด ถ้าเป็นการติดตั้งครั้งแรกจะพบว่ามีข้อความแจ้งว่า Process fail ซึ่งต้องลง Driver ก่อน โดยไปที่ Start menu และเปิด Control Panel ขึ้นมา ไปที่ Device manager จากนั้นดูที่ Ports (COM&LPT)จะพบว่ามีชื่อ Arduino XXX (COMxx)

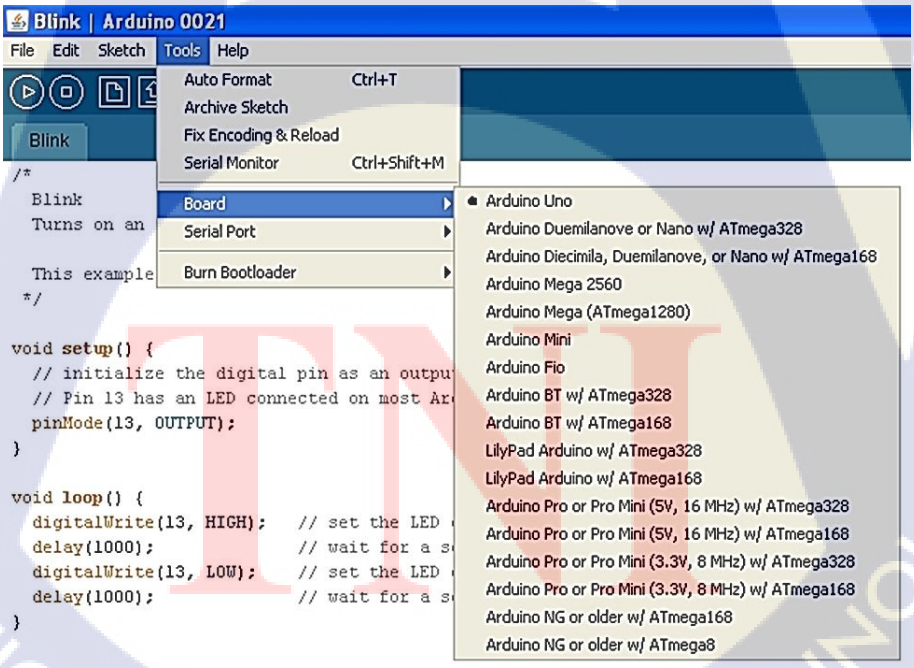


จากนั้นให้คลิกขวาที่ Arduino ครับ จะพบกับ "Update Driver Software" แล้วให้เลือก "Browse my computer for Driver software"



ทำการเลือก Folder ที่เก็บ Arduino IDE ไว้ ซึ่งภายในจะมีFolder ชื่อ "drivers" ในนั้นจะพบกับไฟล์ "arduino.inf" ดับเบิ้ลคลิกเลือกไฟล์นี้แล้วปล่อยให้มันทำงานเป็นอันเสร็จเรียบร้อย โดยทำแค่ครั้งแรกครั้งเดียว

เมื่อต้องการUpload Sketch ลงบอร์ดต้องทำการเลือก COM Port และเลือกรุ่นของบอร์ด Arduino ที่ใช้ให้ถูกต้องก่อน

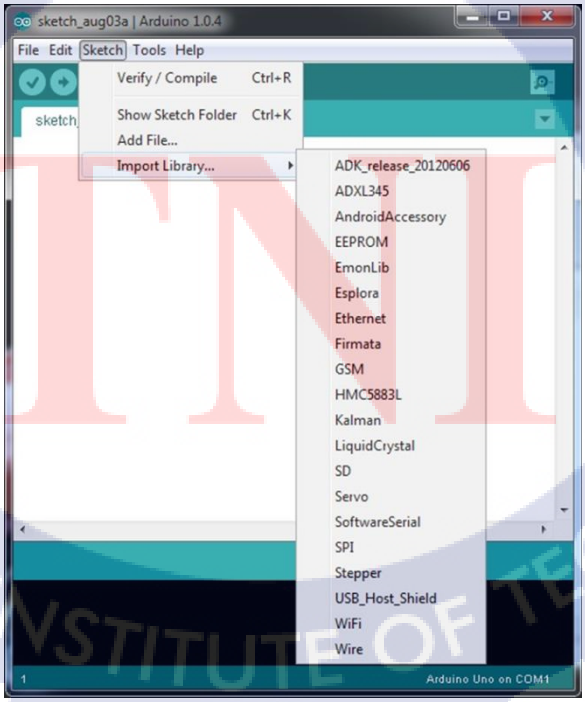


การ Upload ให้ไปที่เครื่องหมายลูกศรที่ชี้ไปทางขวาใต้ Menu ข้างๆ เครื่องหมายถูกเท่านี้ Arduino IDE ก็จะ Upload sketch ของเราลงบอร์ดระหว่างการ Upload จะเห็น LED 2 ดวงกระพริบถี่ๆ สลับกันถ้าทำทุกอย่างถูกต้องจะเห็น LED 13 กระพริบติดดับอย่างละ 1 วินาที

การติดตั้ง Library บน Arduino IDE

การติดตั้ง library ที่เรา download มาจากเว็บไซต์อย่างเช่น www.github.comเรียกว่า "Third Party Library" เมื่อซื้อShield หรือ Module ซักอันมาจากเว็บ Arduitronicsต้องลง Library เพื่อให้มันใช้งานได้อย่างสะดวกอย่างที่ผู้ออกแบบ Shield ตั้งใจไว้ เช่น motor shield, SD Module และ DHT11 เป็นต้น

โดยปกติเราจะได้ไฟล์มาเป็น .zip ก็ให้เรา unzip ไปไว้ใน folder ที่ชื่อเดียวกับไฟล์ที่ download มาจากนั้นย้ายไปเก็บไว้ใน folder ชื่อ libraries ซึ่งจะอยู่ภายใน folder ของ Arduino IDE ถ้าไม่เคยdownload library ใหม่นี้มาก่อน จะต้องสร้าง Folder ใหม่แล้วย้ายไปเก็บไว้ที่ Folder ที่สร้างขึ้น และให้ไปที่ Sketch -> Show Sketch Folder และBrowserเลือก Folder Libraries ที่ต้องการ เมื่อดำเนินการเสร็จเรียบร้อยแล้วให้ปิดและเปิดโปรแกรมใหม่ เพราะ Arduino IDE จะค้นหาและแสดง Libraries เฉพาะตอนเปิดโปรแกรมใหม่เท่านั้นถ้าลง Library ได้ถูกต้อง เวลาไปที่ Sketch -> Imported Library... จะเจอชื่อ Folder ของ Library ที่เราสร้างขึ้น ก็เป็นอันพร้อมใช้งาน



ภาคผนวก ค.
โปรแกรมสำหรับอุปกรณ์วัดความเร็วในการเคลื่อนที่

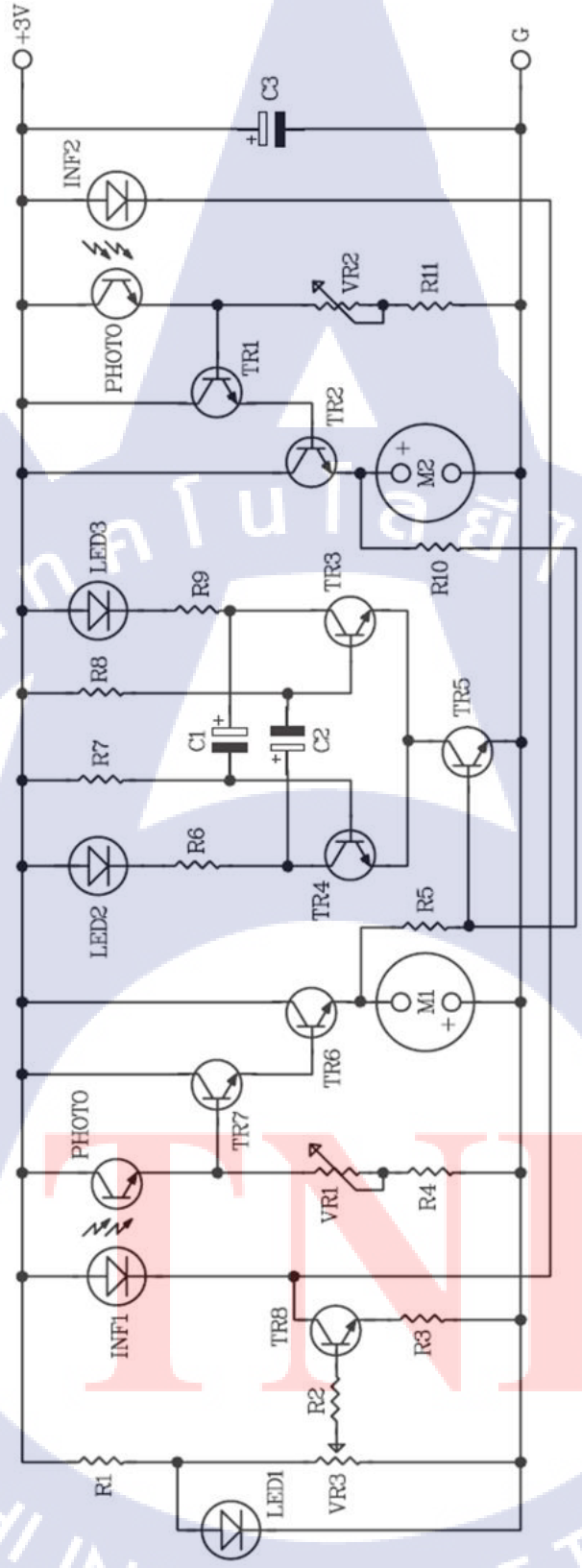
WheelSpeedometer.ino

```
/*
sample calculations
tire radius ~ 69 millimeters
circumference = pi * 2 * radius =~ 433.54 millimeters
distance = circumference / 5 = 86.71
max speed of 56 kph =~ 15,646.6 millimeters/second
max rps =~ 180.45
*/

#define sensor 8                    // pin connected to sensor

//storage variables
float radius = 69;                  // tire radius (in mm.)
booleansensorValue, signal = 1;
inttime = 0;                        // time between one full rotation (in ms)
inttimex, counter = 0;
float circumference, distance, kph = 0.00;

/*****************************************************************/
voidsetup()
{
   circumference = 2 * 3.14 * radius;
   distance = circumference / 5;
   pinMode(sensor,INPUT);
   Serial.write(12);                //clear
   cli();                           //stop interrupts

//set timer1 interrupt at 1kHz
   TCCR1A = 0;                      // set entire TCCR1A register to 0
   TCCR1B = 0;                      // same for TCCR1B
   TCNT1  = 0;                      //initialize counter value to 0

// set timer count for 1khz increments
   OCR1A = 1999;                    // (16*10^6) / (1000*8)  1
   TCCR1B |= (1 << WGM12);          // turn on CTC mode
   TCCR1B |= (1 << CS11);           // Set CS11 bit for 8 prescaler
   TIMSK1 |= (1 << OCIE1A);         // enable timer compare interrupt

   sei();                           //allow interruptsand END TIMER SETUP
   Serial.begin(9600);              // start serial connection
}

/*****************************************************************/
voidloop()
{
   Display_KPH();                   // print kph once a second
   delay(1000);
}

/*****************************************************************/
voiddisplay_KPH()
{
   Serial.write(12);                //clear
   Serial.print(timex);
   Serial.write(" Speed = ");
   Serial.print(kph);
   Serial.write(" KPH ");
}
```

```
/*********************************************************************/
ISR(TIMER1_COMPA_vect)
{
   sensorValue = digitalRead(sensor);
   time += 1;
   if (signal == 1 &sensorValue == 0)
   {
      signal = 0;
      counter += 1;
      if (counter == 5)
      {
         timex = time;
         kph = (3.6 * 5 * float(distance)) / float(time);
         counter = 0;
         time = 0;
     }
   }
   else
   {
      signal = sensorValue;
      if (time >= 4000)
      {
         timex = 0;
         kph = 0;
         counter = 0;
         time = 0;
      }
   }
}
```

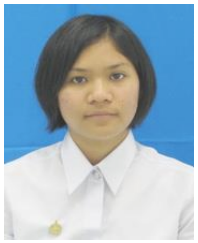
ภาคผนวก ง.

วงจรหุ่นยนต์ TACON วิ่งตามเส้น



วงจรหุ่นยนต์ TACON วิ่งตามเส้น

ประวัติผู้วิจัย



ชื่อ-นามสกุล นางสาวณฐภัทร ลวดเงิน

วัน เดือน ปีเกิด 8 มกราคม 2533

ประวัติการศึกษา

ระดับอุดมศึกษา คณะวิศวกรรมศาสตร์ สาขาวิศวกรรมคอมพิวเตอร์ พ.ศ. 2551
สถาบันเทคโนโลยีไทย-ญี่ปุ่น

ระดับมัธยมศึกษา มัธยมศึกษาตอนปลาย (สายวิทย์-คณิต) พ.ศ. 2550
โรงเรียนสิรินธร

ระดับประถมศึกษา ประถมศึกษา พ.ศ. 2544
โรงเรียนอนุบาลสุรินทร์

ทุนการศึกษา - ไม่มี -

ประวัติการฝึกอบรม RDC 2009

ผลงานที่ได้รับการตีพิมพ์ - ไม่มี -