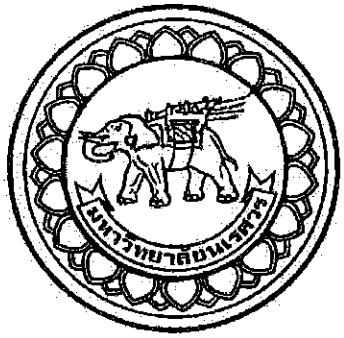


การควบคุมความเร็วของหุ่นยนต์โดยใช้ฟัซซีลอจิกเมื่อโหลดมีการเปลี่ยนแปลง

MOBILE ROBOT SPEED CONTROL USING FUZZY LOGIC UNDER VARIABLE LOAD

นางสาวประภาศรี คำลือ รหัส 55360925

นายนัฐพงศ์ ใบวุฒิ รหัส 55364053

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2558

ใบรับรองปริญญานิพนธ์

ชื่อหัวข้อโครงงาน	**การควบคุมความเร็วของหุ่นยนต์โดยใช้ฟัซซีลอจิกเมื่อโหลดมีการเปลี่ยนแปลง**
ผู้ดำเนินโครงงาน	นางสาวประภาศรี คำลือ รหัส 55360925
	นายนัฐพงศ์ ใบวุฒิ รหัส 55364053
ที่ปรึกษาโครงงาน	ผู้ช่วยศาสตราจารย์ ดร. มุฑิตา สงฆ์จันทร์
สาขาวิชา	วิศวกรรมไฟฟ้า
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2558

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมไฟฟ้า

.....ที่ปรึกษาโครงงาน
(ผู้ช่วยศาสตราจารย์ ดร.มุฑิตา สงฆ์จันทร์)

.....กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.ศุภวรรณ พลพิทักษ์ชัย)

.....กรรมการ
(ดร.สราวุฒิ วัฒนวงค์พิทักษ์)

ชื่อหัวข้อโครงงาน	การควบคุมความเร็วของหุ่นยนต์โดยใช้ฟัซซีลอจิกเมื่อโหลดมีการเปลี่ยนแปลง
ผู้ดำเนินโครงงาน	นางสาวประภาศรี คำลือ รหัส 55360925
	นายนัฐพงศ์ ใบวุฒิ รหัส 55364053
ที่ปรึกษาโครงงาน	ผู้ช่วยศาสตราจารย์ ดร. มุฑิตา สงฆ์จันทร์
สาขาวิชา	วิศวกรรมไฟฟ้า
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2558

.....

บทคัดย่อ

โครงงานนี้นำเสนอการควบคุมความเร็วของหุ่นยนต์โดยใช้การควบคุมแบบฟัซซีเป็นการอนุมานแบบแมมดานิ เพื่อให้ความเร็วของหุ่นยนต์มีค่าคงที่ตามที่กำหนดเมื่อโหลดของหุ่นยนต์มีการเปลี่ยนแปลงค่าตั้งแต่ 0 กิโลกรัมถึง 5 กิโลกรัม โดยค่าอินพุตของฟัซซีเป็นค่าการเปลี่ยนแปลงโหลดน้ำหนักที่ส่งค่ามาจากโหลดเซลล์ชนิดเสตรนเกจ และค่าเอาท์พุตของฟัซซีเป็นค่าดิวตี้ไซเคิลที่ใช้ในการควบคุมความเร็วของมอเตอร์ โดยใช้เซนเซอร์วัดความเร็วรอบมอเตอร์ และหลอดไฟแอลอีดีแสดงสถานะไฟติดเมื่อหุ่นยนต์มีความเร็วคงที่ตามที่กำหนด จากผลการทดลองเห็นได้ว่าการใช้ตัวควบคุมแบบฟัซซีในการควบคุมความเร็วของหุ่นยนต์ เมื่อโหลดมีการเปลี่ยนแปลงนั้นสามารถควบคุมความเร็วของหุ่นยนต์ให้มีค่าคงที่ตามที่กำหนดได้

Project title	Mobile Robot Speed Control using Fuzzy Logic under Variable Load
Name	Miss.Prapasri Khumlue ID. 55360925
	Mr.Nattapong Baiwut ID. 55364053
Project adviser	Asst. Prof. Mutita Songjun, Ph.D.
Major	Electrical Engineering
Department	Electrical and Computer Engineering
Academic year	2015

.....

Abstract

This project presents the mobile robot speed control using fuzzy logic under variable load. The fuzzy is designed to use mamdani is fuzzy inference method which is common and easy. The process is to control the robot is speed at the desired constant when the load is varied between 0 to 5 kilograms. The input of the fuzzy controller is the load variable measured by load cell. The output is the duty cycle used to control the motor is speed. The results show that it is capable to use fuzzy logic method to control the robot at the desired speed when the load is varied.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี ผู้วิจัยใคร่ขอขอบพระคุณบิดามารดาซึ่งมีส่วนช่วยในด้านกำลังทรัพย์และส่วนช่วยในด้านกำลังใจที่ทำให้ฝ่าฟันอุปสรรคต่างๆที่เกิดขึ้นระหว่างการดำเนินงานวิจัยครั้งนี้ ให้ผ่านไปได้อย่างราบรื่น

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร.มุฑิตา สงฆ์จันทร์ สำหรับคำปรึกษาและชี้แนะแนวทางในการทำโครงงานการออกแบบตัวควบคุมฟัซซีลอจิก รวมถึงข้อมูลและทฤษฎีต่างๆที่เกี่ยวข้องกับการออกแบบตัวควบคุมฟัซซีลอจิก ซึ่งได้นำมาประกอบในปริญญานิพนธ์ฉบับนี้

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร.ศุภวรรณ พลพิทักษ์ชัย สำหรับคำปรึกษาและชี้แนะแนวทางในการทำโครงงาน รวมถึงข้อมูลและทฤษฎีต่างๆ ซึ่งได้นำมาประกอบในปริญญานิพนธ์ฉบับนี้

ขอขอบพระคุณ ดร.สราวุฒิ วัฒนวงค์พิทักษ์ สำหรับคำปรึกษาและชี้แนะแนวทางในการทำโครงงานการออกแบบตัวควบคุมฟัซซีลอจิก รวมถึงข้อมูลและทฤษฎีต่างๆที่เกี่ยวข้องกับการออกแบบตัวควบคุมฟัซซีลอจิก ซึ่งได้นำมาประกอบในปริญญานิพนธ์ฉบับนี้

ท้ายที่สุด ผู้วิจัยใคร่ขอขอบคุณผู้ที่มีพระคุณที่ไม่ได้กล่าวถึงทุกท่าน ที่ต่างมีส่วนร่วมในการชี้แนะ ให้ข้อมูลและให้ความรู้เกี่ยวกับปริญญานิพนธ์ฉบับนี้จนสำเร็จลุล่วงออกมาตามวัตถุประสงค์ที่ผู้วิจัยต้องการมา ณ ที่นี้ด้วย

นางสาวประภาศรี คำลือ
นายนัฐพงศ์ ใบวุฒิ

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญตาราง (ต่อ)

ตารางที่ หน้า

สารบัญรูป

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของโครงงาน

ในระบบขนส่งภายในโรงงานอุตสาหกรรม หรือระบบขนส่งสายพานลำเลียง ต้องใช้การขนส่งที่มีประสิทธิภาพ มีการควบคุมระบบให้เสถียรภาพ เพื่อลดต้นทุน เวลา และค่าใช้จ่าย ซึ่งปัจจัยเหล่านี้ขึ้นอยู่กับความเร็วของระบบในการขนส่ง และความเร็วของระบบนี้ก็มีผลมาจากโหลดน้ำหนักของสินค้า หากสินค้ามีน้ำหนักมาก ความเร็วในการขนส่งของระบบก็จะช้าลง ทำให้เสียเวลาในการขนส่ง ในปัจจุบันเทคโนโลยีด้านคอมพิวเตอร์ มีความก้าวหน้าอย่างมาก ซึ่งช่วยในการทำงานได้อย่างแม่นยำ ลดค่าใช้จ่าย รวมถึงลดเวลาในการปฏิบัติงานได้เป็นอย่างมากซึ่งระบบจะถูกควบคุมด้วยระบบฝังตัว หรือสมองกลฝังตัว (Embedded system) คือระบบประมวลผลที่ใช้ชิป หรือไมโครโปรเซสเซอร์ที่ออกแบบมาโดยเฉพาะ ดังนั้น ระบบสมองกลฝังตัวจะมีการทำงานร่วมกับอุปกรณ์ภายนอกอื่นๆอย่างกว้างขวาง จนสามารถนำไปประยุกต์ใช้ในการสร้างหุ่นยนต์ โดยใช้ความรู้ด้านการเขียนโปรแกรมในการทำงานของหุ่นยนต์ได้ จึงเห็นว่าเทคโนโลยีสมัยนี้มีบทบาทความสำคัญที่สามรถนำมาประยุกต์ใช้งานเพื่อแก้ปัญหาเรื่องการควบคุมความเร็วของรถยนต์ให้คงที่ เมื่อน้ำหนักที่บรรทุกมีการเปลี่ยนแปลง

ดังนั้นจึงจัดทำโครงงานการควบคุมความเร็วของหุ่นยนต์โดยใช้ฟัซซี เมื่อโหลดมีการเปลี่ยนแปลงขึ้น และสร้างหุ่นยนต์จำลองที่สามารถรับน้ำหนักได้ไม่เกิน 5 กิโลกรัม ที่สามารถควบคุมความเร็วมอเตอร์ของหุ่นยนต์ด้วยไมโครคอนโทลเลอร์เพื่อให้มีความเร็วคงที่ตามค่าที่กำหนด เมื่อโหลดมีการเปลี่ยนแปลง

1.2 วัตถุประสงค์ของโครงงาน

เพื่อสร้างหุ่นยนต์ที่สามารถควบคุมความเร็วได้โดยใช้ตัวควบคุมแบบฟัซซีเมื่อโหลดของหุ่นยนต์มีการเปลี่ยนแปลง

1.3 ขอบเขตของโครงงาน

1. สร้างหุ่นยนต์ที่มีขนาดความกว้าง 18 เซนติเมตร และความยาว 27 เซนติเมตร
2. สร้างหุ่นยนต์ที่มีความเร็วไม่เกิน 2 เมตร/วินาที โดยใช้มอเตอร์กระแสตรงขนาด 12 โวลต์
3. สร้างหุ่นยนต์ที่สามารถบรรทุกน้ำหนักได้ตั้งแต่ 0 - 5 กิโลกรัม
4. ใช้วิธีควบคุมแบบฟัซซีลอจิกผ่านไมโครคอนโทรลเลอร์ตระกูล AVR เบอร์ ATmega2560
5. โหลดเซลล์ที่ใช้เป็นประเภทสเตรนเกจ

1.4 ขั้นตอนและแผนการดำเนินงาน

ตาราง 1.1 ขั้นตอนและแผนการดำเนินงาน

รายละเอียด	พ.ศ. 2558					พ.ศ. 2559				
	ส.ค	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค	ก.พ.	มี.ค.	เม.ย.	พ.ค.
1. รวบรวมข้อมูล										
2. ศึกษาการทำงานของหุ่นยนต์และการใช้ฟัซซี										
3. สร้างตัวหุ่นยนต์										
4. เขียนโปรแกรมควบคุมหุ่นยนต์และทำการทดสอบ										
5. ปรับปรุงแก้ไขโปรแกรมควบคุมการทำงาน										
6. สรุปผลการดำเนินงาน										
7. จัดทำปริญญานิพนธ์ฉบับสมบูรณ์										

1.5 ประโยชน์ที่ได้รับจากโครงงาน

สามารถนำงานไปใช้พัฒนา และประยุกต์ใช้กับระบบขนส่งภายในโรงงานอุตสาหกรรมระบบสายพานลำเลียง และระบบควบคุมที่ต้องการให้ความเร็วคงที่ เพื่อประหยัดเวลาและลดต้นทุนในการขนส่งได้

1.6 งบประมาณ

1. ค่าแผงวงจรต่างๆ	3,200	บาท
2. ค่าล้อและมอเตอร์	1,330	บาท
3. ค่าวัสดุอุปกรณ์โครงสร้าง	1,000	บาท
4. ค่าแบตเตอรี่	780	บาท
5. ค่าเอกสาร	1,200	บาท
6. ค่าวัสดุอื่นๆ	1,000	บาท
รวมเป็นเงินทั้งสิ้น (แปดพันห้าร้อยสิบบาทถ้วน)	8,510	บาท

หมายเหตุ: ถัวเฉลี่ยทุกรายการ

บทที่ 2

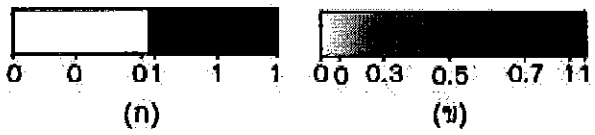
ทฤษฎีและหลักการที่เกี่ยวข้อง

ในบทนี้จะรวบรวมหลักการทำงาน และทฤษฎีขององค์ประกอบที่มีความจำเป็นต่อการทำงานของหุ่นยนต์ควบคุมความเร็ว ซึ่งในแต่ละองค์ประกอบนั้นจะมีการทำงานที่สัมพันธ์กันของโครงสร้างของหุ่นยนต์ควบคุมความเร็ว ทั้งระบบขับเคลื่อนหุ่นยนต์ และระบบควบคุมความเร็ว

2.1 ทฤษฎีฟัซซี

โดยปกติแล้วผู้เชี่ยวชาญจะใช้สามัญสำนึกในการแก้ปัญหา ยิ่งไปกว่านั้นภาษาที่ใช้จะมีความคลุมเครือ ยกตัวอย่าง เช่น "ท่าทางมอเตอร์จะรับงานหนักไปหน่อย แต่ก็จะให้ทำงานต่อไปอีกสักนิด" หรือ "This motor seems to be a little bit overloaded, but it can still work for a while" (ฟัซซีลอจิกใช้ภาษาอังกฤษเป็นหลักในระบบ) ถึงแม้ว่าการสื่อสารดังกล่าวจะมีความคลุมเครือในความหมายเมื่อพิจารณาในเชิงปริมาณตัวเลข แต่ผู้เชี่ยวชาญในด้านเดียวกันหรือผู้ที่ทำงานด้วยก็สามารถเข้าใจและทำงานตามที่เนื้อหาของการสื่อสารได้ ฐานความรู้ที่ผู้เชี่ยวชาญใช้ดังกล่าวมีความแตกต่างเมื่อนำไปใช้ในระบบคอมพิวเตอร์ ที่ซึ่งมีการใช้ข้อมูลในระบบที่มีความแน่นอนหรือชัดเจน ตรรกศาสตร์คลุมเครือหรือฟัซซีลอจิก (fuzzy logic) ใช้ในการอธิบายความคลุมเครือหรือความไม่ชัดเจนดังกล่าว ทฤษฎีฟัซซีเซต (fuzzy set theory) ใช้ทฤษฎีของเซตในการแทนระดับความคลุมเครือ ดังนั้นปริมาณทุกๆ อย่างในระบบทางวิศวกรรม (และระบบอื่นๆ) เช่นอุณหภูมิ ความสูง ความเร็ว ระยะทาง ความงาม ความดัน ฯลฯ สามารถถูกอธิบายด้วยระดับของความคลุมเครือได้ ยกตัวอย่างเช่น This room is very hot – ห้องร้อนมากๆ, RC car is not very fast – รถบังคับวิ่งไม่ค่อยเร็ว, Korat is quite a short distance from Bangkok – โคราชค่อนข้างใกล้กับกรุงเทพการใช้ระดับความคลุมเครือดังกล่าว ทำให้เราไม่สามารถระบุความเป็นสมาชิกของกลุ่มนั้นๆได้อย่างเด็ดขาด ห้องที่ร้อนจะต้องมีอุณหภูมิเท่าไรถึงจะเรียกว่าร้อนมากๆได้ ปกติความสูงแค่ไหนที่จะทำให้เราตัดสินว่าเป็นคนสูง ระบบตรรกะแบบดั้งเดิมสามารถที่จะระบุความเป็นสมาชิก (membership) โดยใช้ตัวเลขที่ชัดเจน (crisp number) ซึ่งจะต้องทำการแบ่งระบบออกเป็นสองกลุ่ม คือใช่ (ร้อน, สูง, เร็ว) กับไม่ใช่ (เย็น เตี้ย ช้า) กล่าวคือ ข้อมูลสามารถเป็นสมาชิกของกลุ่มได้เพียง

แค่ใช่หรือไม่ใช่ ยกตัวอย่างเช่น อุณหภูมิมากกว่า 25 องศาถือเป็นร้อน อุณหภูมิต่ำกว่า 25 องศาถือเป็นเย็น โดยการใช้ระบบดังกล่าวเราสามารถอธิบายระบบได้เพียงสองอย่าง ดังนั้น อุณหภูมิ 25 26 หรือ 40 องศาถือเป็นร้อน ในขณะที่อุณหภูมิ 24 23 หรือ 0 องศาถือเป็นเย็น ซึ่งในความเป็นจริง



รูปที่ 2.1 (ก) ตรรกะบูลีน (ข) ตรรกะหลายระดับ

แล้วอุณหภูมิ 24 องศาอาจจะไม่ถือเป็น 'เย็น' ในขณะที่อุณหภูมิ 26 ไม่น่าที่จะถือเป็น 'ร้อน' ได้เสมอไป ฟัซซีลอจิกนำเสนอแนวคิดในแบบมนุษย์ ซึ่งทำการจำลองภาษาคำพูด การตัดสินใจหรือสามัญสำนึกของมนุษย์ ทำให้ระบบ มีความฉลาดเหมือนมนุษย์ได้ (intelligent system) ฟัซซี (fuzzy) หรือตรรกะหลายระดับ (multi-valued logic) (ดูรูปที่ 2.1) ถูกนำเสนอโดยนักตรรกศาสตร์และปรัชญาชาวโปแลนด์ชื่อแจน วูคาซีวิช (Jan Lukasiewicz) ในช่วงศตวรรษที่ 1930 ต่อมาในปี 1965 ศาสตราจารย์ลอตฟี ซีเดย์ (Lotfi Zadeh) ซึ่งในขณะนั้นเป็นหัวหน้าภาควิชาวิศวกรรมอิเล็คทรอนิกส์ ณ มหาวิทยาลัยแคลิฟอเนียร์เบิร์กเลย์ ประเทศสหรัฐอเมริกา ได้ตีพิมพ์งานในเรื่อง "ฟัซซีเซต (Fuzzy Sets)" ซึ่งทำให้งานทางฟัซซีลอจิกเป็นที่รู้จัก และได้รับความนิยมอย่างกว้างขวางในเวลาต่อมา ฟัซซีลอจิกเป็นเพียงส่วนหนึ่งจากทฤษฎีฟัซซีเซต ฟัซซีลอจิกจึงแตกต่างไปจากลอจิกบูลีนที่ซึ่งมีเพียงใช่-ไม่ใช่ หรือศูนย์-หนึ่ง แต่ฟัซซีลอจิกเป็นตรรกะหลายระดับ ซึ่งเกี่ยวข้องกับการกำหนดค่าระดับความเป็นสมาชิก (degree of membership) โดยใช้ค่าตัวเลขตั้งแต่ศูนย์ ถึงหนึ่ง (ค่าศูนย์ยังคงแทนความไม่ใช่สมาชิกของกลุ่ม ในขณะที่ค่าหนึ่งแทนความใช่สมาชิก ในขณะที่ค่าระหว่าง ศูนย์จนถึงหนึ่งแสดงระดับความเป็นสมาชิกของกลุ่มจากน้อยไปมาก)

2.1.1 ข้อดีของฟัซซีลอจิก

ฟัซซีลอจิกมีคุณลักษณะเด่นหลายๆ อย่าง ทำให้มีการนำเอาฟัซซีลอจิกมาประยุกต์ใช้อย่างมากมาย และอย่างมีประสิทธิภาพ โดยเฉพาะงานทางด้านระบบควบคุม ข้อดีของฟัซซีลอจิกสรุปคร่าวๆ ได้ดังนี้

1. ฟัซซีลอจิกเป็นระบบที่มีเสถียรภาพสูง ไม่จำเป็นจะต้องใช้งานกับระบบที่มีอินพุตที่มีค่าแน่นอนหรือปราศจากสัญญาณรบกวน กล่าวคือระบบสามารถรองรับอินพุตที่มีความคลุมเครือได้อย่างหลากหลาย
2. ฟัซซีลอจิกประมวลผลด้วยการใช้กฎที่กำหนดหรือนิยามด้วยผู้ใช้ (หรือผู้สร้างระบบซึ่งคือผู้เชี่ยวชาญนั่นเอง) ดังนั้นจึงเป็นการสะดวกในการปรับแต่งระบบเพื่อเพิ่มประสิทธิภาพ
3. ฟัซซีลอจิกไม่มีข้อจำกัดของจำนวนอินพุตหรือเอาท์พุต ทำให้การออกแบบระบบสามารถทำได้หลากหลายสามารถใช้ตัวตรวจจับที่ไม่มีความแม่นยำมากนักและมีราคาถูกได้พร้อมๆ กันหลายๆตัว เพื่อเพิ่มประสิทธิภาพของระบบในขณะที่ความยุ่งยากและราคารวมของระบบไม่เพิ่มขึ้น
4. ฟัซซีลอจิกมีโครงสร้างที่สามารถแบ่งแยกเป็นหน่วยประมวลผลย่อยๆได้ ทำให้ได้ระบบที่มีการกระจายการทำงานง่ายต่อการดูแลและปรับปรุงแก้ไข
5. ฟัซซีลอจิกสามารถใช้กับงานที่ไม่เป็นเชิงเส้น (nonlinear) ได้ ทำให้ลดภาระการคำนวณหาแบบจำลองระบบทางคณิตศาสตร์ที่ซับซ้อน

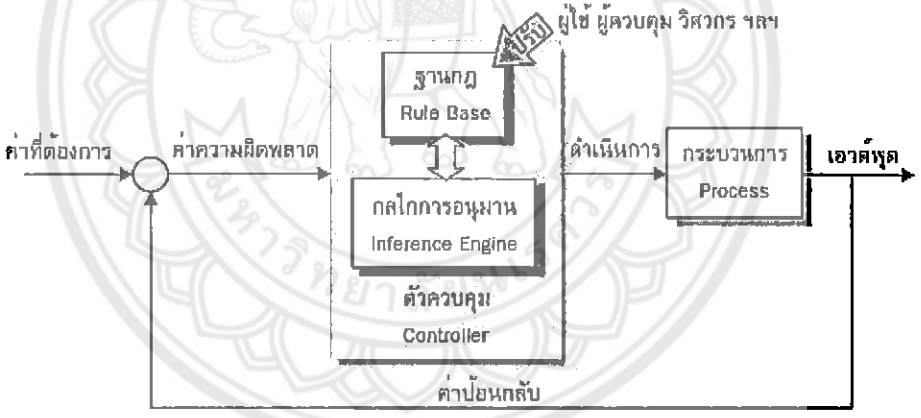
2.1.2 ฟัซซีลอจิกกับตัวควบคุมในงานทางวิศวกรรม

ในหัวข้อนี้จะได้กล่าวถึงฟัซซีลอจิกเบื้องต้น โดยใช้ตัวอย่างการออกแบบตัวควบคุมแบบฟัซซี (fuzzy controller) สำหรับการอธิบายเนื้อหาโดยปกติแล้วเราจะคุ้นเคยกับการออกแบบตัวควบคุมแบบสัดส่วน-ปริพันธ์-อนุพันธ์ (PID : Proportional-Integral Derivative) ด้วยวิธีต่างๆ ได้อย่างไม่ยุ่งยาก รายละเอียดของหัวข้อนี้จะเน้นการออกแบบตัวควบคุมแบบฟัซซีอย่างง่ายๆ

ตัวควบคุมแบบฟัซซีถูกนำไปประยุกต์ใช้งานอย่างแพร่หลายในเครื่องใช้ไฟฟ้าต่างๆ เช่นเครื่องซักผ้า กล้องวิดีโอ หม้อหุงข้าว ตู้เย็น รวมไปถึงกระบวนการในอุตสาหกรรมแบบต่างๆ ตัวควบคุมแบบฟัซซีใช้หลักของฟัซซีลอจิกในการควบคุมองค์ประกอบต่างๆ ตัวของฟัซซีลอจิกเองสามารถ “คำนวณด้วยคำพูดแทนที่จะเป็นตัวเลข” เช่น “มากขึ้นนิด” หรือ “ลดลงหน่อย” ไม่ใช่ “มากขึ้น 10” หรือ “ลดลง 5.2” เป็นต้น ดังนั้นตัวควบคุมแบบฟัซซี จึงกล่าวได้ว่าสามารถ “ควบคุมด้วยประโยคแทนที่จะเป็นสมการคณิตศาสตร์” เช่น “อินพุตกำลังลดลง ให้ปรับ เอาท์พุตมากขึ้นหน่อย” ไม่ใช่ “อินพุตมีค่าเท่ากับ 2 ให้ปรับเอาท์พุตเท่ากับ 7” จะเห็นได้ว่าตัวควบคุมแบบฟัซซีมีลักษณะของการใช้ฐานกฎ (rule base) ที่มาจากการควบคุมด้วยคนจริงๆ ได้ ตัวควบคุมแบบนี้จึงมี

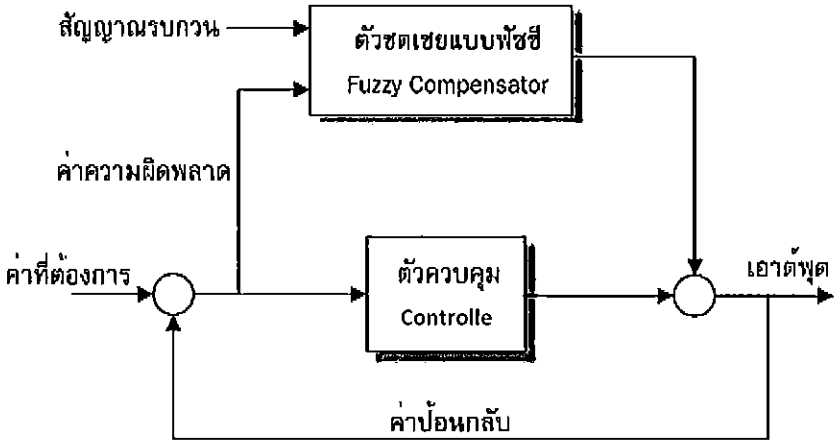
ประโยชน์มาก ในการประยุกต์ใช้กับงานเชิงปฏิบัติการจริงด้วยผู้เชี่ยวชาญ รายละเอียดของหัวข้อนี้จะได้ทำการระบุและอธิบายถึงแนวทางการออกแบบตัวควบคุมต่างๆ ในเชิงวิศวกรรมในตัวควบคุมแบบฐานกฎทั่วๆไป กลยุทธ์ในการควบคุมนั้นจะอยู่ในรูปแบบภาษาธรรมชาติเหมือนกับที่วิศวกรทั่วๆไปใช้พูดสั่งงาน กฎต่างๆ ที่ใช้จึงต้องถูกคัดแยกออกจากส่วนที่เป็นสมการคณิตศาสตร์ (ในตัวควบคุมแบบดั้งเดิม) ตัวอย่างรูปแบบการใช้งานตัวควบคุมแบบฟัซซีมี ดังต่อไปนี้

1. การควบคุมโดยตรง (direct control) ตัวอย่างของตัวควบคุมแบบฟัซซีดังกล่าวแสดงในรูปที่ 2.2 ที่ซึ่งเป็นการควบคุมโดยตรง จะเห็นได้ว่าตัวควบคุมแบบฟัซซีจะอยู่ในส่วนหน้าก่อนที่เอาท์พุตจะถูกป้อนกลับ เอาท์พุตที่ได้จากกระบวนการจะถูกเปรียบเทียบกับค่าอินพุตที่ตั้งไว้ ถ้ามีค่าความผิดพลาดเกิดขึ้น นั่นคือ เอาท์พุตไม่ตรงหรือไม่สอดคล้องกับค่าอินพุตที่ต้องการ ตัวควบคุมจะดำเนินการอย่างใดอย่างหนึ่ง ตามกลวิธีที่กำหนดหรือออกแบบไว้ใน



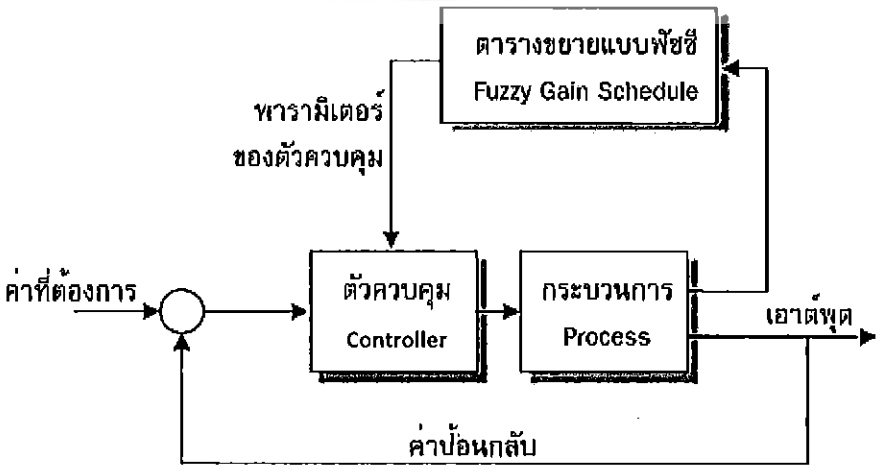
รูปที่ 2.2 การควบคุมโดยตรง

2. การควบคุมแบบไปข้างหน้า (feedforward control) รูปที่ 2.3 แสดงการนำเอาฟัซซีลอจิกมาเป็นตัวชดเชยการทำงานของตัวควบคุมในการควบคุมแบบไปข้างหน้า ตัวชดเชยแบบฟัซซีใช้สัญญาณรบกวนเป็นข้อมูลในการตัดสินใจว่าจะทำการชดเชย ให้กับตัวควบคุมขนาดไหน ตัวควบคุมในระบบอาจจะเป็นตัวควบคุมแบบสัดส่วน-ปริพันธ์-อนุพันธ์ (PID : Proportional-Integral Derivative) แบบเชิงเส้น ในขณะที่ตัวชดเชยแบบฟัซซีจะทำการชดเชยการทำงานของตัวควบคุมในลักษณะที่ไม่เป็นเชิงเส้น



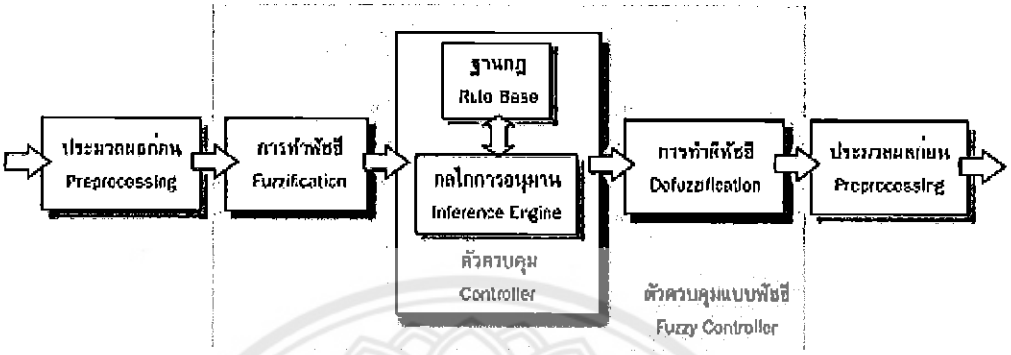
รูปที่ 2.3 การควบคุมแบบไปข้างหน้า

3. การควบคุมค่าพารามิเตอร์เชิงปรับตัว (parameter adaptive control) ในกรณีที่ระบบที่ไม่เป็นเชิงเส้นมีจุดทำงานเปลี่ยนแปลงไปจากค่าเริ่มต้นที่ตั้งไว้ เราสามารถที่จะปรับเปลี่ยนพารามิเตอร์ต่างๆ ของตัวควบคุมให้สอดคล้องกับจุดทำงานใหม่ได้ด้วยการจัดตารางการขยายแบบฟัซซี (fuzzy gain scheduling) ตัวควบคุม ที่มีการจัดตารางการขยายแบบฟัซซีจะประกอบไปด้วยตัวควบคุมแบบเชิงเส้น ที่ซึ่งมีค่าของพารามิเตอร์เปลี่ยนแปลงไปจากจุดทำงานเดิม อินพุตที่วัดจากตัวตรวจจับจะถูกใช้เป็นตัวแปรการจัดตาราง (scheduling variable) ที่ซึ่งใช้ในการปรับค่าพารามิเตอร์ของตัวควบคุมเดิม การปรับค่าดังกล่าวจะอยู่ในรูปของตารางค้นหา (look-up table) รูปที่ 2.4 แสดงแผนผังทั่วไปของการควบคุมค่าพารามิเตอร์เชิงปรับตัวแบบฟัซซี



รูปที่ 2.4 การควบคุมค่าพารามิเตอร์เชิงตัวปรับ

รายละเอียดในหัวข้อนี้จะได้กล่าวถึงการออกแบบสร้างตัวควบคุมแบบฟัซซี เพื่อใช้เป็นตัวควบคุมเชิงเส้น การออก แบบตัวควบคุมฟัซซีมีข้อดีตรงที่ไม่จำเป็นจะต้องทำการคำนวณเหมือนกับเช่นวิธีราก-โลคัส (root-locus) วิธีผลตอบสนองความถี่ (frequency response)



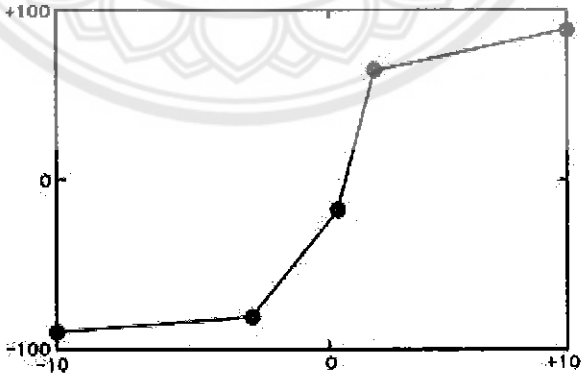
รูปที่ 2.5 โครงสร้างตัวควบคุมแบบฟัซซี

หรือวิธีวางตำแหน่งโพล (pole placement) กฎที่ใช้ในตัวควบคุมแบบฟัซซีสามารถไม่เป็นเชิงเส้นได้ รายละเอียดต่อไปนี้จะได้กล่าวถึงองค์ประกอบต่างๆ ของตัวควบคุมแบบฟัซซีตาม แนวทางสำหรับออกแบบในเชิงวิศวกรรม รูปที่ 2.5 แสดงโครงสร้างของตัวควบคุมแบบฟัซซี องค์ประกอบในการประมวลผลก่อนและหลัง เป็นการปรับสภาพอินพุตและเอาท์พุตที่จะใช้กับตัวควบคุมแบบฟัซซีให้มีความเหมาะสมรายละเอียดของแต่ละองค์ประกอบสรุปได้ดังต่อไปนี้

1. การประมวลผลก่อน (preprocessing) เป็นขั้นตอนที่ใช้ในการเตรียมอินพุตจากโลกจริง ให้มีความเหมาะสมที่จะใช้กับตัวควบคุมในโลกของฟัซซี (ไม่รวมขั้นตอนการทำให้เป็นฟัซซี) โดยปกติแล้วอินพุตของระบบ จะเป็นค่าเชิงตัวเลขที่วัดหรือออกมาจากเครื่องมือต่างๆ และไม่ได้มีค่าในรูปภาษา จึงจำเป็นจะต้องมีการประมวลผลก่อน เพื่อปรับค่าอินพุตเหล่านี้ให้มีความเหมาะสม ตัวอย่างของการประมวลผลก่อน เช่น
   - แปลงค่าจากสัญญาณแอนาล็อกให้เป็นสัญญาณดิจิตอล
   - ปัดค่าตัวเลขให้อยู่รูปที่ระบบรองรับ (เช่นปัดเป็นจำนวนเต็ม)
   - ค่าให้เป็นบรรทัดฐาน (normalization) ในย่านเฉพาะที่ต้องการ
   - กรองหรือกำจัดสัญญาณรบกวน
   - คำนวณหาค่าอนุพันธ์หรือปริพันธ์ (ใช้เป็นอินพุตเพิ่มเติม)

รายละเอียดของขั้นตอนการประมวลผลก่อนข้างต้นมีผลต่อประสิทธิภาพ หรือการทำงานของตัวควบคุมแบบฟัซซีโดยตรง ค่าสัญญาณดิจิตอลที่ละเอียดย่อมทำให้ตัวควบคุมทำงานได้อย่างราบเรียบว่าค่าที่หยาบ การปรับค่าให้เป็นบรรทัดฐานอาจส่งผลต่อการกำหนดตัวแปรในระบบฟัซซี ('small' 'medium' และ 'large' ฯลฯ) ได้ ดังตัวอย่างแสดงในรูปที่ 2.6 อินพุตแต่ละช่วงจะถูกปรับค่าให้เป็นบรรทัดฐานที่แตกต่างกัน ทำให้เกิดความไม่ต่อเนื่องของอินพุต (แสดงด้วยจุดกลม) ส่วนค่าอื่นๆ ที่ได้จากการประมวลผลก่อนไม่ว่าจะเป็นค่าอนุพันธ์ (derivation) หรือปริพันธ์ (integration) ทำให้จำนวนอินพุตของตัวควบคุมเพิ่มมากขึ้น แน่นอนว่า จำนวนกฎที่ต้องออกแบบสำหรับตัวควบคุมจะเพิ่มมากขึ้นด้วย

2. การทำฟัซซี (fuzzification) ค่าอินพุตที่ได้จากการประมวลผล ก่อนจะถูกแปลงให้เป็นค่าความเป็นสมาชิกจากฟังก์ชันสมาชิกต่างๆ ที่มีอยู่ในระบบ แล้วทำการรวมผลลัพธ์ของอินพุตนั้น ตามเงื่อนไข (ตัวแปรภาษา) ที่ถูกออกแบบไว้
3. ฐานกฎ (rule base) กฎในระบบฟัซซีถือเป็นหัวใจในการดำเนินการควบคุม กฎดังกล่าวสามารถมาจากเงื่อนไขที่หลากหลาย รวมไปถึงสามารถให้ผลลัพธ์ที่มากกว่า 1 ผลลัพธ์ได้ ตัวควบคุมที่มีอินพุตและเอาท์พุต มากกว่าหนึ่งจะเรียกว่า MIMO (Multi-Input Multi-Output) ในขณะที่ตัวควบคุมที่มีเพียงหนึ่งอินพุตและเอาท์พุตจะเรียกว่า SISO (Single-Input Single-Output) โดยปกติแล้ว ระบบที่เป็น SISO จะทำการควบคุม



รูปที่ 2.6 การปรับค่าให้เป็นบรรทัดฐานแบบไม่เชิงเส้นของค่าอินพุต

สัญญาณค่าความผิดพลาดเพียงอย่างเดียว ในบางกรณีอาจจะมีการใช้ค่าอัตราการเปลี่ยนแปลงหรือค่าสะสม ของค่าความผิดพลาดร่วมด้วย แต่จะยังคงเรียกว่าเป็นอินพุต

เดียว เนื่องจากทั้งอัตราการเปลี่ยนแปลงหรือ ค่าสะสมดังกล่าวนั้นมาจากอินพุตค่าความผิดพลาดเพียงค่าเดียว แนวคิดของการใช้ฐานกฎในฟัซซีลอจิกทำ ให้ระบบที่ได้มีความใกล้เคียงกับการทำงานจริงของมนุษย์ หรือกล่าวได้ว่าเป็นผู้เชี่ยวชาญนั่นเอง

4. กลไกการอนุมาน (inference engine) กฎต่างๆ ที่กำหนดไว้จะถูกอนุมานเป็นผลลัพธ์ในการตัดสินใจของ ระบบ เมื่อระบบตัดสินใจได้แล้ว การดำเนินการที่สอดคล้องกับการตัดสินใจนั้นก็จะดำเนินต่อไป ยกตัวอย่าง เช่นระบบตรวจจับได้ว่าอุณหภูมิจากตัวตรวจจับที่ 1 กำลัง 'ร้อนขึ้น' อย่าง 'รวดเร็ว' ระบบจะทำการพิจารณา ค่าอินพุตพร้อมกับตรวจสอบกับกฎการทำงาน ที่สอดคล้องกับเงื่อนไขดังกล่าว แล้วทำการอนุมานหรือตัดสินใจว่าจะทำการเปิดเครื่องทำความเย็น 'แรงที่สุด' เป็นต้น ผลลัพธ์การตัดสินที่ได้ยังคงอยู่ในเทอมของค่าเชิงภาษา ที่ซึ่งจะถูกแปลงเป็นค่าที่ใช้งานจริงด้วยขั้นตอนต่อไป
5. การทำดีฟัซซี (defuzzification) ผลลัพธ์เชิงภาษาที่ได้จากกลไกการอนุมานจะอยู่ในรูป เช่น เปิดเครื่องทำความเย็น 'แรงที่สุด' หรือลดเครื่องทำความร้อน 'ลงพอประมาณ' ฯลฯ ผลลัพธ์ดังกล่าวจะถูกแปลงให้เป็นค่า ที่สอดคล้องกับการทำงานจริงของระบบ เช่น เปิดเครื่องทำความเย็นเพิ่มขึ้น 25% เป็นต้น
9. การประมวลผลตาม (postprocessing) เอาท์พุตที่ได้จากระบบอาจจะต้องถูกปรับให้เหมาะสมกับการนำไปใช้งาน ไม่ว่าจะเป็นการทำให้เป็นบรรทัดฐาน (normalization) ในย่านที่ใช้งานจริง เช่น แปลงค่า 0 - 100% เป็นแรงดันขนาด -5 ถึง +5 โวลต์ สำหรับควบคุมให้เครื่องทำความเย็นเปิดปิดตามปริมาณที่ต้องการ

2.1.3 กฎของฟัซซี

กฎของฟัซซีเป็นวิธีการนำเอาความรู้ของมนุษย์มาใส่ในระบบฟัซซีลอจิก กฎของฟัซซีคือกลุ่มของประโยคเงื่อนไข ถ้า-แล้ว หรือ IF-THEN ในรูปแบบต่อไปนี้

ถ้า x เท่ากับ A แล้ว y เท่ากับ B หรือ IF x is A THEN y is B

โดยที่ x และ y เป็นตัวแปรภาษาและ A และ B เป็นค่าเชิงภาษา โดยปกติแล้วกฎของฟัซซีจะครอบคลุมค่าของ ตัวแปรที่อยู่ในส่วนเงื่อนไข IF ยกตัวอย่างเช่นระบบควบคุมอุณหภูมิที่ซึ่งมีค่าของตัวแปรอุณหภูมิที่เป็นไปได้คือ {'เย็น' 'กำลังดี' 'ร้อน'} ดังนั้นส่วนเงื่อนไขในกฎของฟัซซีจะครอบคลุมค่าดังกล่าวเช่น

กฎ 1: ถ้า อุณหภูมิเท่ากับ เย็น แล้ว เอาท์พุต เป็น ให้ความร้อน

กฎ 2: ถ้า อุณหภูมิเท่ากับ ร้อน แล้ว เอาท์พุต เป็น ให้ความเย็น

กฎ 3: ถ้า อุณหภูมิเท่ากับ กำลังดี แล้ว เอาท์พุต เป็น ไม่เปลี่ยนแปลง

กฎของฟัซซีประกอบไปด้วยสองส่วนหลักคือส่วน IF และส่วน THEN ในทฤษฎีดั้งเดิม เมื่อค่าเงื่อนไขใน IF เป็นจริง ส่วน THEN จะถูกประเมิน แต่ในทฤษฎีฟัซซีค่าเงื่อนไขใน IF จะมีความเป็นฟัซซีในระดับหนึ่ง ส่วน THEN จะถูกประเมินค่าด้วยค่าระดับความเป็นสมาชิก ซึ่งจะให้ค่าที่สัมพันธ์ในระดับนั้นๆ ด้วยค่าเงื่อนไขในส่วนของ IF ยังสามารถมีได้หลายค่า (เช่นเดียวกันกับส่วน THEN) ดังรูปแบบต่อไปนี้

IF x is A

AND y is B

OR z is C

THEN p is D

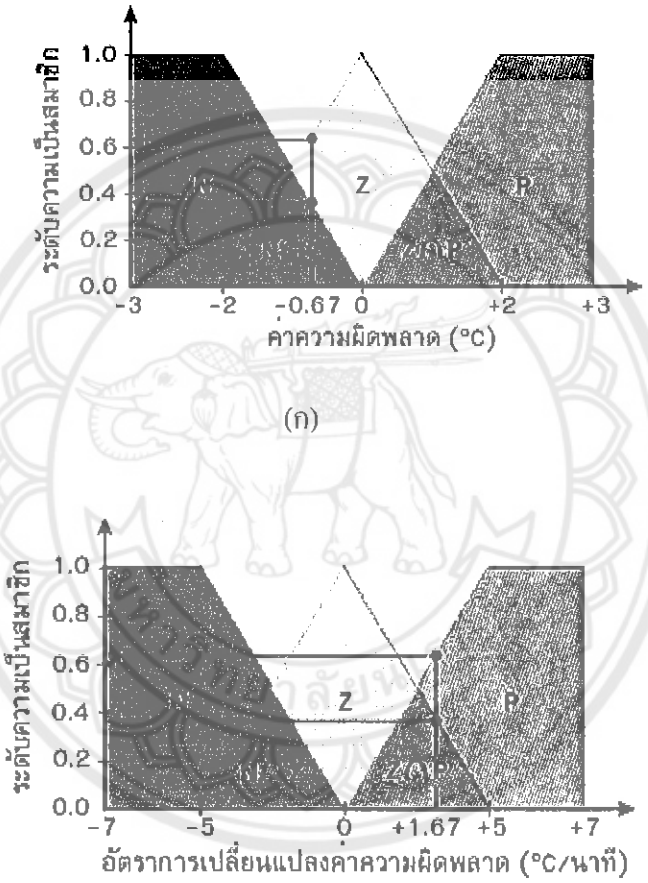
Q is E

ทุกเงื่อนไขในส่วน IF จะถูกประเมินพร้อมๆ กันและรวมกันด้วยปฏิบัติการทางเซต (เช่น AND หรือ OR) โดยปกติ แล้วเรามักจะจำกัดจำนวนค่าเงื่อนไขในระบบไม่ให้มีมากเกินไป โดยการเลือกใช้กฎที่จำเป็นเท่านั้น (บางกฎอาจจะไม่สามารถมีโอกาสเกิดขึ้น) เพราะจะทำให้เพิ่มความยุ่งยากในการออกแบบกฎของฟัซซีต่อไปภายหลัง สังเกตว่า จำนวนเงื่อนไขดังกล่าวจะขึ้นอยู่กับค่าของตัวแปรภาษาภายในระบบนั่นเอง

2.1.4 การอนุมานฟัซซีแบบแมมดานิ (Mamdani)

การอนุมานฟัซซีแบบแมมดานิเป็นวิธีที่นิยมมากวิธีหนึ่ง วิธีการอนุมานนี้นำเสนอเป็นครั้งแรกในปี 1975 โดย ศาสตราจารย์มอมดานี (Ebrahim Mamdani) แห่งมหาวิทยาลัยลอนดอน ซึ่งในครั้งแรกที่นำเสนอนั้นได้ใช้ในการควบคุมเครื่องจักรไอน้ำและหม้อต้มไอน้ำ (boiler) ในงานที่นำเสนอนั้นมีการประยุกต์ใช้กฎของฟัซซีที่สร้างจากผู้เชี่ยวชาญ ขบวนการอนุมานฟัซซีแบบแมมดานิ ประกอบไปด้วย 4 ขั้นตอน คือ การทำฟัซซี , การประเมินกฎของฟัซซี , การรวมกฎ และการทำดีฟัซซี ดังรายละเอียดต่อไปนี้ (พิจารณาระบบควบคุมอุณหภูมิเป็นตัวอย่างในการอธิบาย)

1. การทำฟัซซี (fuzzification) การทำฟัซซี คือการคำนวณหาค่าระดับความเป็นสมาชิกของเซตค่าตัวแปรเชิงภาษาของตัวแปรในระบบ ในขั้นตอนแรกของการอนุมานฟัซซีจะต้องทำการหาค่าระดับความเป็นสมาชิกของเซตดังกล่าวของตัวแปรอินพุต ที่ซึ่งค่าของตัวแปรอินพุตที่เข้ามาสู่ในระบบ จะอยู่ในรูปของค่าเชิงตัวเลข หลังจากนั้นแล้วค่าระดับความเป็นสมาชิกของอินพุตค่านั้นๆ จะสามารถหาได้จากฟังก์ชันสมาชิกการทำฟัซซี



รูปที่ 2.7 (ก) ระดับความเป็นสมาชิกของค่าความผิดพลาดที่ -0.67 องศาเซลเซียส ให้ค่าระดับความเป็นสมาชิกของ 'ลบ' เท่ากับ 0.36 และระดับความเป็นสมาชิกของ 'ศูนย์' เท่ากับ 0.62 (ข) อัตราการเปลี่ยนแปลงของ ค่าความผิดพลาดที่ +1.67 องศาเซลเซียส/นาที ให้ค่าระดับความเป็นสมาชิกของ 'ศูนย์' เท่ากับ 0.35 และระดับ ความเป็นสมาชิกของ 'บวก' เท่ากับ 0.64

ของตัวแปรอินพุตจะขึ้นอยู่กับกฎของฟัซซีด้วยเช่นกัน เนื่องจากสำหรับอินพุตค่าหนึ่งๆ จะมีผลต่อกฎของฟัซซีบางข้อเท่านั้น พิจารณาการคำนวณหาค่าระดับความเป็นสมาชิกของอินพุต จากรูปที่ 2.7 จากค่าความผิดพลาดที่ -0.67 องศาเซลเซียส ซึ่งเป็นค่าที่อยู่ทั้งเซต N (เซต 'ลบ') และเซต Z (เซต 'ศูนย์') นั่นคือสำหรับ ฟังก์ชันสมาชิกของค่าความผิดพลาดที่กำหนด ค่า -0.67 องศาเซลเซียส มีทั้งความเป็นลบและความเป็นศูนย์ โดยที่มีระดับความเป็นสมาชิกของเซต N (หรือความเป็นลบ) เท่ากับ 0.36 และมีระดับความเป็นสมาชิกของเซต Z (หรือความเป็นศูนย์) เท่ากับ 0.62 ตัวอย่างนี้แสดงความเป็นฟัซซีของค่าความผิดพลาด -0.67 องศาเซลเซียส นี้อย่างชัดเจน นั่นคือค่าความผิดพลาดเป็นสมาชิกของทั้งความเป็นลบและความเป็นศูนย์ แต่มีระดับความเป็นศูนย์ มากกว่าความเป็นลบ (จากค่าระดับความเป็นสมาชิก 0.36 และ 0.62) เช่นเดียวกันกับอัตราการเปลี่ยนแปลงของค่าความผิดพลาดที่ +1.67 องศาเซลเซียส/นาที ซึ่งจากฟังก์ชันสมาชิกที่กำหนด ค่านี้มีทั้งความเป็นศูนย์และความเป็นบวก คือเป็นค่าที่อยู่ทั้งในเซต Z ('ศูนย์') และเซต P ('บวก') ด้วยค่าระดับความเป็นสมาชิกเท่ากับ 0.35 และ 0.64 ตามลำดับ (ค่าระดับความเป็นสมาชิกบ่งบอกว่าอัตราการเปลี่ยนแปลงของค่าความผิดพลาด +1.67 องศาเซลเซียส/นาที มีความเป็นบวกมากกว่าความเป็นศูนย์) สังเกตว่าตัวแปรค่าความผิดพลาดมีระดับความเป็น สมาชิกของ 'บวก' เท่ากับ 0.0 ซึ่งหมายความว่าไม่ได้มีความเป็นบวกเลยเนื่องจากมีค่าเป็นลบ ในทำนองเดียวกัน ตัวแปรอัตราการเปลี่ยนแปลงของค่าความผิดพลาดมีระดับความเป็นสมาชิกของ 'ลบ' เท่ากับ 0.0 ซึ่งหมายความว่าไม่ได้มีความเป็นลบเลยเนื่องจากมีค่าเป็นบวก สรุปค่าระดับความเป็นสมาชิกของทั้งสองอินพุตในรูปฟังก์ชันสมาชิกได้ดังนี้

$$\mu(Error = N)(-0.67) = 0.36 \quad (2.1)$$

$$\mu(Error = Z)(-0.67) = 0.62 \quad (2.2)$$

$$\mu(Error = P)(-0.67) = 0.00 \quad (2.3)$$

$$\mu(ErrorRate = N)(1.67) = 0.00 \quad (2.4)$$

$$\mu(ErrorRate = Z)(1.67) = 0.35 \quad (2.5)$$

$$\mu(ErrorRate = P)(1.67) = 0.64 \quad (2.6)$$

2. การประเมินค่ากฎของฟัซซี (fuzzy rule evaluation) หลังจากคำนวณหาค่าระดับความเป็นสมาชิกของ อินพุตทั้งหมดได้แล้ว ขั้นตอนต่อไปคือการประเมินค่าของตัวแปรที่ได้ในกฎของฟัซซี การประเมินค่ากฎดังกล่าวจะเป็นส่วน IF จุดประสงค์เพื่อทำการประเมินว่าค่าเงื่อนไขจากอินพุตนั้นจะทำให้กฎใดต้องกระทำ ในส่วน THEN ต่อไป ซึ่งอาจจะมีกฎในเงื่อนไขดังกล่าวมากกว่าหนึ่งกฎพร้อมๆกัน เนื่องมาจากระบบมีอินพุตมากกว่าหนึ่ง (นั่นคือค่าความผิดพลาดและอัตราการเปลี่ยนแปลงของค่าความผิดพลาด) เงื่อนไขของแต่ละอินพุตจะถูกประเมินค่าด้วยตัวกระทำของฟัซซีเซต เช่น AND หรือ OR เพื่อให้ได้ผลลัพธ์สุดท้ายเป็นค่าตัวเลขที่สามารถนำไปประเมินค่าส่วน THEN ที่ซึ่งภายหลังจะถูกนำไปประเมินเพื่อหาค่าระดับความเป็นสมาชิกของตัวแปรเอาท์พุตในขั้นตอนต่อไป พิจารณาตัวกระทำ OR จากทฤษฎีเซตจะได้ว่า

$$\mu A \cup B(x) = max\left[\mu A(x), \mu B(x)\right] \tag{2.7}$$

อย่างไรก็ดี ตัวกระทำ OR สามารถมีนิยามได้หลายอย่าง ยกตัวอย่างเช่น ตัวกระทำ OR ในกล่องเครื่องมือของแมทแลปฟัซซีลอจิก (MATLAB Fuzzy Logic Toolbox) จะมีทั้งการใช้ฟังก์ชัน max ข้างต้น และฟังก์ชันทางสถิติเรียกว่า probor หรือผลรวมเชิงพีชคณิต (algebraic sum) ดังนั้น

$$\mu A \cup B(x) = probor\left[\mu A(x), \mu B(x)\right] \tag{2.8}$$

$$= \mu A(x) + \mu B(x) - \mu A(x) \times \mu B(x) \tag{2.9}$$

เช่นเดียวกันกับตัวกระทำ AND ซึ่งในกล่องเครื่องมือของแมทแลปฟัซซีลอจิก (MATLAB Fuzzy Logic Toolbox) มีทั้งการใช้ฟังก์ชัน min และ ฟังก์ชันผลคูณ (prod) ดังนี้

$$\mu A \cap B(x) = min\left[\mu A(x), \mu B(x)\right] \tag{2.10}$$

หรือ

$$\mu A \cap B(x) = prod\left[\mu A(x), \mu B(x)\right] \tag{2.11}$$

$$= \mu A(x) \times \mu B(x) \tag{2.12}$$

ในบางกรณี การใช้ฟังก์ชันของตัวกระทำของเซตที่แตกต่างกัน อาจจะให้ผลเชิงตัวเลขที่แตกต่างกันได้ หลังจากประเมินค่าของแต่ละเงื่อนไขและรวมเงื่อนไข ในกรณีที่มีมากกว่า 1 เงื่อนไขในส่วนของ IF แล้ว ผลที่ได้จะถูกนำไปประเมินผลว่ากฎข้อใดที่ต้องถูกพิจารณาในส่วน THEN ต่อไป พิจารณาตัวอย่างระบบ ควบคุมอุณหภูมิซึ่งมีกฎของฟัซซีทั้งหมด 9 ข้อ เงื่อนไขของอินพุตตัวที่หนึ่งได้แก่ Error = -0.67 องศาเซลเซียส ซึ่งให้ค่าระดับความเป็นสมาชิกของ N และ Z ที่ไม่เท่ากับศูนย์ เงื่อนไขดังกล่าวมีค่ามากกว่าศูนย์และอยู่ในส่วน IF ของกฎข้อ 1, 2, 4, 5, 7 และ 8 ในขณะที่เงื่อนไขของอินพุตตัวที่สองได้แก่ ErrorRate = +1.67 องศาเซลเซียส/นาที ซึ่งให้ค่าระดับความเป็น

สมาชิกของ Z และ P ที่ไม่เท่ากับศูนย์และอยู่ในส่วน IF ของกฎข้อที่ 4, 5, 6, 7, 8 และ 9 เมื่อทำการ AND (ใช้ฟังก์ชัน min) เงื่อนไขทั้งสองแล้วจะได้ว่าเงื่อนไขทั้งสองที่มีค่าไม่เป็นศูนย์ก็คือ เงื่อนไขในข้อ 4 ,5, 7 และ 8 โดยสามารถสรุปได้ดังนี้

(1) IF $(Error = \mathrm{N})$ AND $(ErrorRate = \mathrm{N})$ THEN $Output = \mathrm{C}$

IF $(0.36$ AND $0.00) = 0.00$ THEN $Output = \mathrm{C}$

(2) IF $(Error = \mathrm{Z})$ AND $(ErrorRate = \mathrm{N})$ THEN $Output = \mathrm{H}$

IF $(0.62$ AND $0.00) = 0.00$ THEN $Output = \mathrm{H}$

(3) IF $(Error = \mathrm{P})$ AND $(ErrorRate = \mathrm{N})$ THEN $Output = \mathrm{H}$

IF $(0.00$ AND $0.00) = 0.00$ THEN $Output = \mathrm{H}$

(4) IF $(Error = \mathrm{N})$ AND $(ErrorRate = \mathrm{Z})$ THEN $Output = \mathrm{C}$

IF $(0.36$ AND $0.35) = 0.35$ THEN $Output = \mathrm{C}$

(5) IF $(Error = \mathrm{Z})$ AND $(ErrorRate = \mathrm{Z})$ THEN $Output = \mathrm{NC}$

IF $(0.62$ AND $0.35) = 0.35$ THEN $Output = \mathrm{NC}$

(6) IF $(Error = \mathrm{P})$ AND $(ErrorRate = \mathrm{Z})$ THEN $Output = \mathrm{H}$

IF $(0.00$ AND $0.35) = 0.00$ THEN $Output = \mathrm{H}$

(7) IF $(Error = \mathrm{N})$ AND $(ErrorRate = \mathrm{P})$ THEN $Output = \mathrm{C}$

IF $(0.36$ AND $0.64) = 0.36$ THEN $Output = \mathrm{C}$

(8) IF $(Error = \mathrm{Z})$ AND $(ErrorRate = \mathrm{P})$ THEN $Output = \mathrm{C}$

IF $(0.62$ AND $0.64) = 0.62$ THEN $Output = \mathrm{C}$

(9) IF $(Error = \mathrm{P})$ AND $(ErrorRate = \mathrm{P})$ THEN $Output = \mathrm{H}$

IF $(0.00$ AND $0.64) = 0.00$ THEN $Output = \mathrm{H}$

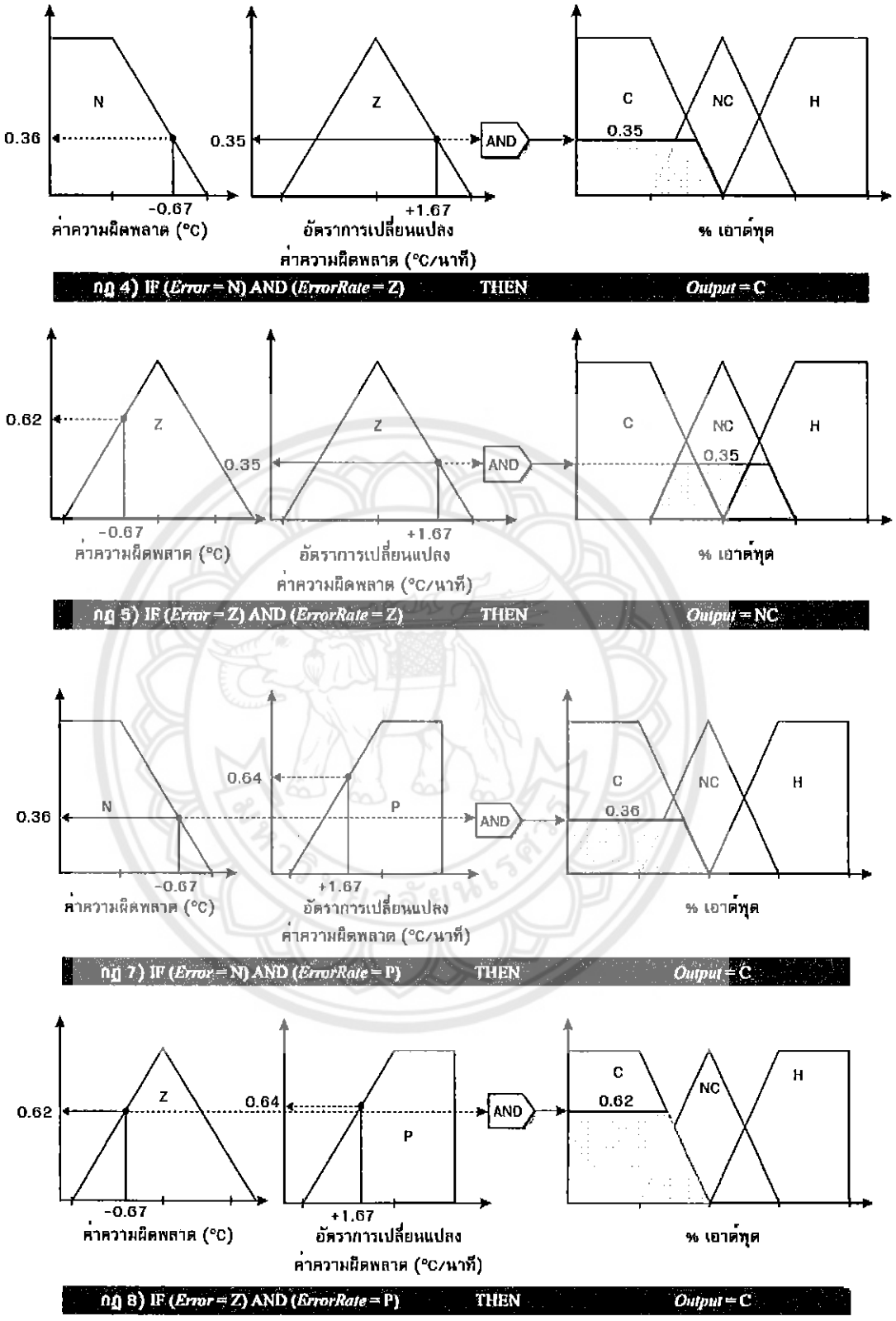
จากค่า Error = -0.67 องศาเซลเซียส และ ErrorRate = 1.67 องศาเซลเซียส/นาที มีผลให้ส่วน THEN ซึ่งก็คือตัวแปรเอาท์พุตของกฎข้อ 4 ,5, 7 และ 8 ถูกประเมินค่าในขั้นตอนต่อไป ค่าระดับความเป็นสมาชิกจากเงื่อนไขอินพุตใน ส่วน IF จะเป็นตัวบอกว่าตัวแปรเอาท์พุตจะมีรูปร่างของระดับความเป็นสมาชิกอย่างไร โดยฟังก์ชันสมาชิก ของเอาท์พุตจะถูกตัดยอด (clipped) หรือถูกปรับขนาด (scaled) ตามผลค่าระดับความเป็นสมาชิกของส่วน เงื่อนไขอินพุต IF นั่นเอง ดังแสดงในรูปที่ 2.8 ถึงแม้ว่าการตัดยอดฟังก์ชันสมาชิกของตัวแปรเอาท์พุตจะทำให้เกิดการสูญเสียข้อมูลบางส่วน แต่

วิธีการดังกล่าวเป็นวิธีที่เร็วและง่ายสำหรับการคำนวณ รวมไปถึงการนำไปใช้ประมวลผลในขั้นตอนต่อไปอีกด้วย รูปที่ 2.9 เปรียบเทียบระหว่างวิธีการตัดยอดและวิธีการปรับขนาด

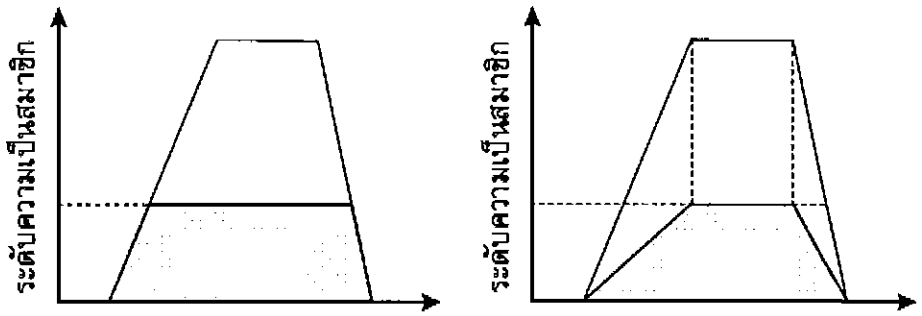
3. การรวมกฎ (aggregation) หลังจากกฎต่างๆ ถูกประเมินค่าแล้ว กฎที่มีผลไม่เท่ากับศูนย์จะถูกรวมเข้าด้วยกันโดยการรวมผลลัพธ์ของฟังก์ชันสมาชิกที่ผ่านการประเมินค่า (ถูกตัดยอดหรือปรับขนาด) ทั้งหมดเข้าด้วยกันเป็นเซตเดียวสำหรับแต่ละตัวแปรเอาท์พุต การรวมกฎจะใช้ตัวกระทำยูเนียน รูปที่ 2.10 แสดงการรวมกฎดังกล่าวจากระบบควบคุมอุณหภูมิที่ค่าความผิดพลาดเท่ากับ -0.67 องศาเซลเซียส และอัตราการเปลี่ยนแปลงค่าความผิดพลาดเท่ากับ +1.67 องศาเซลเซียส ในขั้นตอนต่อไปจะนำผลการรวมกฎนี้ไปแปลงเป็นค่าตัวเลขเดี่ยวเพื่อนำ เอาไปใช้ในการประมวลผลต่อไป

4. การทำดีฟัซซี (defuzzification) จากขั้นตอนแรกมาจนถึงขั้นตอนนี้ ค่าต่างๆในระบบเป็นค่าฟัซซี ไม่ว่าจะเป็นอินพุต กฎต่างๆ หรือเอาท์พุต แต่ว่าสำหรับทุกระบบค่าของเอาท์พุตจะต้องถูกแปลงให้อยู่ในรูปที่สามารถใช้งานได้จริงเช่นค่าสัญญาณแรงดัน ค่าสัญญาณควบคุม ฯลฯ ซึ่งค่าเหล่านี้ไม่สามารถเป็นค่าฟัซซี ได้ เพราะค่าฟัซซีจะเป็นที่เข้าใจภายระบบฟัซซีเท่านั้น ดังนั้นค่าสุดท้ายจากเอาท์พุตของระบบจะต้องเป็นค่า ชัดเจน (crisp value) การทำดีฟัซซีคือขั้นตอนในการแปลงค่าจากผลการรวมกฎให้อยู่ในรูปของค่าชัดเจน วิธีการทำดีฟัซซีนั้นมีหลายแบบ วิธีหนึ่งที่เป็นที่นิยมใช้งานกันอย่างแพร่หลายคือวิธีหาจุดศูนย์ถ่วง (centroid หรือ center of gravity, COG) ค่า COG ของฟัซซีเซต A ในช่วง [a, b] สามารถหาได้จากความสัมพันธ์ดังต่อไปนี้

$$COG = \frac{\int_a^b \mu_A(x)\, x\, dx}{\int_a^b \mu_A(x)\, dx} \qquad (2.13)$$

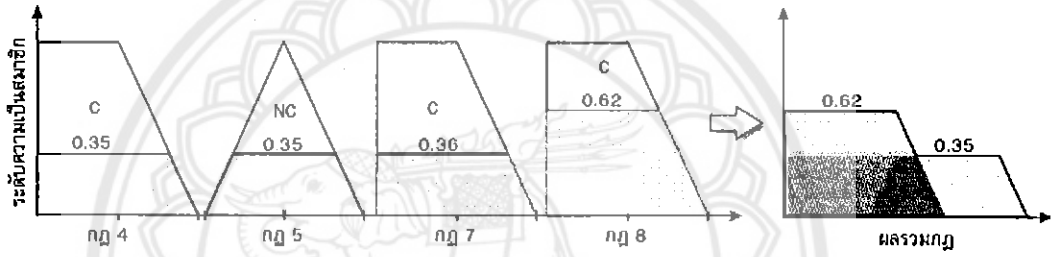


รูปที่ 2.8 การอนุมานแบบแมมดานิ



(ก) (ข)

รูปที่ 2.9 การประเมินค่าฟังก์ชันสมาชิก (ก) วิธีตัดยอด (ข) วิธีปรับขนาด



รูปที่ 2.10 ผลการรวมกฎของ Error = -0.67°C และ ErrorRate = +1.67°C

ในทางปฏิบัติ การคำนวณ COG สามารถหาได้จากข้อมูลการชักตัวอย่างดังนี้

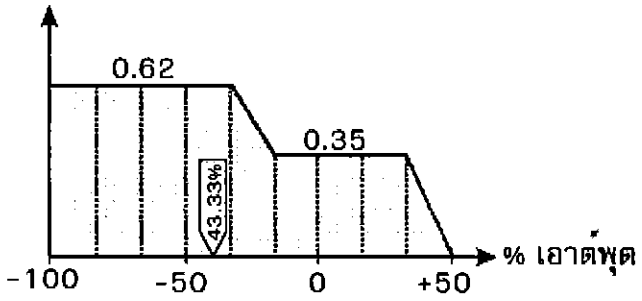
$$COG = \frac{\sum_{x=a}^{b} \mu_A(x)\, x}{\sum_{x=a}^{b} \mu_A(x)} \qquad (2.14)$$

พิจารณาเอาท์พุตของระบบควบคุมอุณหภูมิในรูปที่ 14 ค่า COG สามารถคำนวณได้ดังนี้

$$COG = \frac{(-100-83.33-66.67-50-33.33)\times 0.62 + (-16.67+0+16.67+33.33)\times 0.35}{0.62+0.62+0.62+0.62+0.62+0.35+0.35+0.35+0.35}$$

$$= -43.33$$

ค่าเอาท์พุตที่ได้จากการทำดีฟัซซีเท่ากับ -43.33 เปอร์เซ็นต์ ให้ความหมายว่าระบบต้องเปิดเครื่องทำความเย็นที่ระดับ 43.33 เปอร์เซ็นต์



รูปที่ 2.11 การทำดีฟัซซีควบคุมอุณหภูมิ

2.2 ไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์(Microcontroller) คือชิปประมวลผลอย่างหนึ่งซึ่งจะทำหน้าที่ประมวลผลตามโปรแกรมหรือชุดคำสั่งเหมือนกับไมโครโพรเซสเซอร์ โครงสร้างภายในจะเป็นวงจรรวมขนาดใหญ่ประกอบไปด้วย หน่วยคำนวณทางคณิตศาสตร์และลอจิก บัสข้อมูล บัสควบคุม บัสที่อยู่ พอร์ตขนาน พอร์ตอนุกรม รีจิสเตอร์ หน่วยความจำ วงจรนับ วงจรจับเวลาและวงจรอื่นๆ รวมกันอยู่ภายใจชิปหรือไอซี ไมโครคอนโทรลเลอร์ถูกออกแบบมาเพื่อใช้ในงานควบคุมสามารถติดต่อกับอุปกรณ์อินพุตและเอาท์พุตได้สะดวกใช้งานง่าย สามารถทำงานได้โดยใช้ชิปเดียว มีคำสั่งที่สนับสนุนในการเขียนโปรแกรมควบคุมและสามารถเข้าถึงข้อมูลระดับบิตได้

2.2.1 โครงสร้างโดยทั่วไปของไมโครคอนโทรลเลอร์ สามารถแบ่งออกได้เป็น 5 ส่วนใหญ่ๆ ดังต่อไปนี้

1. หน่วยประมวลผลกลางหรือซีพียู (CPU: Central Processing Unit)

2. หน่วยความจำ (Memory) สามารถแบ่งออกเป็น 2 ส่วน คือ หน่วยความจำที่มีไว้สำหรับเก็บโปรแกรมหลัก (Program Memory) เปรียบเสมือนฮาร์ดดิสก์ของเครื่องคอมพิวเตอร์ตั้งโต๊ะ คือข้อมูลใดๆที่ถูกเก็บไว้ในนี้จะไม่สูญหายไปแม้ไม่มีไฟเลี้ยง อีกส่วนหนึ่งคือหน่วยความจำข้อมูล (Data Memory) ใช้ในการคำนวณของซีพียูและเป็นที่พักข้อมูลชั่วคราวขณะทำงาน แต่หากไม่มีไฟเลี้ยง ข้อมูลก็จะหายไปคล้ายกับหน่วยความจำแรม (RAM) ในเครื่องคอมพิวเตอร์ทั่วๆไป แต่สำหรับไมโครคอนโทรลเลอร์สมัยใหม่หน่วยความจำข้อมูลจะมีทั้งที่เป็นหน่วยความจำแรม ซึ่งข้อมูลจะหายไปเมื่อไม่มีไฟเลี้ยง และเป็นอีอีพรอม (EEPROM: Electrically Erasable PROMs) ซึ่งสามารถเก็บข้อมูลได้แม้ไม่มีไฟเลี้ยง

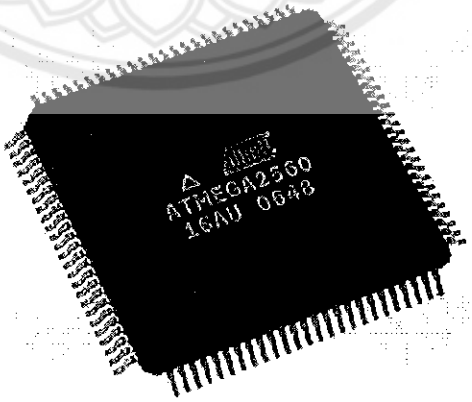
3. ส่วนติดต่อกับอุปกรณ์ภายนอกหรือพอร์ต(Port) มี 2 ลักษณะ คือ พอร์ตอินพุต(Input Port)มีหน้าที่รับสัญญาณเข้าและพอร์ตเอาท์พุต(Output Port)มีหน้าที่ส่งสัญญาณออกไปยังอุปกรณ์ภายนอก

4. ช่องทางเดินของสัญญาณหรือบัส(BUS) คือ เส้นทางการแลกเปลี่ยนสัญญาณข้อมูลระหว่างซีพียูหน่วยความจำและพอร์ตเป็นลักษณะของสายสัญญาณจำนวนมากอยู่ภายในตัวไมโครคอนโทรลเลอร์โดยแบ่งเป็นบัสข้อมูล(Data Bus) บัสแอดเดรส(Address Bus) และบัสควบคุม (Control Bus)

5. วงจรกำเนิดสัญญาณนาฬิกา นับเป็นส่วนประกอบที่สำคัญมากอีกส่วนหนึ่ง เนื่องจากการทำงานที่เกิดขึ้นในตัวไมโครคอนโทรลเลอร์จะขึ้นอยู่กับการกำหนดจังหวะ หากสัญญาณนาฬิกามีความถี่สูงจังหวะการทำงานก็จะสามารถทำได้ถี่ขึ้น ส่งผลให้ไมโครคอนโทรลเลอร์ตัวนั้นมีความเร็วในการประมวลผลสูงตามไปด้วย

2.2.2 ไมโครคอนโทรเลอร์ตระกูล AVR เบอร์ ATmega2560

ไมโครคอนโทรลเลอร์ AVR เป็นไอซีไมโครคอนโทรลเลอร์ของบริษัท Atmel มีสถาปัตยกรรมภายในเป็นแบบ RISC (reduced instruction set computer) โดยใช้สัญญาณนาฬิกาเพียง 1 ลูกในการปฏิบัติงานใน 1 คำสั่งโดยจะประกอบด้วยหน่วยความจำโปรแกรมภายในที่เป็นแบบแฟลชโปรแกรมข้อมูลได้แบบ In-System programmable



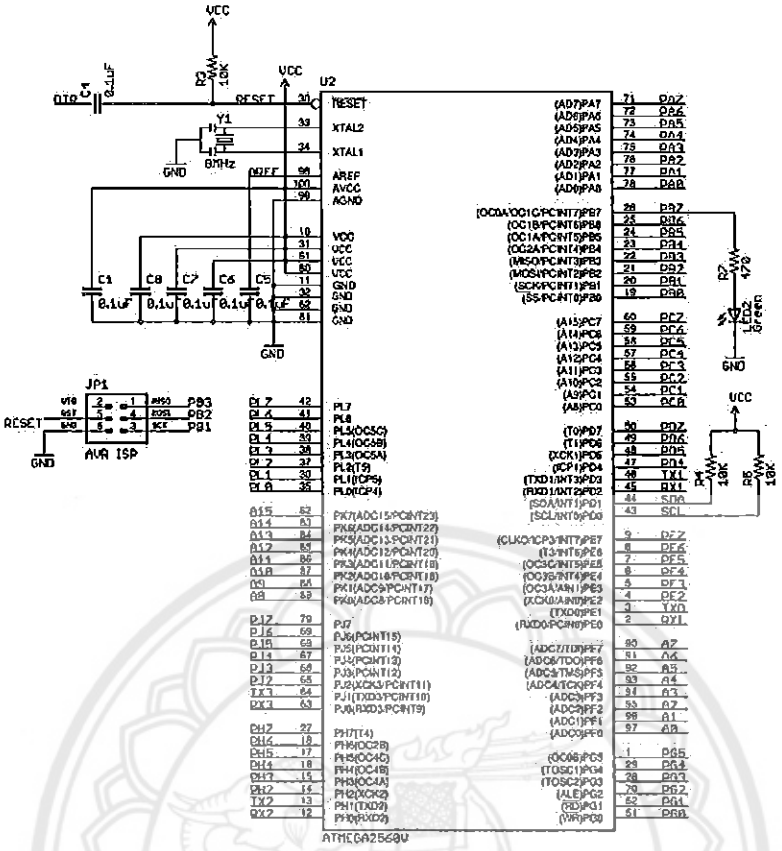
รูปที่ 2.12 ไมโครคอนโทรลเลอร์ตระกูล AVR เบอร์ ATmega2560

คุณสมบัติของ ATmega2560

1. หน่วยความจำแบบแฟลช(Flash) สำหรับบันทึกหน่วยความจำโปรแกรมหลัก (Program Memory) ขนาด 256 กิโลไบท์
2. หน่วยความจำแบบอีอีพรอม (EEPROM) สำหรับบันทึกหน่วยความจำข้อมูล (Data Memory) ขนาด 4 กิโลไบท์
3. หน่วยความจำแบบเอสแรม(SRAM) ขนาด 8 กิโลไบท์
4. ทำงานที่แรงดัน 2.7 - 5.5 โวลต์
5. ระบบกำเนิดความถี่สัญญาณแบบ PWM จำนวน 12 ช่อง ขนาด 16 บิต
6. มีวงจร Internal RC Clock 8 เมกะเฮิรตซ์
7. ทำงานที่ความถี่ 16 เมกะเฮิรตซ์
8. ระบบการสื่อสารข้อมูลดิจิตอลแบบอะซิงโครนัส (UART) จำนวน 4 ช่อง
9. ระบบการสื่อสารข้อมูลดิจิตอลแบบซิงโครนัส (SPI) จำนวน 1 ช่อง
10. มี 4 ช่อง USART
11. ไทม์เมอร์/เคาน์เตอร์ ขนาด 8 บิต จำนวน 2 ช่อง
12. ไทม์เมอร์/เคาน์เตอร์ ขนาด 16 บิต จำนวน 4ช่อง
13. มี ISP(In System Programming) สำหรับโปรแกรม

โครงสร้างภายนอกและตำแหน่งขา

มีพอร์ต อินพุท/เอาท์พุต 100 PIN TQFP ใช้งานทั่วไปจำนวน 86 ขา เป็นดิจิตอล อินพุท/เอาท์พุต จำนวน 54 ช่อง (5 โวลท์TTL LOGIC) และ อนาล็อก จำนวน 16 ช่อง (เป็นอนาล็อกเป็นดิจิตอล ขนาด 10 บิต 16 ช่อง) 4 ช่องเป็น UART (เป็น HARDWARE SERIAL PORT) แบบ 5 โวลท์TTL LOGIC



รูปที่ 2.13 ตำแหน่งขาของชิปATmega2560

2.2.3 ภาษาที่ใช้ในการเขียนคำสั่งควบคุมไมโครคอนโทรลเลอร์

ภาษาที่ใช้สำหรับการเขียนโปรแกรมบนไมโครคอนโทรลเลอร์แบ่งได้เช่นเดียวกับการเขียนโปรแกรมบนคอมพิวเตอร์คือภาษาระดับสูงและภาษาระดับต่ำ ภาษาระดับสูงเช่น ภาษาซี, เบสิค ข้อดีคือเขียนง่าย แก้ไขเปลี่ยนแปลง หรือเพิ่มเติมได้ง่าย ส่วนข้อเสียก็คือการทำงานจะช้า ขนาดโปรแกรมที่เขียนมีขนาดใหญ่ ภาษาระดับต่ำ ซึ่งก็คือ ภาษาแอสแซมบลี ข้อดีคือ ตัวคอมไพล์แจกฟรี ขนาดโปรแกรมหลังจากคอมไพล์แล้วมีขนาดเล็ก โปรแกรมมีความเร็ว แต่ข้อเสียก็คือเขียนยากเพราะลักษณะภาษาไม่ค่อยสื่อความหมายแก้ไขเปลี่ยนแปลงยาก ภาษาแต่ละภาษาก็มีข้อดีข้อเสียแตกต่างกันไป ซึ่งในโครงงานนี้ได้เลือกใช้ภาษาซีในการเขียนโปรแกรมไมโครคอนโทรลเลอร์

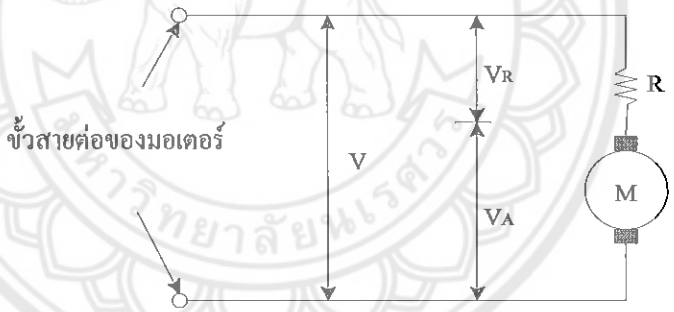
2.3 มอเตอร์ไฟฟ้ากระแสตรง

มอเตอร์ไฟฟ้าคือเครื่องกลไฟฟ้า (Electromechanical Energy) ที่ทำหน้าที่เปลี่ยนพลังงานไฟฟ้า (Electric Energy) เป็นพลังงานกล (Mechanical Energy) ในรูปแบบของการเคลื่อนที่แบบหมุน มอเตอร์ไฟฟ้ามีโครงสร้างเบื้องต้นที่สำคัญ 2 ส่วนคือ ส่วนแม่เหล็กถาวร และส่วนของขดลวดตัวนำ ซึ่งโครงสร้างคล้ายกับเครื่องกำเนิดไฟฟ้าโดยหลักการทำงานของมอเตอร์ไฟฟ้าอาศัยสนามแม่เหล็ก 2 ชุดที่เกิดขึ้นได้แก่ สนามแม่เหล็กถาวร สนามแม่เหล็กไฟฟ้าของขดลวดตัวนำ ส่งผลให้เกิดการผลักดันกันขึ้นของสนามแม่เหล็กทั้งสองทำให้ขดลวดตัวนำที่วางอยู่กลางแม่เหล็กถาวรเกิดการหมุนเครื่องไปได้

2.3.1 หลักการเบื้องต้นของมอเตอร์ไฟฟ้ากระแสตรง

1. แรงดันมอเตอร์

พิจารณาแรงดันที่ป้อนและความต้านทานของโรเตอร์ด้วย วงจรภายในของมอเตอร์เขียนได้ดังรูปที่ 2.14



รูปที่ 2.14 วงจรภายในของมอเตอร์ไฟฟ้ากระแสตรง

โดยสมมติให้ทุ่นโรเตอร์ไม่มีความต้านทานอยู่เลย อนุกรมกับความต้านทานซึ่งในที่นี้ก็คือความต้านทานของขดลวดนั่นเอง แรงดันที่ขั้วต่อสายของมอเตอร์ก็คือผลบวกระหว่างแรงดันที่ทุ่นโรเตอร์ (V_A) และ แรงดันตกคร่อมความต้านทานขดลวด (V_R)

แรงดัน V_A ถูกเรียกว่า แรงเคลื่อนเหนี่ยวนำป้อนกลับ (BACK EMF) ซึ่งเกิดขึ้นในโรเตอร์ขณะที่หมุนแรงดันที่เกิดขึ้นนี้เป็นไปตากกฎของการเหนี่ยวนำแม่เหล็กไฟฟ้าจากการเคลื่อนที่ของตัวนำในสนามแม่เหล็ก สัมพันธ์กับแรงเคลื่อนเหนี่ยวนำแม่เหล็ก และ ความเร็วในการเคลื่อนที่ของตัวนำ แรงดันที่เกิดขึ้นจะมีขั้วตรงกันข้ามกับแรงดันที่ป้อนให้กับมอเตอร์ และแปรผันตรงกับ

ความเร็วในการหมุน ผลบวกของแรงดันที่ทุ่นโรเตอร์ (V_A) และแรงดันตกคร่อมขดลวด (V_R) ต้องเท่ากับแรงดันที่ป้อนให้กับมอเตอร์ (V)

$$V = V_A + V_R \quad \text{[โวลต์]} \tag{2.15}$$

2. กระแสมอเตอร์

เมื่อพิจารณาตั้งแต่มอเตอร์หยุดนิ่ง ความเร็วมีค่าเป็นศูนย์ ดังนั้น $V_A = 0$, $V_R = V$ กระแสที่ไหลในมอเตอร์หาได้จาก

$$I = \frac{V_R}{R} \quad \text{[แอมแปร์]} \tag{2.16}$$

เมื่อมอเตอร์เริ่มหมุนจะมีความเร็ว และ V_A เพิ่มขึ้นเป็นเส้นตรงตามความเร็ว V_R ซึ่งมีค่าเท่ากับความแตกต่างระหว่าง V_A และ V จะเริ่มลดลงกระแส I ก็จะเริ่มลดลงเช่นกันขณะที่มอเตอร์ยังมีความเร่งอยู่ ความเร็วจะเพิ่มขึ้น แรงบิดจะลดลงจนกว่าจะถึงจุดซึ่งแรงบิดของมอเตอร์รับภาระโหลดได้สมดุลพอดี ขณะที่มอเตอร์ไม่มีโหลด และ หมุนอย่างอิสระจะมีเพียงค่าความฝืดของแบริ่ง และ แรงต้านอากาศทำให้ V_A เกือบเท่ากับค่า V

3. แรงผลักดันในกระแสแม่เหล็ก

แรงผลักตัวนำให้เคลื่อนที่ในสนามแม่เหล็ก ซึ่งจะมากหรือน้อยเพียงใดนั้นย่อมขึ้นอยู่กับความสัมพันธ์ระหว่างกระแสไฟฟ้าที่ให้ไหลผ่านตัวนำในสนามแม่เหล็ก กับเส้นแรงแม่เหล็กในช่องว่างอากาศระหว่างขั้วเหนือ ขั้วใต้ ดังนี้

$$F = B \cdot \ell \cdot I \cdot Z \quad \text{[นิวตัน]} \tag{2.17}$$

เมื่อ F คือ แรงผลักตัวนำ [นิวตัน/ตัวนำ]

B คือ ความหนาแน่นของเส้นแรงเหล็กในสนามแม่เหล็ก [เทสลา]

ℓ คือ ความยาวของตัวนำในสนามแม่เหล็ก [เมตร]

I คือ กระแสไฟฟ้าที่ให้ไหลผ่านตัวนำในสนามแม่เหล็ก [แอมแปร์]

Z คือ จำนวนตัวนำทั้งหมดในสนามแม่เหล็ก [ตัวนำ]

4. ทอร์คของตัวนำ

ทอร์ค (Torqe) คือ โมเมนต์หมุนหรือแรงบิด เป็นผลคูณระหว่างแรงกับแขนแรง ดังนี้

$$T = F \cdot R = F_A \cdot r = B \cdot \ell \cdot I \cdot Z \cdot R \quad \text{[นิวตันเมตร]} \tag{2.18}$$

เมื่อ T คือ ทอร์ค : แรงบิด : โมเมนต์หมุน [นิวตันเมตร]

F คือ แรงฉุดหรือแรงผลักตัวนำ [นิวตัน]

R คือ รัศมีของอาร์เมเจอร์ : แขนแรง [เมตร]

F_A คือ แรงดึงสายพานสำหรับหมุนขับงาน [นิวตัน]

r คือ รัศมีของล้อขับสายพาน : แขนแรง [เมตร]

5. ทอร์คของมอเตอร์

จากสมการที่ (2.18)

$$T = F \cdot R = B \cdot \ell \cdot I \cdot Z \cdot R$$

$$B = \frac{\phi}{A} \qquad (2.19)$$

เมื่อ B คือ ความหนาแน่นของเส้นแรงแม่เหล็ก [เทสลา]

ϕ คือ เส้นแรงแม่เหล็ก/ขั้ว [เวเบอร์]

A คือ พื้นที่หน้าตัดของแกนขั้วแม่เหล็ก [ตารางเมตร]

I คือ กระแสไฟฟ้าที่ให้ไหลผ่านตัวนำ [แอมแปร์/ตัวนำ]

จะเห็นได้อย่างชัดเจนว่า ทอร์คของมอเตอร์เป็นปฏิภาคโดยตรงกับเส้นแรงแม่เหล็ก และกระแสของมอเตอร์

6. สมการไฟฟ้าของมอเตอร์

$$E_C = \frac{\phi \cdot P \cdot n}{60} \cdot \frac{z}{a}$$

เมื่อ $K' = P \cdot \frac{z}{60 \cdot a}$ (2.20)

$$E_C = K' \cdot \phi \cdot n \qquad (2.21)$$

เมื่อ E_C คือ แรงดันไฟฟ้าเหนี่ยวนำกลับ [โวลต์]

ϕ คือ เส้นแรงแม่เหล็ก/ขั้ว [เวเบอร์]

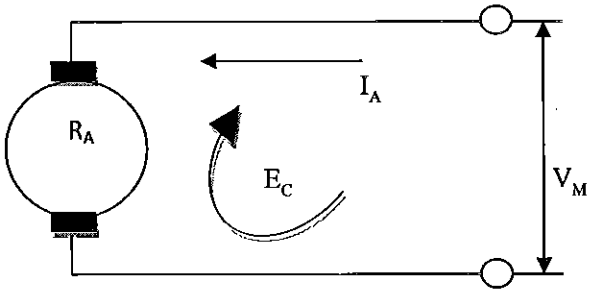
P คือ จำนวนขั้วแม่เหล็ก [ขั้ว]

n คือ จำนวนรอบที่มอเตอร์หมุน [รอบ/นาที]

z คือ จำนวนตัวนำบนอาร์เมเจอร์ [ตัวนำ]

a คือ จำนวนวงจรไฟฟ้าคู่ขนานบนอาร์เมเจอร์ [ตัวนำ]

K' คือ ค่าคงที่ของมอเตอร์ (โดยลักษณะสร้าง)



รูปที่ 2.15 วงจรไฟฟ้าของมอเตอร์

จากรูปที่ 2.15 จะได้สมการ ดังนี้

$$V_M = E_C + I_A \cdot R_A = E_C + V_A \tag{2.22}$$

$$E_C = V_M - I_A \cdot R_A = V_M + V_A \tag{2.23}$$

$$V_A = I_A \cdot R_A = V_M - E_C \tag{2.24}$$

$$I_A = \frac{V_M - E_C}{R_A} \tag{2.25}$$

จากสมการที่ (2.21)

$$E_C = K' \cdot \phi \cdot n$$

จะได้

$$n = \frac{E_C}{K' \cdot \phi} = \frac{V_M - V_A}{K' \cdot \phi} \tag{2.26}$$

2.3.2 การจำแนกประเภทมอเตอร์ไฟฟ้ากระแสตรง

มอเตอร์กระแสตรงสามารถจำแนกชนิดออกได้ ทั้งตามลักษณะของการต่อขดลวดสนามแม่เหล็ก (หรือการพันขดลวดสนามแม่เหล็ก) และการไหลของกระแสผ่านขดลวดสนามแม่เหล็ก ดังนี้

1. มอเตอร์แบบอนุกรม (Series motor)

คุณสมบัติทั่วไป : ประกอบด้วยขดลวดสนามแม่เหล็กที่มีความต้านทานต่ำ พันด้วยขดลวดทองแดงเส้นใหญ่บนแกนขั้วแม่เหล็กจำนวนน้อยรอบ เช่นเดียวกับขดลวดกระแสของแอมมิเตอร์ ต่อเป็นอนุกรมกับอาร์เมเจอร์และแรงดันเมน ลักษณะงานใช้หมุนขับงานที่ต้องการ

ทอร์คเริ่มหมุนสูง และความเร็วรอบเปลี่ยนแปลงตลอดเวลา เช่น รถไฟฟ้า รถราง เครน ลิฟท์ คอนเวเยอร์ ฯลฯ

คุณสมบัติทางเทคนิค : สมรรถนะในการหมุนขับโหลด เมื่อโหลดเปลี่ยนแปลง กระแสอาร์เมเจอร์ และเส้นแรงแม่เหล็กในสนามแม่เหล็กจะเปลี่ยนแปลงไปด้วย มีผลทำให้ความเร็วรอบของมอร์เตอร์เปลี่ยนแปลงไปในที่สุด ดังนั้นจึงกล่าวได้ว่า ความเร็วรอบของมอเตอร์อนุกรมจะเปลี่ยนแปลงไปตามการเปลี่ยนแปลงของโหลด คือ เมื่อโหลดเพิ่มขึ้น ความเร็วรอบจะลดลง และเมื่อโหลดลดลง ความเร็วรอบจะเพิ่มขึ้น

2. มอเตอร์แบบขนาน (Shunt motor)

คุณสมบัติทั่วไป : ประกอบด้วยขดลวดสนามแม่เหล็กที่มีความต้านทานค่อนข้างสูง ซึ่งใช้ลวดทองแดงเส้นเล็กๆ พันบนแกนขั้วแม่เหล็กหลายๆรอบ เช่นเดียวกับขดลวดแรงดัน (Voltage หรือ Potential Coil) ของโวลต์มิเตอร์ ต่อขนานกับอาร์เมเจอร์ และต่ออนุกรมกับตัวต้านทานที่ปรับค่าได้ (รีโอสตาท : Rheostat) แล้วต่อขนานกับสายเมน ลักษณะงานเป็นมอเตอร์ที่ใช้หมุนขับเครื่องจักรกลที่ต้องการความเร็วรอบคงที่

คุณสมบัติทางเทคนิค : สมรรถนะในการหมุนขับโหลดความเร็วรอบจะลดลงเพียงเล็กน้อยเมื่อโหลดเพิ่มขึ้น มอเตอร์จะหมุนด้วยความเร็วรอบค่อนข้างจะคงที่ตราบเท่าที่มอเตอร์ยังคงต่ออยู่กับแรงดันเมนที่คงที่ ดังนั้นจึงกล่าวได้ว่า "มอเตอร์ขนานเป็นมอเตอร์ที่หมุนด้วยความเร็วรอบคงที่ (Constant Speed Machine)"

3. มอเตอร์กระแสตรงแบบผสม (Compound motor) มอเตอร์ไฟฟ้ากระแสตรงแบบผสมนี้ จะนำคุณลักษณะที่ดีของมอเตอร์ไฟฟ้ากระแสตรงทั้งแบบขนาน และแบบอนุกรมมารวมกัน มอเตอร์ไฟฟ้ากระแสตรงแบบผสม มีคุณลักษณะพิเศษคือมีแรงบิดสูง (High staring torque) แต่ความเร็วรอบคงที่ตั้งแต่ยังไม่มีโหลดจนกระทั่งมีโหลดเต็มที่ มอเตอร์แบบผสมมีวิธีการต่อขดลวดขนานหรือขดลวดชันท์อยู่ 2 วิธี

3.1 ใช้ต่อขดลวดแบบชันท์ขนานกับอาเมเจอร์เรียกว่า ชอทชันท์ (Short Shunt Compound Motor)

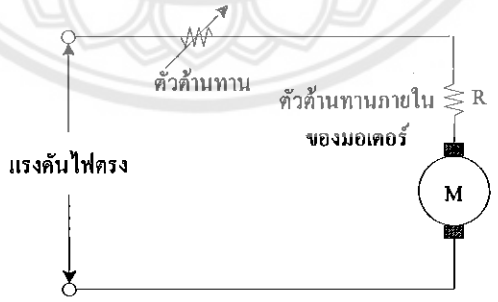
3.2 ต่อขดลวด ขนานกับขดลวดอนุกรมและขดลวดอาเมเจอร์เรียกว่าลองชั้นท์ คอมเปาวด์ มอเตอร์ (Long shunt motor)

2.3.3 การทำงานของมอเตอร์ไฟฟ้ากระแสตรง

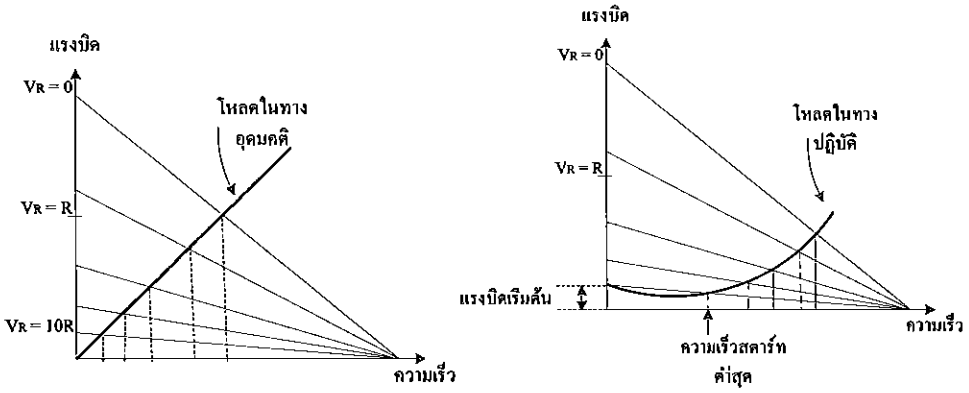
มอเตอร์ไฟฟ้ากระแสตรงประกอบด้วยแม่เหล็กขั้ว 2 ขั้ววางอยู่ระหว่างขดลวดตัวนำเมื่อมีการผ่านกระแสไฟฟ้าเข้าไปยังขดลวดในสนามแม่เหล็กจะทำให้เกิดแรงแม่เหล็กซึ่งมีสัดส่วนแรงกับกระแสแรง โดยแรงจะเกิดขึ้นเป็นมุมฉากกับกระแสและสนามแม่เหล็ก ขณะที่ทิศทางของแรงกลับตรงกันข้ามกันถ้าหากกระแสของสนามแม่เหล็กไหลย้อนกลับจะทำให้เกิดการเปลี่ยนแปลงของกระแสและสนามแม่เหล็กเป็นผลทำให้เกิดทิศทางของแรงเปลี่ยนไป ด้วยคุณสมบัตินี้ทำให้มอเตอร์กระแสตรงกลับทิศการหมุนได้ ซึ่งสนามแม่เหล็กของมอเตอร์ส่วนหนึ่งเกิดจากแม่เหล็กถาวรซึ่งจะยึดติดกับแผ่นเหล็กหรือเหล็กกล้า โดยปกติส่วนนี้จะเป็นส่วนที่ยึดอยู่กับที่และขดลวดเหนี่ยวนำจะพันอยู่กับส่วนที่เป็นแกนของมอเตอร์

2.3.4 การควบคุมความเร็วของมอเตอร์ไฟฟ้ากระแสตรง

1. การควบคุมแบบปรับค่าได้ เป็นรูปแบบพื้นฐานที่สุดของการควบคุมมอเตอร์คือใช้ตัวต้านทานปรับค่าได้อนุกรมกับมอเตอร์โดยกำหนดให้ตัวต้านทานที่ปรับค่าได้จะเป็นตัวกำหนดความเร็วในการหมุนของมอเตอร์การบังคับแบบนี้ไม่มีประสิทธิภาพเพราะกำลังไฟสูญเสียไปในตัวความต้านทานมักนิยมใช้กับมอเตอร์ขนาดตัวเล็กๆการบังคับแบบนี้ให้คุณสมบัติการสตาร์ทที่ดี(ให้แรงบิดสูงที่ความเร็วต่ำ)แต่จะให้ความเร็วที่สูงมากเมื่อมอเตอร์อยู่ในภาวะที่มีโหลดน้อยๆดังนั้นการบังคับแบบนี้มีประโยชน์เฉพาะภาวะที่แรงต้านคงที่ เช่น การบังคับความเร็วของเครื่องจักรเย็บผ้า เป็นต้น

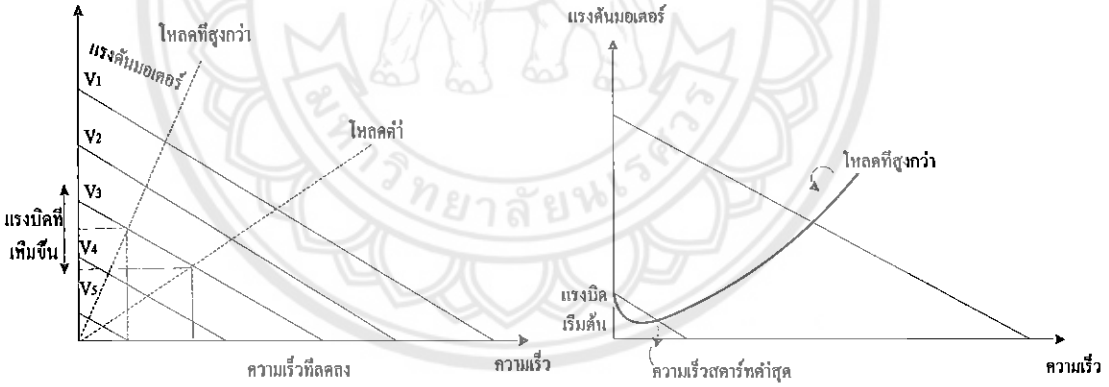


รูปที่ 2.16 วงจรควบคุมความเร็วมอเตอร์กระแสตรงแบบใช้ตัวต้านทานอนุกรม



รูปที่ 2.17 กราฟแสดงคุณสมบัติของวงจรควบคุมความเร็วของมอเตอร์กระแสตรงแบบใช้ตัวต้านทานอนุกรม

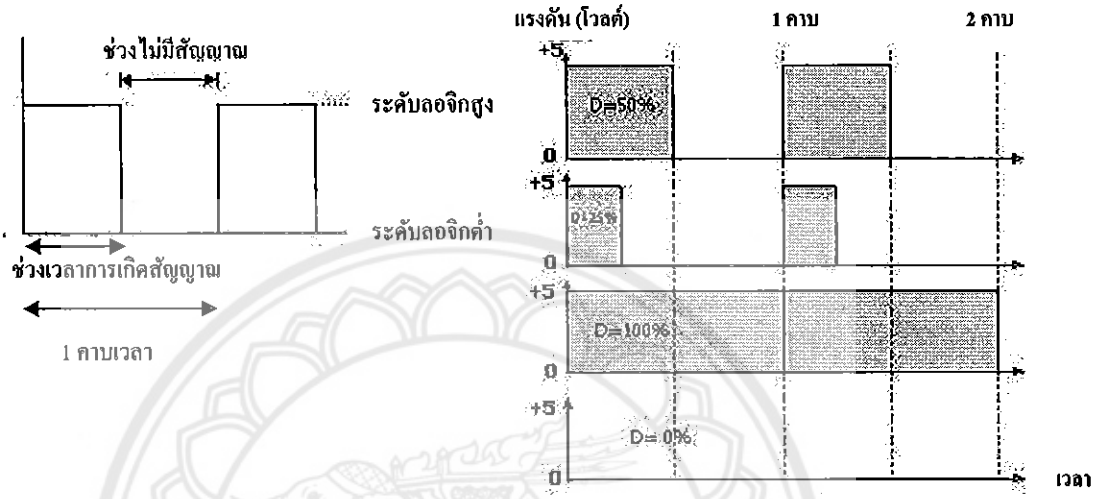
2. การควบคุมด้วยวิธีเปลี่ยนค่าแรงดัน วิธีการนี้ดีกว่าวิธีการแรกแต่จะซับซ้อนกว่าต้องที่ใช้อุปกรณ์อิเล็กทรอนิคส์ที่อัตราขยายแบบกำลังสูง และ มอเตอร์จะถูกป้อนด้วยแรงดันที่เปลี่ยนแปลงค่าได้ จากแหล่งจ่ายที่มีอิมพีแดนซ์ต่ำ ข้อดีของการควบคุมวิธีนี้คือถ้าความเร็วลดลงจากผลของแรงบิดแรงดันที่ป้อนให้กับมอเตอร์จะเพิ่มขึ้นเพื่อรักษาระดับความเร็ว ส่วนข้อเสียจากการควบคุมวิธีนี้คือ เมื่อมอเตอร์มีความเร็วต่ำแรงดันที่ป้อนให้กับมอเตอร์จะมีค่าต่ำเช่นกัน



รูปที่ 2.18 กราฟแสดงคุณสมบัติของการควบคุมความเร็วของมอเตอร์กระแสตรงโดยการเปลี่ยนค่าแรงดัน

3. การควบคุมแบบ PMW (Pluse Width Modulation) การมอดูเลชั่นทางความกว้างของพัลส์ PWM (Pulse Width Modulation) จะเป็นการปรับเปลี่ยนที่สัดส่วน และความกว้างของสัญญาณพัลส์ โดยความถี่ของสัญญาณพัลส์จะไม่มีการเปลี่ยนแปลง หรือเป็นการเปลี่ยนแปลงที่ค่าของดิวตี้ไซเกิล (dutycycle) นั้นเอง ซึ่งค่าของดิวตี้ไซเคิล คือช่วงความกว้างของพัลส์ที่มีสถานะลอจิกสูง โดยคิดสัดส่วนเป็นเปอร์เซ็นต์จากความกว้างของพัลส์ทั้งหมด ยกตัวอย่างเช่น ถ้าหากค่าดิวตี้ไซเคิลมีค่าเท่ากับเท่ากับ 50 เปอร์เซ็นต์ ก็หมายถึงใน 1 รูปสัญญาณพัลส์จะมีช่วงของ

สัญญาณที่เป็นสถานะลอจิกสูงอยู่ครึ่งหนึ่ง และสถานะลอจิกต่ำอยู่อีกครึ่งหนึ่ง และในทำนองเดียวกันถ้าหากค่าดิวตี้ไซเคิลมีค่ามาก หมายความว่าความกว้างของพัลส์ที่เป็นสถานะลอจิกสูงจะมีความกว้างมากขึ้น หากค่าดิวตี้ไซเคิลมีค่าเท่ากับ 100 เปอร์เซ็นต์ ก็หมายความว่าจะไม่มีสถานะลอจิกต่ำเลย



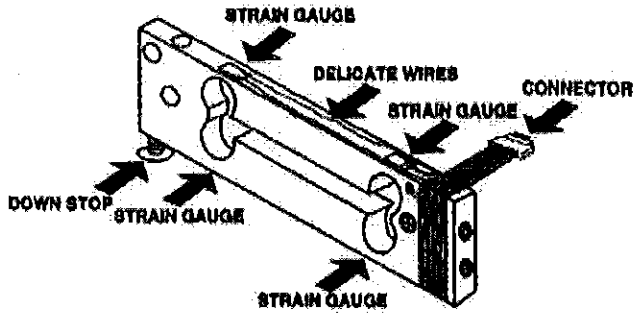
รูปที่ 2.19 ความกว้างของพัลท์ขนาดต่างๆ และค่าดิวตี้ไซเคิล ของช่วงพัล**ท์ที่มีความถี่คงที่**

2.4 โหลดเซลล์

โหลดเซลล์ คือ เซนเซอร์ที่สามารถแปลงค่าแรงกด หรือแรงดึง เป็นสัญญาณทางไฟฟ้าได้ เหมาะสำหรับการทดสอบคุณสมบัติทางกลของชิ้นงาน (Mechanical Properties of Parts) โหลดเซลล์ถูกนำไปใช้ในอุตสาหกรรมหลากหลายประเภท ได้แก่ การชั่งน้ำหนัก การทดสอบแรงกดของชิ้นงาน การทดสอบความแข็งแรงของชิ้นงาน การทดสอบการเข้ารูปชิ้นงาน(Press fit) ใช้สำหรับงานทางด้านวัสดุ โลหะ ทดสอบโลหะ ชิ้นส่วนรถยนต์ วิศวกรรมโยธา ทดสอบคอนกรีต ทดสอบไม้ ฯลฯ

2.4.1 ประเภทของโหลดเซลล์ที่ใช้งานทั่วไป

1. โหลดเซลล์แบบสเตรนเกจ (Strain Gauge Load cell) ซึ่งใช้ในการทำ โครงการนี้ หลักการของ โหลดเซลล์ ประเภทนี้ก็คือ เมื่อมีน้ำหนักมากระทำ ความเครียด (Strain) จะเปลี่ยนเป็นความต้านทานทางไฟฟ้าใน สัดส่วนโดยตรงกับแรงที่มากระทำ ปกติแล้วมักจะใช้เกจวัดความเครียด 4 ตัว



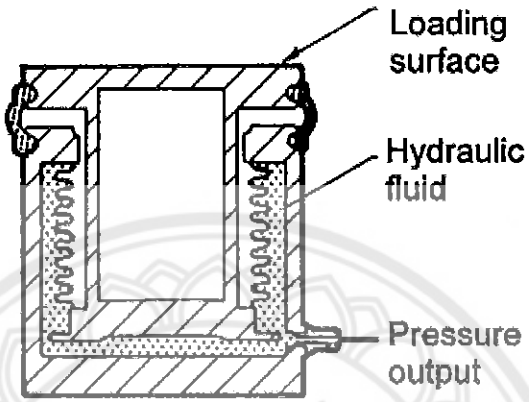
รูปที่ 2.20 การติดตั้งสเตรนเกจภายในโหลดเซลล์

การวัดโดยเกจตัวต้านทานทั้งสี่จะเชื่อมต่อเข้าด้วยกันเพื่อใช้แปลงแรงที่กระทำ กับตัวของมันไม่ว่าจะเป็นแรงกด หรือแรงดึงส่ง สัญญาณออกมาเป็นแรงดันไฟฟ้า โดยที่แรงดันไฟฟ้าที่ได้จะมีหน่วยเป็น มิลลิโวลต์/โวลต์ หมายความว่า ถ้าจ่าย แรงดัน 10 โวลต์ ให้กับ โหลดเซลล์ ที่มีสเปค 2 มิลลิโวลต์/โวลต์ ที่ฟูลโหลด (Full load) สมมุติว่าน้ำหนักเป็น 2,000 กิโลกรัม ดังนั้นเมื่อมีแรงกระทำ ต่อ โหลดเซลล์ ที่น้ำหนัก ฟูลโหลด(Full load) สัญญาณที่จะได้ก็จะได้เท่ากับ 20 มิลลิโวลต์

ในทางปฏิบัติการใช้งานโหลดเซลล์ เพียงแค่จ่ายแรงดันไฟฟ้ากระแสตรงให้กับโหลดเซลล์ตามขั้วอินพุตที่กำหนดของโหลดเซลล์ แรงดันเอาท์พุตที่ได้ก็จะแปรผันตรงกับแรงที่มากระทำต่อโหลดเซลล์ ซึ่งแรงดันไฟฟ้าที่ได้จะยังคงมีค่าน้อยอยู่ จึงต้องนำไปผ่านวงจรขยายสัญญาณก่อนเพื่อนำไปใช้ในกระบวนการต่อไป โดยส่งสัญญาณเข้าไปยังกระบวนการประมวลผลแรงดันที่ได้จากการขยายสัญญาณเพื่อทาการวิเคราะห์ข้อมูลในไมโครคอนโทรลเลอร์จากนั้นจึงส่งไปยังกระบวนการแสดงผลทางจอ LCD โดยโหลดเซลล์ที่จำหน่ายในปัจจุบันมีหลายชนิดหลายแบบ ขึ้นอยู่กับว่าจะนำไปใช้งานแบบไหน เช่น โหลดเซลล์แบบลิงค์ (Link-Type Load Cell) ใช้ในงานรับแรงดึง,โหลดเซลล์แบบวงแหวน (Ring-Type Load Cell) ใช้ในงานรับแรงกด,โหลดเซลล์แบบเชียร์-เวบ (Shear-Web-Type Load Cell) ใช้ในงานที่เป็นลักษณะของแรงเฉือน,โหลดเซลล์แบบคาน (Beam-Type Load Cell) ใช้งานในลักษณะเป็นคานรับแรง

2. โหลดเซลล์แบบไฮดรอลิก (Hydraulic Load Cell) ลักษณะของการทำงานก็คือจะวัดน้ำหนักจากการเปลี่ยนแปลงความดันของของเหลวภายในระบบเมื่อมีแรงมากระทำที่แท่นรับน้ำหนักในโหลดเซลล์แบบไฮดรอลิก ที่มีแผ่นไดอะแฟรม โดยแรงจะถูกส่งผ่านลูกสูบเป็นผลให้ของเหลวภายในช่องแผ่นไดอะแฟรมถูกกดอัด ซึ่งการวัดแรงที่เกิดขึ้นสามารถวัดได้จากความดัน

ของ ของเหลวความสัมพันธ์ ระหว่างแรงกระทำกับแรงดันของของเหลวนี้ มีลักษณะเป็นแบบเชิงเส้นและไม่ขึ้นกับอุณหภูมิและปริมาณของของเหลวในกระบอกสูบ โดยปกติโหลดเซลล์แบบนี้จะความแม่นยำ (Accuracy) ในการวัดอยู่ที่ประมาณ 0.3 เปอร์เซ็นต์ ที่ ขนาดเต็ม (Full scale) ซึ่งระดับความแม่นยำนี้ก็เป็นที่ยอมรับได้ในงานอุตสาหกรรมทั่วไป



รูปที่ 2.21 การทำงานของโหลดเซลล์แบบไฮดรอลิก (Hydraulic Load Cell)

3. โหลดเซลล์แบบนิวแมติก (Pneumatic Load cell) ซึ่งจะทำงานโดย ใช้หลักการสมดุลแรงเช่นเดียวกับแบบไฮดรอลิก แต่ต่างกันที่ โหลดเซลล์แบบนี้จะมีความแม่นยำกว่าแบบไฮดรอลิกมากเพราะว่า มีการใช้ช่องว่างหลายช่อง ในการหน่วงความดันของของเหลวเพื่อลดแรงสั่นสะเทือน โหลดเซลล์แบบนี้ มักจะใช้วัดสิ่งของที่มีน้ำหนักไม่มากนักในงาน อุตสาหกรรมที่ต้องการความสะอาดและความปลอดภัยสูงสำหรับ จุดเด่นของโหลดเซลล์แบบนี้ คือ สามารถทนแรงกระแทกได้สูงและไม่ไวต่อการเปลี่ยนแปลงของอุณหภูมินอกจากนี้ ในระบบนิวเมติก จะไม่ใช้ของเหลวในเครื่องมือวัด เหมือนกับระบบไฮดรอลิก ทำให้ไม่มีของเหลวมาปนเปื้อนโดนสิ่งที่ต้องการจะวัดในกรณีที่ไดอะแฟรมมีการแตกร้าว

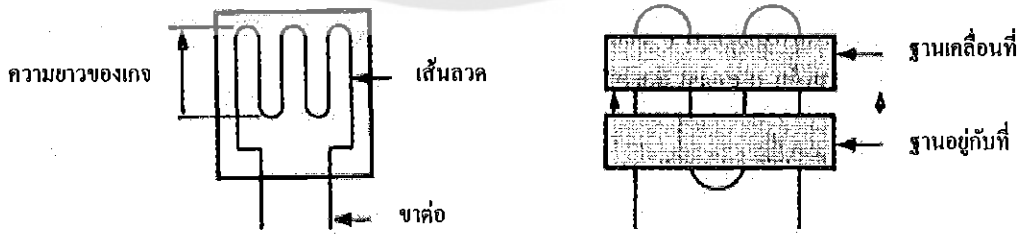
4. โหลดเซลล์แบบไพโซรีซิสทีฟ (PiezoreSistive) ซึ่งมีการทำงานเหมือนกับเกจ วัดความเครียด แต่ไพโซรซิสทีฟ สามารถผลิตสัญญาณออกมาได้ในระดับสูงจึงเหมาะสำหรับ เครื่องชั่งน้ำหนักที่ไม่ซับซ้อนในการวัดเนื่องจากสามารถต่อเข้าโดยตรงกับ ส่วนแสดงผล อย่างไรก็ตามเครื่องมือวัดลักษณะนี้ได้รับความนิยมลดลงเรื่อย ๆ เพราะตัวขยายสัญญาณที่มีคุณภาพดีนั้นมีราคาถูกลง นอกจากนี้ โหลดเซลล์แบบไพโซรีซิสทีฟ (PiezoreSistive) ยังมีข้อเสียคือความสัมพันธ์ระหว่างสัญญาณที่ออกกับน้ำหนักที่ วัดมีลักษณะไม่เป็นเชิงเส้นสำหรับข้อเสีย ขอโหลดเซลล์ แบบ

นี้มีคือความเร็วในการตอบสนองต่ำและต้องใช้งานในสภาวะแวดล้อมที่สะอาด ปลอดความชื้น อีกทั้งยังจะต้องมีการควบคุมอากาศหรือไนโตรเจนภายในเครื่องให้เหมาะสม

5.โหลดเซลล์แบบแมกเนโตสเตร็กทีฟ (Magnetostrictive) การทำงานของเซนเซอร์แบบนี้ขึ้นอยู่กับ การเปลี่ยนแปลงในการแผ่สัญญาณ แม่เหล็กของแม่เหล็กถาวรที่อยู่ภายใต้แรงที่มากระทำแรงทำให้เกิดการผิดรูป ของสนามแม่เหล็กและจะให้เกิดสัญญาณที่เป็นสัดส่วนโดยตรงต่อแรงที่มากระทำ ซึ่งจะใช้ หลักการการเหนี่ยวนำสนามแม่เหล็กนั่นเองครับโดยอุปกรณ์ลักษณะนี้จะตรวจวัด การเคลื่อนที่ของแกนแม่เหล็ก และวัดการเหนี่ยวนำของขดลวดแม่เหล็กไฟฟ้าที่เปลี่ยนไป ในที่นี้การเคลื่อนที่ของแกนเหล็กจะแปรผันโดยตรงกับน้ำหนักที่วัดนั่นเอง สำหรับโหลดเซลล์รูปแบบนี้มีความทนทานมากและยังคงมีใช้อยู่มากโดยเฉพาะอย่าง ยิ่งในอุตสาหกรรมรีดโลหะ ข้อดีของโหลดเซลล์แบบนี้คือ สามารถที่จะใช้ในพื้นที่ที่อันตราย (Hazardous Area) เช่น พวกโรงงานที่มีวัตถุไวไฟต่างๆ เนื่องจาก โหลดเซลล์ แบบนี้ไม่ต้องใช้ไฟฟ้าในการวัด

ซึ่งในโครงงานนี้เลือกใช้โหลดเซลล์ประเภทสเตรนเกจแบบคาน (Beam-Type Load Cell) เป็นตัวตรวจรู้น้ำหนัก ประกอบด้วยสเตรนเกจ 2 ตัวติดอยู่ที่คานด้านบน และอีก 2 ตัวติดอยู่ที่คานด้านล่าง โดยสเตรนเกจทั้ง 4 ตัวต่อกันเป็นวงจรวีทสโตนบริดจ์ โดยมีรายละเอียดดังนี้

สเตรนเกจสามารถแบ่งออกเป็น 2 แบบ คือ แบบยึดติด (Bonded Strain Gage) และแบบไม่ยึดติด (Unbounded Strain Gage) โดยสเตรนเกจทั้งสองชนิดจะมีลักษณะโครงสร้างและการทางานที่คล้ายกันคือทาด้วยเส้นลวดเล็กๆขดไปขดมาและนำไปติดกับวัตถุที่ต้องการตรวจวัดความเครียด



(ก) (ข)

รูปที่ 2.22 สเตรนเกจ (ก) แบบยึดติด (ข) แบบไม่ยึดติด

เมื่อสเตรนเกจถูกยืดออก ความยาวของเส้นลวดจะเพิ่มขึ้นในขณะที่พื้นที่หน้าตัดจะลดลง ทำให้ความต้านทานของเส้นลวดเพิ่มขึ้น เนื่องจากความต้านทานโลหะตัวนำจะแปรค่า

โดยตรงตามความยาวและแปรผกผันกับพื้นที่หน้าตัด โดยเขียนความสัมพันธ์ทางคณิตศาสตร์ได้ดังนี้

$$R = \frac{\rho L}{A} \tag{2.27}$$

R คือ ค่าความต้านทานของขดลวดตัวนำ [โอห์ม]

ρ คือ ค่าสัมประสิทธิ์ความต้านทานของลวดตัวนำที่ใช้ทาสเตรนเกจ [โอห์ม-เมตร]

L คือ ความยาวของขดลวดตัวนำ [เมตร]

A คือ พื้นที่หน้าตัดของลวดตัวนำ [ตารางเมตร]

เมื่อพิจารณาถึงการเปลี่ยนแปลงคุณลักษณะของลวดตัวนำเมื่อได้รับแรงกระทำแล้ว จะพบว่ามีการเปลี่ยนแปลงเกิดขึ้นสองประการคือ ความยาวของลวดตัวนำเปลี่ยนไปจากเดิมและความต้านทานของลวดตัวนำก็เปลี่ยนไปจากเดิม เพราะฉะนั้นถ้านำค่าทั้งสองชนิดนี้ไปทำการเทียบสัดส่วนกันก็จะได้ค่าตัวประกอบชนิดหนึ่งซึ่งมีชื่อเรียกว่า ค่าตัวประกอบเกจ (Gage factor) ซึ่งเขียนเป็นสมการได้ดังนี้

$$K = \frac{\Delta R / R}{\Delta L / L} \tag{2.28}$$

K คือ ค่าตัวประกอบเกจ

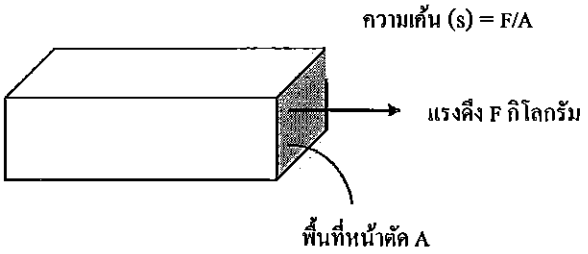
ΔR คือ ค่าความต้านทานที่เปลี่ยนแปลงไปของลวดตัวนำหลังจากถูกแรงกระทำ

R คือ ค่าความต้านทานของลวดตัวนาเริ่มแรกก่อนถูกแรงกระทำ

ΔL คือ ค่าความยาวที่เปลี่ยนแปลงไปของลวดตัวนำหลังจากถูกแรงกระทำ

L คือ ค่าความยาวของลวดตัวนำก่อนถูกแรงกระทำ

และเนื่องจากค่า $\Delta L / L$ ได้รับการกำหนดชื่อทางกลศาสตร์ว่าค่าคงตัวความเครียด แทนด้วยตัวแปร ε จึงสามารถเขียนสมการได้เป็น $K = \frac{\Delta R / R}{\varepsilon}$ สมมุติว่าแรงทางกลศาสตร์ที่กระทำต่อ วัสดุโครงสร้างดังกล่าวมีค่าเท่ากับ F ซึ่งมีหน่วยเป็นกิโลกรัมต่อตารางเมตร และพื้นที่หน้าตัดของวัสดุดังกล่าวมีค่าเท่ากับ A ตารางเมตร



รูปที่ 2.23 ความเค้นที่เกิดจากพื้นที่หน้าตัดของวัตถุต่อแรงดึง

จะสามารถคำนวณหาค่าความเค้นที่กระทำต่อวัสดุนั้นได้จากสมการ

$$\sigma = \frac{F}{A} \tag{2.29}$$

σ คือ ค่าความเค้นของวัสดุ

F คือ แรงที่กระทำต่อวัสดุ

A คือ พื้นที่หน้าตัดของวัสดุ

โดยเราจะพบว่าความเค้นจะมีการเปลี่ยนแปลง โดยตรงกับความเครียดในช่วงระยะหนึ่ง แต่เมื่อผ่านช่วงนี้ไปแล้วความสัมพันธ์เป็นเชิงเส้น ความชันของกราฟในช่วงดังกล่าวจะมีค่าเท่ากับค่าความเค้น (Stress) หารด้วยค่าความเครียด (Strain) มีชื่อเรียกว่าค่า มอดูลัสความยืดหยุ่น (Modulus Of Elasticity) ซึ่งมีค่าเท่ากับ

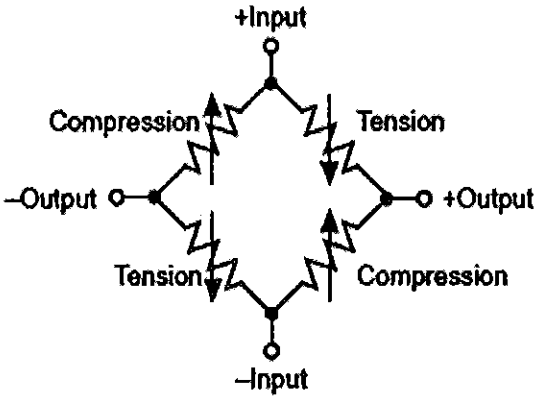
$$E = \frac{\sigma}{\varepsilon} \tag{2.30}$$

E คือ ค่ามอดูลัสความยืดหยุ่น

σ คือ ความเค้นของวัสดุ

ε คือ ค่าความเครียดของวัสดุ

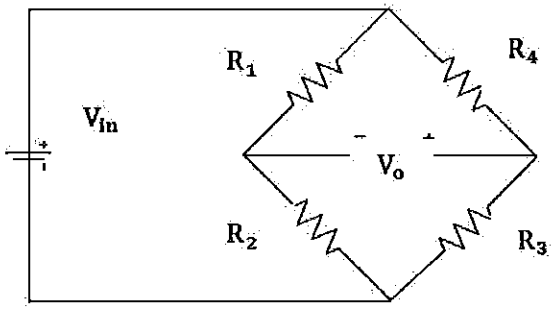
วงจรวีทสโตนบริดจ์ เนื่องจากค่าความต้านทานที่เปลี่ยนไปมีค่าค่อนข้างต่ำ ดังนั้นในทางปฏิบัติจึงนิยมนำสเตรนเกจมาใช้งานโดยต่อวงจรแบบวีทสโตนบริดจ์ ดังในรูป



รูปที่ 2.24 วงจรการบีบอัดของสเตรนเกจในวงจรวีทสโตนบริดจ์

เมื่อเราป้อนแรงดันให้แก่วงจรบริดจ์ระหว่างขั้วอินพุตบวกและอินพุตลบ ในสภาวะที่ยังไม่มีแรงมากระทำหรือยังไม่มีน้ำหนักมากระทำต่อโหลดเซลล์ ค่าความต้านทานของสเตรนเกจภายในจะทีค่าเท่ากันทำให้วงจรบริดจ์อยู่ในสภาวะสมดุล แรงดันเอาท์พุตที่ออกมาระหว่างขั้วเอาท์พุตบวกและเอาท์พุตลบจะมีค่าเป็นศูนย์ และเมื่อมีแรงมากระทำหรือมีน้ำาหนักมากระทำต่อโหลดเซลล์ จะทำให้สเตรนเกจยืดออกหรืองอเข้าจะทำให้ค่าความต้านทานภายในสเตรนเกจของแต่ละตัวนั้นเปลี่ยนค่าไปทำให้วงจรบริดจ์อยู่ในสภาวะที่ไม่สมดุล ทำให้สามารถวัดแรงดันที่เอาท์พุตออกมาได้ ยิ่งมีน้ำหนักหรือวัตถุที่มากระทำต่อโหลดเซลล์มากเพียงใดก็จะทำให้ค่าความต้านทานของสเตรนเกจนั้นเปลี่ยนค่าไปมากขึ้นและยังทำให้แรงดันเอาท์พุตมีค่ามากขึ้นด้วย อย่างไรก็ตามแรงดันเอาท์พุตที่ได้จากวงจรบริดจ์มีค่าน้อยมากจึงต้องอาศัยวงจรขยายสัญญาณ เพื่อให้แรงดันเอาท์พุตนั้นมีค่าเพิ่มมากขึ้นเพื่อที่จะนำแรงดันที่ได้ไประมวลผลในกระบวนการต่อไปโดยแรงดันเอาท์พุตภายในวงจรวีทสโตนบริดจ์จะเป็นไปตามสมการ

$$V = \left[\frac{R_3}{R_3 + R_4} - \frac{R_2}{R_1 + R_2} \right] V_{in} \tag{2.31}$$



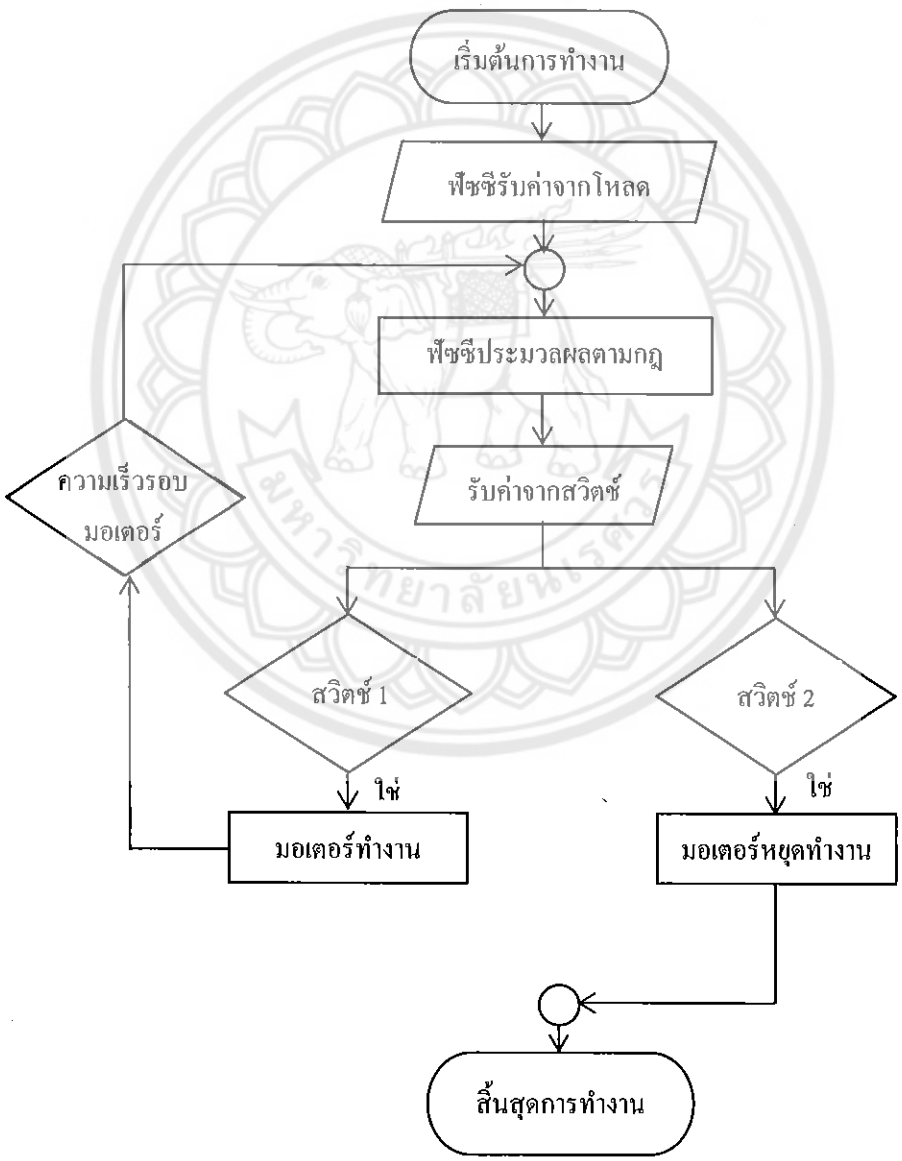
รูปที่ 2.25 วงจรบริดจ์เมื่อเชื่อมต่อกับแหล่งจ่ายไฟ

บทที่ 3

วิธีดำเนินโครงงาน

หลังจากศึกษาหลักการและทฤษฎีที่เกี่ยวข้องของโครงงานในบทที่ผ่านมา สามารถนำหลักการดังกล่าวมาประยุกต์ เพื่อสร้างหุ่นยนต์ควบคุมความเร็วโดยฟัซซีเมื่อโหลดมีการเปลี่ยนแปลง โดยมีขั้นตอนและการดำเนินงาน ดังต่อไปนี้

3.1 ขั้นตอนการทำงานของหุ่นยนต์ควบคุมความเร็ว



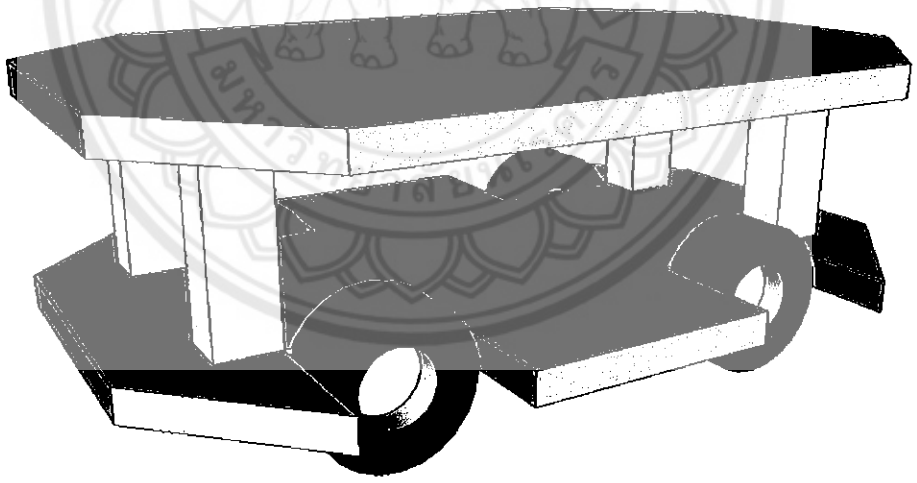
รูปที่ 3.1 แผนผังขั้นตอนการทำงานของหุ่นยนต์ควบคุมความเร็ว

จากรูปที่ 3.1 เมื่อเริ่มต้นการทำงานหุ่นยนต์จะรับค่าอินพุตเป็นน้ำหนักของโหลด จากนั้นระบบควบคุมฟัซซีจะทำการประมวลผลค่าจากอินพุต เพื่อนำไปใช้ในการควบคุมความเร็วของมอเตอร์ แล้วรับค่าอินพุตจากสวิตช์เพื่อให้หุ่นยนต์ทำงานตามคำสั่ง เมื่อกดสวิตช์ 1 มอเตอร์จะทำงานทำให้หุ่นยนต์เคลื่อนที่ไปตามคำสั่งแล้วตรวจสอบเงื่อนไขความเร็วรอบของมอเตอร์ว่ามีความเร็วรอบตามที่กำหนดหรือไม่ แล้วส่งค่าป้อนกลับไปยังระบบประมวลผลฟัซซี และเมื่อกดสวิตช์ 2 มอเตอร์จะหยุดทำงาน ทำให้ระบบสิ้นสุดการทำงาน

3.2 โครงสร้างหุ่นยนต์ควบคุมความเร็ว

3.2.1 ออกแบบโครงสร้างหุ่นยนต์ควบคุมความเร็ว

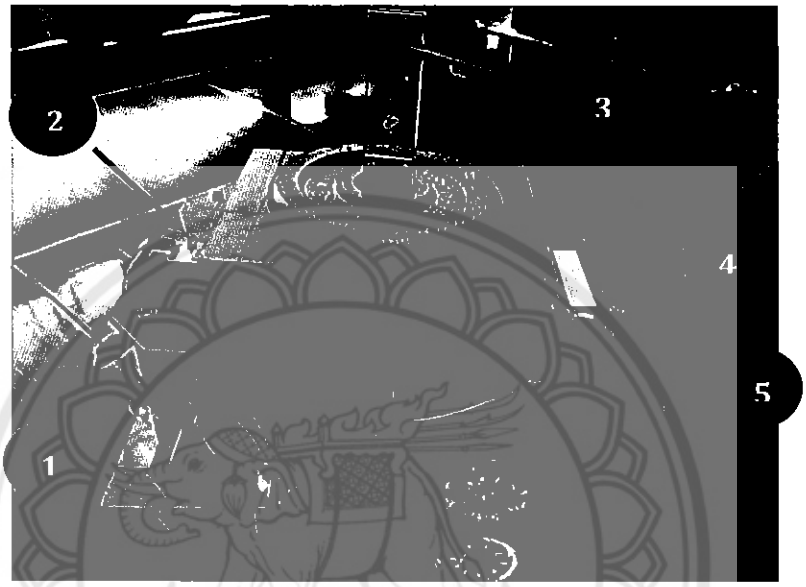
การออกแบบโครงสร้างหุ่นยนต์ควบคุมความเร็วจะแบ่งโครงสร้างออกเป็น 2 ส่วนคือส่วนที่ 1 ด้านล่างจะวางแผงวงจรควบคุมและแบตเตอรี่ ส่วนที่ 2 ด้านบนใช้วางโหลดเซลล์ ดังแสดงในรูปที่ 3.2



รูปที่ 3.2 การออกแบบโครงสร้างหุ่นยนต์ควบคุมความเร็ว

3.2.2 โครงสร้างของหุ่นยนต์ควบคุมความเร็วเมื่อสร้างเสร็จ

ทำการสร้างโครงหุ่นยนต์โดยใช้อะคริลิคมาประกอบชิ้นส่วนและยึดติดเข้าด้วยกัน จากนั้นนำแผงวงจรควบคุมกับตัวเซ็นเซอร์มาติดตั้งที่ชั้นล่าง และโหลดเซลล์, หลอดไฟแอลอีดี และจอแอลซีดีมาติดตั้งที่ชั้นบน ดังแสดงในรูปที่ 3.3



รูปที่ 3.3 โครงสร้างของหุ่นยนต์ควบคุมความเร็ว

(1) เซ็นเซอร์ตรวจวัดความเร็วรอบมอเตอร์

(2) หลอดไฟแอลอีดีใช้แสดงสถานะความเร็วคงที่ของหุ่นยนต์

(3) ชั้นโหลดเซลล์ ใช้ตรวจวัดน้ำหนักของโหลด

(4) จอแอลซีดีแสดงค่าน้ำหนักของโหลด และความเร็วรอบของมอเตอร์

(5) ชั้นแผงวงจรควบคุมและแบตเตอร์รี่

3.3 ออกแบบฮาร์ดแวร์ของหุ่นยนต์ควบคุมความเร็ว

ส่วนประกอบที่เป็นฮาร์ดแวร์ (Hardware) ของหุ่นยนต์ควบคุมความเร็วมีดังนี้

(1) แผงวงจรไมโครคอนโทรลเลอร์

(2) แผงวงจรขับมอเตอร์

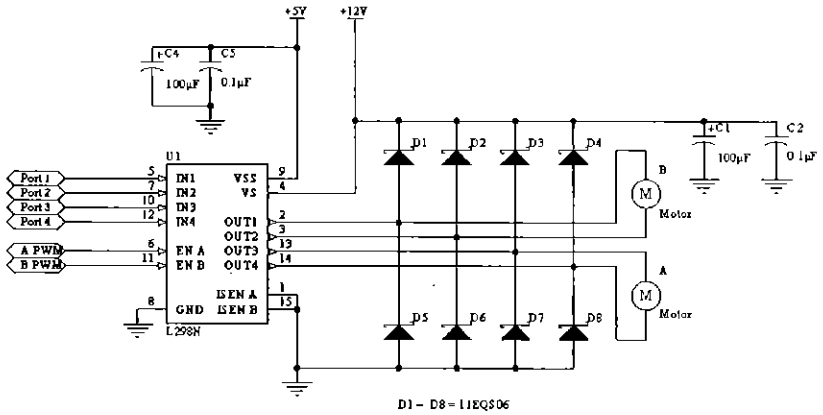
(3) แผงวงจรสวิตช์ควบคุมการทำงานของหุ่นยนต์

3.3.1 แผงวงจรไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ในโครงงานนี้เลือกใช้ไมโครคอนโทรลเลอร์ตระกูล AVR เนื่องจากเป็นไมโครคอนโทรลเลอร์ที่นำมาประยุกต์ในการใช้งานได้อย่างสะดวก ซึ่งไมโครคอนโทรลเลอร์ตระกูล AVR ที่นำมาใช้คือไมโครคอนโทรลเลอร์เบอร์ ATmega2560 เนื่องจากเป็นแพลตฟอร์ม (Platform) ของอินพุต/เอาท์พุต (I/O) ขั้นพื้นฐานที่พอเพียงกับการใช้งานและการเรียนรู้ โดยตัวแผงวงจรมีชุดคำสั่งที่ใช้ควบคุมพอร์ต อินพุต/เอาท์พุต ไม่ว่าจะเป็นพอร์ตดิจิตอล พอร์ตอนาลอกพีดับเบิลยูเอ็มและพอร์ตอนุกรม ซึ่งแผงวงจรไมโครคอนโทรลเลอร์นี้ทำให้คอมพิวเตอร์สามารถรับสัญญาณจากภายนอก และส่งสัญญาณไปควบคุมอุปกรณ์ภายนอกได้อย่างมีประสิทธิภาพมากกว่าการใช้เครื่องคอมพิวเตอร์ ตัวแผงวงจรออกแบบจากไมโครคอมพิวเตอร์ชิพเดี่ยว และมีโปรแกรมพัฒนาสำหรับเขียนโปรแกรมให้สามารถรับสัญญาณจากสวิตช์หรือตัวรับรู้และควบคุมหลอดไฟมอเตอร์ หรืออุปกรณ์อื่น ๆ และสามารถทำงานได้อิสระหรือทำงานติดต่อกับโปรแกรมที่ทำงานบนเครื่องคอมพิวเตอร์ได้

3.3.2 แผงวงจรขับมอเตอร์

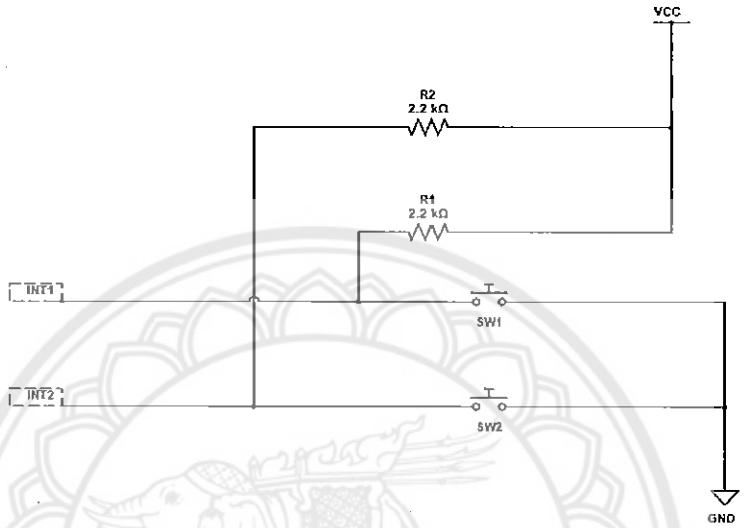
วงจรขับมอเตอร์ที่ใช้เป็นไอซีเบอร์ L298N เป็นตัวขับเคลื่อนหลักสามารถใช้ในการขับความร้อนต่ำ มีวงจรเรกูเลตในตัว ใช้สำหรับขับดีซีมอเตอร์ ซึ่งจะขับดีซีมอเตอร์ได้ 2 ตัวพร้อมกัน สามารถควบคุมการหมุนกลับทิศทางได้แบบอิสระ รองรับแรงดันไฟฟ้ากระแสตรงได้กว้าง 7-35 โวลต์ มีกระแสสูงสุดได้ 2 แอมป์ต่อข้าง และกำลังสูงสุดที่สามารถใช้ได้ 20 วัตต์ ซึ่งมีวงจรการทำงานดังแสดงในรูปที่ 3.4 ดังนี้



รูปที่ 3.4 วงจรขับมอเตอร์ที่ใช้ไอซีเบอร์ L298N

3.3.3 แผงวงจรสวิตช์ควบคุมการทำงานของหุ่นยนต์

การออกแบบวงจรสวิตช์นี้ใช้ตัวต้านทานขนาด 2.2 กิโลโอห์มและสวิตช์ อย่างละ 2 ตัว เมื่อกดปุ่มสวิตช์ 1 เริ่มการทำงาน คือปุ่มกดสำหรับให้หุ่นยนต์เริ่มการทำงาน กดปุ่มสวิตช์ 2 ให้หุ่นยนต์หยุดการทำงาน ดังแสดงในรูปที่ 3.6 ดังนี้



รูปที่ 3.5 วงจรสวิตช์ควบคุมการทำงานของหุ่นยนต์

3.4 ออกแบบตัวควบคุมแบบฟัซซี

โครงสร้างการทำงานของระบบควบคุมนี้ถูกออกแบบด้วยโปรแกรมแมทแลปเพื่อจำลองการควบคุมความเร็วของหุ่นยนต์โดยมีการรับค่าอินพุตจากค่าการเปลี่ยนแปลงน้ำหนักของโหลดจากโหลดเซลล์และส่งเป็นค่าเอาท์พุตในการควบคุมความเร็วของมอเตอร์ จึงทำการออกแบบตัวควบคุมแบบฟัซซีได้เป็น 3 ขั้นตอนหลักดังนี้

(1) การกำหนดอินพุตเอาท์พุตของระบบ

(2) การสร้างฐานกฎฟัซซี

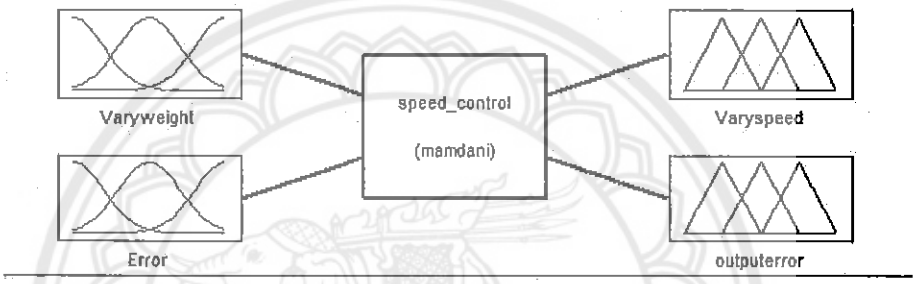
(3) การประมวลผลฐานกฎฟัซซี

ได้ทำการออกแบบตัวควบคุมแบบฟัซซี 3 รูปแบบ เพื่อดูว่าการออกแบบครั้งไหนเหมาะสมกับการนำมาใช้งานจริง

3.4.1 การออกแบบตัวควบคุมฟัซซีแบบที่ 1

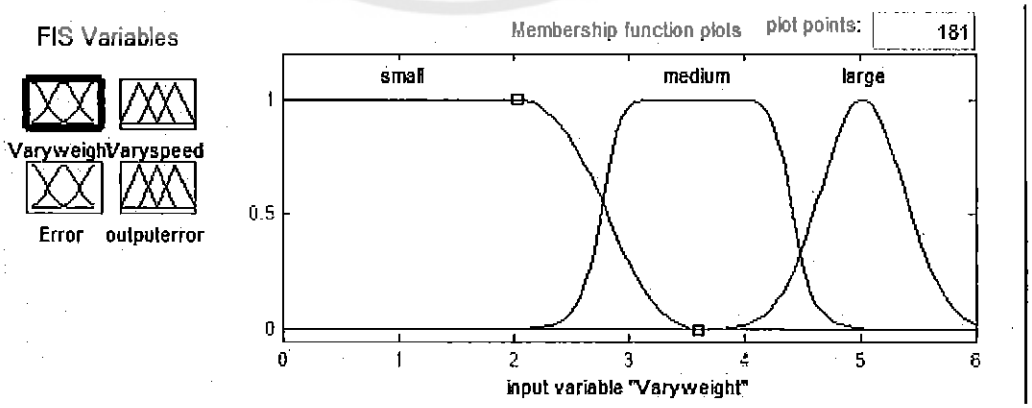
(1) การกำหนดอินพุตและเอาท์พุตของระบบ

การกำหนดค่าอินพุตได้นำค่าการเปลี่ยนแปลงของโหลดน้ำหนักกับค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักมากำหนดเป็นฟัซซีเซตจะได้อินพุต 2 ตัวและจะได้เอาท์พุตเป็นค่าการเปลี่ยนแปลงความเร็วของมอเตอร์กับค่าเอาท์พุตความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนัก ดังแสดงในรูปที่ 3.6 ดังนี้



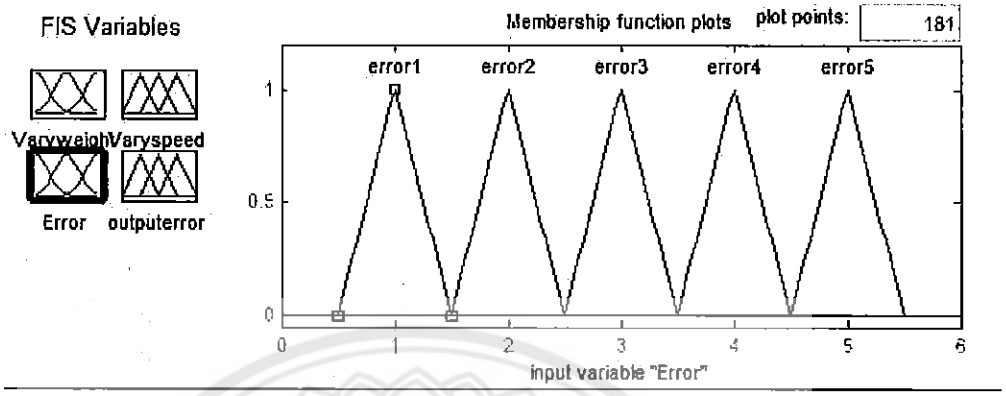
รูปที่ 3.6 รูปแบบการอนุมานอินพุตและเอาท์พุตของฟัซซี

อินพุตค่าการเปลี่ยนแปลงของโหลดน้ำหนักซึ่งแบ่งฟัซซีเซตออกเป็น small, medium และ large มีขอบเขตของการเปลี่ยนแปลงโหลดน้ำหนักตั้งแต่ 0 กิโลกรัม ถึง 6 กิโลกรัม ดังแสดงในรูปที่ 3.7 ดังนี้



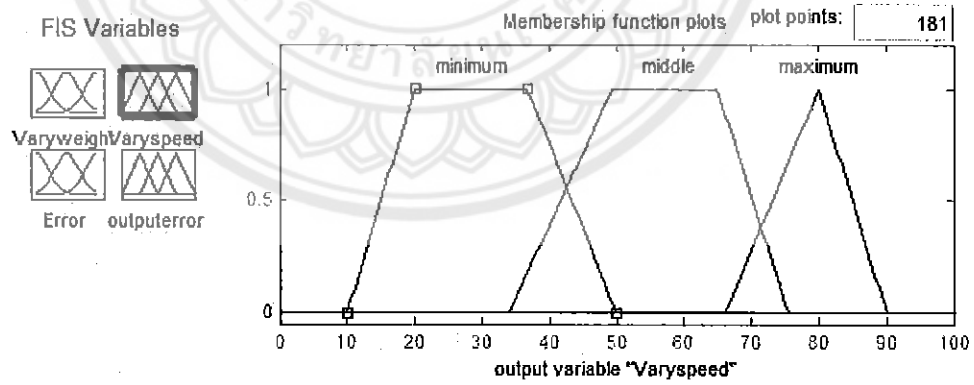
รูปที่ 3.7 ฟัซซีเซตสำหรับปริมาณอินพุตค่าการเปลี่ยนแปลงของโหลดน้ำหนักแบบที่ 1

อินพุตค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักซึ่งแบ่งฟัซซีเซตออกเป็น error1, error2, error3, error4 และ error5 มีขอบเขตของค่าความคลาดเคลื่อนตั้งแต่ 0 ถึง 6 ดังแสดงในรูปที่ 3.8 ดังนี้



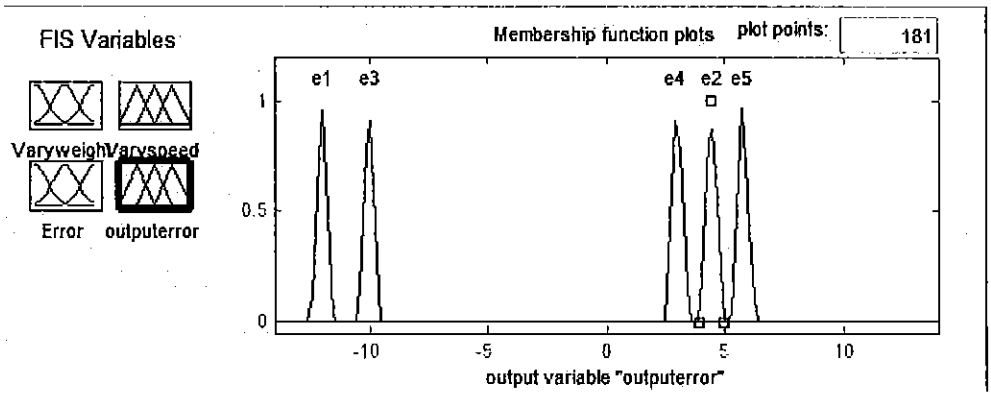
รูปที่ 3.8 ฟัซซีเซตสำหรับปริมาณอินพุตค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักแบบที่ 1

เอาท์พุตค่าการเปลี่ยนแปลงความเร็วของมอเตอร์ซึ่งแบ่งฟัซซีเซตออกเป็น minimum, middle และ maximum มีขอบเขตของค่าการเปลี่ยนแปลงความเร็วของมอเตอร์ตั้งแต่ 0 ถึง 100 ดังแสดงในรูปที่ 3.9 ดังนี้



รูปที่ 3.9 ฟัซซีเซตสำหรับปริมาณเอาท์พุตค่าการเปลี่ยนแปลงความเร็วของมอเตอร์แบบที่ 1

เอาท์พุตค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักซึ่งแบ่งฟัซซีเซตออกเป็น e1,e2,e3,e4 และ e5 มีขอบเขตของค่าความคลาดเคลื่อนตั้งแต่ -14 ถึง 14 ดังแสดงในรูปที่ 3.10 ดังนี้



รูปที่ 3.10 ฟัซซีเซตสำหรับปริมาณเอาท์พุตค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนัก แบบที่ 1

(2) การสร้างฐานกฎฟัซซี

เมื่อทราบปริมาณอินพุตแล้ว ต้องมีกฎฟัซซีเพื่อที่จะสามารถเชื่อมโยงจากปริมาณอินพุตไปยังปริมาณเอาต์พุตได้ เขียนอยู่ในรูปแบบความสัมพันธ์ดังนี้

"IF x is A THEN y is B"

กฎการควบคุมความเร็วของหุ่นยนต์ที่สร้างขึ้นมีดังนี้

(1) if (Varyweight is small) then (Varyspeed is minimum)

(2) if (Varyweight is medium) then (Varyspeed is middle)

(3) if (Varyweight is large) then (Varyspeed is maximum)

กฎการควบคุมค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนัก

(1) if (Error is error1) then (outputerror is e1)

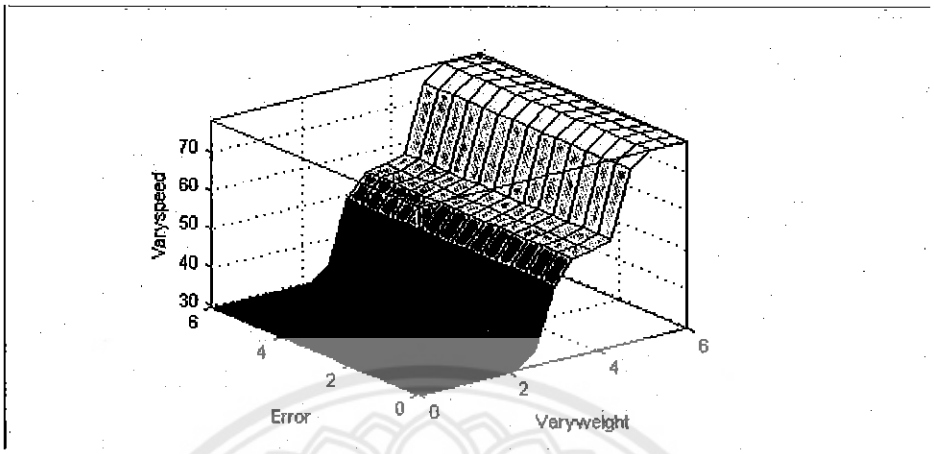
(2) if (Error is error2) then (outputerror is e2)

(3) if (Error is error3) then (outputerror is e3)

(4) if (Error is error4) then (outputerror is e4)

(5) if (Error is error5) then (outputerror is e5)

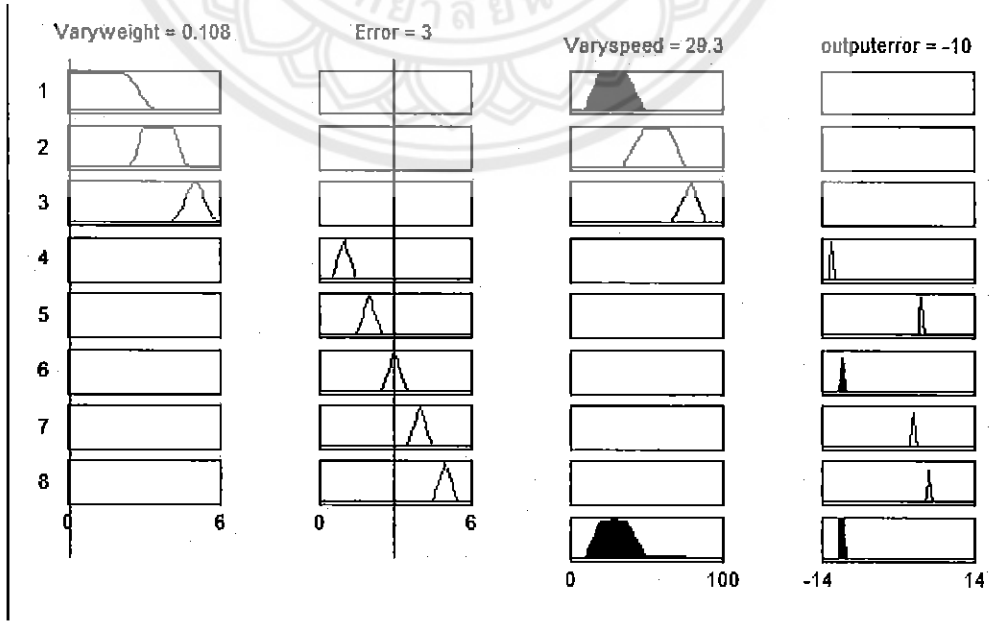
จากกฎการควบคุมที่สร้างมาข้างต้นสามารถเรียกดูมุมมองพื้นผิวของกฎจากกราฟดังแสดงในรูปที่ 3.11 ดังนี้



รูปที่ 3.11 มุมมองพื้นผิวของกฎการควบคุมความเร็วของหุ่นยนต์แบบที่ 1

(3) การประมวลผลฐานกฎฟัซซี

เมื่อนำข้อมูลค่าอินพุตการเปลี่ยนแปลงของโหลดน้ำหนักที่ 3 กิโลกรัม และค่าอินพุตความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักที่ 3 มาวิเคราะห์ทำการคำนวณกฎการควบคุมความเร็วของหุ่นยนต์และค่าความคลาดเคลื่อน จะได้ดังรูปที่ 3.12 ดังนี้



รูปที่รูปที่ 3.12 การประมวลผลค่าเอาท์พุตด้วยวิธีแมมดานิแบบที่ 1

สรุปการพยากรณ์ข้อมูลเบื้องต้น ดังตารางที่ 3.1 ดังนี้

ตารางที่ 3.1 ข้อมูลการพยากรณ์ค่าความเป็นไปได้ของเอาท์พุตแบบที่ 1

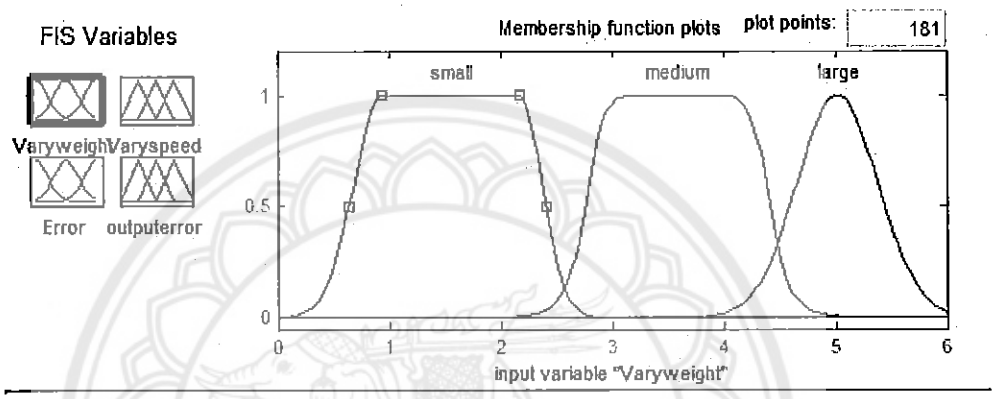
ค่าอินพุตของการเปลี่ยนแปลงโหลด	ค่าอินพุตของความคลาดเคลื่อน	ค่าเอาท์พุตของการเปลี่ยนแปลงความเร็ว	ค่าเอาท์พุตของความคลาดเคลื่อน
0	0	29	0
1	1	29	-12
2	2	29	4
3	3	52	-10
4	4	56	3
5	5	78	6

เมื่อระบบรับค่าอินพุตของการเปลี่ยนแปลงโหลดกับค่าอินพุตของความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักตามตาราง จะได้ค่าเอาท์พุตของการเปลี่ยนแปลงความเร็ว และค่าเอาท์พุตของความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักที่ประมวลผลออกมาเป็นไปตามตาราง เพื่อนำค่าเอาท์พุตที่ได้ไปควบคุมความเร็วของมอเตอร์ให้คงที่ แต่การออกแบบนี้เกิดปัญหาเมื่อการเปลี่ยนแปลงโหลดเริ่มต้นเป็น 0 กิโลกรัม เอาท์พุตของการเปลี่ยนแปลงความเร็วจะประมวลผลออกมาเป็นการเปลี่ยนแปลงช่วงเดียวกับ 1 กิโลกรัมและ 2 กิโลกรัม

3.4.2 การออกแบบตัวควบคุมฟัซซีแบบที่ 2

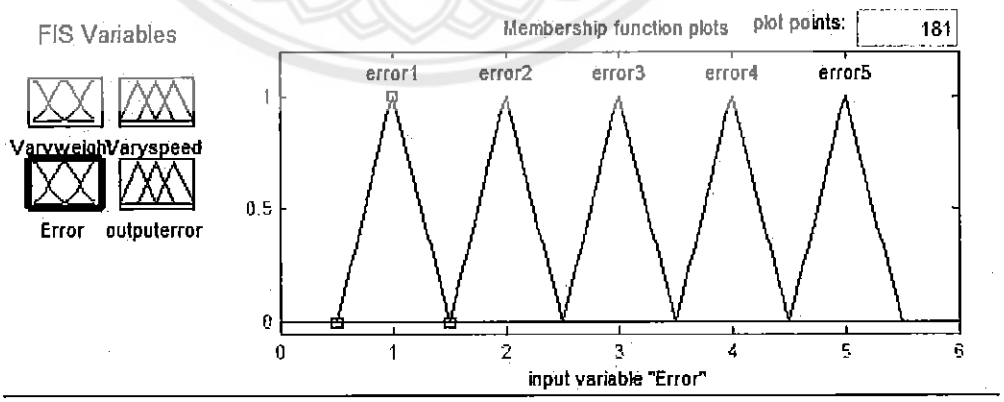
(1) การกำหนดอินพุตและเอาท์พุตของระบบ

อินพุตค่าการเปลี่ยนแปลงของโหลดน้ำหนักซึ่งแบ่งฟัซซีเซตออกเป็น small, medium และ large มีขอบเขตของการเปลี่ยนแปลงโหลดน้ำหนักตั้งแต่ 0 กิโลกรัม ถึง 6 กิโลกรัม ดังแสดงในรูปที่ 3.13 ดังนี้



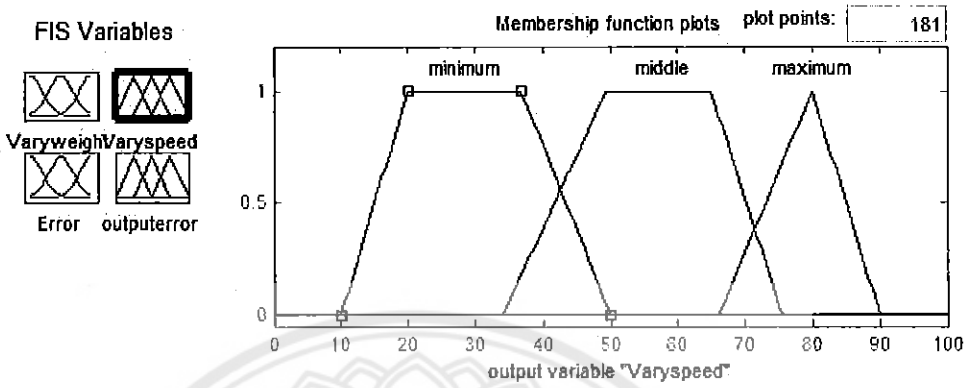
รูปที่ 3.13 ฟัซซีเซตสำหรับปริมาณอินพุตค่าการเปลี่ยนแปลงของโหลดน้ำหนักแบบที่ 2

อินพุตค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักซึ่งแบ่งฟัซซีเซตออกเป็น error1, error2, error3, error4 และ error5 มีขอบเขตของค่าความคลาดเคลื่อนตั้งแต่ 0 ถึง 6 ดังแสดงในรูปที่ 3.14 ดังนี้



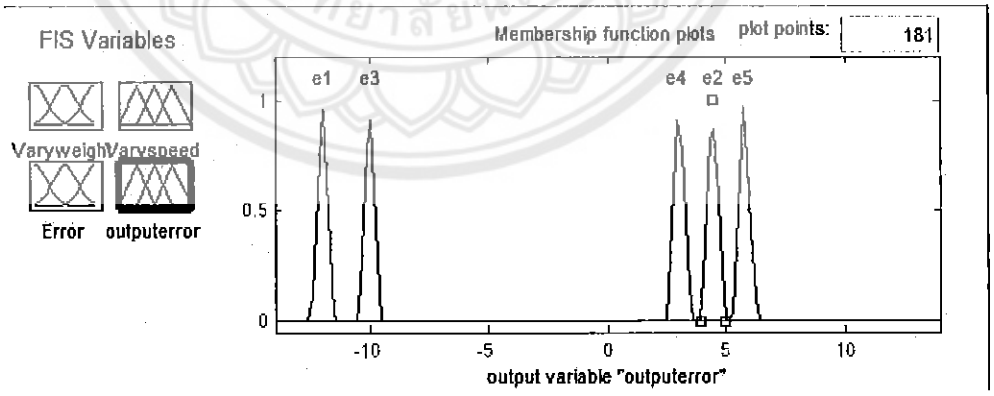
รูปที่ 3.14 ฟัซซีเซตสำหรับปริมาณอินพุตค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักแบบที่ 2

เอาท์พุตค่าการเปลี่ยนแปลงความเร็วของมอเตอร์ซึ่งแบ่งฟัซซีเซตออกเป็น minimum, middle และ maximum มีขอบเขตของค่าการเปลี่ยนแปลงความเร็วของมอเตอร์ตั้งแต่ 0 ถึง 100 ดังแสดงในรูปที่ 3.15 ดังนี้



รูปที่ 3.15 ฟัซซีเซตสำหรับปริมาณเอาท์พุตค่าการเปลี่ยนแปลงความเร็วของมอเตอร์แบบที่ 2

เอาท์พุตค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักซึ่งแบ่งฟัซซีเซตออกเป็น e1,e2,e3,e4 และ e5 มีขอบเขตของค่าความคลาดเคลื่อนตั้งแต่ -14 ถึง 14 ดังแสดงในรูปที่ 3.16 ดังนี้



รูปที่ 3.16 ฟัซซีเซตสำหรับปริมาณเอาท์พุตค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักแบบที่ 2

(2) การสร้างฐานกฎฟัซซี

เมื่อทราบปริมาณอินพุตแล้ว ต้องมีกฎฟัซซีเพื่อที่จะสามารถเชื่อมโยงจากปริมาณอินพุตไปยังปริมาณเอาต์พุตได้ เขียนอยู่ในรูปแบบความสัมพันธ์ดังนี้

"IF x is A THEN y is B"

กฎการควบคุมความเร็วของหุ่นยนต์ที่สร้างขึ้นมีดังนี้

(1) if (Varyweight is small) then (Varyspeed is minimum)

(2) if (Varyweight is medium) then (Varyspeed is middle)

(3) if (Varyweight is large) then (Varyspeed is maximum)

กฎการควบคุมค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนัก

(1) if (Error is error1) then (outputerror is e1)

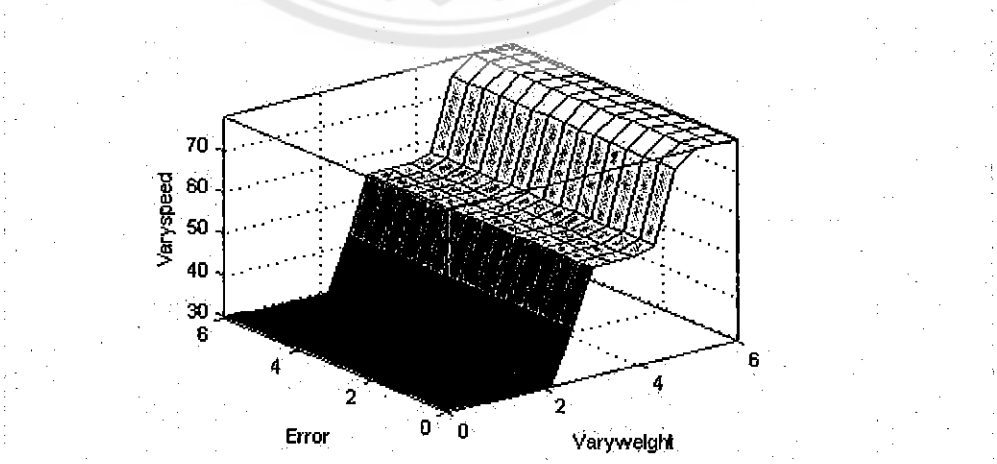
(2) if (Error is error2) then (outputerror is e2)

(3) if (Error is error3) then (outputerror is e3)

(4) if (Error is error4) then (outputerror is e4)

(5) if (Error is error5) then (outputerror is e5)

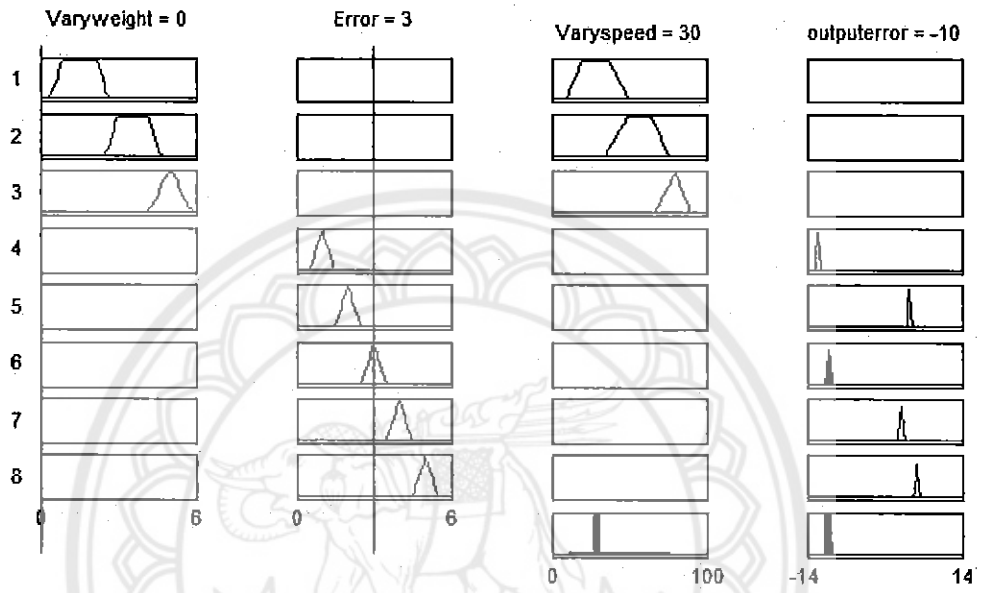
จากกฎการควบคุมที่สร้างมาข้างต้นสามารถเรียกดูมุมมองพื้นผิวของกฎจากกราฟดังแสดงในรูปที่ 3.17 ดังนี้



รูปที่ 3.17 มุมมองพื้นผิวของกฎการควบคุมความเร็วของหุ่นยนต์แบบที่ 2

(3) การประมวลผลฐานกฎฟัซซี

เมื่อนำข้อมูลค่าอินพุตการเปลี่ยนแปลงของโหลดน้ำหนักที่ 3 กิโลกรัม และค่าอินพุตความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักที่ 3 มาวิเคราะห์ทำการคำนวณกฎการควบคุมความเร็วของหุ่นยนต์และค่าความคลาดเคลื่อน จะได้ดังรูปที่ 3.18 ดังนี้



รูปที่ 3.18 การประมวลผลค่าเอาท์พุตด้วยวิธีแมมดานิแบบที่ 2

สรุปการพยากรณ์ข้อมูลเบื้องต้น ดังตารางที่ 3.2 ดังนี้

ตารางที่ 3.2 ข้อมูลการพยากรณ์ค่าความเป็นไปได้ของเอาท์พุตแบบที่ 2

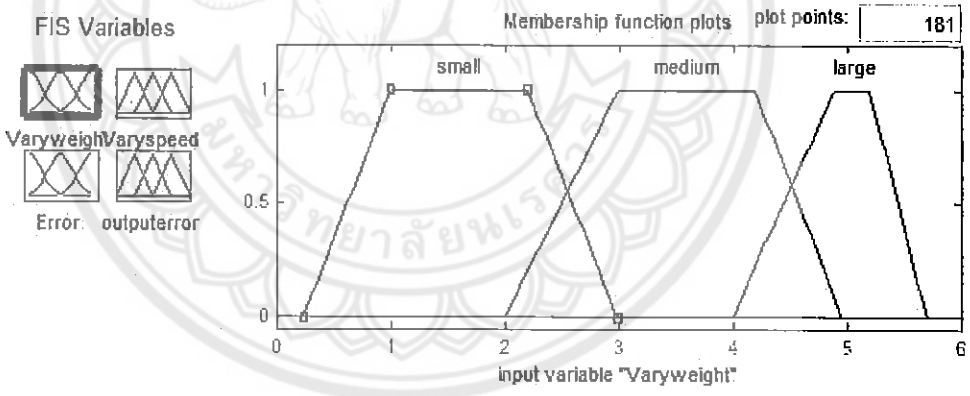
ค่าอินพุตของการเปลี่ยนแปลงโหลด	ค่าอินพุตของความคลาดเคลื่อน	ค่าเอาท์พุตของการเปลี่ยนแปลงความเร็ว	ค่าเอาท์พุตของความคลาดเคลื่อน
0	0	30	0
1	1	29	-12
2	2	29	4
3	3	55	-10
4	4	56	3
5	5	78	6

เมื่อระบบรับค่าอินพุตของการเปลี่ยนแปลงโหลดกับค่าอินพุตของความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักตามตาราง จะได้ค่าเอาท์พุตของการเปลี่ยนแปลงความเร็ว และค่าเอาท์พุตของความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักที่ประมวลผลออกมาเป็นไปตามตาราง เพื่อนำค่าเอาท์พุตที่ได้ไปควบคุมความเร็วของมอเตอร์ให้คงที่ แต่การออกแบบนี้เกิดปัญหาเมื่อการเปลี่ยนแปลงโหลดเริ่มต้นเป็น 0 กิโลกรัม เอาท์พุตของการเปลี่ยนแปลงความเร็วจะประมวลผลออกมาเป็นการเปลี่ยนแปลงช่วงเดียวกับ 1 กิโลกรัมและ 2 กิโลกรัม

3.4.3 การออกแบบตัวควบคุมฟัซซีครั้งที่ 3

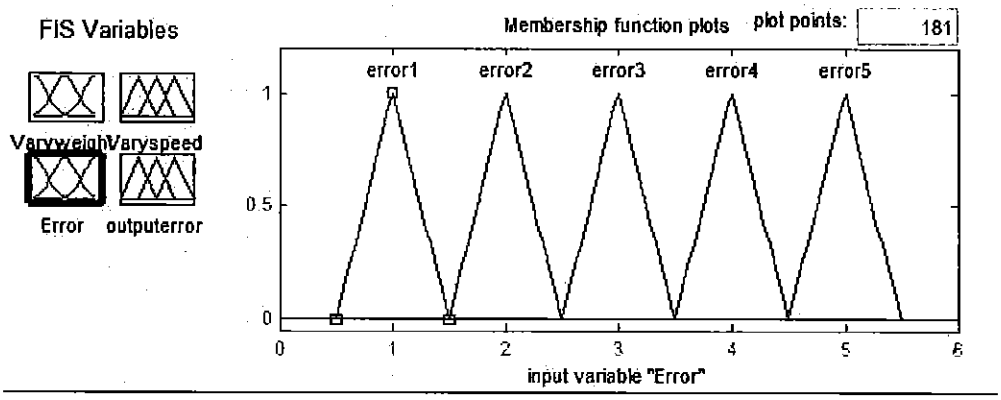
(1) การกำหนดอินพุตและเอาท์พุตของระบบ

อินพุตค่าการเปลี่ยนแปลงของโหลดน้ำหนักซึ่งแบ่งฟัซซีเซตออกเป็น small, medium และ large มีขอบเขตของการเปลี่ยนแปลงโหลดน้ำหนักตั้งแต่ 0 กิโลกรัม ถึง 6 กิโลกรัม ดังแสดงในรูปที่ 3.19 ดังนี้



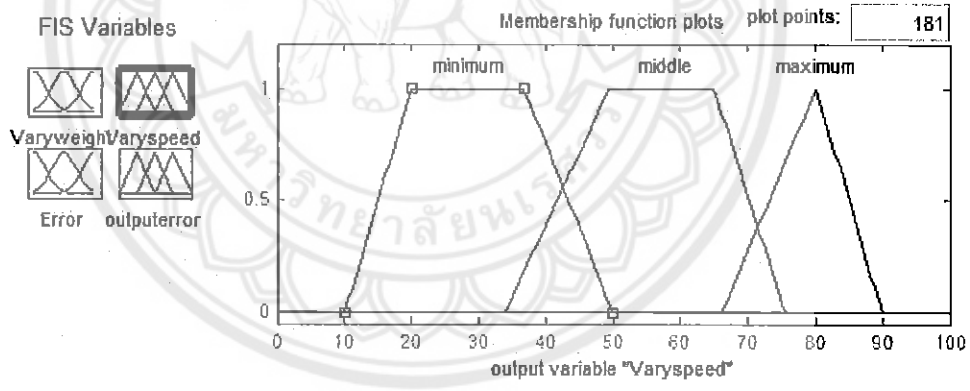
รูปที่ 3.19 ฟัซซีเซตสำหรับปริมาณอินพุตค่าการเปลี่ยนแปลงของโหลดน้ำหนักแบบที่ 3

อินพุตค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักซึ่งแบ่งฟัซซีเซตออกเป็น error1, error2, error3, error4 และ error5 มีขอบเขตของค่าความคลาดเคลื่อนตั้งแต่ 0 ถึง 6 ดังแสดงในรูปที่ 3.20 ดังนี้



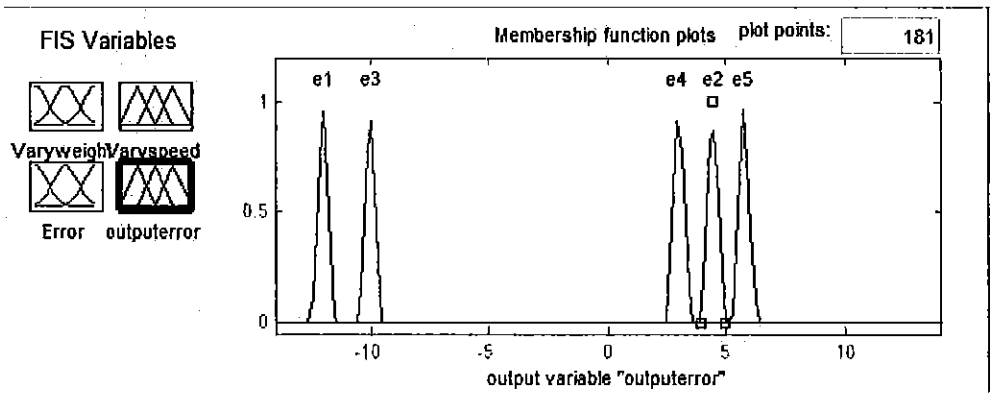
รูปที่ 3.20 ฟัซซีเซตสำหรับปริมาณอินพุตค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักแบบที่ 3

เอาท์พุตค่าการเปลี่ยนแปลงความเร็วของมอเตอร์ซึ่งแบ่งฟัซซีเซตออกเป็น minimum, middle และ maximum มีขอบเขตของค่าการเปลี่ยนแปลงความเร็วของมอเตอร์ตั้งแต่ 0 ถึง 100 ดังแสดงในรูปที่ 3.21 ดังนี้



รูปที่ 3.21 ฟัซซีเซตสำหรับปริมาณเอาท์พุตค่าการเปลี่ยนแปลงความเร็วของมอเตอร์แบบที่ 3

เอาท์พุตค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักซึ่งแบ่งฟัซซีเซตออกเป็น e1,e2,e3,e4 และ e5 มีขอบเขตของค่าความคลาดเคลื่อนตั้งแต่ -14 ถึง 14 ดังแสดงในรูปที่ 3.22 ดังนี้



รูปที่ 3.22 ฟัซซีเซตสำหรับปริมาณเอาท์พุตค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักแบบที่ 3

(2) การสร้างฐานกฎฟัซซี

เมื่อทราบปริมาณอินพุตแล้ว ต้องมีกฎฟัซซีเพื่อที่จะสามารถเชื่อมโยงจากปริมาณอินพุตไปยังปริมาณเอาต์พุตได้ เขียนอยู่ในรูปแบบความสัมพันธ์ดังนี้

"IF x is A THEN y is B"

กฎการควบคุมความเร็วของหุ่นยนต์ที่สร้างขึ้นมีดังนี้

(1) if (Varyweight is small) then (Varyspeed is minimum)

(2) if (Varyweight is medium) then (Varyspeed is middle)

(3) if (Varyweight is large) then (Varyspeed is maximum)

กฎการควบคุมค่าความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนัก

(1) if (Error is error1) then (outputerror is e1)

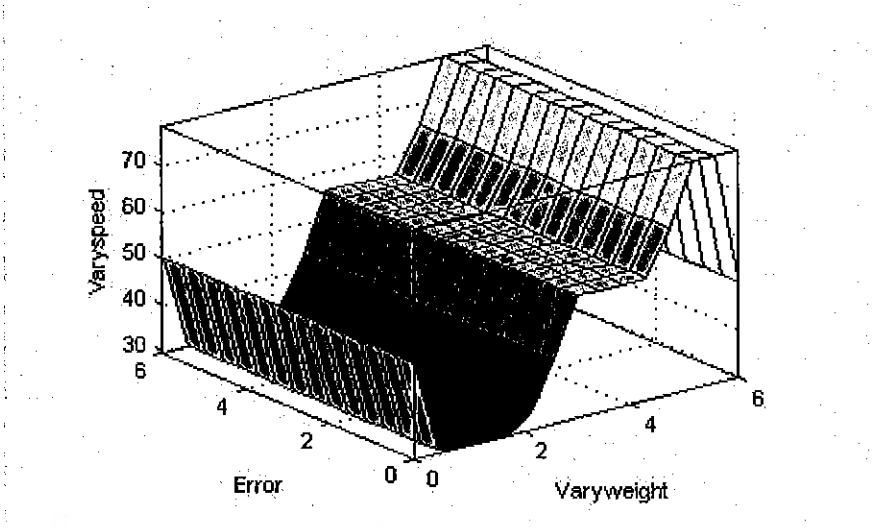
(2) if (Error is error2) then (outputerror is e2)

(3) if (Error is error3) then (outputerror is e3)

(4) if (Error is error4) then (outputerror is e4)

(5) if (Error is error5) then (outputerror is e5)

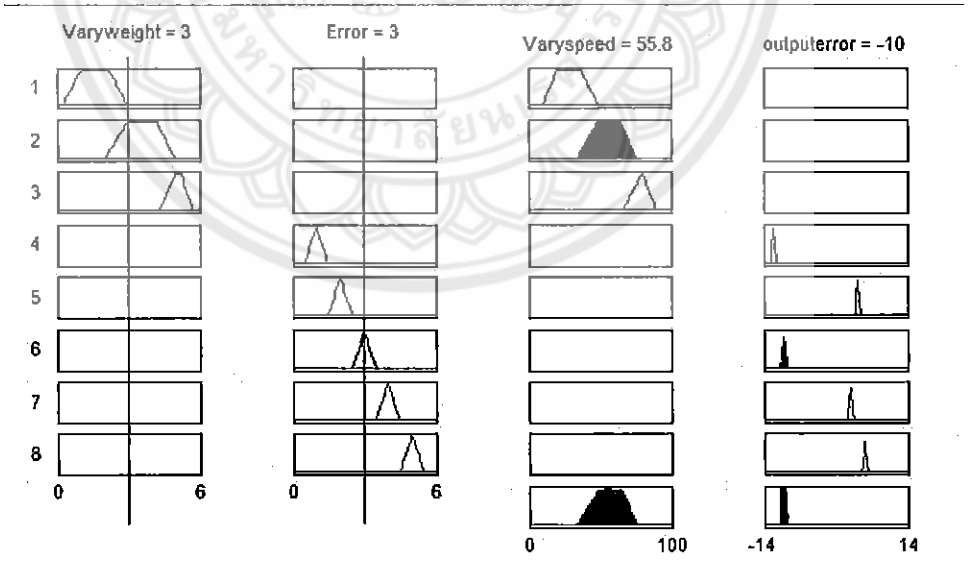
จากกฎการควบคุมที่สร้างมาข้างต้นสามารถเรียกดูมุมมองพื้นผิวของกฎจากกราฟดังแสดงในรูปที่ 3.23 ดังนี้



รูปที่ 3.23 มุมมองพื้นผิวของกฎการควบคุมความเร็วของหุ่นยนต์แบบที่ 3

(3) การประมวลผลฐานกฎฟัซซี

เมื่อนำข้อมูลค่าอินพุตการเปลี่ยนแปลงของโหลดน้ำหนักที่ 3 กิโลกรัม และค่าอินพุตความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักที่ 3 มาวิเคราะห์ทำการคำนวณกฎการควบคุมความเร็วของหุ่นยนต์และค่าความคลาดเคลื่อน จะได้ดังรูปที่ 3.24 ดังนี้



รูปที่ 3.24 การประมวลผลค่าเอาท์พุตด้วยวิธีแมมดานิแบบที่ 3

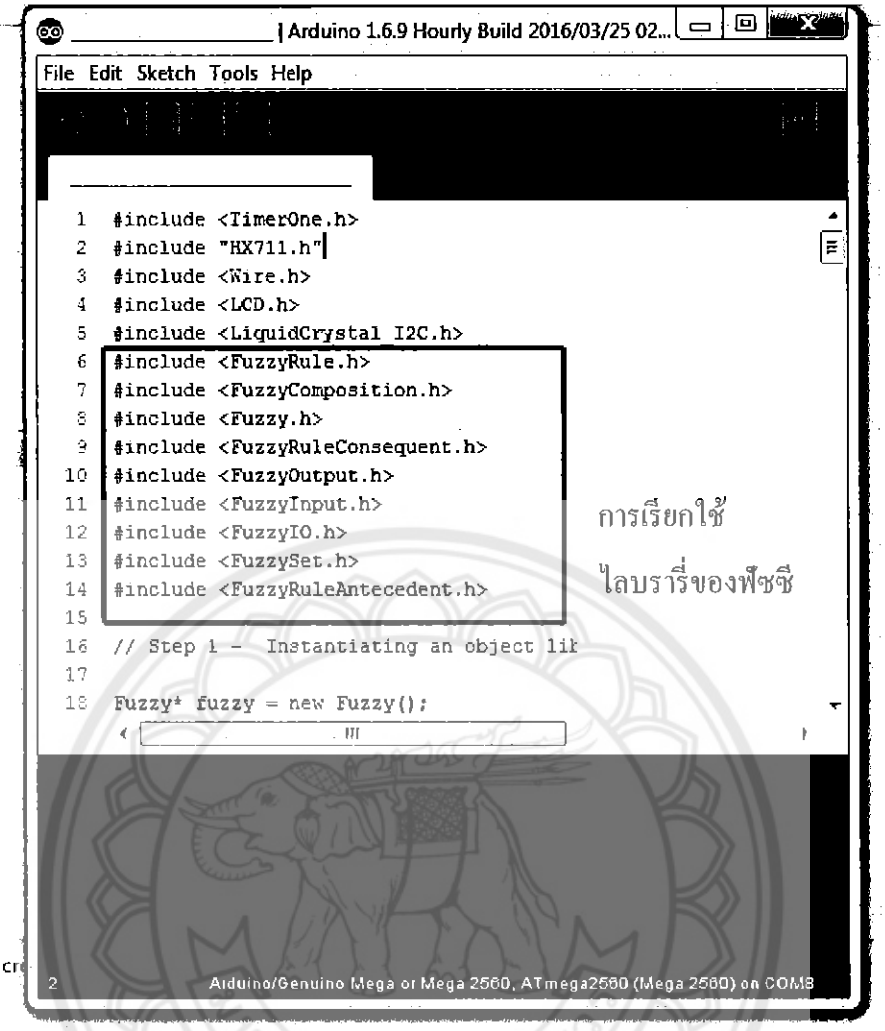
สรุปการพยากรณ์ข้อมูลเบื้องต้น ดังตารางที่ 3.3 ดังนี้

ตารางที่ 3.3 ข้อมูลการพยากรณ์ค่าความเป็นไปได้ของเอาท์พุตแบบที่ 3

ค่าอินพุตของการเปลี่ยนแปลงโหลด	ค่าอินพุตของความคลาดเคลื่อน	ค่าเอาท์พุตของการเปลี่ยนแปลงความเร็ว	ค่าเอาท์พุตของความคลาดเคลื่อน
0	0	0	0
1	1	28	-12
2	2	28	4
3	3	60	-10
4	4	60	3
5	5	80	6

เมื่อระบบรับค่าอินพุตของการเปลี่ยนแปลงโหลดกับค่าอินพุตของความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักตามตาราง จะได้ค่าเอาท์พุตของการเปลี่ยนแปลงความเร็ว และค่าเอาท์พุตของความคลาดเคลื่อนของความเร็วแต่ละโหลดน้ำหนักที่ประมวลผลออกมาเป็นไปตามตาราง เพื่อนำค่าเอาท์พุตที่ได้ไปควบคุมความเร็วของมอเตอร์ให้คงที่ การออกแบบตัวควบคุมฟัซซีในรูปแบบนี้มีความเหมาะสมกับการนำไปใช้งานจริง เนื่องจากเมื่อมีการเปลี่ยนแปลงโหลดเริ่มต้นที่ 0 กิโลกรัม เอาท์พุตที่ได้จะไม่อยู่ในช่วงของการเปลี่ยนแปลงโหลดน้ำหนักที่ 1 กิโลกรัม และ 2 กิโลกรัม

การนำตัวควบคุมฟัซซีไปใช้งานจริง โดยเรียกใช้ไลบรารี่ของฟัซซีในโปรแกรมอาดูยโน ดังแสดงในรูปที่ 3.25 ดังนี้



รูปที่ 3.25 การเรียกใช้งานไลบรารี่ของฟัซซี

บทที่ 4

ผลการทดลอง

การทดลองในบทนี้เป็นการทดลองหาค่าเวลาหน่วงของหุ่นยนต์เมื่อโหลดมีการเปลี่ยนแปลงเพื่อเปรียบเทียบค่าเวลาหน่วงที่ความเร็วต่างกันโดยได้แบ่งการทดลองดังนี้

1) การทดลองความเร็วสูงสุดของหุ่นยนต์
2) การทดลองระยะเวลาหน่วงของหุ่นยนต์เมื่อโหลดมีการเปลี่ยนแปลง
3) กราฟแสดงการเปรียบเทียบค่าความเร็วที่มีผลต่อระยะเวลาหน่วงของหุ่นยนต์

4.1 การทดลองความเร็วสูงสุดของหุ่นยนต์

ก่อนทำการทดลองได้ทำการชั่งน้ำหนักของตัวหุ่นยนต์มีน้ำหนัก 2.12 กิโลกรัม เริ่มทำการทดลองให้หุ่นยนต์ใส่โหลดน้ำหนักเริ่มต้นตามตารางโดยให้เริ่มวิ่งจากจุดหยุดนิ่งไปจนกว่าความเร็วของหุ่นยนต์จะคงที่ แล้ววัดความเร็วที่คงที่สูงสุดของแต่ละโหลดน้ำหนัก

ตารางที่ 4.1 ความเร็วสูงสุดของหุ่นยนต์

โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	ความเร็วสูงสุด (เมตร/วินาที)
0	1.11
1	1.04
2	0.95
3	0.88
4	0.82
5	0.68

จากผลการทดลองทำให้รู้ว่าโหลดน้ำหนักแต่ละค่าสามารถรองรับความเร็วสูงสุดได้เป็นไปตามดังตารางที่ 4.1 และทำให้สามารถกำหนดค่าความเร็วของหุ่นยนต์ให้คงที่ตามที่กำหนดได้

4.2 การทดลองระยะเวลาหน่วงของหุ่นยนต์เมื่อโหลดมีการเปลี่ยนแปลง

4.2.1 การทดลองที่ 1 เมื่อโหลดมีการเปลี่ยนแปลง 1 กิโลกรัม

กำหนดให้หุ่นยนต์ใส่โหลดน้ำหนักเริ่มต้นตามตารางโดยให้เริ่มวิ่งจากจุดหยุดนิ่งไปจนกว่าความเร็วของหุ่นยนต์จะคงที่ จากนั้นใส่โหลดเพิ่มอีก 1 กิโลกรัม แล้วดูเวลาที่หุ่นยนต์หน่วงจนกว่าหุ่นยนต์จะกลับมาความเร็วคงที่ตามที่กำหนดไว้

ทดลองที่ความเร็ว 0.67 เมตร/วินาที

ตารางที่ 4.2 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 1 กิโลกรัม ที่ความเร็ว 0.67 เมตร/วินาที

โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	1	1.92	1.80	1.85	1.86	1.73	1.83
1	2	น้อยมาก	น้อยมาก	น้อยมาก	น้อยมาก	น้อยมาก	น้อยมาก
2	3	1.88	1.83	1.76	1.69	1.84	1.80
3	4	1.99	1.95	2.01	1.98	2.10	2.01
4	5	0.91	1.00	1.03	0.99	0.97	0.98

ทดลองที่ความเร็ว 0.57 เมตร/วินาที

ตารางที่ 4.3 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 1 กิโลกรัม ที่ความเร็ว 0.57 เมตร/วินาที

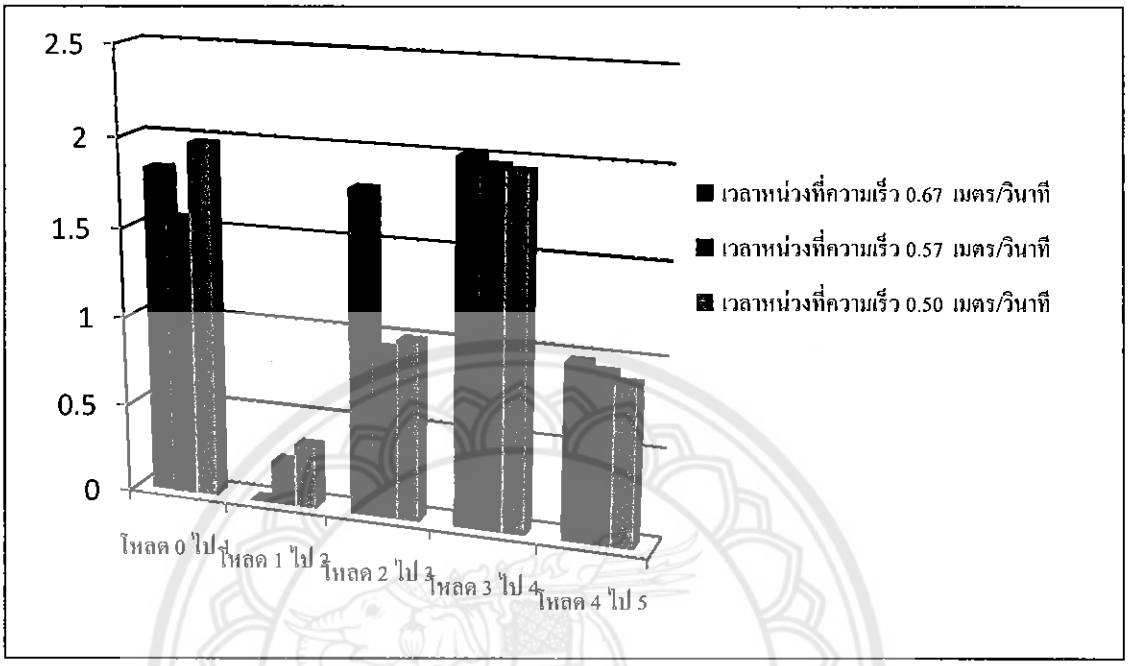
โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	1	1.45	1.60	1.49	1.46	1.78	1.56
1	2	น้อยมาก	0.75	น้อยมาก	น้อยมาก	0.45	0.24
2	3	0.91	0.96	0.90	1.01	0.90	0.94
3	4	1.99	1.98	1.93	1.97	2.00	1.97
4	5	0.96	0.88	0.97	1.03	0.87	0.94

ทดลองที่ความเร็ว 0.50 เมตร/วินาที

ตารางที่ 4.4 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 1 กิโลกรัม ที่ความเร็ว 0.50 เมตร/วินาที

โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	1	1.86	2.01	2.12	1.91	2.02	1.98
1	2	0.88	น้อยมาก	น้อยมาก	น้อยมาก	0.86	0.35
2	3	0.98	0.97	1.01	0.96	1.07	1.00
3	4	2.00	1.95	2.01	1.87	1.92	1.95
4	5	0.77	0.86	0.93	0.90	1.00	0.89

กราฟแสดงการเปรียบเทียบระยะเวลาหน่วงเมื่อโหลดมีการเปลี่ยนแปลง 1 กิโลกรัมที่ความเร็ว 0.67 เมตร/วินาที , 0.57 เมตร/วินาที และ 0.50 เมตร/วินาที



รูปที่ 4.1 กราฟการเปรียบเทียบระยะเวลาหน่วงเมื่อโหลดมีการเปลี่ยนแปลง 1 กิโลกรัม ที่ความเร็ว 0.67 เมตร/วินาที , 0.57 เมตร/วินาที และ 0.50 เมตร/วินาที

จากการทดลองระยะเวลาหน่วงของหุ่นยนต์เมื่อโหลดมีการเปลี่ยนแปลง 1 กิโลกรัม จะเห็นได้ว่าเมื่อความเร็ว 0.67 เมตร/วินาที ที่โหลดเริ่มต้น 2 กิโลกรัม จะมีระยะเวลาการหน่วงไม่ใกล้เคียงกับความเร็วอื่น แต่ที่โหลดเริ่มต้นอื่นๆจะมีระยะเวลาการหน่วงของแต่ละความเร็วใกล้เคียงกัน

4.2.2 การทดลองที่ 2 เมื่อโหลดมีการเปลี่ยนแปลง 2 กิโลกรัม

กำหนดให้หุ่นยนต์ใส่โหลดน้ำหนักเริ่มต้นตามตารางโดยให้เริ่มวิ่งจากจุดหยุดนิ่งไปจนกว่าความเร็วของหุ่นยนต์จะคงที่ จากนั้นใส่โหลดเพิ่มอีก 2 กิโลกรัม แล้วดูเวลาที่หุ่นยนต์หน่วงจนกว่าหุ่นยนต์จะกลับมาความเร็วคงที่ตามที่กำหนดไว้

ทดลองที่ความเร็ว 0.67 เมตร/วินาที

ตารางที่ 4.5 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 2 กิโลกรัม ที่ความเร็ว 0.67 เมตร/วินาที

โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	2	1.36	1.06	1.13	1.15	1.20	1.18
1	3	1.05	1.11	0.90	0.97	1.10	1.03
2	4	1.83	2.03	2.03	1.97	2.09	1.99
3	5	1.95	1.86	2.12	2.08	1.97	2.00

ทดลองที่ความเร็ว 0.57 เมตร/วินาที

ตารางที่ 4.6 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 2 กิโลกรัม ที่ความเร็ว 0.57 เมตร/วินาที

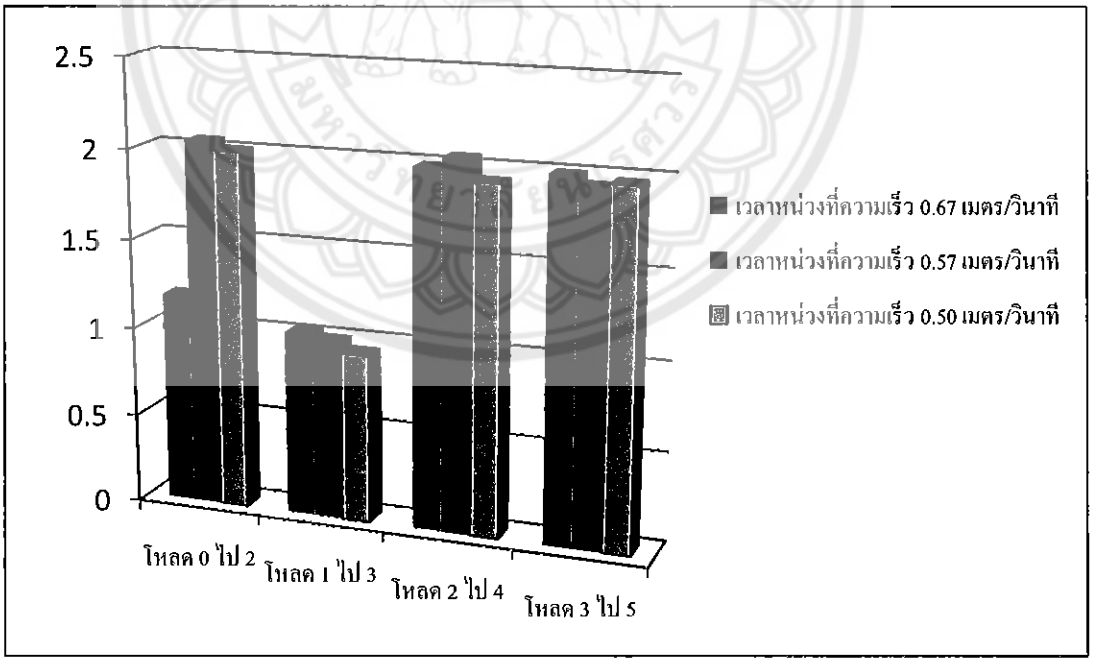
โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	2	1.96	2.02	2.11	2.05	2.13	2.05
1	3	1.03	0.93	1.01	0.95	1.08	1.00
2	4	2.01	2.11	2.05	1.98	2.10	2.05
3	5	1.90	1.93	1.87	2.00	2.05	1.95

ทดลองที่ความเร็ว 0.50 เมตร/วินาที

ตารางที่ 4.7 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 2 กิโลกรัม ที่ความเร็ว 0.50 เมตร/วินาที

โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	2	1.85	2.09	1.96	2.03	2.05	2.00
1	3	0.86	0.96	0.93	0.91	1.04	0.94
2	4	1.93	1.90	1.98	1.89	1.94	1.93
3	5	1.96	1.95	1.92	2.04	1.96	1.97

กราฟแสดงการเปรียบเทียบระยะเวลาหน่วงเมื่อโหลดมีการเปลี่ยนแปลง 2 กิโลกรัมที่ความเร็ว 0.67 เมตร/วินาที , 0.57 เมตร/วินาที และ 0.50 เมตร/วินาที



รูปที่ 4.2 กราฟการเปรียบเทียบระยะเวลาหน่วงเมื่อโหลดมีการเปลี่ยนแปลง 2 กิโลกรัม ที่ความเร็ว 0.67 เมตร/วินาที , 0.57 เมตร/วินาที และ 0.50 เมตร/วินาที

จากการทดลองระยะเวลาหน่วงของหุ่นยนต์เมื่อโหลดมีการเปลี่ยนแปลง 2 กิโลกรัม จะเห็นว่าที่โหลดเริ่มต้น 0 กิโลกรัม ระยะเวลาการหน่วงที่ความเร็ว 0.67 เมตร/วินาที จะไม่ใกล้เคียงกับความเร็วอื่น แต่ที่โหลดเริ่มต้น 1 กิโลกรัม, 2 กิโลกรัม และ 3 กิโลกรัม ระยะเวลาการหน่วงของแต่ละความเร็วจะใกล้เคียงกัน

4.2.3 การทดลองที่ 3 เมื่อโหลดมีการเปลี่ยนแปลง 3 กิโลกรัม

กำหนดให้หุ่นยนต์ใส่โหลดน้ำหนักเริ่มต้นตามตารางโดยให้เริ่มวิ่งจากจุดหยุดนิ่งไปจนกว่าความเร็วของหุ่นยนต์จะคงที่ จากนั้นใส่โหลดเพิ่มอีก 3 กิโลกรัม แล้วดูเวลาที่หุ่นยนต์หน่วงจนกว่าหุ่นยนต์จะกลับมาความเร็วคงที่ตามที่กำหนดไว้

ทดลองที่ความเร็ว 0.67 เมตร/วินาที

ตารางที่ 4.8 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 3 กิโลกรัม ที่ความเร็ว 0.67 เมตร/วินาที

โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	3	1.01	1.05	1.16	1.08	1.00	1.06
1	4	2.05	1.99	1.98	2.07	2.11	2.04
2	5	1.98	1.99	1.95	2.06	2.04	2.00

ทดลองที่ความเร็ว 0.57 เมตร/วินาที

ตารางที่ 4.9 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 3 กิโลกรัม ที่ความเร็ว 0.57 เมตร/วินาที

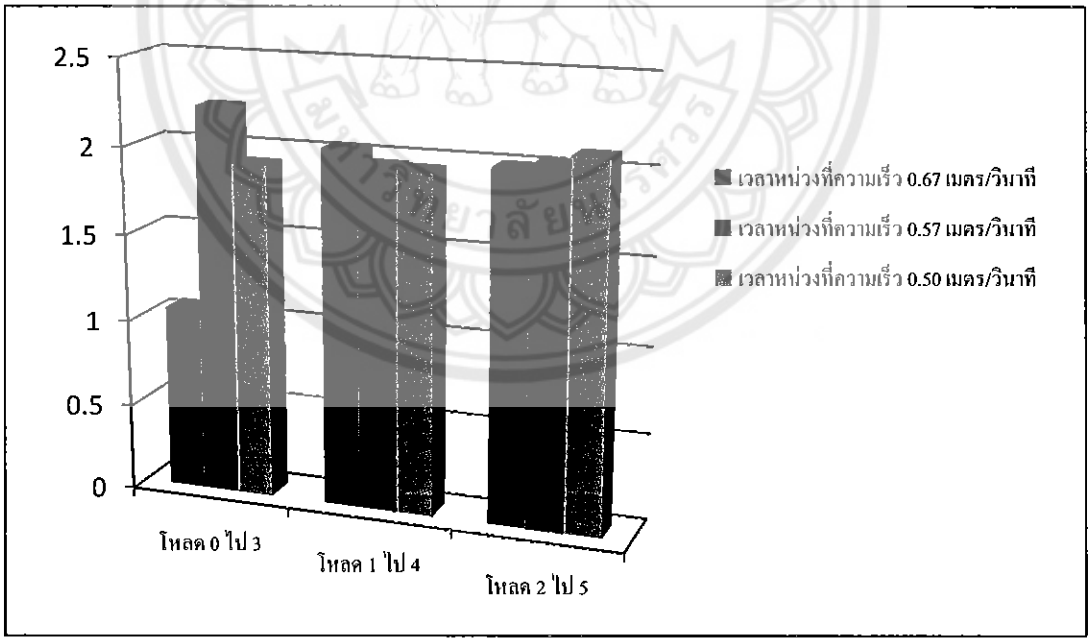
โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	3	2.06	2.45	2.20	2.30	2.13	2.23
1	4	2.13	1.93	1.93	1.85	1.97	1.96
2	5	2.00	2.05	2.01	1.97	2.12	2.03

ทดลองที่ความเร็ว 0.50 เมตร/วินาที

ตารางที่ 4.10 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 3 กิโลกรัม ที่ความเร็ว 0.50 เมตร/วินาที

โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	3	1.78	1.96	1.95	2.09	1.84	1.92
1	4	1.93	1.87	1.90	2.07	1.99	1.95
2	5	2.10	2.15	2.09	2.12	1.98	2.09

กราฟแสดงการเปรียบเทียบระยะเวลาหน่วงเมื่อโหลดมีการเปลี่ยนแปลง 3 กิโลกรัมที่ความเร็ว 0.67 เมตร/วินาที , 0.57 เมตร/วินาที และ 0.50 เมตร/วินาที



รูปที่ 4.3 กราฟการเปรียบเทียบระยะเวลาหน่วงเมื่อโหลดมีการเปลี่ยนแปลง 3 กิโลกรัม ที่ความเร็ว 0.67 เมตร/วินาที , 0.57 เมตร/วินาที และ 0.50 เมตร/วินาที

จากการทดลองระยะเวลาหน่วงของหุ่นยนต์เมื่อโหลดมีการเปลี่ยนแปลง 3 กิโลกรัม จะเห็นได้ว่าที่โหลดเริ่มต้น 0 กิโลกรัม ระยะเวลาการหน่วงของแต่ละความเร็วจะไม่ใกล้เคียงกัน แต่ที่โหลดเริ่มต้น 1 กิโลกรัมและ 2 กิโลกรัม ระยะเวลาการหน่วงของแต่ละความเร็วจะใกล้เคียงกัน

4.2.4 การทดลองที่ 4 เมื่อโหลดมีการเปลี่ยนแปลง 4 กิโลกรัม

กำหนดให้หุ่นยนต์ใส่โหลดน้ำหนักเริ่มต้นตามตารางโดยให้เริ่มวิ่งจากจุดหยุดนิ่งไปจนกว่าความเร็วของหุ่นยนต์จะคงที่ จากนั้นใส่โหลดเพิ่มอีก 4 กิโลกรัม แล้วดูเวลาที่หุ่นยนต์หน่วงจนกว่าหุ่นยนต์จะกลับมาความเร็วคงที่ตามที่กำหนดไว้

ทดลองที่ความเร็ว 0.67 เมตร/วินาที

ตารางที่ 4.11 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 4 กิโลกรัม ที่ความเร็ว 0.67 เมตร/วินาที

โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	4	1.98	1.87	1.91	1.98	1.96	1.94
1	5	1.99	2.04	2.13	2.01	2.14	2.06

ทดลองที่ความเร็ว 0.57 เมตร/วินาที

ตารางที่ 4.12 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 4 กิโลกรัม ที่ความเร็ว 0.57 เมตร/วินาที

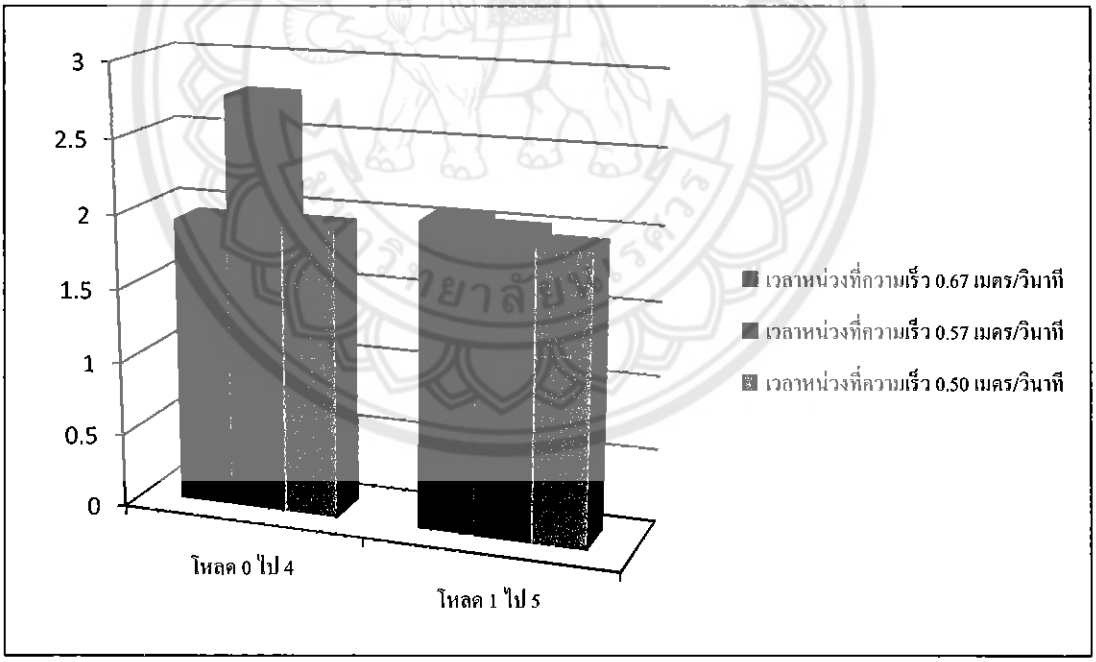
โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	4	2.80	2.81	2.75	2.91	2.65	2.78
1	5	1.99	2.05	2.10	1.90	2.04	2.02

ทดลองที่ความเร็ว 0.50 เมตร/วินาที

ตารางที่ 4.13 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 4 กิโลกรัม ที่ความเร็ว 0.50 เมตร/วินาที

โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	4	1.87	1.90	2.03	1.96	1.97	1.95
1	5	1.97	1.90	1.86	2.03	1.97	1.95

กราฟแสดงการเปรียบเทียบระยะเวลาหน่วงเมื่อโหลดมีการเปลี่ยนแปลง 4 กิโลกรัมที่ความเร็ว 0.67 เมตร/วินาที , 0.57 เมตร/วินาที และ 0.50 เมตร/วินาที



รูปที่ 4.4 กราฟการเปรียบเทียบระยะเวลาหน่วงเมื่อโหลดมีการเปลี่ยนแปลง 4กิโลกรัม ที่ความเร็ว 0.67 เมตร/วินาที , 0.57 เมตร/วินาที และ 0.50 เมตร/วินาที

จากการทดลองระยะเวลาหน่วงของหุ่นยนต์เมื่อโหลดมีการเปลี่ยนแปลง 4 กิโลกรัม จะเห็นได้ว่าที่ความเร็ว 0.57 เมตร/วินาที โหลดเริ่มต้น 0 กิโลกรัมจะมีระยะเวลาการหน่วงที่ไม่ใกล้เคียงกับความเร็วอื่น แต่ที่ความเร็ว 0.67 เมตร/วินาที และ 0.50 เมตร/วินาที จะมีระยะเวลาการหน่วงใกล้เคียงกัน

4.2.5 การทดลองที่ 5 เมื่อโหลดมีการเปลี่ยนแปลง 5 กิโลกรัม

กำหนดให้หุ่นยนต์ใส่โหลดน้ำหนักเริ่มต้นตามตารางโดยให้เริ่มวิ่งจากจุดหยุดนิ่งไปจนกว่าความเร็วของหุ่นยนต์จะคงที่ จากนั้นใส่โหลดเพิ่มอีก 5 กิโลกรัม แล้วดูเวลาที่หุ่นยนต์หน่วงจนกว่าหุ่นยนต์จะกลับมาความเร็วคงที่ตามที่กำหนดไว้

ทดลองที่ความเร็ว 0.67 เมตร/วินาที

ตารางที่ 4.14 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 5 กิโลกรัม ที่ความเร็ว 0.67 เมตร/วินาที

โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	5	2.03	1.90	1.87	1.96	1.99	1.95

ทดลองที่ความเร็ว 0.57 เมตร/วินาที

ตารางที่ 4.15 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 5 กิโลกรัม ที่ความเร็ว 0.57 เมตร/วินาที

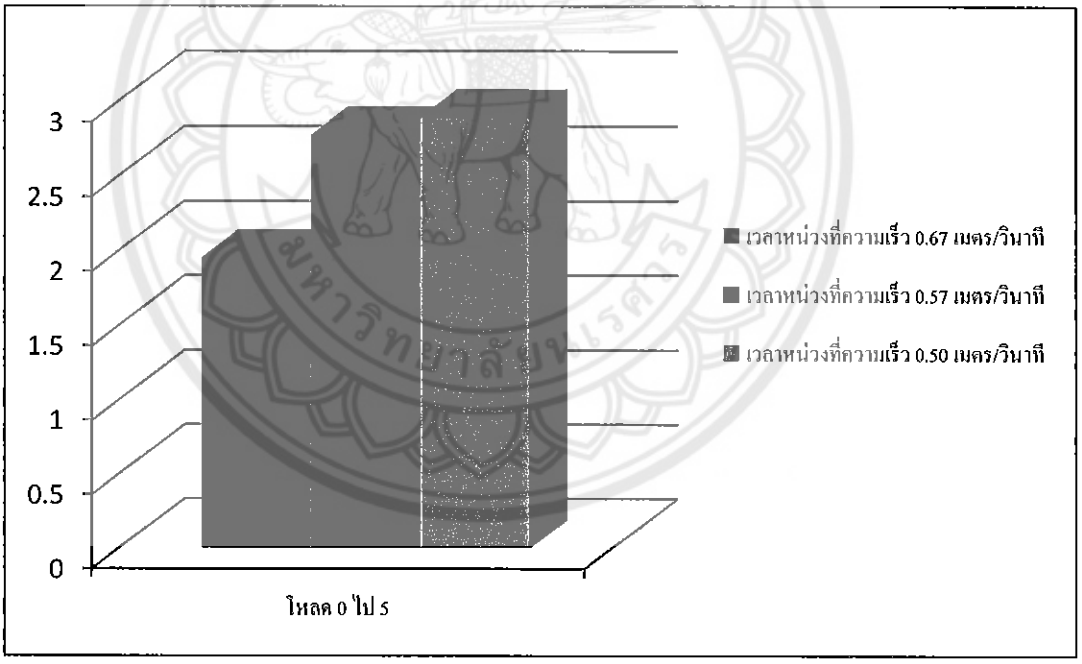
โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	5	2.63	2.86	2.75	2.76	2.83	2.77

ทดลองที่ความเร็ว 0.50 เมตร/วินาที

ตารางที่ 4.16 ระยะเวลาการหน่วงเมื่อโหลดมีการเปลี่ยนแปลงน้ำหนัก 5 กิโลกรัม ที่ความเร็ว 0.50 เมตร/วินาที

โหลดน้ำหนักเริ่มต้น (กิโลกรัม)	โหลดน้ำหนักรวม (กิโลกรัม)	ระยะเวลาที่หน่วง (วินาที)					
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5	เฉลี่ย
0	5	2.86	2.81	3.05	2.90	2.85	2.89

กราฟแสดงการเปรียบเทียบระยะเวลาหน่วงเมื่อโหลดมีการเปลี่ยนแปลง 5 กิโลกรัมที่ความเร็ว 0.67 เมตร/วินาที , 0.57 เมตร/วินาที และ 0.50 เมตร/วินาที

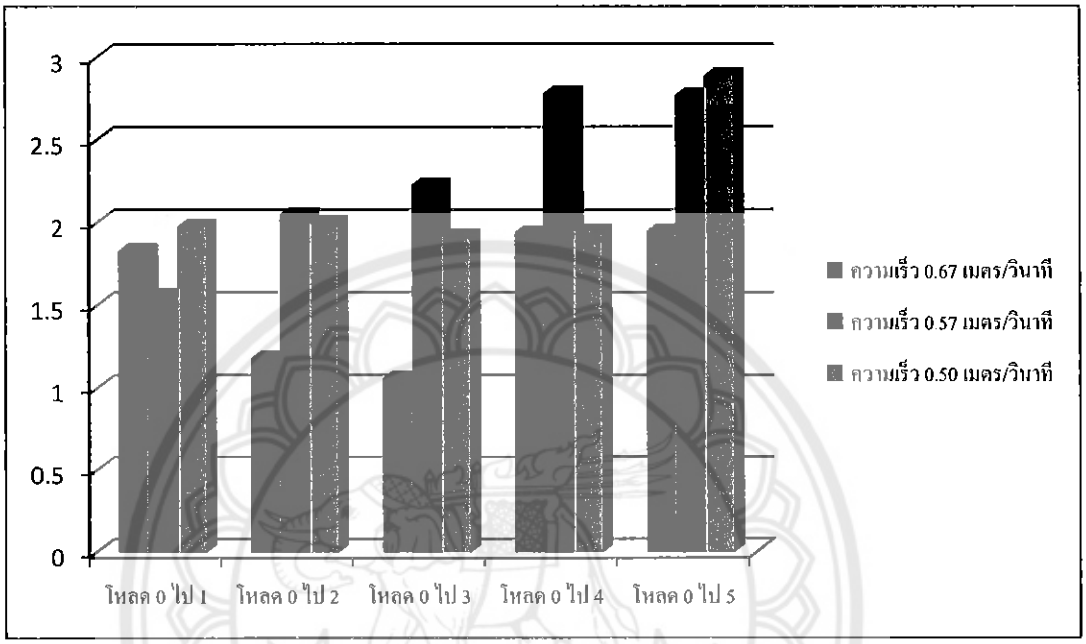


รูปที่ 4.5 กราฟการเปรียบเทียบระยะเวลาหน่วงเมื่อโหลดมีการเปลี่ยนแปลง 5 กิโลกรัม ที่ความเร็ว 0.67 เมตร/วินาที , 0.57 เมตร/วินาที และ 0.50 เมตร/วินาที

จากการทดลองระยะเวลาหน่วงของหุ่นยนต์เมื่อโหลดมีการเปลี่ยนแปลง 5 กิโลกรัม จะเห็นว่าที่ความเร็ว 0.67 เมตร/วินาที จะมีระยะเวลาการหน่วงน้อยที่สุด แต่ที่ความเร็ว 0.57 เมตร/วินาที และที่ความเร็ว 0.50 เมตร/วินาที จะมีระยะเวลาการหน่วงที่ใกล้เคียงกัน

4.3 กราฟแสดงการเปรียบเทียบค่าความเร็วที่มีผลต่อระยะเวลาหน่วงของหุ่นยนต์

จากข้อมูลผลการทดลองยกตัวอย่างที่โหลดเริ่มต้น 0 กิโลกรัม ให้โหลดมีการเปลี่ยนแปลงค่าตั้งแต่ 1 กิโลกรัมถึง 5 กิโลกรัม แสดงผลได้ดังรูปกราฟที่ 4.6 ดังนี้



รูปที่ 4.6 กราฟการเปรียบเทียบค่าความเร็วที่มีผลต่อระยะเวลาหน่วงของหุ่นยนต์

จากกราฟจะเห็นได้ว่าระยะเวลาการหน่วงของหุ่นยนต์มีค่าแตกต่างกัน เนื่องจากความเร็วมีผลทำให้ระยะเวลาการหน่วงเปลี่ยนแปลงตาม คือเมื่อกำหนดให้หุ่นยนต์มีความเร็วที่ 0.67 เมตร/วินาที จะทำให้ระยะเวลาการหน่วงมีค่าไม่ใกล้เคียงกัน แต่เมื่อกำหนดให้หุ่นยนต์มีความเร็วที่ 0.50 เมตร/วินาที จะทำให้ระยะเวลาการหน่วงมีค่าใกล้เคียงกัน

บทที่ 5

สรุปผลการดำเนินงานและข้อเสนอแนะ

จากการทดลองทั้งหมดที่ได้ดำเนินมาทำให้สามารถสรุปผลการดำเนินงานและรู้ถึงปัญหาที่เกิดขึ้นในขณะทำการทดลองเพื่อนำมาปรับปรุงแก้ไขและพัฒนาโครงงานให้มีประสิทธิภาพดังนี้

5.1 สรุปผลการดำเนินงาน

จากผลการทดลองการควบคุมความเร็วของหุ่นยนต์โดยใช้การควบคุมแบบฟัซซี ทำให้ได้ผลการทดลองเป็นไปตามวัตถุประสงค์ คือหุ่นยนต์สามารถควบคุมความเร็วให้คงที่ตามที่กำหนดดังนี้ 0.67 เมตร/วินาที, 0.57 เมตร/วินาที และ 0.50 เมตร/วินาที เมื่อโหลดมีค่าการเปลี่ยนแปลงตั้งแต่ 1 กิโลกรัม จนถึง 5 กิโลกรัม โดยการเคลื่อนที่ของหุ่นยนต์จะเกิดระยะเวลาการหน่วงจากการเปลี่ยนแปลงของโหลดเพื่อให้หุ่นยนต์กลับมาความเร็วคงที่ เมื่อทำการทดลองเปรียบเทียบระยะเวลาการหน่วงของหุ่นยนต์ที่มีค่าการเปลี่ยนแปลงโหลดเท่ากันที่แต่ละความเร็วจะได้ระยะเวลาการหน่วงที่ใกล้เคียงกัน แต่เมื่อเปรียบเทียบระยะเวลาการหน่วงของหุ่นยนต์ที่มีการเปลี่ยนแปลงโหลดต่างกันที่แต่ละความเร็ว จะได้ระยะเวลาการหน่วงที่มีค่าแตกต่างกัน เมื่อเปรียบเทียบความเร็วคงที่สูงและความเร็วคงที่ต่ำที่โหลดต่างกันจะพบว่าที่ความเร็วคงที่สูงระยะเวลาการหน่วงจะมีค่าไม่ใกล้เคียงกัน แต่ที่ความเร็วคงที่ต่ำจะมีค่าระเวลาการหน่วงใกล้เคียงกัน ดังนั้นขนาดของความเร็วจึงมีผลต่อระยะเวลาการหน่วง เนื่องจากความเร็วคงที่สูงจะมีแรงเสียดทานที่กระทำต่อหุ่นยนต์น้อยกว่าความเร็วคงที่ต่ำ ส่งผลให้ระยะเวลาการหน่วงแปรผกผันกับค่าความเร็ว

5.2 ข้อเสนอแนะและแนวทางการแก้ปัญหา

ในการเก็บค่าผลการทดลองอาจมีระยะเวลาหน่วงบางค่าที่ต่างไปจากค่าอื่นๆ เช่น ที่ความเร็ว 0.67 เมตร/วินาที โดยเปลี่ยนโหลดน้ำหนักจาก 2 กิโลกรัม เป็นโหลด 3 กิโลกรัม จะเห็นได้ว่าระยะเวลาการหน่วงสูงกว่าที่ความเร็ว 0.57 เมตร/วินาที และ 0.50 เมตร/วินาที ซึ่งอาจเกิดจากปัญหาและปัจจัยต่างๆ ดังต่อไปนี้

5.2.1 ปัญหาและแนวทางการแก้ปัญหา

(1) โครงสร้างของหุ่นยนต์เมื่อเวลามีการใส่โหลดพบว่ามีปัญหาต่อการวางโหลด ทำให้ระยะเวลาการหน่วงคลาดเคลื่อนได้

แนวทางการแก้ปัญหา เวลารถเคลื่อนตัวต้องทำการใส่โหลดอย่างช้าๆและแม่นยำ

(2) แบตเตอรี่เมื่อใช้เก็บค่าผลการทดลองไปได้สักพัก แรงดันที่ได้จากแบตเตอรี่จะมีค่าลดลง จึงส่งผลให้ระยะเวลาการหน่วงคลาดเคลื่อนได้

แนวทางการแก้ปัญหา ต้องหยุดเก็บผลการทดลอง แล้วนำแบตเตอรี่มาชาร์จให้แบตเตอรี่เต็มเพื่อให้ได้ประสิทธิภาพในการเก็บผลการทดลอง

(3) สถานที่ในการเก็บผลการทดลองพบว่าพื้นผิวที่ขรุขระ จะทำให้หุ่นยนต์เคลื่อนที่ได้ไม่เต็มประสิทธิภาพ เกิดแรงเสียดทานมากขึ้น แล้วทำให้ผลการทดลองคลาดเคลื่อน

แนวทางการแก้ปัญหา ทำการทดลองบนพื้นที่แบบเรียบซึ่งช่วยลดรงเสียดทานที่ทำให้ค่าผลการทดลองคลาดเคลื่อนได้

5.3 การนำไปพัฒนาและต่อยอด

สามารถนำไปสร้างหุ่นยนต์ที่มีโครงสร้างที่ใหญ่ขึ้นและแข็งแรงเพื่อรองรับโหลดน้ำหนักที่มากขึ้นและหุ่นยนต์สามรถเคลื่อนที่ได้หลายรูปแบบเช่น การเคลื่อนที่บนพื้นที่เอียงหรือการเคลื่อนที่ในแนวดิ่ง เป็นต้น

เอกสารอ้างอิง

[1] สมนึก บุญพาไสว. "การวัดและเครื่องมือวัด". กรุงเทพฯ: บริษัท สำนักพิมพ์ท้อป จำกัด ,2550

[2] ผศ.ศุภชัย สุรินทร์วงศ์. "มอเตอร์ไฟฟ้ากระแสตรง". พิมพ์ครั้งที่ 5. กรุงเทพฯ: สมาคมส่งเสริมเทคโนโลยี(ไทย-ญี่ปุ่น), 2541

[3] www.robotshop.com/.../ArduinoMega2560Datasheet.pdf ,สืบค้นเมื่อ พฤศจิกายน 2558

[4] http://www.zerokol.com/2012/09/arduinofuzzy-fuzzy-library-for-arduino.html ,สืบค้นเมื่อ มกราคม 2559

[5] S.K.Harisha,Ramkanth Kumar P,M. Krishna, and S.Sharma (2008),**Fuzzy Logic Reasonning to Control Mobile Robot on Pre-defined Strip Path.**World Academy of Science Engineering and technology.

[6] กันตภณ พริ้วไธสง.การประยุกต์ใช้ฟัซซีลอจิกกับการเคลื่อนที่หลบหลีกสิ่งกีดขวางของหุ่นยนต์,วารสารวิศกรรมศาสตร์ ราชมงคลธัญบุรี.

ภาคผนวก ก

รายละเอียดข้อมูลของ AVR ATMEGA2560

Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V

8-bit Atmel Microcontroller with 16/32/64KB In-System Programmable Flash

SUMMARY

Features

- **High Performance, Low Power Atmel® AVR® 8-Bit Microcontroller**
- **Advanced RISC Architecture**
  - **135 Powerful Instructions – Most Single Clock Cycle Execution**
  - **32 x 8 General Purpose Working Registers**
  - **Fully Static Operation**
  - **Up to 16 MIPS Throughput at 16MHz**
  - **On-Chip 2-cycle Multiplier**
- High Endurance Non-volatile Memory Segments
  - 64K/128K/256KBytes of In-System Self-Programmable Flash
  - 4Kbytes EEPROM
  - 8Kbytes Internal SRAM
  - Write/Erase Cycles:10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/ 100 years at 25°C
  - Optional Boot Code Section with Independent Lock Bits
    - In-System Programming by On-chip Boot Program
    - True Read-While-Write Operation
  - Programming Lock for Software Security
    - Endurance: Up to 64Kbytes Optional External Memory Space
- Atmel® QTouch® library support
  - Capacitive touch buttons, sliders and wheels
  - QTouch and QMatrix acquisition
  - Up to 64 sense channels
- JTAG (IEEE® std. 1149.1 compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - Four 16-bit Timer/Counter with Separate Prescaler, Compare- and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four 8-bit PWM Channels
  - Six/Twelve PWM Channels with Programmable Resolution from 2 to 16 Bits (ATmega1281/2561, ATmega640/1280/2560)
  - Output Compare Modulator
  - 8/16-channel, 10-bit ADC (ATmega1281/2561, ATmega640/1280/2560)
  - Two/Four Programmable Serial USART (ATmega1281/2561, ATmega640/1280/2560)
  - Master/Slave SPI Serial Interface
  - Byte Oriented 2-wire Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
  - 54/86 Programmable I/O Lines (ATmega1281/2561, ATmega640/1280/2560)
  - 64-pad QFN/MLF, 64-lead TQFP (ATmega1281/2561)
  - 100-lead TQFP, 100-ball CBGA (ATmega640/1280/2560)
  - RoHS/Fully Green
- Temperature Range:
  - -40°C to 85°C Industrial
- Ultra-Low Power Consumption
  - Active Mode: 1MHz, 1.8V: 500µA
  - Power-down Mode: 0.1µA at 1.8V
- Speed Grade:
  - ATmega640V/ATmega1280V/ATmega1281V:
    - 0 - 4MHz @ 1.8V - 5.5V, 0 - 8MHz @ 2.7V - 5.5V
  - ATmega2560V/ATmega2561V:
    - 0 - 2MHz @ 1.8V - 5.5V, 0 - 8MHz @ 2.7V - 5.5V
  - ATmega640/ATmega1280/ATmega1281:
    - **0 - 8MHz @ 2.7V - 5.5V, 0 - 16MHz @ 4.5V - 5.5V**
  - ATmega2560/ATmega2561:
    - **0 - 16MHz @ 4.5V - 5.5V**

1. Pin Configurations

Figure 1-1. TQFP-pinout ATmega640/1280/2560

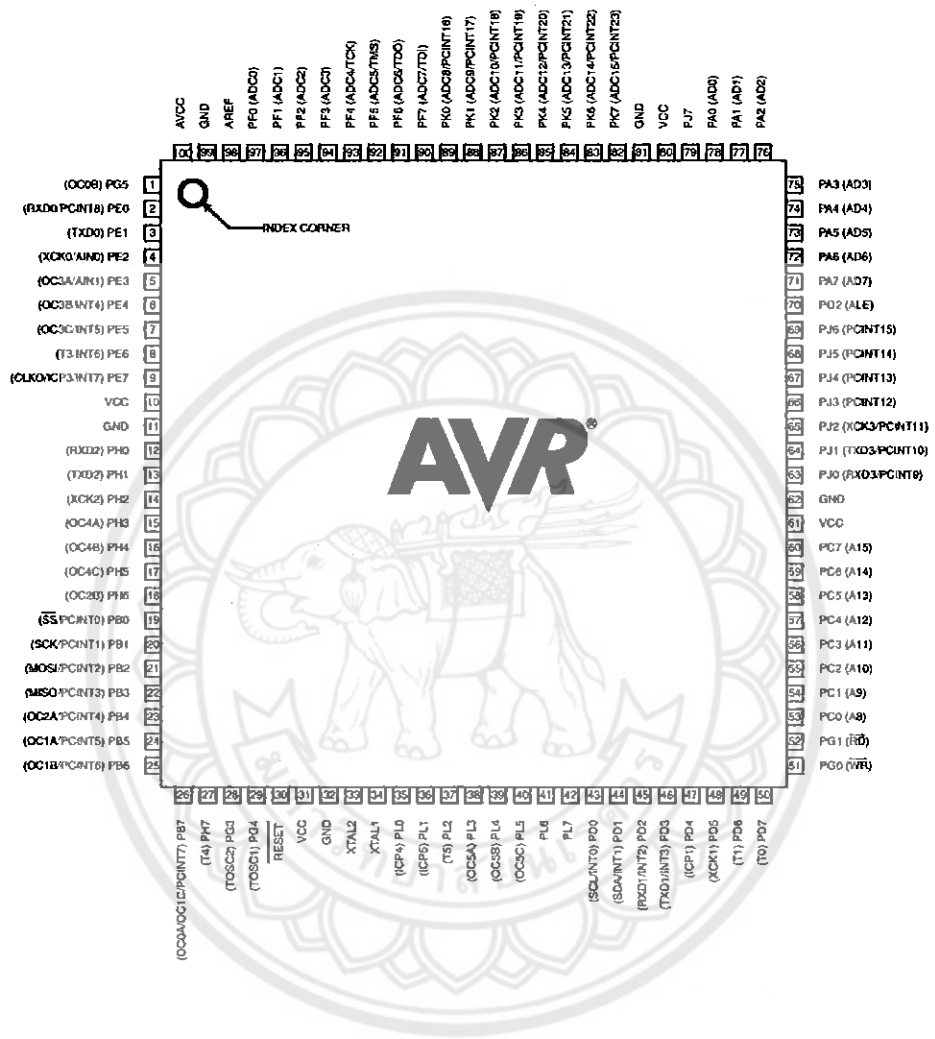


Figure 1-2. CBGA-pinout ATmega640/1280/2560

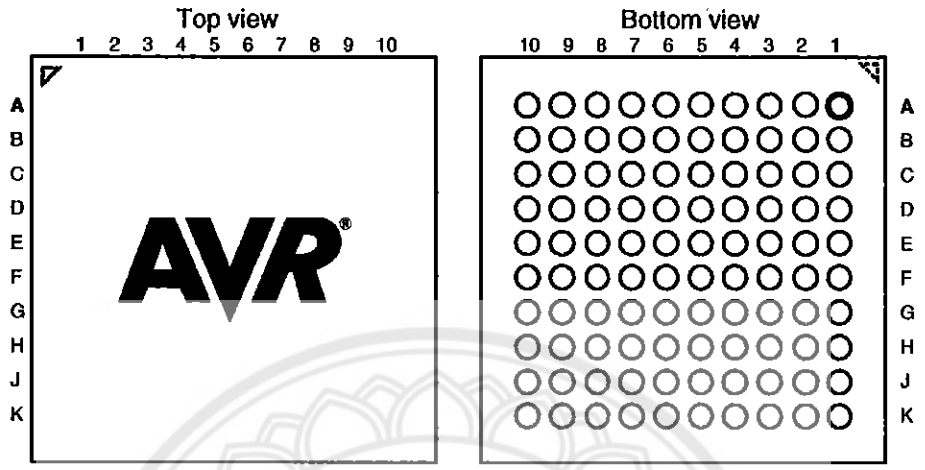
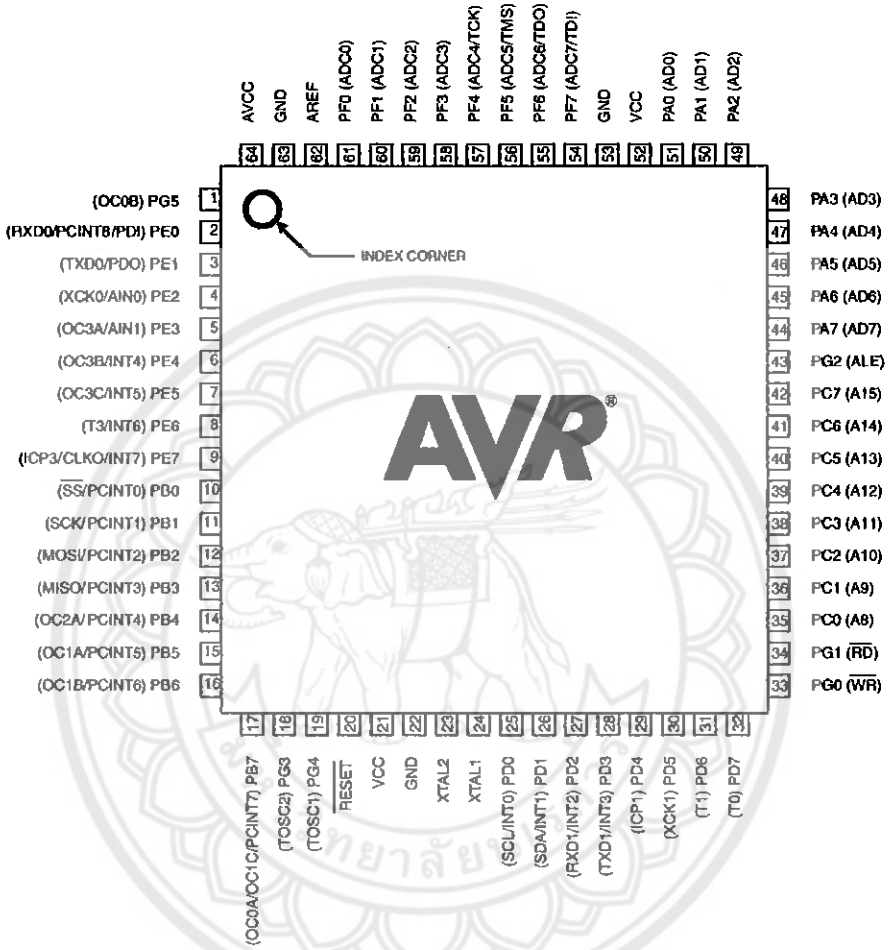


Table 1-1. CBGA-pinout ATmega640/1280/2560

	1	2	3	4	5	6	7	8	9	10
A	GND	AREF	PF0	PF2	PF5	PK0	PK3	PK6	GND	VCC
B	AVCC	PG5	PF1	PF3	PF6	PK1	PK4	PK7	PA0	PA2
C	PE2	PE0	PE1	PF4	PF7	PK2	PK5	PJ7	PA1	PA3
D	PE3	PE4	PE5	PE6	PH2	PA4	PA5	PA6	PA7	PG2
E	PE7	PH0	PH1	PH3	PH5	PJ6	PJ5	PJ4	PJ3	PJ2
F	VCC	PH4	PH6	PB0	PL4	PD1	PJ1	PJ0	PC7	GND
G	GND	PB1	PB2	PB5	PL2	PD0	PD5	PC5	PC6	VCC
H	PB3	PB4	RESET	PL1	PL3	PL7	PD4	PC4	PC3	PC2
J	PH7	PG3	PB6	PL0	XTAL2	PL6	PD3	PC1	PC0	PG1
K	PB7	PG4	VCC	GND	XTAL1	PL5	PD2	PD6	PD7	PG0

Note: The functions for each pin is the same as for the 100 pin packages shown in Figure 1-1 on page 2.

Figure 1-3. Pinout ATmega1281/2561



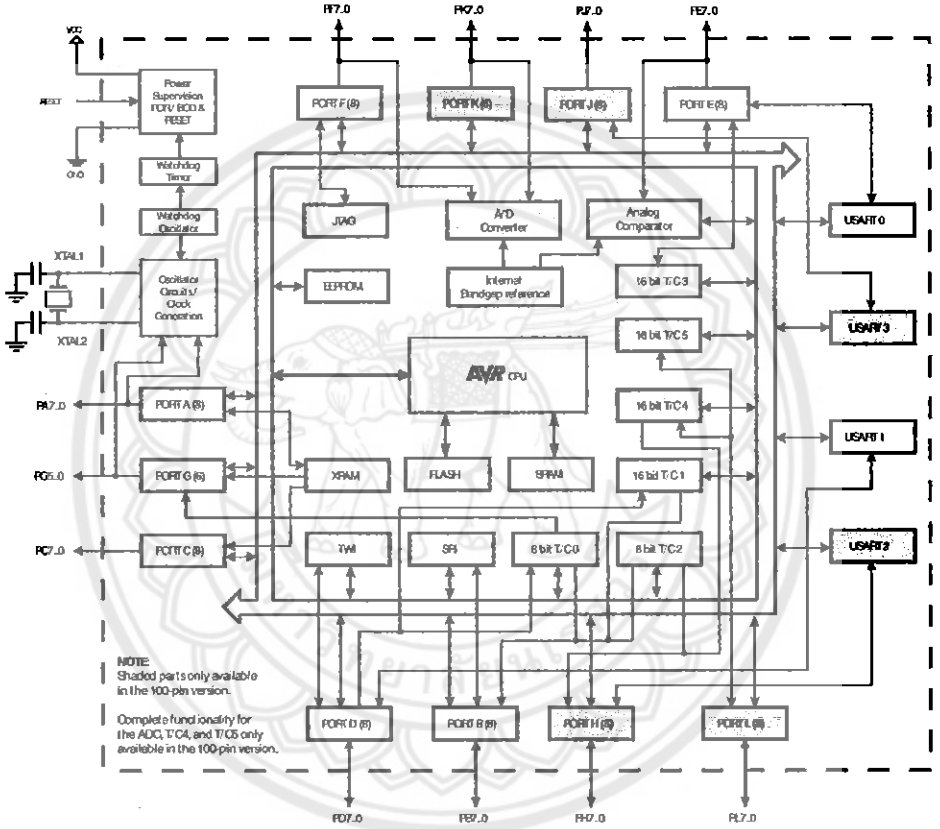
Note: The large center pad underneath the QFN/MLF package is made of metal and internally connected to GND. It should be soldered or glued to the board to ensure good mechanical stability. If the center pad is left unconnected, the package might loosen from the board.

2. Overview

The ATmega640/1280/1281/2560/2561 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega640/1280/1281/2560/2561 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

2.1 Block Diagram

Figure 2-1. Block Diagram



The Atmel® AVR® core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega640/1280/1281/2560/2561 provides the following features: 64K/128K/256K bytes of In-System Programmable Flash with Read-While-Write capabilities, 4Kbytes EEPROM, 8Kbytes SRAM, 54/86 general purpose I/O lines, 32 general purpose working registers, Real Time Counter (RTC), six flexible Timer/Counters with compare modes and PWM, four USARTs, a byte oriented 2-wire Serial Interface, a 16-channel, 10-bit ADC with optional differential input stage with programmable gain, programmable Watchdog Timer with Internal Oscillator, an SPI serial port, IEEE® std. 1149.1 compliant JTAG test interface, also used for accessing the On-chip Debug system and programming and six software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and Interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next Interrupt or Hardware Reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except Asynchronous Timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the Crystal/Resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

Atmel offers the QTouch® library for embedding capacitive touch buttons, sliders and wheels functionality into AVR microcontrollers. The patented charge-transfer signal acquisition offersrobust sensing and includes fully debounced reporting of touch keys and includes Adjacent Key Suppression® (AKS®) technology for unambiguous detection of key events. The easy-to-use QTouch Suite toolchain allows you to explore, develop and debug your own touch applications.

The device is manufactured using the Atmel high-density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega640/1280/1281/2560/2561 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega640/1280/1281/2560/2561 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

2.2 Comparison Between ATmega1281/2561 and ATmega640/1280/2560

Each device in the ATmega640/1280/1281/2560/2561 family differs only in memory size and number of pins. Table 2-1 summarizes the different configurations for the six devices.

Table 2-1. Configuration Summary

Device	Flash	EEPROM	RAM	General Purpose I/O pins	16 bits resolution PWM channels	Serial USARTs	ADC Channels
ATmega640	64KB	4KB	8KB	86	12	4	16
ATmega1280	128KB	4KB	8KB	86	12	4	16
ATmega1281	128KB	4KB	8KB	54	6	2	8
ATmega2560	256KB	4KB	8KB	86	12	4	16
ATmega2561	256KB	4KB	8KB	54	6	2	8

2.3 Pin Descriptions

2.3.1 VCC

Digital supply voltage.

2.3.2 GND

Ground.

2.3.3 Port A (PA7..PA0)

Port A is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port A pins that are externally pulled low will source current if the pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port A also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 75.

2.3.4 Port B (PB7..PB0)

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B has better driving capabilities than the other ports.

Port B also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 76.

2.3.5 Port C (PC7..PC0)

Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port C also serves the functions of special features of the ATmega640/1280/1281/2560/2561 as listed on page 79.

2.3.6 Port D (PD7..PD0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port D also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 80.

2.3.7 Port E (PE7..PE0)

Port E is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port E output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port E pins that are externally pulled low will source current if the pull-up resistors are activated. The Port E pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port E also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 82.

2.3.8 Port F (PF7..PF0)

Port F serves as analog inputs to the A/D Converter.

Port F also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port F output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port F pins that are externally pulled low will source current if the pull-up resistors are activated. The Port F pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PF7(TDI), PF5(TMS), and PF4(TCK) will be activated even if a reset occurs.

Port F also serves the functions of the JTAG interface.

2.3.9 Port G (PG5..PG0)

Port G is a 6-bit I/O port with internal pull-up resistors (selected for each bit). The Port G output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port G pins that are externally pulled low will source current if the pull-up resistors are activated. The Port G pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port G also serves the functions of various special features of the ATmega640/1280/1281/2560/2561 as listed on page 86.

2.3.10 Port H (PH7..PH0)

Port H is a 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port H output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port H pins that are externally pulled low will source current if the pull-up resistors are activated. The Port H pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port H also serves the functions of various special features of the ATmega640/1280/2560 as listed on page 88.

2.3.11 Port J (PJ7..PJ0)

Port J is a 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port J output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port J pins that are externally pulled low will source current if the pull-up resistors are activated. The Port J pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port J also serves the functions of various special features of the ATmega640/1280/2560 as listed on page 90.

2.3.12 Port K (PK7..PK0)

Port K serves as analog inputs to the A/D Converter.

Port K is a 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port K output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port K pins that are externally pulled low will source current if the pull-up resistors are activated. The Port K pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port K also serves the functions of various special features of the ATmega640/1280/2560 as listed on page 92.

2.3.13 Port L (PL7..PL0)

Port L is a 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port L output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port L pins that are externally pulled low will source current if the pull-up resistors are activated. The Port L pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port L also serves the functions of various special features of the ATmega640/1280/2560 as listed on page 94.

2.3.14 $\overline{\text{RESET}}$

Reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in "System and Reset Characteristics" on page 360. Shorter pulses are not guaranteed to generate a reset.

2.3.15 XTAL1

Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

2.3.16 XTAL2

Output from the inverting Oscillator amplifier.

2.3.17 AVCC

AVCC is the supply voltage pin for Port F and the A/D Converter. It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter.

2.3.18 AREF

This is the analog reference pin for the A/D Converter.

ภาคผนวก ข

รายละเอียดของ L298N

L298

DUAL FULL-BRIDGE DRIVER

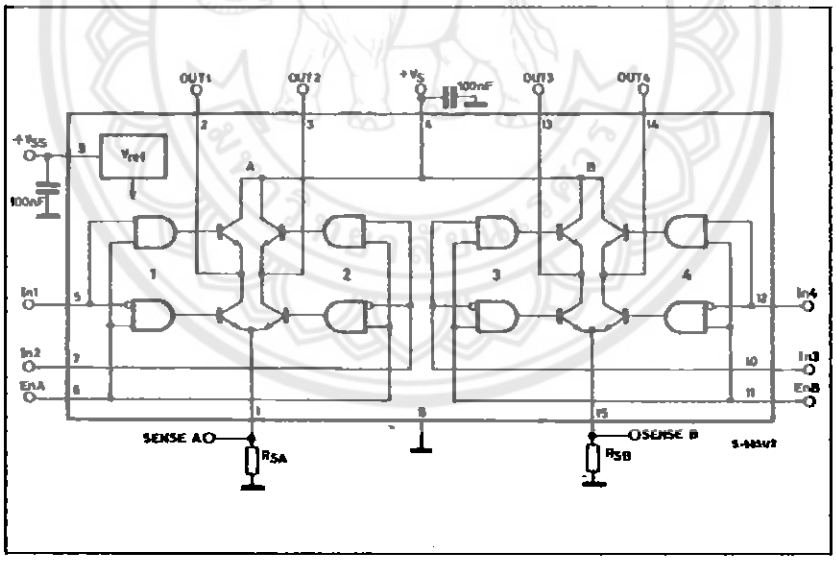
- OPERATING SUPPLY VOLTAGE UP TO 46 V
- TOTAL DC CURRENT UP TO 4 A
- LOW SATURATION VOLTAGE
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)

ORDERING NUMBERS : L298N (Multiwatt Vert.)
L298HN (Multiwatt Horiz.)
L298P (PowerSO20)

DESCRIPTION

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

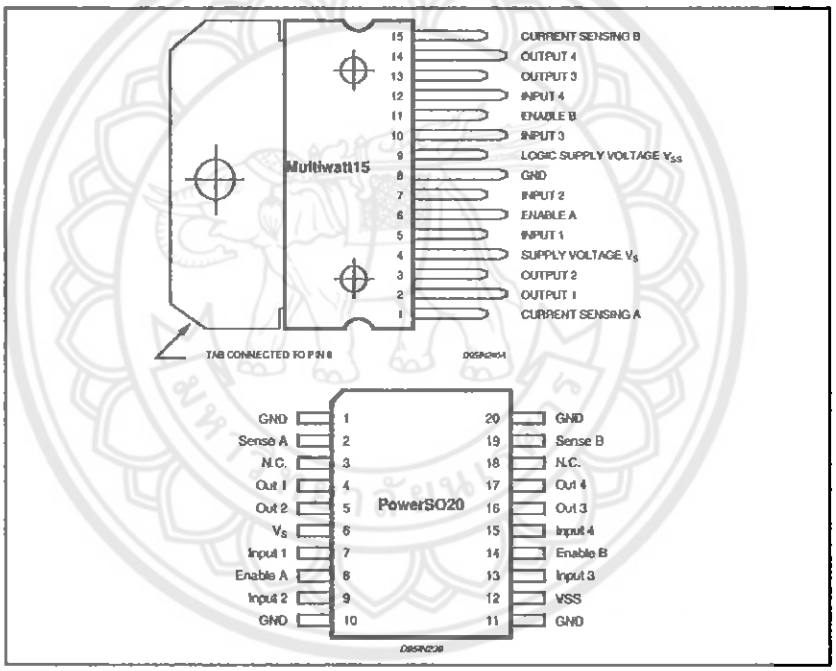
BLOCK DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Power Supply	50	V
V_{SS}	Logic Supply Voltage	7	V
V_i,V_{en}	Input and Enable Voltage	-0.3 to 7	V
I_O	Peak Output Current (each Channel) – Non Repetitive (t = 100µs) –Repetitive (80% on –20% off; t_{on} = 10ms) –DC Operation	 3 2.5 2	 A A A
V_{sens}	Sensing Voltage	–1 to 2.3	V
P_{tot}	Total Power Dissipation (T_{case} = 75°C)	25	W
T_{op}	Junction Operating Temperature	–25 to 130	°C
T_{stg}, T_j	Storage and Junction Temperature	–40 to 150	°C

PIN CONNECTIONS (top view)



THERMAL DATA

Symbol	Parameter		PowerSO20	Multiwatt15	Unit
$R_{th\ j\text{-}case}$	Thermal Resistance Junction-case	Max.	–	3	°C/W
$R_{th\ j\text{-}amb}$	Thermal Resistance Junction-ambient	Max.	13 (*)	35	°C/W

(*) Mounted on aluminum substrate

PIN FUNCTIONS (refer to the block diagram)

MW.15	PowerSO	Name	Function
1;15	2;19	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
2;3	4;5	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
4	6	V_S	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
5;7	7;9	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
6;11	8;14	Enable A; Enable B	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
8	1,10,11,20	GND	Ground.
9	12	VSS	Supply Voltage for the Logic Blocks. A100nF capacitor must be connected between this pin and ground.
10; 12	13;15	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
13; 14	16;17	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.
–	3;18	N.C.	Not Connected

ELECTRICAL CHARACTERISTICS (V_S = 42V; V_{SS} = 5V, T_j = 25°C; unless otherwise specified)

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
V_S	Supply Voltage (pin 4)	Operative Condition		V_{IH} +2.5		46	V
V_{SS}	Logic Supply Voltage (pin 9)			4.5	5	7	V
I_S	Quiescent Supply Current (pin 4)	V_{en} = H; I_L = 0	V_i = L V_i = H		13 50	22 70	mA mA
		V_{en} = L	V_i = X			4	mA
I_{SS}	Quiescent Current from V_{SS} (pin 9)	V_{en} = H; I_L = 0	V_i = L V_i = H		24 7	36 12	mA mA
		V_{en} = L	V_i = X			6	mA
V_{IL}	Input Low Voltage (pins 5, 7, 10, 12)			-0.3		1.5	V
V_{IH}	Input High Voltage (pins 5, 7, 10, 12)			2.3		VSS	V
I_{IL}	Low Voltage Input Current (pins 5, 7, 10, 12)	V_i = L				-10	µA
I_{IH}	High Voltage Input Current (pins 5, 7, 10, 12)	V_i = H ≤ V_{SS} -0.6V			30	100	µA
V_{en} = L	Enable Low Voltage (pins 6, 11)			-0.3		1.5	V
V_{en} = H	Enable High Voltage (pins 6, 11)			2.3		V_{SS}	V
I_{en} = L	Low Voltage Enable Current (pins 6, 11)	V_{en} = L				-10	µA
I_{en} = H	High Voltage Enable Current (pins 6, 11)	V_{en} = H ≤ V_{SS} -0.6V			30	100	µA
$V_{CEsat\ (H)}$	Source Saturation Voltage	I_L = 1A I_L = 2A		0.95	1.35 2	1.7 2.7	V V
$V_{CEsat\ (L)}$	Sink Saturation Voltage	I_L = 1A (5) I_L = 2A (5)		0.85	1.2 1.7	1.6 2.3	V V
V_{CEsat}	Total Drop	I_L = 1A (5) I_L = 2A (5)		1.80		3.2 4.9	V V
V_{sens}	Sensing Voltage (pins 1, 15)			-1 (1)		2	V

ภาคผนวก ค

โค้ดโปรแกรมการทำงานของหุ่นยนต์

```
#include <TimerOne.h>
#include "HX711.h"
#include <Wire.h>
#include <LCD.h>
#include <LiquidCrystal_I2C.h>
#include <FuzzyRule.h>
#include <FuzzyComposition.h>
#include <Fuzzy.h>
#include <FuzzyRuleConsequent.h>
#include <FuzzyOutput.h>
#include <FuzzyInput.h>
#include <FuzzyIO.h>
#include <FuzzySet.h>
#include <FuzzyRuleAntecedent.h>

//กำหนดค่า ในอินพุตฟัซซี//

Fuzzy* fuzzy = new Fuzzy();

 FuzzySet* small = new FuzzySet(0.5, 1,2.2 , 2.5);
 FuzzySet* medium = new FuzzySet(2.3, 3, 4.2, 4.5);
 FuzzySet* large = new FuzzySet(4.3, 5, 5.2, 5.5);

  FuzzySet* error1 = new FuzzySet(0.5, 1, 1, 1.5);
  FuzzySet* error2 = new FuzzySet(1.5, 2, 2, 2.5);
  FuzzySet* error3 = new FuzzySet(2.5, 3, 3, 3.5);
  FuzzySet* error4 = new FuzzySet(3.5, 4, 4, 4.5);
  FuzzySet* error5 = new FuzzySet(4.5, 5, 5, 5.5);

#define I2C_ADDR    0x3F
#define BACKLIGHT_PIN  3
```

```
LiquidCrystal_I2C lcd(I2C_ADDR, 2, 1, 0, 4, 5, 6, 7);
float calibration_factor = 173908.00;
#define zero_factor 8482205
#define DOUT  A3
#define CLK   A2
#define DEC_POINT  2
float offset = 0;
float get_units_kg();

HX711 scale(DOUT, CLK);
unsigned int counter = 0;
int in1Pin      = 11;
int in2Pin      = 10;
int in3Pin      = 9;
int in4Pin      = 8;
int in5Pin      = 7;
int in6Pin      = 6;
int in7Pin      = 5;
int in8Pin      = 4;
int buttonPin1    = 18;
int buttonPin2    = 19;
int in13Pin = 13;
int buttonState1  = 0;
int buttonState2  = 0;
int rotation      = 0;
int STATE1 = HIGH;
int I=0;

void docount()
{
        counter++;
```

```
}

void timerIsr()
{
        Timer1.detachInterrupt();
        Serial.print("Motor Speed: ");
        rotation = (counter / 20);
        Serial.print(rotation, DEC);
        Serial.println(" Rotation per seconds");
        counter = 0;
        Timer1.attachInterrupt( timerIsr );
}

void START()
{
        STATE1 = LOW;
}

void STOP()
{
 STATE1 = HIGH;

        digitalWrite(in1Pin, LOW);
        digitalWrite(in2Pin, LOW);
        digitalWrite(in3Pin, LOW);
        digitalWrite(in4Pin, LOW);
        digitalWrite(in5Pin, LOW);
        digitalWrite(in6Pin, LOW);
        digitalWrite(in7Pin, LOW);
        digitalWrite(in8Pin, LOW);
}
```

```
void setup(){
Serial.begin(9600);
 lcd.begin (16, 2);
lcd.setBacklightPin(BACKLIGHT_PIN, POSITIVE);
lcd.setBacklight(HIGH);

Timer1.initialize(1000000);
attachInterrupt(0, docount, RISING);   Timer1.attachInterrupt( timerIsr );

scale.set_scale(calibration_factor);
scale.set_offset(zero_factor);

pinMode(in1Pin, OUTPUT);
pinMode(in2Pin, OUTPUT);
pinMode(in3Pin, OUTPUT);
pinMode(in4Pin, OUTPUT);
pinMode(in5Pin, OUTPUT);
pinMode(in6Pin, OUTPUT);
pinMode(in7Pin, OUTPUT);
pinMode(in8Pin, OUTPUT);
pinMode(in13Pin, OUTPUT);
pinMode(buttonPin1, INPUT_PULLUP);
pinMode(buttonPin2, INPUT_PULLUP);
 attachInterrupt(digitalPinToInterrupt(buttonPin1), START, FALLING);
attachInterrupt(digitalPinToInterrupt(buttonPin2), STOP, FALLING);

//กำหนดชื่ออินพุตฟัซซี
FuzzyInput* weight = new FuzzyInput(1);
 weight->addFuzzySet(small);
 weight->addFuzzySet(medium);
 weight->addFuzzySet(large);
```

```
fuzzy->addFuzzyInput(weight);

FuzzyInput* errorspeed = new FuzzyInput(2);
errorspeed->addFuzzySet(error1);
errorspeed->addFuzzySet(error2);
errorspeed->addFuzzySet(error3);
errorspeed->addFuzzySet(error4);
errorspeed->addFuzzySet(error5);
fuzzy->addFuzzyInput(errorspeed);

//กำหนดค่าเอาท์พุต//
FuzzyOutput* varyspeed = new FuzzyOutput(1);
FuzzySet* minimum = new FuzzySet(10, 20, 30, 50);
varyspeed->addFuzzySet(minimum);
FuzzySet* middle = new FuzzySet(45, 55, 65, 75);
varyspeed->addFuzzySet(middle);
FuzzySet* maximum = new FuzzySet(70, 80, 80, 90);
varyspeed->addFuzzySet(maximum);
fuzzy->addFuzzyOutput(varyspeed);

FuzzyOutput* outputerror = new FuzzyOutput(2);
FuzzySet* e1 = new FuzzySet(-12, -12, -12, -12);
outputerror ->addFuzzySet(e1);
FuzzySet* e2 = new FuzzySet(4, 4, 4, 4);
outputerror ->addFuzzySet(e2);
FuzzySet* e3 = new FuzzySet(-10, -10, -10, -10);
outputerror ->addFuzzySet(e3);
FuzzySet* e4 = new FuzzySet(3, 3, 3, 3);
outputerror ->addFuzzySet(e4);
FuzzySet* e5 = new FuzzySet(4, 4, 4, 4);
outputerror ->addFuzzySet(e5);
```

```
fuzzy->addFuzzyOutput(outputerror );

//กำหนดกฎของฟัซซี//
//IF weight = small THEN varyspeed = minimum//

FuzzyRuleAntecedent* ifWeightSmall = new FuzzyRuleAntecedent();
ifWeightSmall->joinSingle(small);
FuzzyRuleConsequent* thenVaryspeedMinimum = new FuzzyRuleConsequent();
thenVaryspeedMinimum->addOutput(minimum);
        FuzzyRule* fuzzyRule01 = new FuzzyRule(1, ifWeightSmall,
                thenVaryspeedMinimum);
fuzzy->addFuzzyRule(fuzzyRule01);

//IF weight = medium THEN varyspeed = middle//

FuzzyRuleAntecedent* ifWeightMedium = new FuzzyRuleAntecedent();
ifWeightMedium->joinSingle(medium);
FuzzyRuleConsequent* thenVaryspeedMiddle = new FuzzyRuleConsequent();
thenVaryspeedMiddle->addOutput(middle);
FuzzyRule* fuzzyRule02 = new FuzzyRule(1, ifWeightMedium,
thenVaryspeedMiddle);
fuzzy->addFuzzyRule(fuzzyRule02);

//IF weight = large THEN varyspeed = maximum//

FuzzyRuleAntecedent* ifWeightLarge = new FuzzyRuleAntecedent();
ifWeightLarge->joinSingle(large);
FuzzyRuleConsequent* thenVaryspeedMaximum= new FuzzyRuleConsequent();
thenVaryspeedMaximum->addOutput(maximum);
FuzzyRule* fuzzyRule03 = new FuzzyRule(1,
ifWeightLarge,thenVaryspeedMaximum);
```

```
fuzzy->addFuzzyRule(fuzzyRule03);

//IF errorspeed = error1 THEN outputerror = e1//

FuzzyRuleAntecedent* ifErrorspeedError1 = new FuzzyRuleAntecedent();
ifErrorspeedError1->joinSingle(error1);
FuzzyRuleConsequent* thenOutputspeedE1= new FuzzyRuleConsequent();
thenOutputspeedE1->addOutput(e1);
FuzzyRule* fuzzyRule04 = new FuzzyRule(1, ifErrorspeedError1, thenOutputspeedE1);
fuzzy->addFuzzyRule(fuzzyRule04);

//IF errorspeed = error2 THEN outputerror = e2//

FuzzyRuleAntecedent* ifErrorspeedError2 = new FuzzyRuleAntecedent();
ifErrorspeedError2->joinSingle(error2);
FuzzyRuleConsequent* thenOutputspeedE2= new FuzzyRuleConsequent();
thenOutputspeedE2->addOutput(e2);
FuzzyRule* fuzzyRule05 = new FuzzyRule(1, ifErrorspeedError2, thenOutputspeedE2);
fuzzy->addFuzzyRule(fuzzyRule05);

//IF errorspeed = error3 THEN outputerror = e3//

FuzzyRuleAntecedent* ifErrorspeedError3 = new FuzzyRuleAntecedent();
ifErrorspeedError3->joinSingle(error3);
FuzzyRuleConsequent* thenOutputspeedE3= new FuzzyRuleConsequent();
thenOutputspeedE3->addOutput(e3);
FuzzyRule* fuzzyRule06 = new FuzzyRule(1, ifErrorspeedError3, thenOutputspeedE3);
fuzzy->addFuzzyRule(fuzzyRule06);
```

```
 //IF errorspeed = error4 THEN outputerror = e4//

FuzzyRuleAntecedent* ifErrorspeedError4 = new FuzzyRuleAntecedent();
 ifErrorspeedError4->joinSingle(error4);
 FuzzyRuleConsequent* thenOutputspeedE4= new FuzzyRuleConsequent();
 thenOutputspeedE4->addOutput(e4);
 FuzzyRule* fuzzyRule07 = new FuzzyRule(1, ifErrorspeedError4, thenOutputspeedE4);
 fuzzy->addFuzzyRule(fuzzyRule07);

 //IF errorspeed = error5 THEN outputerror = e5//
FuzzyRuleAntecedent* ifErrorspeedError5 = new FuzzyRuleAntecedent();
 ifErrorspeedError5->joinSingle(error5);
 FuzzyRuleConsequent* thenOutputspeedE5= new FuzzyRuleConsequent();
 thenOutputspeedE5->addOutput(e5);
 FuzzyRule* fuzzyRule08 = new FuzzyRule(1, ifErrorspeedError5, thenOutputspeedE5);
 fuzzy->addFuzzyRule(fuzzyRule08);
}


void loop(){
 buttonState1 = digitalRead(18);
 buttonState2 = digitalRead(19);
 String data = String(get_units_kg() + offset, DEC_POINT);
 float x = data.toFloat();

 //แสดงค่าน้ำหนักและค่าความเร็วบนจอ LCD//

  lcd.setCursor(0, 0);
 lcd.print("Weight");
 lcd.setCursor(8, 0);
 lcd.print(data);
```

```
lcd.setCursor(14, 0);
lcd.print("kg");
  if (data==0)
{
        lcd.setCursor(11, 0);
        lcd.print("0");
        lcd.setCursor(12, 0);
        lcd.print("0");
        lcd.setCursor(13, 0);
        lcd.print("0");
}
        lcd.setCursor(0, 1);
        lcd.print("speed");
        lcd.setCursor(8, 1);
        lcd.print(rotation);
        lcd.setCursor(11, 1);
        lcd.print("Rps");
 if (rotation==0)
{
        lcd.setCursor(9, 1);
        lcd.print("0");
}
int A=x-0;
fuzzy->setInput(1, A);
fuzzy->setInput(2, I);
fuzzy->fuzzify();

 float output1 = fuzzy->defuzzify(1);
 float output2 = fuzzy->defuzzify(2);
```

```
float y =165+output1+output2;
 if (rotation !=14 )
{
          if (x<1){
         I=0;
          }
          if (x>0.5&&x<1.5){
         I=1;
         }
          if (x>1.5&&x<2.5){
          I=2;
         }
        if (x>2.5&&x<3.5){
         I=3;
         }
          if (x>3.5&&x<4.5){
          I=4;
         }
          if (x>4.5&&x<5.5){
          I=5;
         }
}
```

//กำหนดการทำงานของมอเตอร์เมื่อรับคำสั่งจากสวิตช์//

```
if (STATE1 == LOW) {
          digitalWrite(in1Pin, LOW);
          digitalWrite(in2Pin, HIGH);
          analogWrite (in2Pin , y);
          digitalWrite(in3Pin, LOW);
         digitalWrite(in4Pin, HIGH);
```

```
        analogWrite (in4Pin , y);
        digitalWrite(in5Pin, LOW);
        digitalWrite(in6Pin, HIGH);
        analogWrite (in6Pin , y);
        digitalWrite(in7Pin, LOW);
        digitalWrite(in8Pin, HIGH);
        analogWrite (in8Pin , y);
 }
 else {
        digitalWrite(in1Pin, LOW);
        digitalWrite(in2Pin, LOW);
        digitalWrite(in3Pin, LOW);
        digitalWrite(in4Pin, LOW);
        digitalWrite(in5Pin, LOW);
        digitalWrite(in6Pin, LOW);
        digitalWrite(in7Pin, LOW);
        digitalWrite(in8Pin, LOW);
 }

//กำหนดการแสดงไฟ LED//

 if (rotation ==14 )
{
        digitalWrite(in13Pin,HIGH);
}
else if (rotation !=14 )
{
        digitalWrite(in13Pin,LOW);
}
}
float get_units_kg()
```

```
{
        return (scale.get_units() * 0.453592);
}
```

ประวัติผู้ดำเนินโครงงาน

ชื่อ นางสาวประภาศรี คำลือ
ภูมิลำเนา 22/1 ม.6 ต.เชียงม่วน อ.เชียงม่วน
จ. พะเยา
ประวัติการศึกษา
- จบระดับมัธยมศึกษาจากโรงเรียน
เชียงม่วนวิทยาคม
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่
4 สาขาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร
Email: prapasrik55@email.nu.ac.th

ชื่อ นายนัฐพงศ์ ใบวุฒิ
ภูมิลำเนา 88/178 ม.2 ต.ท่าทอง อ.เมือง จ.พิษณุโลก
ประวัติการศึกษา
- จบระดับมัธยมศึกษาจากโรงเรียน
พิษณุโลกพิทยาคม
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปี
ที่ 4 สาขาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร
Email: nattapongb55@email.nu.ac.th