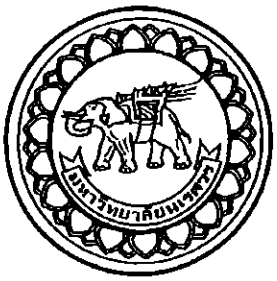
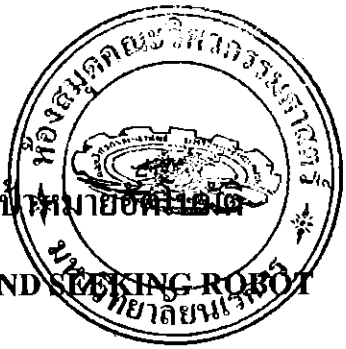




การพัฒนาหุ่นยนต์หลบหลีกสิ่งกีดขวางและค้นหาเป้าหมายอัตโนมัติ

IMPROVEMENT OF AUTONOMOUS AVOIDANCE AND SEEKING ROBOT

นายศีลวัตร มะลิดิน รหัส 49371378

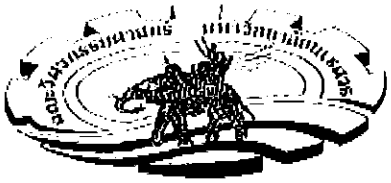
นายอภิชัย บุญมา รหัส 49371460

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2552



ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน	การพัฒนาหุ่นยนต์หลบหลีกสิ่งกีดขวางและค้นหาเป้าหมายอัตโนมัติ		
ผู้ดำเนินโครงงาน	นายศีลวัตร	มะลิดิน	รหัส 49371378
	นายอภิชัย	บุญมา	รหัส 49371460
อาจารย์ที่ปรึกษา	อาจารย์เศรษฐา ตั้งค้าวานิช		
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2552		

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์

คณะกรรมการการสอบโครงงานวิศวกรรม

.....ประธานกรรมการ
(อาจารย์เศรษฐา ตั้งค้าวานิช)

.....กรรมการ
(ดร.อัครพันธ์ วงศ์กังแห)

.....กรรมการ
(อาจารย์ภาณุพงศ์ สอนคม)

หัวข้อโครงงาน	การพัฒนาหุ่นยนต์หลบหลีกสิ่งกีดขวางและค้นหาเป้าหมายอัตโนมัติ		
ผู้ดำเนินโครงงาน	นายศีลวัตร	มะลิดิน	รหัส 49371378
	นายอภิชัย	บุญมา	รหัส 49371460
อาจารย์ที่ปรึกษา	อาจารย์เศรษฐา	ตั้งค้าวานิช	
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2552		

.....

บทคัดย่อ

โครงงานนี้ศึกษาและพัฒนาหุ่นยนต์ค้นหาเป้าหมายและหลบหลีกสิ่งกีดขวางอัตโนมัติ ซึ่งพัฒนาขึ้นเพื่อเพิ่มขีดความสามารถในการค้นหาผู้ประสบภัยในกรณีเกิดเหตุการณ์ฉุกเฉินขึ้น แล้วต้องการค้นหาเป้าหมายในพื้นที่ซึ่งทำให้เจ้าหน้าที่กู้ภัยไม่สามารถเข้าถึงได้ รวมไปถึงเขตพื้นที่อันตรายอาจเนื่องด้วยสารพิษหรือพื้นที่เสี่ยงต่อการระเบิดซึ่งอาจก่ออันตรายกับเจ้าหน้าที่ได้ โดยหุ่นยนต์สามารถเคลื่อนที่ภายในสนามจำลองเพื่อค้นหาเป้าหมายที่สนใจ นอกจากนั้นยังสามารถระบุข้อมูลสภาพแวดล้อมที่จำเป็นของจุดสนใจได้อย่างอัตโนมัติ ด้วยการทำงานร่วมกันระหว่างคอมพิวเตอร์และไมโครคอนโทรลเลอร์ โดยที่คอมพิวเตอร์จะประมวลผลจากภาพที่รับมาจากกล้อง Webcam ด้วยวิธีการ Image subtraction เพื่อหาสิ่งสนใจในภาพและสั่งให้ไมโครคอนโทรลเลอร์ควบคุมการหมุนของมอเตอร์เพื่อเคลื่อนที่หุ่นยนต์ไปยังจุดหมายที่ต้องการ การพัฒนาโปรแกรมนี้ทำงานและทดสอบบนระบบปฏิบัติการไมโครซอฟต์วินโดวส์ XP โครงงานนี้จะอธิบายถึงการทำงานของหุ่นยนต์ที่พัฒนาขึ้นและอุปกรณ์สำคัญที่ใช้ เช่น กล้อง โมดูลวัดระยะทางด้วย Infrared รวมไปถึงอุปกรณ์ต่าง ๆ ที่ใช้ควบคุมการทำงานของหุ่นยนต์ สิ่งที่ได้รับจากโครงงานนี้คือ ทำให้มีหุ่นยนต์ต้นแบบที่สามารถพัฒนาให้หุ่นยนต์ทำงานร่วมกับมนุษย์ เพื่อช่วยเหลือผู้ประสบภัยในสถานการณ์ต่าง ๆ ได้

Project Title	Improvement of autonomous avoidance and seeking robot		
Name	Mr. Seelwat	Malidin	ID. 49371378
	Mr. Apichai	Boonma	ID. 49371460
Project Advisor	Mr. Settha	Tangkavanit	
Major	Computer Engineering		
Department	Electrical and Computer Engineering		
Academic Year	2009		

.....

ABSTRACT

This project studies and development of robots that search for targets and avoid automatic barrier to developing capacity in the search for victims in the case of an emergency incident. Then find the target area rescue workers cannot reach. Including area may be because of toxic or hazardous areas at the risk of explosion could be hazardous to workers. Rescue robot to find targets of interest to avoid obstruction and also provide the environment necessary to run automatically with the collaboration between computer and microcontroller. Processing of digital images by image subtraction and assign work to microcontroller controls the rotation of the direct current motor. Development and test program running on the operating system Microsoft Windows XP, this project will describe the performance of robotic devices developed and being used, such as cameras, sensors Infrared distance measuring module including devices that use the control of robots. This project makes a prototype robot that can be developed to work with human to help victims of events.

กิตติกรรมประกาศ

ทางคณะผู้จัดทำโครงงาน **"การพัฒนาหุ่นยนต์หลบหลีกสิ่งกีดขวางและค้นหาเป้าหมายอัตโนมัติ"** ขอขอบคุณ อาจารย์เศรษฐา ตั้งค้าวานิช ที่ให้ความช่วยเหลือในโครงงานนี้ให้สามารถดำเนินการไปได้ด้วยดี โดยช่วยให้คำแนะนำปรึกษาเกี่ยวกับโครงงานตลอด ทั้งให้ความเอื้อเฟื้อสถานที่ในการทำงานและอุปกรณ์เครื่องมือต่างๆ อีกทั้งอาจารย์ทุกท่านในภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ที่ให้คำแนะนำและช่วยเหลือในครั้งนี้

นายศีลวัตร มะลิดิน

นายอภิชัย บุญมา

สารบัญ

หน้า

บทคัดย่อภาษาไทย ..... ก

บทคัดย่ออังกฤษ ..... ข

กิตติกรรมประกาศ ..... ค

สารบัญ ..... ง

สารบัญตาราง ..... ช

สารบัญรูป ..... ฌ

บทที่ 1 บทนำ

1.1 ที่มาและความสำคัญของโครงงาน 1

1.2 จุดประสงค์ของโครงงาน 1

1.3 ขอบเขตของโครงงาน 1

1.4 แผนการดำเนินงาน 2

1.5 ผลที่คาดว่าจะได้รับ 2

1.6 งบประมาณของโครงงาน 3

บทที่ 2 หลักการและทฤษฎีที่เกี่ยวข้อง

2.1 โครงสร้างหุ่นยนต์และการควบคุมหุ่นยนต์ 4

2.1.1 เฟือง (Gear) 4

2.1.2 เพลา (Shaft) 5

2.1.3 การเชื่อมโลหะ 6

2.2 หลักกการควบคุมหุ่นยนต์ 6

2.2.1 โครงสร้างในการขับเคลื่อนหุ่นยนต์ 7

2.2.2 โครงสร้างอุปกรณ์ควบคุมการเคลื่อนที่ของหุ่นยนต์ 12

2.2.3 กล้อง และ การเชื่อมต่อกล้อง Web Camera 35

2.3 การควบคุมระดับบน (High Level) 24

2.3.1 การประมวลภาพ (Image processing) 24

2.3.2 หลักปัญญาประดิษฐ์ 29

2.3.3 กล้อง และ การเชื่อมต่อกล้อง Web Camera 35

สารบัญ (ต่อ)

หน้า

บทที่ 3 หลักการทำงาน

3.1 การออกแบบโครงสร้างหุ่นยนต์ 37

3.1.1 หลักการออกแบบตัวหุ่นยนต์ 37

3.1.2 โครงสร้างของหุ่นยนต์ 37

3.1.3 ออกแบบโครงล้อของหุ่นยนต์ 37

3.1.4 การออกบแบบล้อของหุ่นยนต์ 41

3.1.5 การออกแบบล้อตีนตะขาบ 44

3.2 การออกแบบระดับล่าง (Low Level) 47

3.2.1 ชุดควบคุมการทำงานโดยไมโครคอนโทรลเลอร์ 47

3.2.2 การติดต่อพอร์ทอนุกรม 48

3.2.3 ชุดควบคุมมอเตอร์แบบ H-Bridge 49

3.2.4 การเชื่อมต่ออุปกรณ์ตรวจจับวัตถุ 53

3.3 การควบคุมระดับบน (High level control) 60

3.3.1 แนวคิดการแก้ปัญหาจะต้องไปที่ไหน 60

3.3.2 การประมวลผลภาพค้นหาเป้าหมาย 64

3.3.3 ส่วนควบคุมติดต่อกับผู้ใช้ (Graphic User Interface, GUI) 70

บทที่ 4 ผลการทดลอง

4.1 การค้นหาเป้าหมายอัตโนมัติ 71

4.1.1 การทดสอบค้นหาตำแหน่งของสีที่สนใจ 71

4.1.2 การทดสอบเลือกค่าเส้นแบ่งที่เหมาะสม 75

4.1.3 การทดสอบค้นหาตำแหน่งของสีที่สนใจและการกำจัดสัญญาณรบกวน 77

4.1.4 การกำจัดสัญญาณรบกวนและเลือกขนาด Kernel ที่เหมาะสม 78

4.1.5 การทดสอบการกำหนดตำแหน่งของผู้ประสบภัย 79

4.1.6 การทดสอบค้นหาตำแหน่งของสีที่สนใจและการกำจัดสัญญาณรบกวน 77

4.2 การเคลื่อนที่หุ่นยนต์และค้นหาเป้าหมายอัตโนมัติ 81

4.2.1 การทดสอบหุ่นยนต์เคลื่อนที่หลบหลีกสิ่งกีดขวาง 81

4.2.2 การทดสอบหุ่นยนต์เคลื่อนที่หลบหลีกสิ่งกีดขวางและค้นหาเป้าหมาย 87

สารบัญ (ต่อ)

หน้า

บทที่ 5 สรุปผล

5.1 สรุปผลของโครงงาน 92

5.2 ปัญหาที่พบ 92

5.2.1 ปัญหาทางด้านโครงสร้างหุ่นยนต์ (ตัวหุ่นยนต์) 92

5.2.2 ปัญหาทางด้านระบบควบคุมระดับล่าง 93

5.2.3 ปัญหาทางด้านระบบควบคุมระดับบน 93

5.3 แนวทางการแก้ปัญหาและข้อเสนอแนะ 94

5.3.1 ปัญหาทางด้านโครงสร้างหุ่นยนต์ (ตัวหุ่นยนต์) 94

5.3.2 ปัญหาทางด้านระบบควบคุมระดับล่าง 94

5.3.3 ปัญหาทางด้านระบบควบคุมระดับบน 95

5.4 แนวทางในการพัฒนาเพิ่มเติม 95

เอกสารอ้างอิง 98

ภาคผนวก 99

สารบัญตาราง

ตารางที่ หน้า

2.1 ตัวอย่างสอนที่สังเกตได้ ... 32

2.2 ชุดตัวอย่างสอนสำหรับการเรียนรู้เบย์อย่างง่าย ... 34

2.3 ชุดตัวอย่างข้อมูลที่ต้องการจำแนก ... 34

3.1 ค่าสถานะการเคลื่อนที่ของหุ่นยนต์ ... 52

4.1 ผลการทดสอบหุ่นยนต์เคลื่อนที่ค้นหาเป้าหมายแบบมีผนังกีดขวางและทางแยก ... 82

4.2 ผลการทดสอบหุ่นยนต์เคลื่อนที่ ค้นหาเป้าหมายแบบมีผนังกีดขวาง ทางแยกและทางตันขวาง ... 83

4.3 ผลการทดสอบหุ่นยนต์เคลื่อนที่แบบมีผนังกีดขวาง ทางแยกและทางตันขวาง รอบที่ 2 หลังจากการชาร์จแบตเตอรี่ ... 84

4.4 ผลการทดสอบหุ่นยนต์เคลื่อนที่แบบมีผนังกีดขวาง ทางแยกซ้าย ... 85

4.5 ผลการทดสอบหุ่นยนต์เคลื่อนที่แบบมีผนังกีดขวาง ทางแยกขวา ... 86

4.6 ผลการทดสอบหุ่นยนต์เคลื่อนที่และค้นหาผู้ประสบภัยจำนวน 1 คน ... 89

4.7 ผลการทดสอบหุ่นยนต์เคลื่อนที่และค้นหาผู้ประสบภัยจำนวน 2 คน ... 90

สารบัญรูป

รูปที่ หน้า

2.1 ลักษณะของเฟืองตรงชนิดขบ 4

2.2 ลักษณะการส่งกำลังของเฟืองตรง 5

2.3 เพลารองรับภาระ 6

2.4 วงจรภายในมอเตอร์กระแสตรง 7

2.5 แสดงการทำงานของมอเตอร์กระแสตรง 8

2.6 แสดงส่วนประกอบของมอเตอร์กระแสตรง 9

2.7 วงจรควบคุมความเร็วของมอเตอร์กระแสตรงโดย PWM 10

2.8 วงจรควบคุมความเร็วของมอเตอร์กระแสตรงโดย PWM 11

2.9 ภาพแสดง H-Bridge Switching เมื่อ S1 และ S3 On พร้อมกัน 11

2.10 ภาพแสดงH-Bridge Switching เมื่อ S2 และ S4 On พร้อมกัน 11

2.11 แสดงลักษณะการสั่งงานมอเตอร์ให้ล้อตีนตะขาบเคลื่อนที่เดินหน้า ถอยหลัง เลี้ยวซ้าย เลี้ยวขวา ตามลำดับจากซ้ายไปขวา 12

2.12 แสดงโครงสร้างเบื้องต้นการควบคุมทิศทางการหมุนของมอเตอร์จากไมโครคอนโทรลเลอร์ 12

2.13 วงจรไมโครคอนโทรลเลอร์ P89V51RD2 13

2.14 รูปแบบการส่งข้อมูลแบบอนุกรม 15

2.15 การสื่อสารแบบอะซิงโครนัสที่ไม่มีพาริตี้บิต 16

2.16 การสื่อสารแบบอะซิงโครนัสที่มีพาริตี้บิต 16

2.17 ภาพแสดงตัวตรวจจับรังสีอินฟราเรด 17

2.18 ภาพแสดง Fresnel lens 18

2.19 ภาพแสดงการทำงานของ PIR 18

2.20 ภาพแสดงการทำงานของ PIR 18

2.21 ภาพแสดง TYPICAL CONFIGURATION OF PYROSENSOR 19

2.22 ภาพแสดง Ultrasonic sensor 19

2.23 ภาพแสดงวงจรชุดรับของ Ultrasonic sensor 20

2.24 ภาพแสดงวงจรชุดส่งของ Ultrasonic 20

2.25 ภาพแสดง Incremental encoder 21

สารบัญรูป (ต่อ)

รูปที่ หน้า

2.26 ระบบควบคุมแบบ PID 21

2.27 ระบบควบคุมแบบดิจิตอล 22

2.28 ตำแหน่งของพิกเซลของภาพ 2 มิติ 24

2.29 การผสมสีของระบบสี RGB 24

2.30 การรวมกันของสีหลักต่างๆ 25

2.31 Cylinder แทนระบบสี HSL และ HSV 25

2.32 เปรียบเทียบระหว่างระบบสี HSL (รูปซ้าย) และ HSV (รูปขวา) 26

2.33 (ก) ภาพต้นแบบ

(ข) กราฟแสดงการแจกแจงความถี่ของพิกเซล 27

2.34 กราฟตัวอย่างการหาขอบภาพ 28

2.35 การหาขอบโดยใช้วิธี Gradient method

(ก) มีค่า SNR = 30 dB

(ข) มีค่า SNR = 20 dB 28

2.36 (ก) ภาพตัวอย่าง

(ข) ผลลัพธ์หลังจากใช้วิธี Laplacian พร้อมกับ LOG Filter

(ค) ผลลัพธ์หลังจากใช้วิธี Laplacian พร้อมกับ PLUS Filter 29

2.37 ปัญหาโลกของบล็อก 29

2.38 การค้นหาแบบแนวกว้างก่อนในปัญหาโลกของบล็อก 30

2.39 การค้นหาแบบแนวลึกก่อนในปัญหาโลกของบล็อก 31

2.40 ตัวอย่างของต้นไม้ตัดสินใจ 33

2.41 สมการวิธีจำแนกประเภทแบบเบย์อย่างง่าย 33

2.42 ภาพแสดงกล้อง Web Cam 35

3.1 แนวคิดเริ่มต้นของการสร้างหุ่นยนต์กู้ภัยอัตโนมัติ 36

3.2 เหล็กกระบอกสี่เหลี่ยมสำหรับทำโครงล้อ 37

3.3 แผ่นเหล็กสำหรับทำที่ประกบล้อหน้า 38

3.4 แผ่นเหล็กสำหรับทำที่ประกบล้อหลัง 38

3.5 แผ่นเหล็กไว้ยึดแผ่นประกบล้อ 38

สารบัญรูป (ต่อ)

รูปที่ หน้า

3.6 การประกอบชิ้นส่วนของฐานล้อ 39
3.7 รูปฐานล้อเมื่อประกอบแล้ว 39
3.8 เหล็กฉากที่จะนำไปยึดเป็นโครงของตัวหุ่นยนต์ 40
3.9 โครงสร้างฐานล้อของหุ่นยนต์ 40
3.10 การติดตั้งชุดบูทกับแผ่นล้อหน้า 40
3.11 สเตอร์รถจักรยานยนต์ขนาด 34 ฟัน 41
3.12 รูปจริงของสเตอร์รถจักรยานยนต์ขนาด 34 ฟัน 41
3.13 เพลาล้อหน้า 42
3.14 การประกอบเพลาเข้ากับดุมและล้อ 42
3.15 ชุดส่วนประกอบของล้อหลัง 42
3.16 การประกอบล้อหน้าเข้ากับโครงของหุ่นยนต์ 43
3.17 การประกอบล้อหลังของหุ่นยนต์ 43
3.18 เฟืองโซ่ขับเคลื่อนและการติดตั้งเฟืองโซ่ 44
3.19 โครงฐานล้อของหุ่นยนต์ 44
3.20 ภาพแสดงการออกแบบส่วนประกอบของล้อตีนตะขาบ 45
3.21 แผ่นเหล็กที่เชื่อมกับโซ่ 45
3.22 ชิ้นส่วนของโซ่ตีนตะขาบ 45
3.23 สายพานรถจักรยานยนต์ที่ตัดแล้ว 46
3.24 ส่วนประกอบของล้อตีนตะขาบ 46
3.25 ล้อเมื่อใส่โซ่ตีนตะขาบ 46
3.26 วงจรไมโครคอนโทรลเลอร์แสดงพอร์ทต่าง ๆ 47
3.27 บอร์ดไมโครคอนโทรลเลอร์และอุปกรณ์ประกอบพร้อมลายปริ้นจาก
โปรแกรม Protel 99SE 48
3.28 วงจรติดต่อผ่านพอร์ทสื่อสารอนุกรม RS-232 48
3.29 การจัดการขาของไอซี L298 49
3.30 การต่อวงจรใช้งานแบบ 1 ช่องของ L298 50
3.31 บอร์ดควบคุมมอเตอร์ H-Bridge 50

สารบัญรูป (ต่อ)

รูปที่ หน้า

3.32 Module ควบคุมทิศทางการหมุนของมอเตอร์ขับเคลื่อน 51

3.33 วงจรขยายกระแสด้วย IC ULN2003A และ IC Not Gate 7404 53

3.34 ภาพการจำลองเหตุการณ์เกิดเหตุภัยพิบัติ 54

3.35 การแสดงการตรวจจับสิ่งกีดขวาง และการติดตั้งอุปกรณ์ตรวจวัดระยะทาง 54

3.36 โมดูลตรวจวัดระยะทางด้วยแสงอินฟราเรด GP2Y0A21YK0F

(ก) ลักษณะภายนอกของโมดูลตรวจวัดระยะทาง

(ข) ขนาดของโมดูลตรวจวัดระยะทาง 55

3.37 โมดูล Analog to Digital เบอร์ PCF8591 8-bit A/D 56

3.38 การทำงานของ SDA และ SCL โมดูล Analog to Digital เบอร์ PCF8591 8-bit A/D 56

3.39 ลำดับของข้อมูลของโมดูล Analog to Digital เบอร์ PCF8591 8-bit A/D 57

3.40 แสดงการวัดสัญญาณการรับส่งข้อ SDA (เส้นบน) และ SCL (เส้นล่าง) 57

3.41 แสดงการทดสอบการรับค่าจากเซนเซอร์ 57

3.42 แสดงการประกอบอุปกรณ์และเชื่อมต่อสายบัสต่าง

(ก) ภาพวงจรที่รวมวงจรทั้งหมดเข้าด้วยกัน

(ข) แสดงการเชื่อมต่อของ MCU และโมดูลต่าง ๆ จากมุมบน

(ค) แสดงเชื่อมต่อบัสต่างๆ เข้าสู่ตัวหุ่นยนต์ 58

3.43 แสดงการติดตั้งกล้อง Webcam กับตัวหุ่นยนต์ 59

3.44 แสดงหุ่นยนต์ประกอบเสร็จสมบรูณ์ 59

3.45 ภาพการจำลองผู้ประสบภัยของการแข่งขันหุ่นยนต์กู้ภัยชิงแชมป์ประเทศไทย ประจำปี 2552 60

3.46 แผนผังการเคลื่อนที่ของหุ่นยนต์ของแต่ละสถานะ 61

3.47 แสดง Flow chart การเคลื่อนที่ของหุ่นยนต์ในสถานะ S1 62

3.48 Flow chart การเคลื่อนที่ของหุ่นยนต์ S2 ไปยัง S3 ไปยัง S1 62

3.49 ภาพจำลองการมองเห็นของหุ่นยนต์เมื่อพบผู้ประสบภัย 63

3.50 แสดง Flow chart การค้นหาเป้าหมายโดยการประมวลผลภาพ 65

3.51 การนำภาพจากแหล่งเก็บข้อมูลเข้าประมวลผลภาพ

(ก) ภาพสถานการณ์จำลองแบบไม่มีผู้ประสบภัย

(ข) ภาพสถานการณ์จำลองแบบมีผู้ประสบภัย 66

สารบัญรูป (ต่อ)

รูปที่ หน้า

3.52 การแปลงภาพ HSV Model ใน RGB Model

(ก) ภาพการแปลงสถานการณ์จำลองแบบไม่มีผู้ประสบภัย

(ข) ภาพการแปลงสถานการณ์จำลองแบบมีผู้ประสบภัย 66

3.53 การแปลงภาพ HSV Model จาก RGB Model

(ก) ภาพแสดงสีจำลองเครื่องแต่งกายผู้ประสบภัย

(ข) ภาพการแปลงเป็น HSV Model แสดงใน RGB Model 67

3.54 การแยกภาพจากการลบกันของภาพจริงกับภาพจำลองสีเครื่องแต่งกายผู้ประสบภัย

(ก) แสดงภาพ H Plane ของ HSV Model

(ข) แสดงภาพ S Plane ของ HSV Model

(ค) แสดงภาพ V Plane ของ HSV Model 67

3.55 การทำ Threshold ที่ค่าลบกันต่ำกว่า 50 ให้เป็น 0 แล้วกลับค่าด้วย Binary Inverse

(ก) แสดงการทำ Threshold ภาพ H Plane ของ HSV Model

(ข) แสดงการทำ Threshold งภาพ S Plane ของ HSV Model

(ค) แสดงการทำ Threshold ภาพ V Plane ของ HSV Model 68

3.56 การทำกรอง Noises ด้วย Median filter ด้วย Mask 3 x 3

(ก) แสดงการทำ Median filter ภาพ H Plane ของ HSV Model

(ข) แสดงการทำ Median filter ภาพ S Plane ของ HSV Model

(ค) แสดงการทำ Median filter ภาพ V Plane ของ HSV Model 68

3.57 การทำกำหนดขอบเขตของผู้ประสบภัยค้นหา 69

3.58 แสดงการหาจุดพิกัดรอบกรอบสีเหลี่ยม ที่แสดงค่า

X Min, Y Min และ X Max, Y Max 69

3.59 ภาพแสดงส่วนของ Graphic User Interface 70

4.1 แสดงการหาตำแหน่งของสีที่สนใจด้วยการลบกันระหว่าง ภาพจำลอง กับสีแดงที่สนใจ 71

4.2 แสดงการหาตำแหน่งของสีที่สนใจด้วยการลบกันระหว่าง ภาพจำลอง กับสีเขียวที่สนใจ 72

4.3 แสดงการหาตำแหน่งของสีที่สนใจด้วยการลบกันระหว่าง ภาพจำลอง กับสีน้ำเงินที่สนใจ.. 73

4.4 แสดงการหาตำแหน่งของสีที่สนใจด้วยการลบกันระหว่าง ภาพจำลอง กับสีน้ำเงินที่สนใจ.. 74

สารบัญรูป (ต่อ)

รูปที่		หน้า
4.5	แสดงภาพจำลองการลบกันระหว่างภาพสองภาพและภาพผลลัพธ์	75
4.6	ภาพแสดงการเลือกค่าเส้นขีดแบ่งที่เหมาะสม	76
4.7	ภาพแสดงการกำจัดสัญญาณรบกวนด้วย filter แบบต่างๆ	77
4.8	แสดงตัวอย่างการเลื่อนของ Kernel 3x3	78
4.9	แสดงตัวอย่าง MEDIAN Filter กับ Kernel ขนาดต่างๆ	79
4.10	แสดงตัวอย่างตรวจนับพิกเซลเป้าหมายที่ค้นหาพบที่ 1000 พิกเซล	80
4.11	แสดงตัวอย่างตรวจนับพิกเซลเป้าหมายที่ค้นหาพบที่ 5000 พิกเซล	81
4.12	แสดงสถานการณ์จำลองการเคลื่อนที่แบบทางตรง และทางแยก มีผนังกีดขวาง	82
4.13	แสดงสถานการณ์จำลองการเคลื่อนที่แบบทางตรง ทางแยก และทางตันขวาง	83
4.14	แสดงสถานการณ์จำลองการเคลื่อนที่แบบทางตรง ทางแยกซ้าย	85
4.15	แสดงสถานการณ์จำลองการเคลื่อนที่แบบทางตรง ทางแยกขวา	86
4.16	ภาพหุ่นยนต์และผู้ประสบภัยจำลอง	88
4.17	แสดงสถานการณ์จำลองการค้นหาผู้ประสบภัย 1 คน	88
4.18	แสดงสถานการณ์จำลองการค้นหาผู้ประสบภัย 2 คน	89

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของโครงงาน

จากภัยธรรมชาติที่เกิดขึ้นไม่ว่าจะเป็น อัคคีภัย อุทกภัย วาตภัย และยังรวมไปถึงสิ่งที่เกิดจากการกระทำของมนุษย์ อย่างเช่น การก่อวินาศกรรมซึ่งเป็นเหตุให้มีผู้ได้รับบาดเจ็บและผู้เสียชีวิตทั้งในพื้นที่ซึ่งเจ้าหน้าที่ไม่สามารถเข้าไปช่วยเหลือผู้ประสบภัยได้โดยสะดวก และในที่สภาวะสภาพแวดล้อมที่ผู้ประสบภัยอาจก่อให้เกิดอันตรายต่อผู้เข้าช่วยเหลือ อย่างเช่น วัตถุมีพิษรังสีอันตราย, วัตถุระเบิด ดังนั้นโครงงานนี้จึงได้จัดทำ หุ่นยนต์กู้ภัยเพื่อช่วยค้นหาผู้ประสบภัยในพื้นที่ที่ยากต่อการเข้าถึง และช่วยบอกให้เจ้าหน้าที่กู้ภัยได้ทราบถึงสภาพแวดล้อมของผู้ประสบภัย ซึ่งเทคโนโลยีที่ทันสมัย ทั้งในด้านระบบสื่อสารไร้สาย ระบบอิเล็กทรอนิกส์ ระบบคอมพิวเตอร์ที่มีประสิทธิภาพนั้นยังไม่เพียงพอเนื่องจากยังคงต้องการผู้ควบคุมหุ่นยนต์ จึงมีความจำเป็นที่จะต้องพัฒนาให้หุ่นยนต์สามารถเคลื่อนที่และค้นหาเป้าหมายได้อย่างอัตโนมัติ เพื่อให้หุ่นยนต์สามารถทำงานได้โดยไม่จำเป็นต้องมีผู้ควบคุม

1.2 จุดประสงค์ของโครงงาน

1.2.1 เพื่อศึกษาการทำงานของไมโครคอนโทรลเลอร์ในการควบคุมมอเตอร์กระแสตรง

1.2.2 ใช้ความรู้เชิงวิศวกรรมทางด้านอิเล็กทรอนิกส์

1.2.3 เพื่อศึกษาการประมวลผลจากภาพดิจิตอล (Digital Image Processing)

1.2.4 เพื่อศึกษาและทดสอบการตัดสินใจแบบอัตโนมัติ สำหรับหุ่นยนต์อัตโนมัติ

1.3 ขอบเขตของโครงงาน

1.3.1 สามารถเขียนโปรแกรมควบคุมการทำงานของไมโครคอนโทรลเลอร์ MCS-51

1.3.2 สามารถเขียนโปรแกรมให้คอมพิวเตอร์ติดต่อกับไมโครคอนโทรลเลอร์ได้

1.3.3 หุ่นยนต์สามารถค้นหาและระบุตำแหน่งของสีที่สนใจในภาพที่รับมาได้

1.3.4 หุ่นยนต์สามารถเคลื่อนที่ในสนามจำลองตามมาตรฐานการแข่งขันหุ่นยนต์กู้ภัยชิงชนะเลิศระดับประเทศและหลบหลีกกำแพงที่เป็นอุปสรรคในการเคลื่อนที่ได้อย่างอัตโนมัติ

1.3.5 หุ่นยนต์สามารถระบุถึงข้อมูลของสภาพแวดล้อมในบริเวณที่สนใจได้อย่างอัตโนมัติ

1.4 แผนการดำเนินงาน

รายการ	พ.ศ. 2552							พ.ศ. 2553		
	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.
1.4.1 ออกแบบโครงสร้างหุ่นยนต์	←	→								
1.4.2 จัดหาวัสดุอุปกรณ์ สำหรับโครงสร้างของหุ่นยนต์	←	—	→							
1.4.3 หาเครื่องมือและสถานที่การทำโครงงาน	←	—	—	→						
1. 4.4 จัดหางบประมาณ	←	→								
1.4.5 ลงมือสร้างหุ่นยนต์				←	—	→				
1.4.6 ทำการทดสอบและแก้ไขโครงสร้างหุ่นยนต์					←	—	→			
1.4.7 ทำการประกอบชุดวงจรควบคุมหุ่นยนต์						←	→			
1.4.8 เขียนโปรแกรมควบคุมการทำงานของหุ่นยนต์								←	→	
1.4.9 ทำการทดสอบและแก้ไขการทำงานของโปรแกรม								←	→	
1.4.10 ทดสอบและปรับปรุงการทำงานของหุ่นยนต์									←	→

1.5 ผลที่คาดว่าจะได้รับ

1.5.1 สามารถควบคุมมอเตอร์ด้วยคอมพิวเตอร์ผ่านชุดไมโครคอนโทรลเลอร์ MCS-51ได้

1.5.2 สามารถบังคับหุ่นยนต์ด้วยระบบคอมพิวเตอร์ ได้อย่างอัตโนมัติ

1.5.3 สามารถค้นหาเป้าหมายที่สนใจได้อย่างอัตโนมัติ

1.5.4 หุ่นยนต์ตรวจจับสิ่งกีดขวางที่เป็นอุปสรรคต่อการเคลื่อนที่และหลบหลีกได้

1.6 งบประมาณของโครงงาน

วัสดุโครงสร้างหุ่นยนต์	5,000 บาท
วงจรควบคุมการทำงาน	1,500 บาท
มอเตอร์กระแสตรง	2,500 บาท
เซนเซอร์วัดระยะทาง	2,500 บาท
กล้อง Webcam	590 บาท
Notebook	27,500 บาท

บทที่ 2
หลักการและทฤษฎี

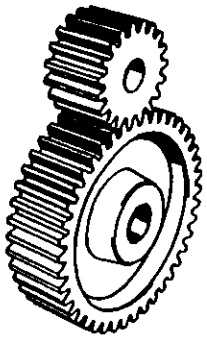
ในการจัดทำโครงการนี้จะประกอบไปด้วยหลักการและทฤษฎีต่าง ๆ เพื่อให้หุ่นยนต์กู้ภัยหลบหลีกสิ่งกีดขวางและค้นหาเป้าหมายอัตโนมัติค้นหาผู้ประสบภัยและระบุตำแหน่งของเป้าหมายได้อย่างมีประสิทธิภาพ โดยประกอบไปด้วยหลักการ ทั้งหลักการสำหรับส่วนโครงสร้างหุ่นยนต์ หลักการควบคุมและหลักการตรวจจับวัตถุเป้าหมาย และอีกส่วนคือ การประมวลผลภาพจากกล้องวีดีโอ รวมถึงการใช้หลักของปัญญาประดิษฐ์ เพื่อประมวลผลเกี่ยวกับความน่าจะเป็นของผู้ประสบภัย เช่น สภาพแวดล้อมที่ผู้ประสบภัยติดอยู่ ทำให้ผู้ช่วยเหลือสามารถประมาณเวลาในการเข้าช่วยเหลือผู้ประสบภัยได้อย่างรวดเร็ว

2.1 โครงสร้างหุ่นยนต์และการควบคุมหุ่นยนต์

การสร้างหุ่นยนต์จะต้องมีส่วนประกอบที่สำคัญ คือ การส่งกำลังจากตัวส่งกำลังไปยังตัวรับกำลังซึ่งในโครงงานนี้ใช้หลักการของมอเตอร์กระแสตรงเพื่อเป็นต้นกำลังไปยังตัวรับกำลังเพื่อขับเคลื่อนหุ่นยนต์ ดังนั้นตัวส่งกำลังและตัวรับกำลังจึงมีความสำคัญในการสร้างโครงสร้างหุ่นยนต์ ซึ่งหัวข้อนี้จะอธิบายหลักการทำงานของตัวรับกำลัง และการพิจารณาเลือกใช้ให้เหมาะสมกับหุ่นยนต์ที่ออกแบบ เพื่อส่งผลหุ่นยนต์ประสิทธิภาพสูงสุด

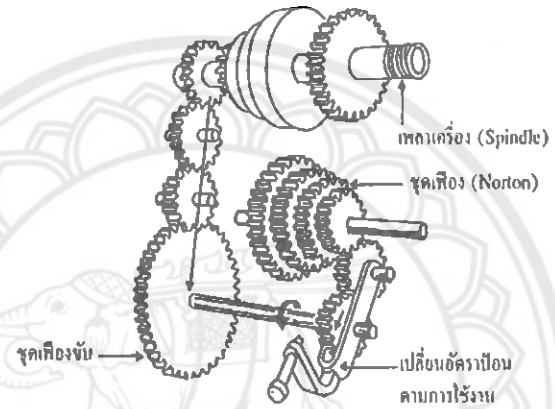
2.1.1 เฟือง (Gear)

เป็นชิ้นส่วนสำคัญในการถ่ายทอดกำลังในการเคลื่อนที่ของหุ่นยนต์ โดยจะแบ่งเป็นหลายประเภท คือเฟืองขบ เฟืองตรง เฟืองโซ่ และเฟืองสะพาน เป็นต้นซึ่งเฟืองขบก็จัดอยู่ในเฟืองตรงเช่นเดียวกัน แต่เฟืองขบก็ยังมีอีกหลายลักษณะ เช่น เฟืองเฉียง เฟืองวงแหวน เฟืองเฉียงก้างปลา และเฟืองหนอน ซึ่งจะอธิบายหน้าที่และลักษณะการทำงานดังต่อไปนี้



รูปที่ 2.1 ลักษณะของเฟืองตรงชนิดขบ

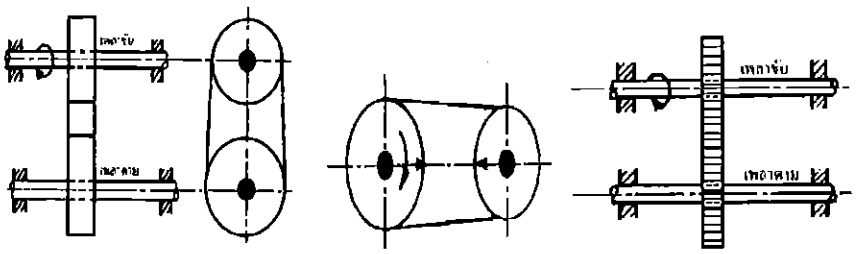
หน้าที่การใช้งานของเฟืองตรงเป็นเฟืองที่ใช้ส่งกำลังกับเพลาที่ขนานกัน เฟืองตรงเหมาะสำหรับการส่งกำลังที่มีความเร็วรอบต่ำหรือความเร็วรอบปานกลางไม่เกิน 20 เมตรต่อนาที เช่น ชุดเฟืองทดลองของเครื่องกลึงเพื่อเดินกลึงอัตโนมัติ หรือชุดเฟืองทดลองของเครื่องจักรการเกษตรที่ความเร็วรอบต่ำๆ ข้อดีของเฟืองตรงขณะใช้งานจะไม่เกินแรงในแนวแกน ประสิทธิภาพในการทำงานสูง หน้ากว้างของเฟืองตรงสามารถเพิ่มได้เพื่อให้เกิดผิวสัมผัสที่มากขึ้น เพื่อลดการสึกหรอให้น้อยลง ข้อเสียของเฟืองตรงขณะใช้งาน คือ ขณะที่เฟืองหมุนจะทำให้เกิดเสียงดังและอาจเกิดความร้อนสูง จึงควรเลือกใช้ให้เหมาะสมกับการใช้งาน ตัวอย่างการใช้งานของเฟืองดังรูปที่ 2.2



รูปที่ 2.2 ลักษณะการส่งกำลังของเฟืองตรง

2.1.2 เพลา (Shaft)

เพลาเป็นชิ้นส่วนเครื่องจักรกลที่หมุนได้ เพลาจะรับโมเมนต์บิดที่ถ่ายภาระมาจากเฟืองล้อสายพาน หรือคัตช์ เพลาจึงสามารถรับภาระบิดและภาระดัด จึงมีการแบ่งเพลาออกเป็น 2 อย่าง คือ เพลาส่งกำลัง และเพลารองรับภาระ เพลาส่งกำลัง (Transmission Shafts) เพลาชนิดนี้ใช้เฉพาะการบิดหรืออาจรับทั้งการบิดและการดัดผสมกันก็ได้ การส่งกำลังจะถ่ายทอดผ่านเพลาโดยอาศัยแผ่นประกบต่อเพลา (Coupling) ผ่านเฟืองผ่านพูลเลย์ผ่านสายพาน จานโซ่ หรือเฟืองโซ่เพลาชนิดนี้มีหน้าที่ส่งถ่ายกำลังจากจุดหนึ่งไปยังอีกจุดหนึ่งโดยผ่านชิ้นส่วนอื่นต่างๆดังนี้ เช่น ส่งถ่ายกำลังมาจากเฟือง มาจากพูลเลย์ มาจากคลัตช์ ซึ่งชิ้นส่วนต่างๆ เหล่านี้ได้ต้นกำลังมาจากมอเตอร์ หรือเครื่องยนต์ ส่วนเพลารองรับภาระ เป็นเพลาชิ้นส่วนเครื่องจักรกลเช่นกัน ขณะใช้งานเพลาชนิดนี้อาจหมุนหรือไม่ก็ได้ แต่ที่สำคัญเพลาชนิดนี้ไม่ได้ส่งกำลังจะทำหน้าที่เป็นตัวรองรับชิ้นส่วนอื่นให้หมุน เช่น เพลาลูกกรอกสายพาน เพลาลูกล้อสลิงต่างๆ ซึ่งเป็นเพลาที่รับภาระน้ำหนักของอุปกรณ์อื่นที่กดทับทำให้สภาพการเสียหายของเพลาเกิดการดัดงอเป็นส่วนใหญ่ เช่น เพลา ล้อรถไฟ เป็นต้น จึงควรเลือกขนาดของเพลาและน้ำหนักของเพลาให้เหมาะสมในการนำมาเป็นเพลาของหุ่นยนต์



รูปที่ 2.3 เพลารองรับภาระ

2.1.3 การเชื่อมโลหะ

กระบวนการเชื่อมไฟฟ้าด้วยลวดเชื่อม คือกระบวนการเชื่อมที่อาศัยความร้อนจากการอาร์คระหว่าง ลวดเชื่อมโลหะมีสารพอกหุ้มกับชิ้นงาน ทำให้ลวดเชื่อมและชิ้นงานบริเวณการอาร์คหลอมละลายรวมตัวกันเป็นแนวเชื่อม และสารพอกหุ้มจะเกิดเป็นก๊าซ และสแล็ค ปกคลุมแนวเชื่อมจากบรรยากาศภายนอก

กฎความปลอดภัยทั่วไปในการเชื่อม

การเชื่อมจะไม่เกิดอันตรายใดๆ ถ้าผู้เชื่อมได้มีการป้องกันและระมัดระวังอยู่ตลอด จนสร้างเป็นนิสัยแห่งความปลอดภัยขึ้น ผู้ปฏิบัติงานควรปฏิบัติตามกฎแห่งความปลอดภัยต่างๆ ดังต่อไปนี้

1. ชุดปฏิบัติงานจะต้องแห้ง ติดไฟยาก รองเท้าต้องมีที่กำบังเม็ดโลหะ ถุงมือหนัง เสื้อหนัง และกระจกหน้ากากที่มีความเข้มพอเหมาะกับงาน
2. บริเวณที่ทำการเชื่อมไม่มีวัสดุติดไฟง่าย วัสดุที่จะเกิดการระเบิดได้ เช่น คาร์บอนเทตตราคลอดไรด์
3. บริเวณที่ทำการเชื่อมจะต้องมีการถ่ายเทอากาศอย่างเพียงพอ
4. เครื่องมือ และอุปกรณ์ในการเชื่อมจะต้องอยู่ในสภาพดีเสมอ
5. ถ้าจำเป็นต้องเชื่อมบนพื้นที่เปียกชื้น ต้องสวมรองเท้ายาง หรือยืนบนแผ่นไม้ที่แห้ง

2.2 หลักการควบคุมหุ่นยนต์

หุ่นยนต์อัตโนมัติ (Autonomous robots) คือ หุ่นยนต์ที่สามารถทำงานตามคำสั่งที่กำหนดได้ภายใต้สภาพแวดล้อมที่ไม่แน่นอน โดยปราศจากการควบคุมจากมนุษย์ การควบคุมหุ่นยนต์จึงแตกต่างจากหุ่นยนต์ที่มีผู้ควบคุมดังนั้นการออกแบบระบบควบคุมจึงมีความแตกต่างทั้งโครงสร้างหุ่นยนต์และระบบควบคุม โดยจะใช้ระบบที่ประมวลผลต่างๆได้อย่างรวดเร็ว เพื่อให้สามารถทำงานได้ใกล้เคียงกับมนุษย์ จึงจำเป็นต้องเลือกอุปกรณ์อย่างเหมาะสมในการประมวลผลหุ่นยนต์อัตโนมัติ ซึ่งจากหุ่นยนต์เคลื่อนที่หลบหลีกสิ่งกีดขวางและค้นหาเป้าหมายอัตโนมัติจึงจะต้อง

พิจารณาอุปกรณ์ให้มีความเหมาะสมโดยจะอธิบายในส่วนโครงสร้างในการขับเคลื่อนหุ่นยนต์และการควบคุมหุ่นยนต์ดังต่อไปนี้

2.2.1 โครงสร้างในการขับเคลื่อนหุ่นยนต์

หุ่นยนต์กู้ภัยต้องเคลื่อนที่ไปสถานที่ที่มีสิ่งกีดขวางและเป็นเส้นทางที่ยากต่อการเคลื่อนที่ ดังนั้นหุ่นยนต์กู้ภัยจึงจัดอยู่ในประเภทของหุ่นยนต์แบบเคลื่อนที่ได้ ใช้โซ่และตีนตะขาบหรือพลูเลย์ กับสายพานเพื่อเป็นตัวขับเคลื่อนให้หุ่นยนต์เคลื่อนที่ไปในพื้นที่ขรุขระได้ สามารถขึ้นทางที่มีความชัน จึงต้องใช้ต้นกำลังที่มีน้ำหนักเบาเช่น มอเตอร์ไฟฟ้ากระแสตรง แทนที่จะใช้เครื่องยนต์ซึ่งน้ำหนัก ขนาดอุปกรณ์และพื้นที่ในการติดตั้งมากดังนั้นการติดตั้งและการควบคุมของหุ่นยนต์ตู้ภัยจึงเลือกที่ใช้มอเตอร์กระแสตรงโดยมีหลักการควบคุมดังต่อไปนี้

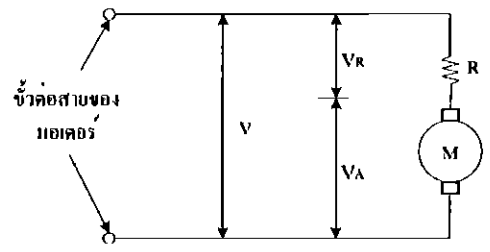
หลักการควบคุมมอเตอร์กระแสตรง

เมื่อป้อนกระแสไฟฟ้าไหลผ่านเข้าไปยังขดลวดสนามแม่เหล็ก จะทำให้เกิดสนามแม่เหล็กซึ่งมีสัดส่วนของสนามแม่เหล็กขึ้นอยู่กับกระแสสนามแม่เหล็ก โดยแรงจะเกิดขึ้นเป็นมุมตั้งฉากกับกระแสและสนามแม่เหล็ก ขณะที่ทิศทางของแรงกลับตรงข้ามกัน ถ้าหากกระแสของสนามแม่เหล็กไหลย้อนกลับจะทำให้เกิดการเปลี่ยนแปลงของกระแส และสนามแม่เหล็กเป็นผลให้ทิศทางของแรงเปลี่ยนไป ด้วยคุณสมบัตินี้ทำให้มอเตอร์กระแสตรงกลับทิศทางการหมุนได้

สนามแม่เหล็กของมอเตอร์ส่วนหนึ่งเกิดขึ้นจากสนามแม่เหล็กถาวรซึ่ง จะถูกยึดกับแผ่นเหล็ก หรือเหล็กกล้า โดยปกติส่วนนี้จะเป็นส่วนที่ยึดอยู่กับที่ และขดลวดเหนี่ยวนำจะพันอยู่กับส่วนที่เป็นแกนหมุนของมอเตอร์

คุณสมบัติของมอเตอร์กระแสตรง

ในการอธิบายคุณสมบัติของมอเตอร์กระแสตรงให้ละเอียดนั้นต้องพิจารณาแรงดันที่ป้อนและความต้านทานของโรเตอร์ด้วย วงจรภายในของมอเตอร์เขียนได้ดังรูปที่ 2.4



รูปที่ 2.4 วงจรภายในมอเตอร์กระแสตรง

โดยสมมุติให้ทุ่นโรเตอร์ไม่มีความต้านทานอยู่เลย อนุกรมกับความต้านทานซึ่งในที่นี้ก็คือความต้านทานของขดลวดนั้นเอง แรงดันที่ขั้วต่อสายของมอเตอร์ก็ คือ ผลบวกระหว่างแรงดันที่ทุ่นโรเตอร์ (V_A) และแรงดันตกคร่อมความต้านทานขดลวด (V_R)

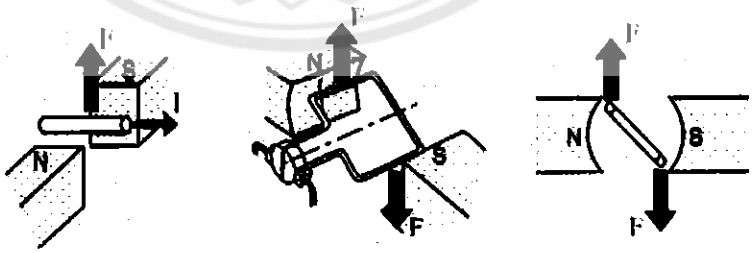
แรงดัน V_A ถูกเรียกว่า แรงเคลื่อนเหนี่ยวนำป้อนกลับ (Back EMF) ซึ่งเกิดขึ้นในโรเตอร์ขณะหมุน แรงดันที่เกิดขึ้นเป็นไปตามกฎเหนี่ยวนำแม่เหล็กไฟฟ้าจากการเคลื่อนที่ของตัวนำไฟฟ้าในสนามแม่เหล็ก สัมพันธ์กับแรงเคลื่อนเหนี่ยวนำแม่เหล็กและความเร็วในการเคลื่อนที่ของตัวนำ แรงดันที่เกิดขึ้นจะมีขั้วตรงข้ามกับแรงดันที่ป้อนมอเตอร์และแปรผันตรงกับความเร็วในการหมุน ผลบวกของแรงดันที่ทุ่นโรเตอร์ (V_A) และแรงดันตกคร่อมขดลวด (V_R) ต้องเท่ากับแรงดันที่ป้อนให้กับมอเตอร์

$$V = V_A + V_R \tag{2.1}$$

เมื่อพิจารณาตั้งแต่มอเตอร์หยุดนิ่ง ความเร็วมีค่าเป็นศูนย์ดังนั้น $V_A = 0$, $V_R = V$ กระแสที่ไหลในมอเตอร์หาได้จาก

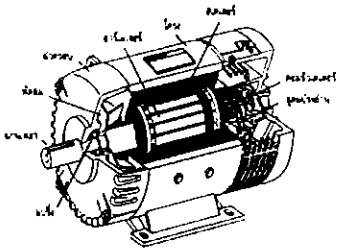
$$I = V_R / R \tag{2.2}$$

เมื่อมอเตอร์เริ่มหมุนจะมีความเร็ว และ V_A เพิ่มขึ้นเป็นเส้นตรงตามความเร็ว V_R ซึ่งมีค่าเท่ากับความแตกต่างระหว่าง V_A และ V จะเริ่มลดลง กระแส I ก็เริ่มลดลงเช่นกันขณะที่มอเตอร์ยังมีความเร่งอยู่ ความเร็วจะเพิ่มขึ้น แรงบิดจะลดลงจนกว่าจะถึงจุดซึ่งแรงบิดของมอเตอร์จะรับภาระโหลดได้พอดี ขณะที่มอเตอร์ไม่มีโหลด และหมุนอย่างอิสระจะมีเพียงค่าความฝืดของแบริ่ง และแรงต้านอากาศทำให้ V_A เกือบเท่า V



รูปที่ 2.5 แสดงการทำงานของมอเตอร์กระแสตรง

ส่วนประกอบของมอเตอร์ไฟฟ้ากระแสตรง



รูปที่ 2.6 แสดงส่วนประกอบของมอเตอร์กระแสตรง

ขดลวดสนามแม่เหล็ก (Field Coil) คือ ขดลวดที่ถูกพันอยู่กับขั้วแม่เหล็กที่ยึดติดกับโครงมอเตอร์ ทำหน้าที่กำเนิดขั้วแม่เหล็กขั้วเหนือ (N) และขั้วใต้ (S) แทนแม่เหล็กถาวร ขดลวดที่ใช้เป็นขดลวดอาบน้ำยาฉนวน สนามแม่เหล็กจะเกิดขึ้นเมื่อจ่ายแรงดันไฟตรงให้มอเตอร์

ขั้วแม่เหล็ก (Pole Pieces) คือ แกนสำหรับรองรับขดลวดสนามแม่เหล็ก ถูกยึดติดกับโครงมอเตอร์ด้านใน ขั้วแม่เหล็กทำมาจากแผ่นเหล็กอ่อนบางๆ อัดซ้อนกัน (Lamination Sheet Steel) เพื่อลดการเกิดกระแสไหลวน (Edy Current) ที่จะทำให้ความเข้มของสนามแม่เหล็กลดลง ขั้วแม่เหล็กทำหน้าที่ให้กำเนิดขั้วสนามแม่เหล็กมีความเข้มสูงสุดแทนขั้วสนามแม่เหล็กถาวร

โครงมอเตอร์ (Motor Frame) คือ ส่วนเปลือกหุ้มภายนอกของมอเตอร์ และยึดให้ส่วนที่อยู่กับที่ (Stator) ของมอเตอร์ไว้ภายในร่วมกับฝาปิดหัวท้ายของมอเตอร์ โครงมอเตอร์ทำหน้าที่เป็นทางเดินของเส้นแรงแม่เหล็กระหว่างขั้วแม่เหล็กให้เกิดสนามแม่เหล็กครบวงจร

อาร์เมเจอร์ (Armature) คือ ส่วนเคลื่อนที่ (Rotor) ถูกยึดติดกับเพลา (Shaft) และรองรับการหมุนด้วยที่รองรับการหมุน (Bearing) ตัวอาร์เมเจอร์ทำจากเหล็กแผ่นบางๆ อัดซ้อนกัน ถูกเซาะร่องออกเป็นส่วนๆ เพื่อไว้พันขดลวดอาร์เมเจอร์ (Armature Winding) ขดลวดอาร์เมเจอร์เป็นขดลวดอาบน้ำยาฉนวน ปลายขดลวดอาร์เมเจอร์ต่อไว้กับคอมมิวเตเตอร์ แรงผลักดันของสนามแม่เหล็กทั้งสอง ทำให้อาร์เมเจอร์หมุนเคลื่อนที่

คอมมิวเตเตอร์ (Commentator) คือ ส่วนเคลื่อนที่อีกส่วนหนึ่ง ถูกยึดติดเข้ากับอาร์เมเจอร์และเพลาร่วมกัน คอมมิวเตเตอร์ทำจากแท่งทองแดงแข็งประกอบเข้าด้วยกันเป็นรูปทรงกระบอก แต่ละแท่งทองแดงของคอมมิวเตเตอร์ถูกแยกออกจากกันด้วยฉนวนไมก้า (Mica) อาร์เมเจอร์ คอมมิวเตเตอร์ทำหน้าที่เป็นขั้วรับแรงดันไฟตรงที่จ่ายมาจากแปรงถ่าน เพื่อส่งไปให้ขดลวดอาร์เมเจอร์

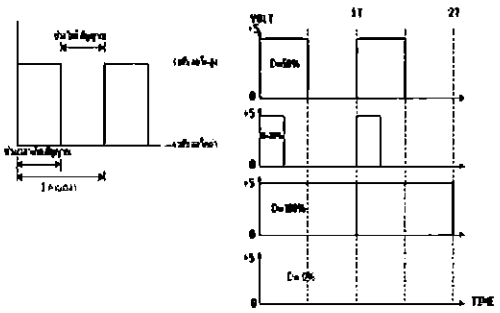
แปรงถ่าน (Brush) คือ ตัวสัมผัสกับคอมมิวเตเตอร์ ทำเป็นแท่งสี่เหลี่ยมผลิตมาจากคาร์บอนหรือแกรไฟต์ผสมผงทองแดง เพื่อให้แข็งและนำไฟฟ้าได้ดี มีสายตัวนำต่อร่วมกับแปรงถ่านเพื่อไปรับแรงดันไฟตรงที่จ่ายเข้ามา แปรงถ่านทำหน้าที่รับแรงดันไฟตรงจากแหล่งจ่ายแล้วจ่ายผ่านไปให้คอมมิวเตเตอร์

การควบคุมความเร็วมอเตอร์

การควบคุมด้วยตัวต้านทานที่ปรับค่าได้ เป็นรูปแบบพื้นฐานที่สุดของการควบคุมมอเตอร์ คือ ใช้ตัวตานทานปรับค่าได้อนุรมกับมอเตอร์โดยตัวต้านทานที่รับค่าได้จะเป็นตัวกำหนดความเร็วในการหมุนของมอเตอร์ ซึ่งการบังคับแบบนี้ไม่มีประสิทธิภาพเพราะกำลังไฟจะสูญเสียไปในตัวต้านทาน มักนิยมใช้กับมอเตอร์ตัวเล็ก การบังคับแบบนี้ให้คุณสมบัติในการสตาร์ทดี ให้แรงบิดสูงที่ความเร็วต่ำแต่จะให้ความเร็วสูงมากเมื่อมอเตอร์อยู่ในภาวะมีโหลดน้อย ดังนั้นการบังคับแบบนี้มีประโยชน์เฉพาะภาวะแรงดันคงที่ เช่น การบังคับความเร็วเครื่องจักรเย็บผ้า

การควบคุมด้วยวิธีเปลี่ยนค่าแรงดัน วิธีการนี้ดีกว่าวิธีการแรกแต่จะซับซ้อนกว่าต้องใช้อุปกรณ์อิเล็กทรอนิกส์ที่มีอัตราขยายกำลังสูง มอเตอร์จะถูกป้อนด้วยแรงดันที่เปลี่ยนแปลงค่าได้จากแหล่งจ่ายที่มีอิมพิแดนซ์ต่ำ ข้อดีของการควบคุมวิธีนี้คือถ้าความเร็วลดลงจากผลของแรงบิดแรงดันที่ป้อนให้กับมอเตอร์จะเพิ่มขึ้นเพื่อรักษาระดับความเร็ว ส่วนข้อเสียจากการควบคุมวิธีนี้คือเมื่อมอเตอร์มีความเร็วต่ำแรงดันที่ป้อนให้กับมอเตอร์จะมีค่าต่ำเช่นกัน

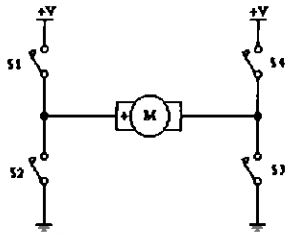
การควบคุมแบบ PWM (Pulse Width Modulation) การมอดูเลชันทางความกว้างของพัลส์ PWM (Pulse Width Modulation) จะเป็นการปรับเปลี่ยนที่สัดส่วน และความกว้างของสัญญาณพัลส์ โดยความถี่ของสัญญาณพัลส์จะไม่มีการเปลี่ยนแปลง หรือเป็นการเปลี่ยนแปลงที่ค่าของดิวตี้ไซเกิล (duty cycle) นั้นเอง ซึ่งค่าของดิวตี้ไซเคิล คือช่วงความกว้างของพัลส์ที่มีสถานะลอจิกสูงโดยคิดสัดส่วนเป็นเปอร์เซ็นต์จากความกว้างของพัลส์ทั้งหมด ยกตัวอย่างเช่น ถ้าหากค่าดิวตี้ไซเคิลมีค่าเท่ากับเท่ากับ 50% ก็หมายถึงใน 1 รูปสัญญาณพัลส์จะมีช่วงของสัญญาณที่เป็นสถานะลอจิกสูงอยู่ครึ่งหนึ่ง และสถานะลอจิกต่ำอยู่อีกครึ่งหนึ่ง และในทำนองเดียวกันถ้าหากค่าดิวตี้ไซเคิลมีค่ามาก หมายความว่าความกว้างของพัลส์ที่เป็นสถานะลอจิกสูงจะมีความกว้างมากขึ้น หากค่าดิวตี้ไซเคิลมีค่าเท่ากับ 100% ก็หมายความว่าจะไม่มีสถานะลอจิกต่ำเลย ซึ่งค่าดิวตี้ไซเคิลสามารถ จะหาได้จากค่าความสัมพันธ์ดังรูปที่ 2.7



รูปที่ 2.7 วงจรควบคุมความเร็วของมอเตอร์กระแสตรงโดย PWM

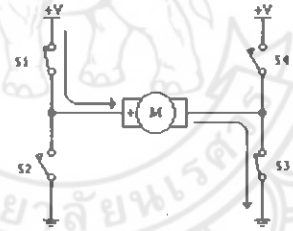
การควบคุมทิศทางการหมุนของมอเตอร์

หลักการของวงจร H-bridge switching จะประกอบไปด้วยสวิทช์ 4 ตัว คือ S1, S2, S3, S4 ซึ่งในรูป จะใช้ DC Motor เป็นโหลดของวงจร ในสภาวะเริ่มต้น สวิทช์ทุกตัว off อยู่ก็จะไม่มีอะไรเกิดขึ้น



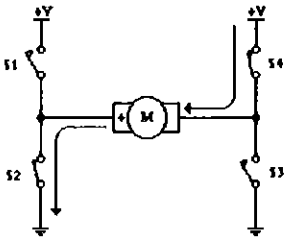
รูปที่ 2.8 ภาพแสดง H-Bridge Switching

เมื่อทำการ ON สวิทช์ S1 และ S3 พร้อมกัน จะเป็นการเชื่อมวงจร ทำให้มีกระแสไฟฟ้าไหลผ่านมอเตอร์จากขั้วบวกของมอเตอร์ไปยังขั้วลบของมอเตอร์ จึงทำให้มอเตอร์สามารถหมุนได้ในทิศทาง forward (จะหมุนแบบตามเข็มนาฬิกาหรือทวนเข็มนาฬิกานั้นขึ้นอยู่กับลักษณะของการพันขดลวดภายในมอเตอร์



รูปที่ 2.9 ภาพแสดง H-Bridge Switching เมื่อ S1 และ S3 On พร้อมกัน

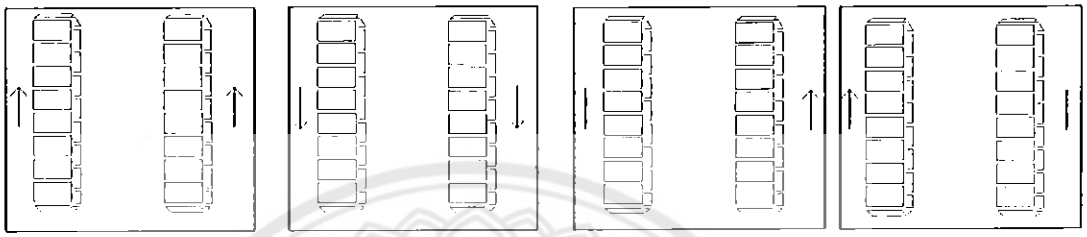
ในทางกลับกัน หากทำการ ON สวิทช์ S2 และ S4 พร้อมกัน จะเป็นการเชื่อมวงจรและทำให้เกิดกระแสไฟฟ้าไหลผ่านมอเตอร์ จากขั้วลบของมอเตอร์ไปยังขั้วบวกของมอเตอร์ จึงทำให้มอเตอร์สามารถหมุนได้และเป็นการหมุนในทิศทาง Backward (กลับทิศทางกับกรณีแรก)



รูปที่ 2.10 ภาพแสดงH-Bridge Switching เมื่อ S2 และ S4 On พร้อมกัน

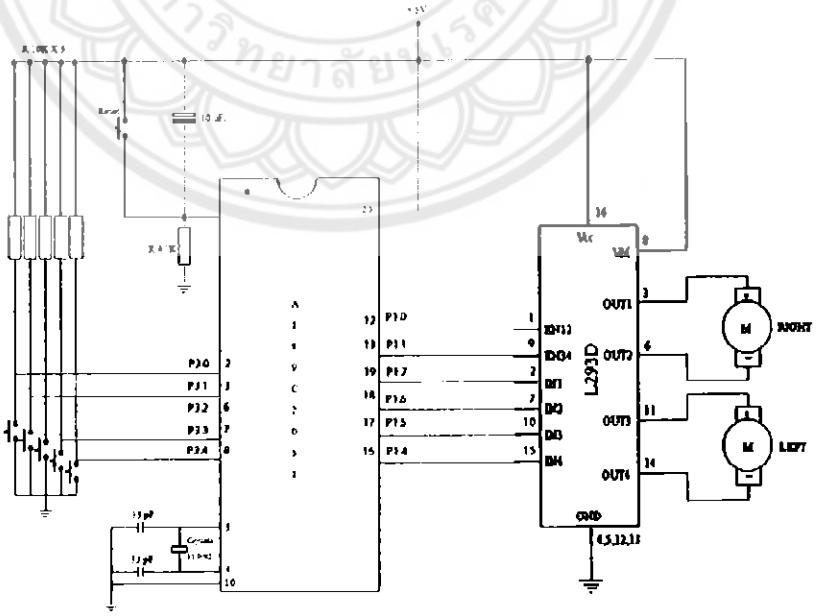
2.2.2 โครงสร้างอุปกรณ์ควบคุมการเคลื่อนที่ของหุ่นยนต์

จากที่ได้กล่าวถึงโครงสร้างของหุ่นยนต์ในหัวข้อที่ผ่านมา การที่จะส่งกำลังของไปที่เฟืองและเพลาของล้อตีนตะขาบของหุ่นยนต์ เพื่อให้หุ่นยนต์สามารถเคลื่อนที่ได้นั้น ถ้าต้องการให้หุ่นยนต์เคลื่อนที่เลี้ยวซ้ายหรือเลี้ยวขวา จะต้องหยุดล้อข้างได้ข้างหนึ่งหรือให้ล้อทั้งสองข้างหมุนตรงข้ามกันโดยต้องมีการส่งคำสั่งไปควบคุมการจ่ายแรงดันและกระแสให้มอเตอร์



รูปที่ 2.11 แสดงลักษณะการสั่งงานมอเตอร์ให้ล้อตีนตะขาบเคลื่อนที่เดินหน้า ถอยหลัง เลี้ยวซ้าย เลี้ยวขวา ตามลำดับจากซ้ายไปขวา

จากหลักการในหัวข้อการกำหนดทิศทางการหมุนของมอเตอร์นั้น จำเป็นที่จะต้องมีส่วนควบคุมที่สำคัญคือ ไมโครคอนโทรลเลอร์ เพื่อทำหน้าที่ในการสั่งการจ่ายกระแสให้กับมอเตอร์จากวงจรควบคุมการหมุนโดยจะแสดงรายละเอียดของไมโครคอนโทรลเลอร์ดังต่อไปนี้



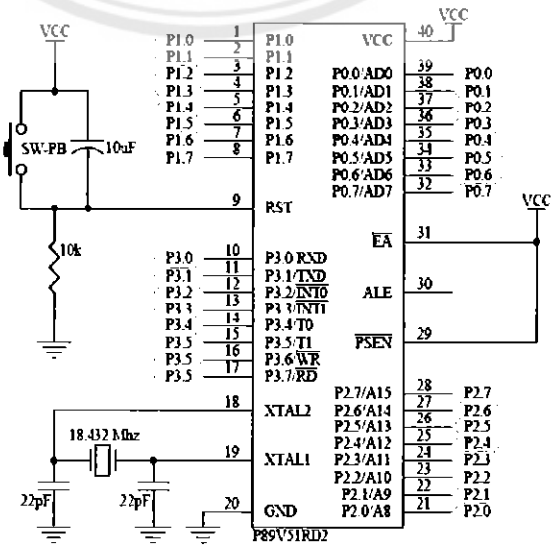
รูปที่ 2.12 แสดงโครงสร้างเบื้องต้นการควบคุมทิศทางการหมุนของมอเตอร์จากไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์เป็นอุปกรณ์อิเล็กทรอนิกส์ที่ทำหน้าที่ประมวลผลภายใน ไมโครคอนโทรลเลอร์ MCS-51 เป็นไมโครคอนโทรลเลอร์ที่ใช้งานได้ง่ายโดยมีโครงสร้างดังต่อไปนี้

ไมโครคอนโทรลเลอร์ MCS-51 (เบอร์ P89V51RD2)

ไมโครคอนโทรลเลอร์ตระกูล MCS-51 เริ่มแรกได้ถูกพัฒนาขึ้นจากบริษัท อินเทล (Intel Corporation) และได้มีการนำไปใช้งานกันอย่างแพร่หลายตั้งแต่ปี 1980 ในช่วงเวลาที่ผ่านมาได้มีบริษัทผู้ผลิตหลายบริษัท เช่น Dallas, Philips, Atmel ได้รับลิขสิทธิ์ในการผลิต และจำหน่าย จากบริษัท อินเทล และบริษัทต่าง ๆ ก็ได้พัฒนาความสามารถของไมโครคอนโทรลเลอร์ MCS-51 รุ่นใหม่ ๆ ให้มีความสามารถ และมีความเร็วเพิ่มขึ้น แต่ยังคงโครงสร้างพื้นฐานที่สำคัญของไมโครคอนโทรลเลอร์ตระกูล 8051 ซึ่งมีรายละเอียดดังนี้

1. เป็นไมโครคอนโทรลเลอร์ที่มีหน่วยประมวลผลกลางแบบ 8 บิต
2. มีคำสั่งคำนวณทางคณิตศาสตร์ และตรรกศาสตร์ (Boolean processor)
3. มีแอดเดรสบัสขนาด 16 บิตทำให้สามารถอ้างตำแหน่งหน่วยความจำโปรแกรม และหน่วยความจำข้อมูลได้ 64 กิโลไบต์
4. มีหน่วยความจำ (RAM) ภายในขนาด 256 ไบต์ (8052/8032)
5. มีพอร์ตอนุกรมทำงานแบบดูเพล็กซ์เต็ม (Full Duplex) 1 พอร์ต
6. มีพอร์ตอินพุต/เอาต์พุตแบบขนานจำนวน 32 บิต
7. มีไทเมอร์ 2 ตัว (8051/8031) หรือ 3 ตัว (8052/8032)
8. มีวงจรควบคุมการเกิดอินเตอร์รัปต์ 6 ประเภท (8052/8032)
9. มีวงจรออสซิลเลเตอร์ภายในตัว



รูปที่ 2.13 วงจรไมโครคอนโทรลเลอร์ P89V51RD2

ไมโครคอนโทรลเลอร์ MCS-51 มีวงจรออสซิลเลเตอร์อยู่ภายใน ดังนั้นในการใช้งานจึงสามารถต่อคริสตอล และตัวเก็บประจุเข้ากับคริสตอลได้โดยตรง โดยความถี่ของคริสตอลที่ต่อเข้ากับไมโครคอนโทรลเลอร์จะเป็นตัวระบุความเร็วในการทำงานโดยตรง ในไมโครคอนโทรลเลอร์ MCS-51 ปกติ 1 แมชชีนไซเคิล (Machine Cycle) จะใช้สัญญาณนาฬิกาจำนวน 12 ลูก และในการทำงานแต่ละคำสั่งไมโครคอนโทรลเลอร์จะใช้เวลาในการทำงาน 1 - 4 แมชชีนไซเคิล ขึ้นอยู่กับความซับซ้อนของคำสั่งนั้น

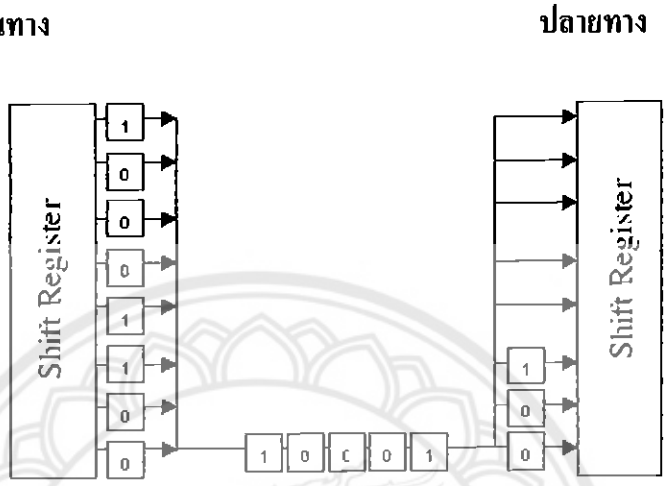
ในปัจจุบันผู้ผลิตได้พัฒนาให้ไมโครคอนโทรลเลอร์สามารถทำงานได้เร็วขึ้น โดยเพิ่มความสามารถในการรองรับคริสตอลความถี่ที่สูงขึ้น รวมไปถึงการปรับปรุงการทำงานภายใน ให้ไมโครคอนโทรลเลอร์ใช้จำนวนสัญญาณนาฬิกาในการสร้างแมชชีนไซเคิลน้อยลง โดยในบางรุ่น 1 แมชชีนไซเคิลใช้สัญญาณนาฬิกาเพียงแค่ 1 ลูกเท่านั้น

สำหรับไมโครคอนโทรลเลอร์ MCS-51 ที่เราจะมาลองเล่นกันนั้นเป็นรุ่น P89V51RD2 ของบริษัท Philips ที่เลือกรุ่นนี้เนื่องจากเป็นรุ่นที่สามารถรองรับการดาวน์โหลดโปรแกรมแบบ ISP (In System Programming) ผ่านพอร์ตอนุกรมได้โดยตรง ไม่ต้องอาศัยอุปกรณ์ หรือวงจรเพิ่มเติมในการดาวน์โหลดโปรแกรม จึงทำให้สามารถใช้งานได้อย่างสะดวก รวมถึงราคาของ P89V51RD2 ที่ไม่แพง เมื่อเทียบกับความสามารถ และประสิทธิภาพของมัน P89V51RD2 สามารถทำงานในโหมด X2 ซึ่ง จะทำให้สามารถทำงานได้เร็วกว่า MCS-51 พื้นฐาน 2 เท่า (1 แมชชีนไซเคิล ใช้สัญญาณนาฬิกา 6 ลูก) เมื่อใช้คริสตอลความถี่ที่เท่ากัน ในการทำงานในโหมด X2 นี้ P89V51RD2 สามารถใช้คริสตอลความถี่สูงสุด 20MHz ส่วนในการทำงานในโหมด X1 สามารถใช้คริสตอลความถี่สูงสุด 40 MHz ภายใน P89V51RD2 มีหน่วยความจำโปรแกรมแบบแฟลชขนาด 64 กิโลไบต์ นอกจากนั้นยังมี หน่วยความจำข้อมูลภายนอกเพิ่มเติมขนาด 1 กิโลไบต์ อยู่ภายในตัวชิพด้วย

การสื่อสารแบบอนุกรม

การส่งข้อมูลแบบขนานนั้น สายส่งข้อมูลแบบขนานที่ใช้สำหรับเชื่อมต่อระหว่างพอร์ต I/O กับอุปกรณ์ภายนอกนอกจากจะมีความยาวได้เพียง 1 หรือ 2 เมตรเท่านั้น ทั้งนี้เนื่องจากค่าคาพาซิแดนซ์ในสายจะจำกัดระยะทางในการส่งข้อมูล แต่ถ้าเราต้องการให้สามารถส่งข้อมูลได้ในระยะทางไกลขึ้น เราก็ต้องนำวงจรขับพิเศษมาใช้ การส่งข้อมูลแบบขนานจะต้องส่งสัญญาณจำนวน 1 เส้นสำหรับข้อมูลในแต่ละบิต ซึ่งทำให้การโอนย้ายข้อมูลแบบขนาน 1 ไบต์มีราคาสูงกว่าการโอนย้ายข้อมูลแบบอนุกรมถึง 8 เท่า เช่นเดียวกันด้วยเหตุผลด้านราคาและความไม่สะดวกที่พบในการโอนย้ายข้อมูลแบบขนาน จึงได้มีอุปกรณ์หลายชนิดที่ใช้ในการสื่อสารแบบอนุกรม ทั้งๆที่ไมโครโปรเซสเซอร์ในเครื่องไมโครคอมพิวเตอร์จะยังคงใช้การโอนย้ายข้อมูลแบบขนาน เมื่อเราศึกษาการโอนย้ายข้อมูลแบบอนุกรม เราก็จะต้องเรียนรู้โพรโตคอลที่เกี่ยวกับการส่งข้อมูลแบบอนุกรม ซึ่งแบ่งออกได้เป็น 3 อย่าง อย่างแรกคือ วิธีการแปลงข้อมูลแบบขนานเป็นแบบอนุกรม

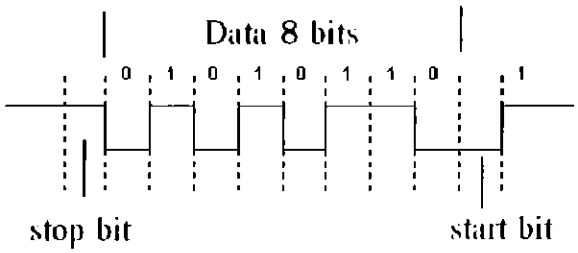
และการแปลงข้อมูลแบบอนุกรมเป็นข้อมูลแบบขนาน อย่างที่สองคือ ชนิดของวงจรและรูปแบบสัญญาณที่ใช้ในการส่งข้อมูลในระยะไกล อย่างที่สามคือรูปแบบของข้อมูลที่ส่งไปและการควบคุมการโอนย้ายข้อมูล



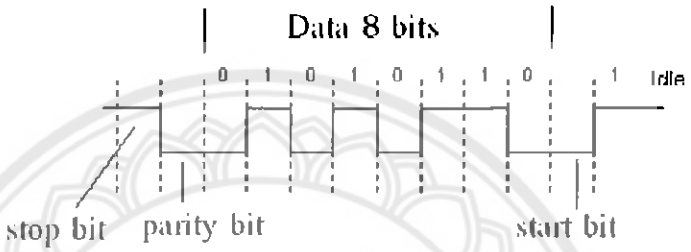
รูปที่ 2.14 รูปแบบการส่งข้อมูลแบบอนุกรม

การเชื่อมต่อแบบอนุกรม UART

การแปลงข้อมูลแบบขนานเป็นข้อมูลแบบอนุกรม โดยเริ่มแรกข้อมูลแบบขนานจะถูกนำไปเก็บไว้ในรีจิสเตอร์ที่เลื่อนค่าได้ (Shift register) จากนั้นเราจะใช้สัญญาณนาฬิกาในการเลื่อนค่าในรีจิสเตอร์ออกมาทีละบิต (โดยจะเลื่อนค่าไปทางขวามือ) โดยบิตแรกที่ถูกเลื่อนออกมาคือ บิต LSB ของข้อมูลและบิตที่สองที่ถูกเลื่อนออกมาก็คือ บิตที่อยู่ถัดไปจากบิต LSB และบิตต่อๆไป สำหรับบิตสุดท้ายที่ถูกเลื่อนออกมาก็คือ บิต MSB ของข้อมูล เมื่อเรานำบิตที่ 8 ของข้อมูลมาใช้ในการตรวจสอบความผิดพลาดในการสื่อสารข้อมูลซึ่งเราเรียกบิตนี้ว่า บิตพาริตี้ (Parity bit) UART ส่วนใหญ่สามารถสร้างและทำการตรวจสอบข้อมูลนั้นว่าเป็นพาริตี้คู่หรือเป็นพาริตี้คี่ได้ ในการสร้างพาริตี้คู่ UART จะทำการเซตหรือเคลียร์ค่าในบิตพาริตี้เพื่อให้ข้อมูลทั้ง 8 บิตนั้นมีเลข 1 จำนวนคู่ตัว และ ในการสร้างพาริตี้คี่ UART จะทำการเซตหรือเคลียร์ค่าในพาริตี้เพื่อให้ข้อมูลทั้ง 8 บิตนั้น มีเลข 1 จำนวนคี่ตัว การส่งข้อมูลของ UART จะเป็นแบบอะซิงโครนัส ซึ่งก็หมายความว่าเวลาระหว่างคำอัตราการส่งข้อมูลของ UART จะไม่ขึ้นกับจังหวะการทำงานของไมโคร โดยไมโครคอนโทรลเลอร์และ UART จะมีวงจรสร้างสัญญาณนาฬิกาของมันเอง แต่ถ้าเราพบว่าไมโครคอนโทรลเลอร์และ UART ทำงานร่วมกันอย่างเข้าจังหวะแต่การทำเช่นนี้ก็เพื่อเป็นการลดส่วนของวงจรฮาร์ดแวร์ที่ใช้สร้างสัญญาณนาฬิกา



รูปที่ 2.15 การสื่อสารแบบอะซิงโคนัสที่ไม่มีพาริตี้บิต



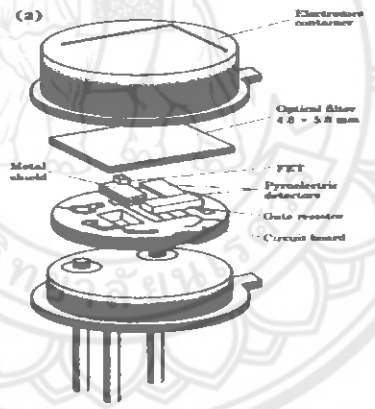
รูปที่ 2.16 การสื่อสารแบบอะซิงโคนัสที่มีพาริตี้บิต

การเชื่อมต่อระหว่างพอร์ตอนุกรม โดยทั่วไปเราจะพบเส้นส่งสัญญาณอนุกรมแบบมาตรฐาน EIA RS-232 มากที่สุดซึ่งเราจะเรียกว่า RS-232 สายส่งสัญญาณ RS-232 นี้ได้ถูกนำไปใช้ในหน่วยแสดงผล เครื่องพิมพ์ โมเด็ม และอุปกรณ์อื่น ๆ ซึ่งจะมีความยาวของสายไม่เกิน 50 ฟุต ซึ่งระบบมาตรฐาน RS-232 ได้กำหนดให้ค่าสัญญาณไฟฟ้าที่มีระดับแรงดันไฟฟ้าเท่ากับ 3 โวลต์หรือสูงกว่า ที่มีค่าทางตรรกะเป็น 1 และกำหนดค่าสัญญาณไฟฟ้าที่มีระดับแรงดันเท่ากับ -3 โวลต์หรือต่ำกว่า มีค่าทางตรรกะเป็น 0 วงจรไอซีที่สร้างสัญญาณเหล่านี้ต้องการแหล่งจ่ายไฟขนาด +12 โวลต์ RS-232 จะใช้สาย 1 เส้นสำหรับส่งข้อมูลและใช้สายอีก 1 เส้นสำหรับรับข้อมูล โดยสัญญาณในแต่ละสายนี้จะถูกอ้างอิงกับกราวนด์ (ขาเบอร์ 7) มาตรฐาน RS-232 นี้ยังได้กำหนดสัญญาณตอบรับเพื่อใช้ในการควบคุมการรับ/ส่งข้อมูลอีกด้วย โดยการสื่อสารอนุกรมนี้จะทำหน้าที่รับ/ส่งข้อมูลกับหน่วยประมวลผลในส่วน High Level ในที่นี้คือ คอมพิวเตอร์ที่มีโปรแกรมขั้นสูงในการทำการประมวลผลภาพและทำการตัดสินใจต่างๆ เพื่อส่งต่อให้ชุดควบคุมระดับล่าง (Microcontroller) ทำให้หุ่นยนต์เคลื่อนที่ได้ต่อไป แต่สิ่งที่จะบอกถึงสภาพของสิ่งแวดล้อมรอบข้างรวมถึงสิ่งกีดขวางนั้นคือ ระบบเซนเซอร์ โดยจะอธิบายในหัวข้อต่อไป

หลักการอุปกรณ์ตรวจจับวัตถุและเป้าหมาย

Passive Infrared Detector เนื่องจากวัตถุประสงค์ของโครงการนี้ นอกเหนือจากการที่หุ่นยนต์สามารถเคลื่อนที่ไปในบริเวณที่มีสภาพแวดล้อมที่เป็นอันตรายแล้ว การค้นหาผู้รอดชีวิตก็เป็นอีกวัตถุประสงค์หนึ่ง ซึ่งผู้จัดทำโครงการทำการศึกษาและเลือกใช้ PIR sensor เนื่องจากเซ็นเซอร์ดังกล่าว สามารถตรวจจับสิ่งมีชีวิตได้ โดยมีรายละเอียดดังต่อไปนี้

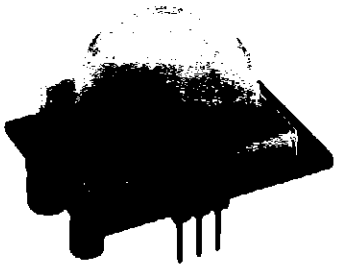
PIR ย่อมาจาก Passive Infrared คือตัวตรวจจับรังสีอินฟราเรดแบบหนึ่ง โดยตัวมันจะทำงานเมื่อตรวจจับพบความเปลี่ยนแปลงของรังสีอินฟราเรด ที่แผ่ออกมาจากตัวคนหรือตัวสัตว์ในขณะที่มีการเคลื่อนไหว ในตัวคนหรือสัตว์จะมีรังสีความร้อนแผ่ออกมารอบ ๆ ตัวในปริมาณที่แน่นอนอยู่จำนวนหนึ่ง เมื่อเกิดการเคลื่อนไหวหรือเคลื่อนที่ก็จะทำให้อุณหภูมิในบริเวณนั้นเกิดการเปลี่ยนแปลง พลังงานความร้อนเกิดการเปลี่ยนแปลง ส่งผลให้คลื่นรังสีความร้อนที่ว่านี้แผ่กระจายออกมา มีความยาวคลื่นประมาณ 0.74-300 ไมโครเมตร อันเป็นแถบความถี่ในย่านอินฟราเรดพอดี ประสิทธิภาพในการตรวจจับแบบกลมสามารถตรวจจับได้ในระยะ 5 เมตร วงจรนี้ใช้ไฟเลี้ยงวงจร 5-15 โวลต์ดีซี กินกระแสสูงสุดประมาณ 35 มิลลิแอมแปร์



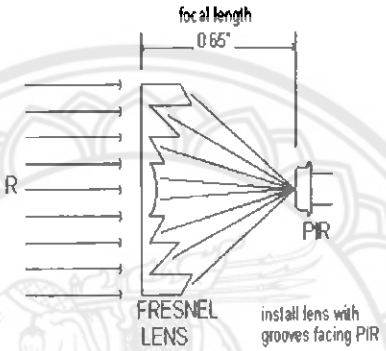
รูปที่ 2.17 ภาพแสดงตัวตรวจจับรังสีอินฟราเรด

ส่วนประกอบหลักของ PIR detector

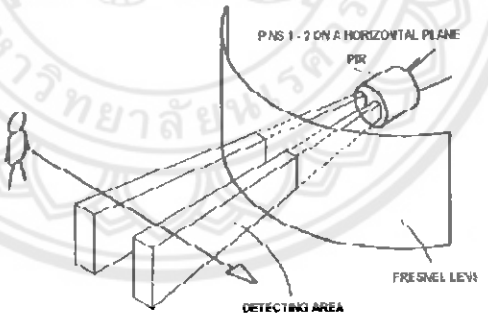
ส่วนประกอบหลักที่ทำหน้าที่ของ PIR detector คล้ายกับตาซึ่งมี Fresnel lens ทำหน้าที่ focus ความร้อนให้ไปตกที่ Pyrosensor, Pyrosensor เปรียบเสมือน จอประสาทตา, Circuit Board เปรียบเสมือนสมองที่ประมวลผลเพื่อส่งสัญญาณไป Microcontroller ของระบบ ซึ่งภาพของส่วนประกอบและการทำงานแสดงในรูปต่อไปนี้



รูปที่ 2.18 ภาพแสดง Fresnel lens



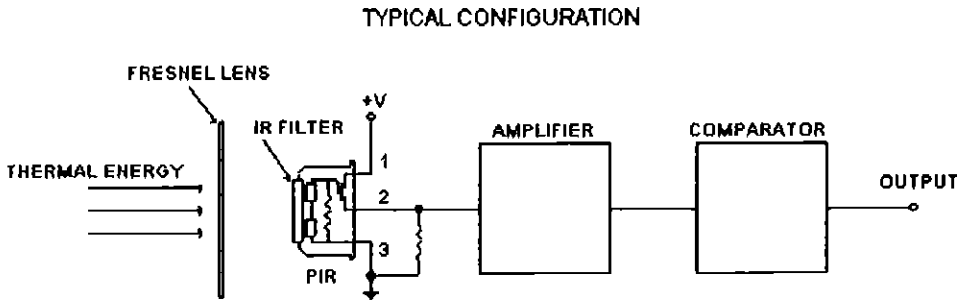
รูปที่ 2.19 ภาพแสดงการทำงานของ PIR



รูปที่ 2.20 ภาพแสดงการทำงานของ PIR

การทำงาน

เมื่อมีคนหรือสัตว์เดินผ่านหน้า PIR จะทำให้ที่ขา S ของ PIR มีพัลส์ลูกเล็ก ๆ เกิดขึ้น เนื่องมาจากตัว PIR จะทำการตรวจจับการเปลี่ยนแปลงความร้อนจากการเปลี่ยนแปลงของรังสีอินฟราเรดที่แผ่ออกมาจากตัวของคนหรือสัตว์ในขณะที่มีการเคลื่อนไหว



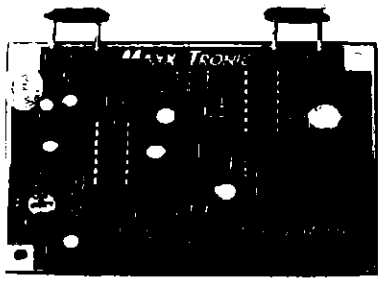
รูปที่ 2.21 ภาพแสดง TYPICAL CONFIGURATION OF PYROSENSOR

Ultrasonic sensor

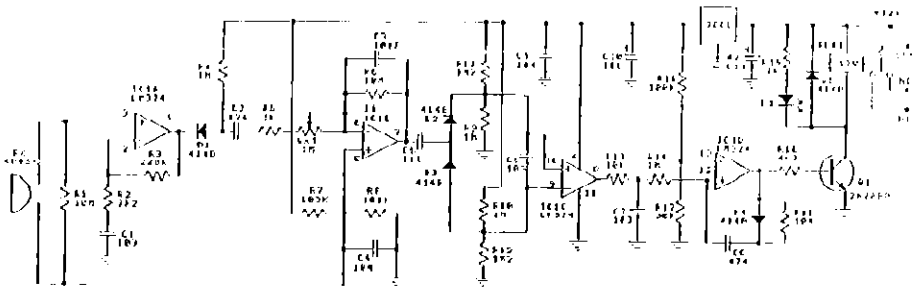
การตรวจจับความเคลื่อนไหวด้วยอัลตราโซนิคนี้จะแบ่งออกได้เป็น 2 ส่วน คือ ส่วนของตัวส่ง (TX) และส่วนของตัวรับ(RX) ในระหว่างการทำงานเซนเซอร์จะทำการส่งสัญญาณเสียงซึ่งเรียกว่า "ซาวด์พาร์เซลส์"(Sound parcels) ให้ขบวนการทางอิเล็กทรอนิกส์ของเวลาทำงานไปเรื่อยๆ จนกระทั่งมีการรับการสะท้อนครั้งแรกที่เกิดขึ้น โดยในส่วนของเครื่องส่งนั้นจะทำงานเป็นวงจรออสซิลเลเตอร์เพื่อกำเนิดความถี่สูงในย่านอัลตราโซนิคขึ้นมา ซึ่งมีความถี่ขนาด 40 กิโลเฮิรตซ์เพื่อกระจายเสียงออกไป ในส่วนของตัวรับนั้นจะทำหน้าที่รับคลื่นที่ได้รับจากการสะท้อนคลื่น

คุณสมบัติของ Ultrasonic sensor

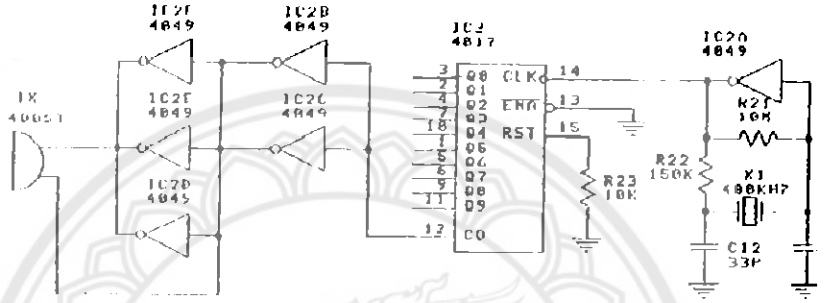
1. สามารถตรวจจับวัตถุทุกชนิดด้วยข้อจำกัดซึ่งน้อยมาก วัตถุทุกชนิดสามารถตรวจจับได้ดีเท่ากันตราบใดที่ขนาดและมุมของการตกกระทบเป็นไปตามที่กำหนด
2. ไม่มีค่าความผิดพลาดที่ต้องแก้ไข ใช้ได้กับทุกสี อย่างไม่มีปัญหาพื้นที่ผิวของวัตถุที่ตรวจจับ ไม่มีอิทธิพลต่อการวัด
3. ทำงานได้ดีแม้ในสภาวะ หมอก ฝุ่น ควัน หรือในที่ซึ่งมีแสงน้อยๆ
4. อัลตราโซนิคดีมากในการตรวจจับวัตถุโปร่งแสงและเป็นมันวาว เช่นกระจก ขวด แผ่นฟอยล์ต่างๆ เป็นต้น



รูปที่ 2.22 ภาพแสดง Ultrasonic sensor



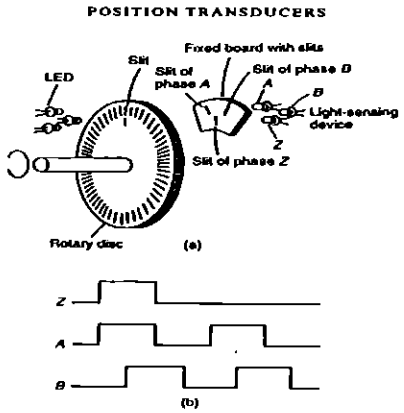
รูปที่ 2.23 ภาพแสดงวงจรชุดรับของ Ultrasonic sensor



รูปที่ 2.24 ภาพแสดงวงจรชุดส่งของ Ultrasonic

หลักการควบคุมป้อนกลับ

เอ็นโค้ดเดอร์ (Encoder) ในโครงงานนี้ได้นำเอาหลักการของ Increment Encoder เข้ามาใช้และอ่านค่าออกมาเพื่อตรวจสอบ ตำแหน่งของการเคลื่อนที่และควบคุมความเร็วของมอเตอร์เพื่อนำมาทำการเปรียบเทียบตำแหน่งการเคลื่อนที่กับค่า input โดยหลักการทำงานของ Encoder มีดังนี้ Increment Encoder มีลักษณะเป็นแผ่นกลมมีแกนอยู่ตรงกลาง และ ที่แผ่นกลม จะมีช่องเล็ก ที่แสงสามารถส่องผ่านได้เป็นจำนวนมากซึ่งเรียกช่องนี้ว่าช่อง slit ซึ่งที่ด้านหนึ่งของแผ่นกลมนี้จะมีหลอด LED ซึ่งเป็นตัวส่งแสง infrared ไปยังตัวรับสัญญาณแสง infrared ซึ่งจะอยู่ในด้านตรงกันข้าม โครงสร้างจะประกอบด้วยตัวกำเนิดแสง,ตัวจับแสงซึ่งถูกคั่นกลางด้วยแผ่นจานกลมๆที่มีการทำรูเจาะไว้รอบๆแผ่น (จำนวนรูจะขึ้นอยู่กับความละเอียดของ incremental encoder) และหน้ากากแยกช่องของสัญญาณพัลส์ A ,B และ Z สัญญาณพัลส์ที่ได้จากเอนโคดเดอร์ชนิดนี้จะประกอบด้วย 3 แทรค (tracks) คือ A,BและZ พัลส์ที่เกิดจาก แทรค A และ B จะเกิดการเหลื่อมกันมีความต่างเฟสกัน 90 องศา เพื่อทำหน้าที่รายงานผลของความเร็วและทิศทางการหมุนของมอเตอร์ให้คอนโทรลเลอร์ดังนี้กรณีพัลส์ A เกิดขึ้นก่อน B คอนโทรลเลอร์จะรับรู้ว่ามอเตอร์กำลังหมุนด้วยทิศทางตามเข็มนาฬิกา ส่วนแทรค Z หรือพัลส์อ้างอิง จะเกิดขึ้น1พัลส์ในการหมุน 1 รอบ ทำหน้าใช้อ้างอิงตำแหน่งของโรเตอร์

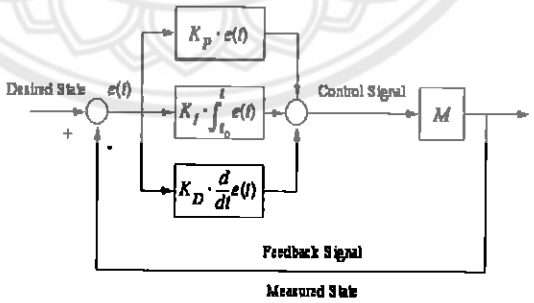


รูปที่ 2.25 ภาพแสดง incremental encoder

Incremental encoder โดยทั่วไปจะไม่นิยมใช้กับระบบเซอร์โวที่มีการควบคุมตำแหน่ง เนื่องจากไม่สามารถจำตำแหน่งเดิมได้กรณีที่มีการปิดเครื่องหรือไฟดับ ซึ่งจะต้องทำการหาจุดอ้างอิงใหม่ทุกครั้ง

การควบคุมแบบป้อนกลับ

ระบบการควบคุมแบบป้อนกลับ เป็นระบบการควบคุมที่นิยมใช้ในงานอุตสาหกรรมต่างๆ เป็นอย่างมาก เนื่องจากคุณสมบัติที่ว่าระบบการควบคุมแบบนี้สามารถปรับค่าการควบคุมได้เมื่ออุปกรณ์ที่ต้องการควบคุมเปลี่ยนไป ซึ่งประสิทธิภาพในการควบคุมก็ให้ผลดีเป็นที่น่าพอใจอีกทั้งการควบคุมก็ทำได้ง่ายและไม่ซับซ้อนจนเกินไป โดยปกติแล้วการควบคุมแบบป้อนกลับสามารถพิจารณาโดยแบ่งการควบคุมออกเป็น 3 ส่วนดังรูป



รูปที่ 2.26 ระบบควบคุมแบบ PID

จากรูปจะเห็นว่าเราสามารถแบ่งการควบคุมแบบป้อนกลับออกเป็นสามส่วนดังสมการ

$$m(t) = K_P e(t) + K_I \int e(t)dt + K_D \frac{de}{dt} \qquad (2.3)$$

ซึ่งเราสามารถแบ่งการควบคุมออกเป็นส่วนๆดังนี้

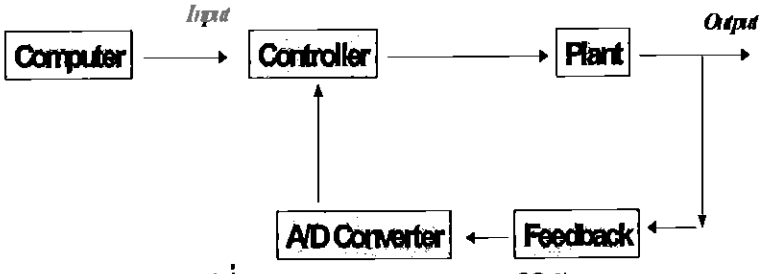
1. ระบบควบคุมแบบสัดส่วน (Proportional Control) การควบคุมแบบสัดส่วนเป็นการนำสัญญาณความคลาดเคลื่อนระหว่างสัญญาณขาเข้าและสัญญาณออกไปคูณกับค่าคงที่ของการควบคุมแบบสัดส่วน (Proportional Gain) แล้วส่งสัญญาณที่ได้ไปขับอุปกรณ์ ซึ่งจะสังเกตได้ว่าหากสัญญาณทั้งสองมีความแตกต่างกันมากๆแล้วระบบจะทำให้เกิดสัญญาณควบคุมมากๆ นั่นจะทำให้อุปกรณ์อยู่ในสภาวะที่เราต้องการได้อย่างรวดเร็ว อย่างไรก็ตามการควบคุมแบบนี้จะทำให้เกิดสภาวะของการสั่นรอบๆจุดของสัญญาณที่เราต้องการ ซึ่งเรียกว่าการเกิด Overshoot โดยหากเราเพิ่มค่าค่าคงที่ควบคุมแบบสัดส่วนมากๆนั้นระบบจะเร็วขึ้นก็จริงแต่อาจเกิดแกว่งของสัญญาณรอบๆจุดที่ต้องการได้

2. ระบบควบคุมแบบสะสม (Integral Control) ในระบบการควบคุมนั้น บางครั้งระบบการควบคุมไม่อาจควบคุมอุปกรณ์ให้ไม่มีความคลาดเคลื่อนของสัญญาณขาเข้าและสัญญาณขาออกได้จึงได้มีการเพิ่มระบบการควบคุม โดยสะสมค่าความคลาดเคลื่อนแล้วนำไปคูณกับค่าคงที่ของการควบคุมแบบสะสม (Integral Control) เพื่อให้เกิดค่าความผิดพลาดน้อยที่สุด

3. ระบบควบคุมแบบความแตกต่าง (Differential Control) ระบบการควบคุมแบบนี้จะหาว่าค่าความเคลื่อนในอดีตกับปัจจุบันมีความแตกต่างกันมากเพียงใด ซึ่งหากมีความแตกต่างมากเมื่อเรานำสัญญาณความแตกต่างไปคูณกับค่าคงที่ของการควบคุมแบบสะสม แล้วก็จะได้สัญญาณที่จะทำให้ระบบนั้นเร็วขึ้นได้

ระบบควบคุมแบบดิจิทัล

เป็นการควบคุมอุปกรณ์โดยใช้เครื่องมือแบบดิจิทัลเช่นเครื่องคอมพิวเตอร์เป็นตัวควบคุมระบบซึ่งแผนผังการควบคุมแบบดิจิทัลแสดงดังรูป



รูปที่ 2.27 ระบบควบคุมแบบดิจิทัล

ในการที่จะสร้างสมการการควบคุมแบบดิจิทัลจากระบบการควบคุมแบบอนาล็อกจะทำได้โดยการแปลงสมการแบบอนาล็อกเป็นสมการแบบดิจิทัล ซึ่งในที่นี้ได้พิจารณาขีดความสามารถของไมโครคอนโทรลเลอร์ที่มีข้อจำกัดต่างๆค่อนข้างมากจะไม่สามารถใช้การคำนวณที่ซับซ้อนได้

การควบคุมแบบ PID ในแบบของสัญญาณอนาล็อกสามารถแสดงดังสมการ 2.3 ในการที่จะแปลงสมการนี้ให้อยู่ในรูปแบบของดิจิทัลสามารถทำได้โดยการใช้สมการอนุพันธ์ดังนี้

ระบบควบคุมแบบสัดส่วน (Proportional Control) สามารถทำได้เหมือนกับในระบบของอนาล็อกนั่นคือเราต้องคูณสัญญาณคลาดเคลื่อนด้วยค่าคงที่ที่เรียกว่า Gain ดังสมการ

$$m(kT) = K_P \times e(kT) \tag{2.4}$$

ระบบควบคุมแบบสะสม (Integral Control) เนื่องจากสมการอนุพันธ์ของการอินทิเกรตคือการหาพื้นที่ใต้เส้นโค้ง โดยที่การหาพื้นที่ใต้เส้นโค้งนี้จะต้องใช้วิธีการอินทีเกรตแบบตัวเลข (Numerical Integration Method) วิธีใดวิธีหนึ่งแต่สำหรับไมโครคอนโทรลเลอร์นั้นยังไม่เหมาะกับการคำนวณที่ซับซ้อน วิธีที่เหมาะสมคือการอินทิเกรตแบบสี่เหลี่ยม (Rectangular Integration) ดังแสดงดังสมการ

$$m(kT) = K_i T(S_{(k-1)T} + e_{kT}) \tag{2.5}$$

ถ้าเราให้

$$K_i T = K_I \tag{2.6}$$

และ

$$S_{kT} = S_{(K-1)T} + e_{kT} \tag{2.7}$$

ดังนั้น

$$m(kT) = K_I S_{kT} \tag{2.8}$$

ระบบควบคุมแบบแตกต่าง (Difference Control) ในการหาค่าความแตกต่างของสัญญาณความผิดพลาดเราจะใช้วิธี Backward Difference Formula ดังสมการ

$$m(kT) = K_D[e(kT) - e([k-1]T)] \tag{2.9}$$

เราจึงได้สมการของการควบคุมแบบ Digital PID ดังสมการ

$$m(kT) = K_P e(kT) + K_I S_{kT} + K_D[e(kT) - e([k-1]T)] \tag{2.10}$$

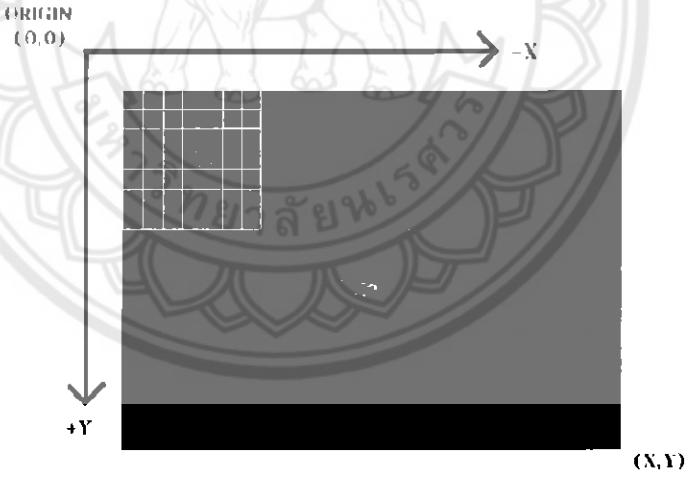
จากหลักการทั้งหมดที่กล่าวมาสามารถสร้างหุ่นยนต์ที่พร้อมจะทำการเคลื่อนที่แต่อีกส่วนที่สำคัญก็คือ การประมวลผลภาพและการใช้หลักการปัญญาประดิษฐ์ เพื่อให้หุ่นยนต์อัตโนมัติสามารถประมวลผลและตัดสินใจในการเคลื่อนที่และค้นหาเป้าหมายที่ดีที่สุด และเพื่อลดการใช้มนุษย์ในการปฏิบัติภารกิจ

2.3 การควบคุมระดับบน (High Level)

2.3.1 การประมวลภาพ (Image processing)

2.3.1.1 การเก็บข้อมูลรูปภาพแบบดิจิทัล ภาพที่เก็บในคอมพิวเตอร์จะถูกจัดเก็บในรูปของพิกเซล (Pixel) ซึ่งหมายถึง ส่วนประกอบมูลฐานของภาพ ซึ่งจะเก็บค่าของจุดสีหรือความเข้มของสีตามประเภทต่างๆที่ต้องการเก็บเช่น RGB HSL HSV เป็นต้น เมื่อซูมภาพเข้าไปใกล้ๆจะเห็นเป็นจุดสีหรือช่องสี่เหลี่ยมที่มีสีเดียวต่อหนึ่งช่อง แต่ละจุดนั่นคือ พิกเซล (Pixel) นั่นเอง หากภาพมีความละเอียดมากก็จะมีจำนวนพิกเซลมากขึ้นด้วย

ภาพ 2 มิติที่เก็บในคอมพิวเตอร์จะมีการบันทึกตำแหน่งของพิกเซลหรือจุดสีในรูปแบบของโคออร์ดิเนต (X, Y) ซึ่ง X และ Y จะมีจุดกำเนิดอยู่ที่มุมบนซ้ายของภาพและมีค่าเพิ่มขึ้นไปทางซ้ายและล่างไปหามุมล่างขวาของภาพนั้นๆ



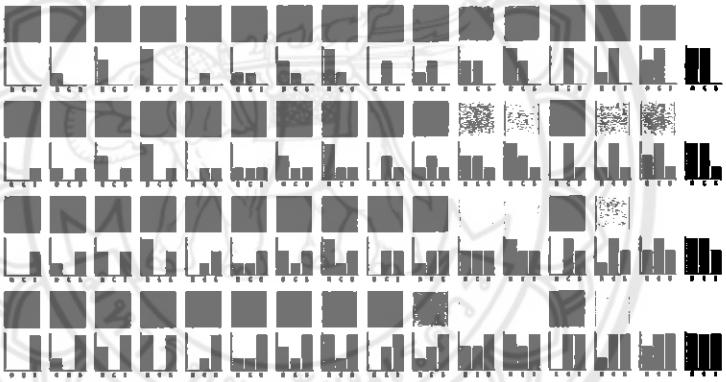
รูปที่ 2.28 ตำแหน่งของพิกเซลของภาพ 2 มิติ

2.3.1.2 แบบจำลองระบบสี

แบบจำลองระบบสี RGB ระบบสี RGB เป็นระบบสีที่เกิดจากการรวมกันของแสงสีแดง เขียว และน้ำเงินโดยมีการรวมกันแบบ Additive ซึ่งโดยปกติจะนำไปใช้กับระบบเครื่องใช้ไฟฟ้า เช่น ทีวี คอมพิวเตอร์

รูปที่ 2.29 การผสมสีของระบบสี RGB

การผสมสีของระบบสี RGB มีลักษณะคือ สีหลัก คือ แดง(Red) เขียว(Green) น้ำเงิน(Blue) สีรอง เกิดจากสีหลัก 2 สีรวมกัน คือ ฟ้า(Cyan) เกิดจาก เขียว+น้ำเงิน, ม่วง(Magenta) เกิดจาก แดง+น้ำเงิน และ เหลือง(Yellow) เกิดจาก แดง+เขียว สีขาว เกิดจากสีหลักรวมกัน สีดำ เกิดจากไม่มีสีใดเลย ความเข้มของสีหลัก จะเป็นตัวกำหนดความเหลือบของสีขาว หรือความสว่างของสี และหากความเข้มของแต่ละสีเท่ากัน จะเป็นลักษณะของการเหลือบของสีเทา



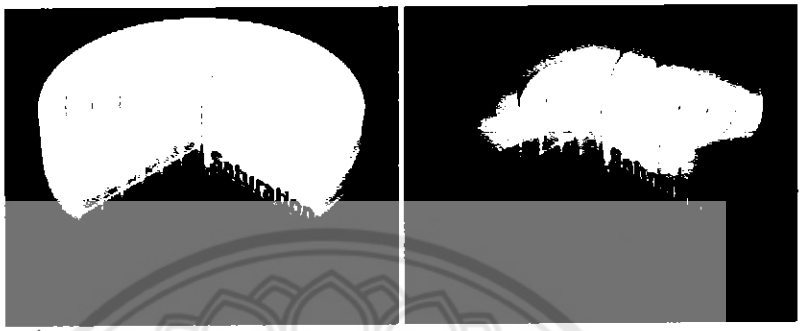
รูปที่ 2.30 การรวมกันของสีหลักต่างๆ

แบบจำลองระบบสี HSL และ HSV

ระบบสี HSL และ HSV จะมีลักษณะคล้ายกับระบบสี RGB โดยที่ HSL หมายถึง สี (hue) ความอิ่มตัว (saturation) และ น้ำหนักความสว่าง (lightness) ส่วน HSV หมายถึง สี (hue) ความอิ่มตัว (saturation) และ ค่า (value) ส่วน HSI และ HSB จะใช้ ความเข้ม (intensity) และความสว่าง (brightness) ซึ่งจะมีลักษณะเดียวกันกับระบบสี HSL

รูปที่ 2.31 Cylinder แทนระบบสี HSL และ HSV

ทั้งระบบสี HSL และ HSV จะใช้ ทรงกระบอก เพื่อแทนสีต่างๆ โดยมีแกนกลางที่ด้านล่างจะเป็นสีดำและด้านบนจะเป็นสีขาว และระหว่างทั้งสองจะเป็นสีเทา โดยมุมที่ทำกับแกนกลางหมายถึง สี และระยะที่ห่างจากแกนหมายถึง ความอิ่มตัว และระยะตามแนวแกนหมายถึง ค่า หรือความสว่าง หรือ น้ำหนักความสว่าง



รูปที่ 2.32 เปรียบเทียบระหว่างระบบสี HSL (รูปซ้าย) และ HSV (รูปขวา)

2.3.1.3 Image Segmentation

จุดประสงค์ที่สำคัญของกระบวนการทาง Image Processing คือ การแยกลักษณะสำคัญ (Feature) หรือ วัตถุ (Object) จากพื้นของภาพ (Background) ดังนั้นกระบวนการที่สำคัญของการแยกลักษณะสำคัญหรือวัตถุจากพื้นของภาพก็คือ Image Segmentation อันเป็นพื้นฐานในการวิเคราะห์และการแปลภาพต่อไป

Image Segmentation หมายถึง การทำให้ภาพแยกเป็นส่วนย่อยๆ โดยอาศัยหลักการทางคณิตศาสตร์บางอย่าง เราสามารถพิจารณาแบ่งการทำ Image Segmentation ที่ได้รับความนิยมมากที่สุดคือ การใช้ค่าขีดแบ่ง (Thresholding) และการหาขอบภาพ (Edge Detection)

การใช้ค่าขีดแบ่ง เทคนิคนี้จะใช้พารามิเตอร์ I_b เรียกว่า ค่าขีดแบ่งความสว่าง (brightness threshold) โดยจะนำพารามิเตอร์ที่เรากำหนดเปรียบเทียบกับค่าสีในแต่ละพิกเซลของภาพ ซึ่งหากค่าสีของพิกเซลนั้นมีค่ามากกว่า I_b แสดงว่าพิกเซลนั้นเป็นวัตถุที่เราสนใจ แต่หากน้อยกว่า I_b ก็จะแสดงว่าพิกเซลนั้นเป็นพื้นหลัง ดังสมการต่อไปนี้ (ให้ a[m,n] คือ ค่าสีของพิกเซลในแถวที่ m และหลักที่ n)

If a[m,n] >= I_b **a[m,n] = object = 1**

Else **a[m,n] = background = 0**

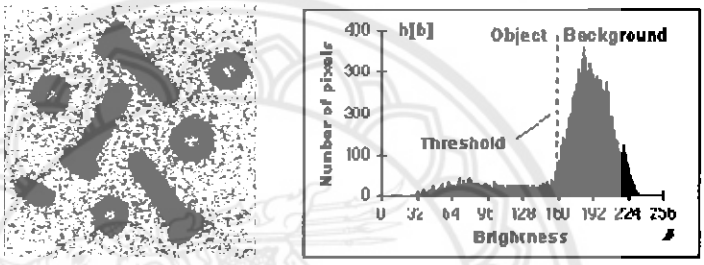
การใช้ค่าขีดแบ่งนั้นมีหลายรูปแบบในการกำหนดค่าขีดแบ่ง เนื่องจากค่าขีดแบ่งนั้นจะเป็นค่าที่กำหนดแบ่งระหว่างวัตถุที่เราสนใจและพื้นหลัง โดยที่รูปแบบการกำหนดค่าขีดแบ่งที่นิยมมีดังนี้

ค่าขีดแบ่งคงที่ (Fixed threshold)

เราจะใช้กับรูปที่มีลักษณะ high-contrast นั่นคือมีระดับสีมีความแตกต่างกันมาก เช่น ภาพขาว-ดำ เราจะใช้ค่า threshold แบบคงที่ เช่น 128 (สมมุติให้มี scale สีระหว่าง 0 ถึง 255) ซึ่งจะสามารถแบ่งภาพออกเป็น 2 ส่วนได้อย่างมีประสิทธิภาพมากที่สุด

ค่าขีดแบ่งจากกราฟฮิสโตแกรม (Histogram-derived threshold)

แต่ในการทำงานส่วนใหญ่แล้วค่าขีดแบ่งจะถูกเลือกจากกราฟแสดงการแจกแจงความถี่ของพิกเซล (Histogram) ที่มีค่าความสว่างต่างๆ แสดงกราฟดังรูปที่ 2.33



(ก) (ข)

รูปที่ 2.33 (ก) ภาพต้นแบบ (ข) กราฟแสดงการแจกแจงความถี่ของพิกเซล

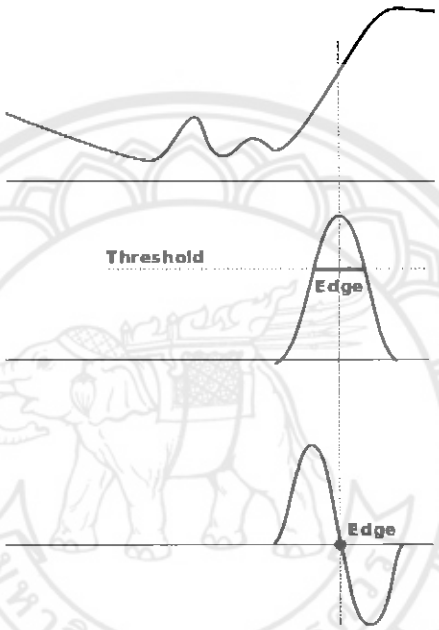
เมื่อทำการหาค่าขีดแบ่งจากกราฟได้แล้วเราจะให้พิกเซลที่ต่ำกว่าค่าขีดแบ่ง(a[m,n] < ϑ) เราจะให้เป็นพิกเซลวัตถุ และในทางตรงกันข้ามถ้าพิกเซลใดที่มีค่าสูงกว่าค่าขีดแบ่ง เราจะให้เป็นพิกเซลพื้นหลังเช่นเดย

การทำ Convolution การประมวลผลภาพโดยการใช้ตัวกรองทางตำแหน่ง หรือ Spatial-filtering นั้นเป็นการปรับปรุงภาพทีละจุด โดยใช้ตัวกรองที่เราเรียกว่า Kernel ซึ่ง Kernel นี้ก็คือ set ของตัวเลข หรือ เมตริกซ์ขนาด 3x3 5x5 หรือ 7x7 หน่วยเป็น Pixel เราจะใช้ Kernel ที่มีตัวเลขค่าต่างๆนั้นกระทำทั่วทั้งภาพ ซึ่งผลลัพธ์ที่ได้จะต่างกัน ถ้าใน Kernel มีตัวเลขที่ต่างกัน โดยวิธีการกระทำนั้นเราจะนำ Kernel ขนาด 3x3 เข้าไปทาบทีละจุดแล้วเอาเลขใน Kernel กับเลขเดิมของภาพในตำแหน่งที่ทาบอยู่มาคูณกันจากนั้นนำผลลัพธ์ที่ได้ทั้งหมดมาบวกกัน เมื่อได้ผลรวมทั้งหมดแล้วเราจะเอามาหารด้วยผลบวกของเลขใน Kernel ทั้งหมด นำผลลัพธ์ที่ได้จากการคำนวณทั้งหมดใส่แทนช่องตรงกลางที่ Kernel ทับอยู่ แล้ววน Kernel ไปทับช่องถัดไป ทำอย่างนี้วนไปจนครบทั้งภาพ ซึ่งทั้งหมดที่เราทำนี้คือการเน้นขอบให้ภาพที่ต้องการชัดขึ้นหรือลด noise ที่มีอยู่ลงไปเพื่อง่ายต่อการนำไปใช้งานต่อไป

การหาขอบภาพ การหาขอบภาพเป็นการหาเส้นรอบวัตถุที่อยู่ในภาพ เมื่อทราบเส้นรอบวัตถุ เราจะสามารถคำนวณพื้นที่ (ขนาด) หรือรู้ชนิดของวัตถุนั้นได้ อย่างไรก็ตาม การหาขอบภาพที่

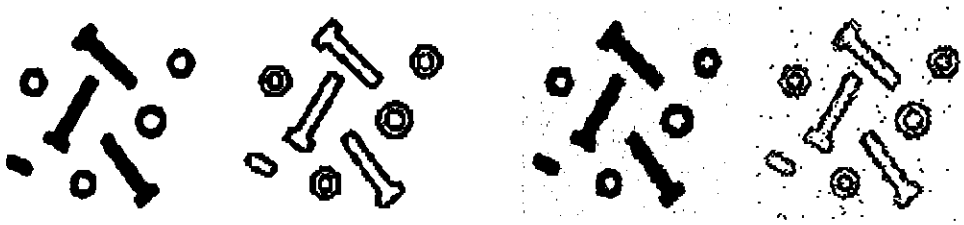
ถูกต้องสมบูรณ์ไม่ใช่เป็นเรื่องที่ง่าย โดยเฉพาะอย่างยิ่งการหาขอบของภาพที่มีคุณภาพต่ำ มีความแตกต่างระหว่างพื้นหน้ากับพื้นหลังน้อย หรือมีความสว่างไม่สม่ำเสมอทั่วภาพ

ขอบภาพเกิดจากความแตกต่างของความเข้มแสงจากจุดหนึ่งไปยังอีกจุดหนึ่ง หากความแตกต่างนี้มีค่ามาก ขอบภาพก็จะเห็นได้ชัด ถ้าความแตกต่างมีค่าน้อย ขอบภาพก็จะไม่ชัดเจน ซึ่งวิธีการหาขอบนั้นมีด้วยกันหลายวิธี แต่อย่างไรก็ตามสามารถแบ่งได้เป็น 2 กลุ่มหลัก คือวิธี Gradient และวิธี Laplacian



รูปที่ 2.34 กราฟตัวอย่างการหาขอบภาพ

วิธี Gradient จะหาขอบโดยการหาจุดต่ำสุดและจุดสูงสุดในรูปของอนุพันธ์อันดับหนึ่งของภาพ เราจะสามารถหาขอบได้ด้วยการคิดคำนวณจากพื้นสี โดยจุดที่เป็นขอบจะอยู่ในส่วนที่เหนือค่าขีดแบ่งดังรูปที่ 2.34 (B) จึงอาจทำให้เส้นขอบที่ได้มีลักษณะหนา ตัวอย่างวิธีการหาขอบของกลุ่มนี้ เช่น Roberts, Prewitt, Sobel และ Canny เป็นต้น



รูปที่ 2.35 การหาขอบโดยใช้วิธี Gradient method (ก) มีค่า SNR = 30 dB และภาพ (ข) SNR = 20

เทคนิคนี้จะมีประสิทธิภาพมากที่สุดเมื่อภาพนั้นต้องมี SNR 30 dB ขึ้นไป สังเกตได้จากรูปที่มีค่า SNR 20 dB ที่มี noise จำนวนมาก แต่อย่างไรก็ตาม ปัญหานี้ก็สามารถจัดการได้ด้วยการกำจัด noise ด้วยการ Blur ภาพก่อนที่จะทำวิธี Gradient

Laplacian method จะหาขอบโดยใช้อนุพันธ์อันดับ 2 โดยใช้จุดที่ค่า y เป็น 0 (Zero-crossing) เป็นขอบของภาพ ดังรูปที่ 2.9 (C) ซึ่งวิธีนี้จะใช้เวลาในการคำนวณมากกว่า Gradient method ตัวอย่างวิธีการหาขอบของกลุ่มนี้ เช่น Laplacian of Gaussian และ Marrs-Hildreth เป็นต้น

ตำแหน่งของขอบนั้นขึ้นอยู่กับตำแหน่งของแบบจำลองของรูปที่ค่า Laplacian เปลี่ยนเครื่องหมาย (บวก ลบ) จึงเรียกว่า zero crossing โดยที่การทำแบบนี้อาจจะก่อให้เกิด noise จากกระบวนการนี้ เราจึงจำเป็นต้องป้องกันการเพิ่มขึ้นของ noise ด้วยเทคนิค Log filter และ Plus filter อีกที

(ก) SNR = 20 dB (ข) LOG filter (ค) PLUS filter

รูปที่ 2.36 (ก) ภาพตัวอย่าง (ข) ผลลัพธ์หลังจากใช้วิธี Laplacian พร้อมกับ LOG Filter (ค) ผลลัพธ์หลังจากใช้วิธี Laplacian พร้อมกับ PLUS Filter

2.3.2 หลักปัญญาประดิษฐ์

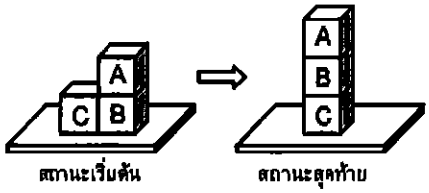
ปัญญาประดิษฐ์ (Artificial Intelligence – AI) เป็นวิชาที่ว่าด้วยการศึกษาเพื่อให้เข้าใจถึงความฉลาดและสร้างระบบคอมพิวเตอร์ที่ชาญฉลาดและนำมาทำงานแทนหรือช่วยมนุษย์ทำงาน

2.3.2.1 Search

การแก้ปัญหาส่วนใหญ่ในปัญญาประดิษฐ์ จะมองปัญหาในรูปของการค้นหา (search) งานหลายๆอย่างในปัญญาประดิษฐ์ใช้พื้นฐานของการค้นหาทั้งสิ้น ในการแก้ปัญหาโดยหลักการนี้เราจะสร้างปริภูมิ (space) ขึ้นมาหนึ่งปริภูม สมาชิกแต่ละตัวนี้แทนตัวเลือกของคำตอบ จากนั้นก็ทำการค้นหาด้วยวิธีการค้นหาอย่างใดอย่างหนึ่ง ซึ่งวิธีในการค้นหามีหลายวิธียกตัวอย่างดังต่อไปนี้

การค้นหาแบบบอด (Blind search)

ส่วนนี้จะเริ่มจากการค้นหาแนวกว้างก่อนตัวอย่างที่ใช้อธิบายคือปัญหาโลกของบล็อก ซึ่งแสดงในรูปด้านล่างนี้



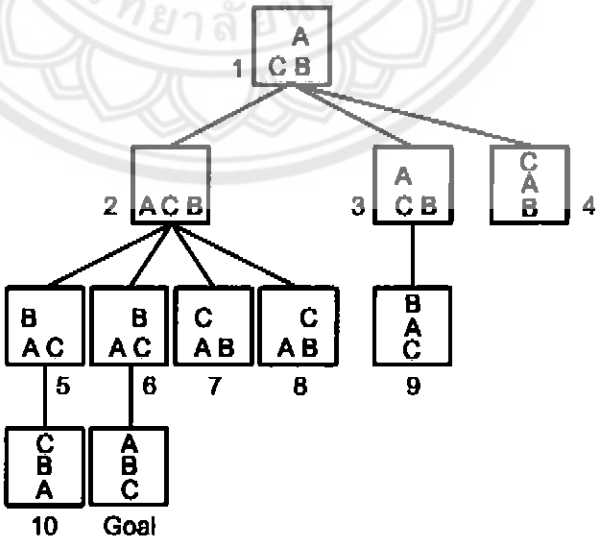
รูปที่ 2.37 ปัญหาโลกของบล็อก

ปัญหาคือกำหนดสถานะเริ่มต้นของการจัดเรียงตัวของบล็อก ต้องการจัดเรียงใหม่ให้ได้ตามสถานะสุดท้าย โดยที่ X และ Y เป็นตัวแปรและสามารถแทนที่ด้วย 'A','B' หรือ 'C' เช่นเมื่อใช้ตัวกระทำการกับสถานะเริ่มต้น จะได้ว่าถ้าบล็อก 'A' ไม่มีบล็อกอื่นทับ แล้ววาง 'A' บนโต๊ะได้

การค้นหาแนวกว้างก่อน

ในการค้นหาแนวกว้างก่อน (breadth-first search) นี้สถานะทุกตัวที่อยู่ในระดับเดียวกันจะถูกตรวจสอบก่อนสถานะที่อยู่ระดับถัดไป วิธีการของการค้นหาแบบนี้ทำโดยเริ่มจากการสร้างสถานะลูกของสถานะเริ่มต้นก่อน แล้วตรวจสอบว่ามีสถานะใดที่เป็นวาระสุดท้ายหรือไม่ ถ้าหากว่ามีก็เป็นอันสิ้นสุดการค้นหา ถ้าไม่พบก็สร้างสถานะลูกของสถานะเหล่านั้นต่อไปอีก ทำเช่นนี้ไปเรื่อยๆ จนกว่าจะพบสถานะสุดท้ายหรือจนไม่สามารถสร้างสถานะลูกได้อีก

ในกรณีของปัญหาโลกของบล็อก เริ่มจากสถานะเริ่มต้น เมื่อเราค้นหาด้วยการค้นหาแนวกว้างก่อนก็จะได้สถานะที่เกิดขึ้นดังรูป ในตัวอย่างนี้กำหนดว่าลำดับของแถวไม่มีความสำคัญ และในการสร้างสถานะจะไม่สร้างซ้ำเดิม

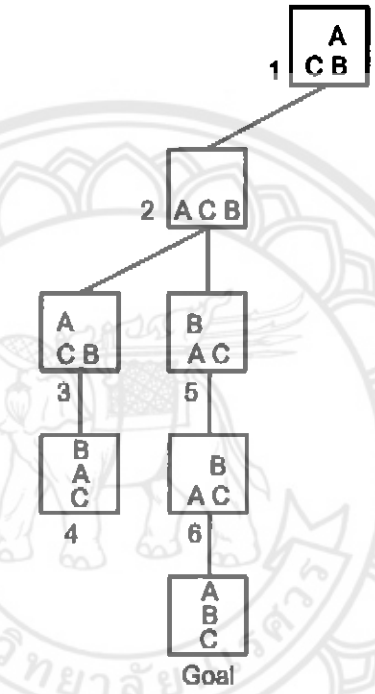


รูปที่ 2.38 การค้นหาแบบแนวกว้างก่อนในปัญหาโลกของบล็อก

ตัวเลข 1, 2, 3 ... ในรูปด้านบนแสดงลำดับของสถานะที่ถูกสร้างขึ้น และ 'Goal' แสดงสถานะเป้าหมายหรือคำตอบ โดยเริ่มจากสถานะที่ 1 ซึ่งเป็นสถานะเริ่มต้น จากนั้นใช้ตัวกระทำการและแทนที่ตัวแปร X และ Y ในตัวกระทำการทั้งสองให้เป็น 'A', 'B' หรือ 'C' ตามลำดับ

การค้นหาแนวลึกก่อน

การค้นหาแนวลึกก่อน (depth-first search) จะสร้างสถานะในแนวลึกทางด้านมุมซ้ายล่างก่อน ถ้าสถานะตามแนวดิ่งถูกสร้างหรือกระจายจนหมดและยังไม่ได้คำตอบก็จะไล่กลับขึ้นด้านบน



รูปที่ 2.39 การค้นหาแบบแนวลึกก่อนในปัญหาโลกของบล็อก

การค้นหาแบบฮิวริสติก

การค้นหาประเภทฮิวริสติกนี้จะใช้ความรู้แบบหนึ่งที่เรียกว่า ฮิวริสติก มาช่วยในการค้นหาให้มีประสิทธิภาพมากขึ้น โดยฮิวริสติกตัวนี้จะช่วยชี้แนะว่ากระบวนการค้นหาควรจะเลือกเส้นทางใดหรือสถานะใดเพื่อทำการค้นหาต่อไปให้ได้คำตอบอย่างมีประสิทธิภาพ

อัลกอรึทึมดีสุดก่อน (best-first search) จะเก็บสถานะทุกตัวโดยไม่มีการตัดทิ้งไป การไม่ตัดทิ้งจึงทำให้อัลกอริทึมนี้ไม่พลาดเส้นทางที่นำไปสู่คำตอบ โดยในแต่ละขั้นตอนจะเลือกสถานะที่มีค่าฮิวริสติกดีสุด โดยพิจารณาสถานะทุกตัวที่ยังไม่ถูกกระจาย (สถานะที่ยังไม่ได้สร้างลูก)

อัลกอรึทึม A* (A* Search) เป็นการขยายอัลกอริทึมดีสุดก่อนโดยพิจารณาเพิ่มเติมถึงต้นทุนจากสถานะเริ่มต้นมายังสถานะปัจจุบันเพื่อใช้คำนวณค่าฮิวรอสติกด้วย ในกรณีของอัลกอริทึม A* เราต้องการหาค่าต่ำสุดของฟังก์ชัน f(s) ของสถานะ s นิยามดังนี้

$$f(s) = g(s) + h'(s) \tag{2.11}$$

โดยที่ g คือฟังก์ชันที่คำนวณต้นทุนจากสถานะเริ่มต้นมายังสถานะปัจจุบัน h' คือฟังก์ชันที่ประมาณต้นทุนจากสถานะปัจจุบันไปยังคำตอบ ดังนั้น f จึงเป็นฟังก์ชันที่ประมวลต้นทุนจากสถานะเริ่มต้นไปยังคำตอบ (ยิ่งน้อยยิ่งดี)

เรามองได้ว่าฟังก์ชัน h' คือฟังก์ชันฮิวริสติกที่เราเคยใช้ในการค้นหาแบบอื่นๆก่อนหน้านี้ เราใส่เครื่องหมาย ' เพื่อแสดงว่าฟังก์ชันนี้เป็นฟังก์ชันประมาณของฟังก์ชันจริง ส่วน g เป็นฟังก์ชันที่คำนวณต้นทุนจริงจากสถานะเริ่มต้นมายังสถานะปัจจุบัน (จึงไม่ได้ใส่เครื่องหมาย ') เพราะเราสามารถหาต้นทุนจริงได้เนื่องจากได้ค้นหาจากสถานะเริ่มต้นจนมาถึงสถานะปัจจุบันแล้ว f ก็เป็นเพียงแค่ฟังก์ชันประมาณโดยการรวมต้นทุนทั้งสองคือ h' และ g'

อัลกอริทึม A* จะทำการค้นหาโดยวิธีเดียวกันกับอัลกอริทึมดีสุดก่อนทุกประการ ยกเว้นฟังก์ชันฮิวริสติกที่ใช้เปลี่ยนมาเป็น f (ต่างจากอัลกอริทึมดีสุดก่อนที่ใช้ h') และด้วยการใช้ f อัลกอริทึม A* จึงให้ความสำคัญกับสถานะหนึ่งๆ 2 ประการคือ (1) สถานะที่ดีต้องมี h' ดีคือต้นทุนเพื่อจะนำไปสู่คำตอบหลังจากนี้ต้องน้อย และ (2) ต้นทุนที่จ่ายไปแล้วกว่าจะถึงสถานะนี้ (g) ต้องน้อยด้วย เราจึงได้ว่า A* จะค้นหาเส้นทางที่ให้ต้นทุนโดยรวมน้อยสุดตามค่า f ซึ่งต่างจากอัลกอริทึมดีสุดก่อนที่เน้นความสำคัญของสถานะที่ต้นทุนหลังจากนี้ที่จะนำไปสู่คำตอบต้องน้อยโดยไม่สนใจว่าต้นทุนที่จ่ายไปแล้วกว่าจะนำมาถึงสถานะนี้ต้องเสียไปเท่าไหร่

2.3.2.2 Machine Learning

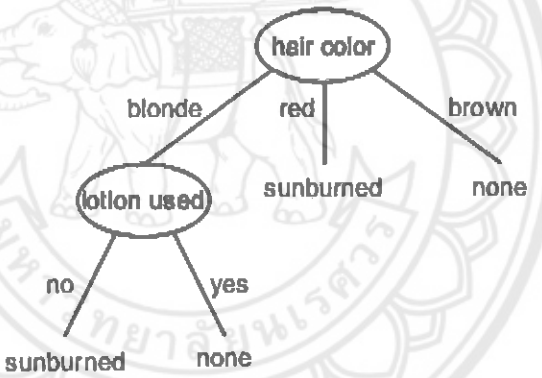
การเรียนรู้ต้นไม้ตัดสินใจ (Decision tree learning) เป็นการเรียนรู้ที่ใช้ในการแทนความรู้อยู่ในรูปของต้นไม้ตัดสินใจ ใช้สำหรับจำแนกประเภทของตัวอย่าง วิธีการเรียนรู้คล้ายกับการเรียนรู้เวอร์ชันสเปซโดยเริ่มจากการป้อนตัวอย่างเข้าไปในระบบ ซึ่งตัวอย่างที่ป้อนให้เป็นตัวอย่างบวกกับตัวอย่างลบก็ได้และนอกจากนั้นเรายังสามารถป้อนตัวอย่างที่มากกว่า 2 ประเภทได้

ตารางที่ 2.1 ตัวอย่างสอนที่สังเกตได้

ประเภท

คุณสมบัติ →	Name	Hair	Height	Weight	Lotion	Result
ค่า	Sarah	blonde	average	light	no	sunburned
	Dana	blonde	tall	average	yes	none
	Alex	brown	short	average	yes	none
	Annie	blonde	short	average	no	sunburned
	Emily	red	average	heavy	no	sunburned
	Pete	brown	tall	heavy	no	none
	John	brown	average	heavy	no	none
	Katie	blonde	short	light	Yes	none

แถวแรกสุดในตารางแสดงคุณสมบัติ ของข้อมูลซึ่งประกอบด้วยชื่อ สีผม ส่วนสูง น้ำหนัก และการใช้โลชัน ส่วนสดมภ์สุดท้ายแทนประเภทของตัวอย่าง คุณสมบัติ Name ไว้สำหรับอ้างอิงตัวอย่างและไม่มีผลต่อการจำแนกข้อมูล เราจึงจะไม่ใช้ Name ในการเรียนรู้เหล่านี้



รูปที่ 2.40 ตัวอย่างของต้นไม้ตัดสินใจ

ต้นไม้ตัดสินใจประกอบด้วยบัพ (Node) และกิ่ง (Link) ที่ต่อกับบัพ บัพที่ปลายสุดเรียกว่าบัพใบ (leaf node) หรือเรียกย่อๆว่าใบ บัพแสดงคุณสมบัติและกิ่งแสดงค่าของคุณสมบัตินั้น ใบ (leaf) แสดงประเภท การสร้างต้นไม้ตัดสินใจทำได้โดยสร้างบัพทีละบัพเพื่อตรวจสอบคุณสมบัติของตัวอย่าง แล้วแยกตัวอย่างลงตามค่าของกิ่ง ทำจนตัวอย่างในใบแต่ละใบอยู่ในประเภทเดียวกัน

การเรียนรู้แบบเบส์ (Bayesian Learning)

การเรียนรู้แบบเบย์ เป็นวิธีการเรียนรู้ที่ใช้หลักการของความน่าจะเป็น ซึ่งมีพื้นฐานมาจากทฤษฎีของเบย์ (Bayes theorem) เข้ามาช่วยในการเรียนรู้ จุดมุ่งหมายก็เพื่อต้องการสร้างโมเดลที่อยู่ในรูปของความน่าจะเป็น ซึ่งเป็นค่าที่บันทึกได้จากการสังเกต จากนั้นนำโมเดลมาหาว่าสมมติฐานใดถูกต้องที่สุดโดยใช้ความน่าจะเป็นเข้ามาช่วย ความรู้ก่อนหน้า หมายถึง ความรู้ที่เรามีเกี่ยวกับสมมติฐานแต่ละตัวก่อนที่เราจะเก็บข้อมูล เมื่อใช้งานเราจะนำความน่าจะเป็นของข้อมูลที่เก็บได้มา

ปรับสมมติฐานซ้ำอีกครั้ง ข้อดีของวิธีการเรียนรู้แบบนี้ก็คือเราสามารถใช้ข้อมูลและความรู้ก่อนหน้า (Prior knowledge) เข้ามาช่วยในการเรียนรู้ได้ซึ่งพบว่าวิธีนี้ให้ประสิทธิภาพในการเรียนรู้ได้ดีไม่ด้อยกว่าวิธีการเรียนรู้ประเภทอื่น

การนำวิธีการเรียนรู้เบย์อย่างง่ายไปใช้ มีวิธีการดังต่อไปนี้ คือ

- หาค่าความน่าจะเป็นของค่าที่พบในแต่ละกลุ่มโดยนำค่า P(a1, a2, …, an | vj) มาคูณกับค่าความน่าจะเป็นของกลุ่มนั้นๆ คือ P(vj) ได้เท่ากับ v_{NB}
- นำค่าที่ได้ มาเปรียบเทียบกัน กลุ่มที่มีค่าความน่าจะเป็นสูงสุด คือ คำตอบ

ดังนั้นเราจะได้ว่าวิธีการจำแนกประเภทแบบเบย์อย่างง่าย ดังสมการในรูปที่ 2.41

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \times \prod_{i=1}^{n} P(a_i \mid v_j)$$

รูปที่ 2.41 สมการวิธีจำแนกประเภทแบบเบย์อย่างง่าย

ตารางที่ 2.2 ชุดตัวอย่างสอนสำหรับการเรียนรู้เบย์อย่างง่าย

ประเภท

คุณสมบัติ → / ค่า

Name	Hair	Height	Weight	Lotion	Result
Sarah	blonde	average	light	no	sunburned
Dana	blonde	tall	average	yes	none
Alex	brown	short	average	yes	none
Annie	blonde	short	average	no	sunburned
Emily	red	average	heavy	no	sunburned
Pete	brown	tall	heavy	no	none
John	brown	average	heavy	no	none
Katie	blonde	short	light	Yes	none

สมมติว่าตัวอย่างที่ต้องการจำแนกประเภท คือ ผลลัพธ์ในตารางที่ 2.2

ตารางที่ 2.3 ชุดตัวอย่างข้อมูลที่ต้องการจำแนก

Name	Hair	Height	Weight	Lotion	Result
Judy	blonde	average	heavy	no	?

ค่าความน่าจะเป็นของประเภท Sunburned และ none คำนวณได้จากสมการในรูปที่ 2.41 กรณีความน่าจะเป็นของประเภท Sunburned แทนด้วย v_{NB} = + จะได้ว่า

$$P(+)P(\text{blonde}|+)P(\text{average}|+)P(\text{heavy}|+)P(\text{no}|+)=\frac{3}{8}x\frac{2}{3}x\frac{2}{3}x\frac{1}{3}x\frac{3}{3}=\frac{1}{8}$$

โดยที่ P(+) หมายถึง ความน่าจะเป็นของประเภท Sunburned ซึ่งผลลัพธ์ที่ได้เป็นประเภท Sunburned จำนวน 3 คน จากทั้งหมด 8 คน

P(blonde|+) หมายถึง ความน่าจะเป็นของคนที่มีผมสี blonde ในประเภท Sunburned ซึ่งมีจำนวน 2 คน จากทั้งหมด 3 คน

P(average|+) หมายถึง ความน่าจะเป็นของคนที่มีความสูงเฉลี่ย ในประเภท Sunburned ซึ่งมีจำนวน 2 คน จากทั้งหมด 3 คน

P(heavy|+) หมายถึง ความน่าจะเป็นของคนที่มีน้ำหนักในประเภท Sunburned ซึ่งมีจำนวน 1 คน จากทั้งหมด 3 คน

P(no|+) หมายถึง ความน่าจะเป็นของคนที่ไม่มีโลชัน ในประเภท Sunburned ซึ่งมีจำนวน 3 คน จากทั้งหมด 3 คน

กรณีความน่าจะเป็นของประเภท none แทนด้วย v_{NB} = - จะได้ว่า

$$P(-)P(\text{blonde}|-)P(\text{average}|-)P(\text{heavy}|-)P(\text{no}|-)=\frac{5}{8}x\frac{2}{5}x\frac{1}{5}x\frac{2}{5}x\frac{2}{5}=\frac{1}{125}$$

โดยที่ P(-) หมายถึง ความน่าจะเป็นของประเภท none ซึ่งผลลัพธ์ที่ได้เป็นประเภท none จำนวน 5 คน จากทั้งหมด 8 คน

P(blonde|-) หมายถึง ความน่าจะเป็นของคนที่มีผมสี blonde ในประเภท none ซึ่งมีจำนวน 2 คน จากทั้งหมด 5 คน

P(average|-) หมายถึง ความน่าจะเป็นของคนที่มีความสูงเฉลี่ย ในประเภท none ซึ่งมีจำนวน 1 คน จากทั้งหมด 5 คน

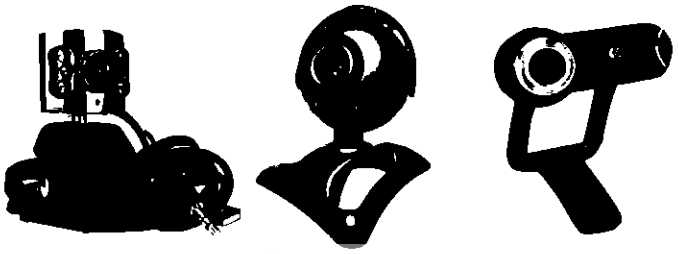
P(heavy|-) หมายถึง ความน่าจะเป็นของคนที่มีน้ำหนักในประเภท none ซึ่งมีจำนวน 2 คน จากทั้งหมด 5 คน

P(no|-) หมายถึง ความน่าจะเป็นของคนที่ไม่มีโลชัน ในประเภท none ซึ่งมีจำนวน 2 คน จากทั้งหมด 5 คนดังนั้นคำตอบที่ได้ คือ v_{NB} = + หมายถึง ประเภท Sunburned เนื่องจากประเภท Sunburned มีค่าความน่าจะเป็นมากที่สุด

2.3.3 กล้อง และ การเชื่อมต่อกล้อง Web Camera

เว็บแคม (Webcam ย่อมาจาก Web Camera) คือ กล้องวีดีโอที่ถ่ายทอดภาพนิ่งหรือภาพวิดีโอผ่านระบบเครือข่าย เว็บไซต์ โปรแกรม ถือเป็นอุปกรณ์นำข้อมูล (อินพุต) ที่สามารถจับ

ภาพเคลื่อนไหวให้ไปปรากฏในจอภาพ และสามารถส่งภาพเคลื่อนไหวหรือภาพนิ่งนี้ไปให้คนอีกฟากหนึ่งเห็นตัวเราเคลื่อนไหวได้เหมือนอยู่ต่อหน้า ปัจจุบันมีทั้งแบบที่เชื่อมต่อกับคอมพิวเตอร์ผ่านสายยูเอสบี และเชื่อมต่อแบบไร้สาย โดยตัวอย่างของกล้อง Web Camera แสดงในรูปที่ 2.42



รูปที่ 2.42 ภาพแสดงกล้อง Web Cam

การนำทฤษฎีต่างๆที่กล่าวมาทั้งหมดจะสามารถนำมาใช้เป็นแนวทาง ในการออกแบบหุ่นยนต์หลบหลีกสิ่งกีดขวางและค้นหาเป้าหมายอัตโนมัติ การออกแบบหุ่นยนต์จำเป็นที่จะต้องคำนึงถึงสิ่งต่าง ๆ มากมาย ไม่ว่าจะขนาด น้ำหนัก ประสิทธิภาพของหุ่นยนต์ เป็นต้น ซึ่งวิธีการออกแบบและสร้างหุ่นยนต์รวมถึงวิธีการทำงานของหุ่นยนต์ต่างๆนั้นจะอธิบายในบทต่อไป

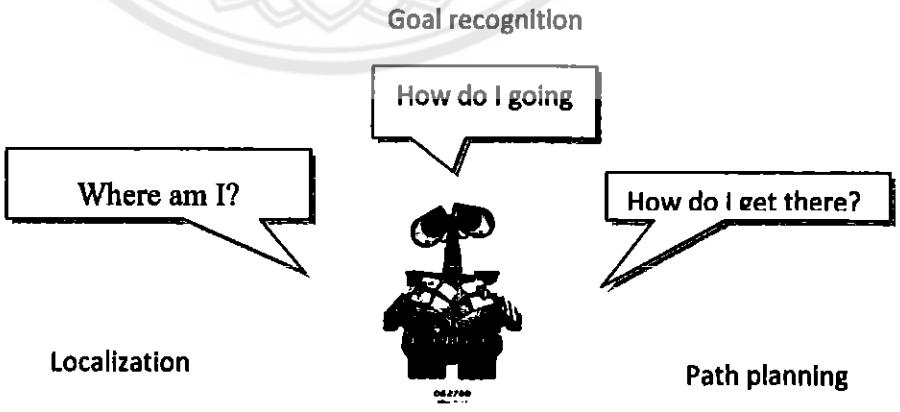
บทที่ 3
หลักการทำงาน

การออกแบบและการสร้างหุ่นยนต์กู้ภัยหลบหลีกสิ่งกีดขวางและค้นหาเป้าหมายอัตโนมัติ โดยการเริ่มออกแบบส่วนที่เป็นโครงสร้างหุ่นยนต์ เพื่อใช้ในการขับเคลื่อนเพื่อเข้าพื้นที่เสี่ยงอันตรายเพื่อค้นหาผู้ประสบภัย ดั้งนั้นการออกแบบหุ่นยนต์จึงต้องใช้หลักการทางด้านวิศวกรรมและหลักการในการใช้วัสดุและอุปกรณ์ที่เหมาะสมกับการทำงานของหุ่นยนต์ รวมถึงต้องคำนึงถึงเรื่องราคาของวัสดุอุปกรณ์ที่เลือกใช้ อายุการใช้งานและความมีเสถียรภาพในการเข้าช่วยเหลือ การวิเคราะห์ค่าความผิดพลาดที่เกิดขึ้น ความแม่นยำของข้อมูลที่ระบุตำแหน่งของผู้ประสบภัยจึงเป็นเรื่องที่สำคัญในการสร้างหุ่นยนต์ ดังนั้นจากหลักการและทฤษฎีในบทที่สองสามารถใช้ในการวิเคราะห์การเลือกใช้งานได้อย่างเหมาะสมเมื่อสร้างโครงสร้างหุ่นยนต์เรียบร้อยแล้วจึงเริ่มออกแบบระบบควบคุมหุ่นยนต์ โดยมีขั้นตอนที่จะกล่าวต่อไปนี้

การออกแบบหุ่นยนต์จะแบ่งออกเป็น 2 ส่วน คือ

1. การสร้างหุ่นยนต์
2. การออกแบบส่วนควบคุม แบ่งออกเป็น 2 ส่วนย่อย คือ
   - 2.1 การออกแบบระดับล่าง (Low Level)
   - 2.2 การออกแบบระดับบน (High Level)

แนวคิดเริ่มต้นของการสร้างหุ่นยนต์กู้ภัยอัตโนมัติ



รูปที่ 3.1 แนวคิดเริ่มต้นของการสร้างหุ่นยนต์กู้ภัยอัตโนมัติ

จากรูปที่ 3.1 จะเห็นได้ว่าหุ่นยนต์จะไม่รู้ว่าหุ่นยนต์อยู่ที่ไหน ไม่รู้ว่ากำลังจะไปทำอะไร และไม่รู้ว่าจะเคลื่อนที่ไปได้อย่างไร ดังนั้นการออกแบบจึงจำเป็นที่จะต้องสร้างหุ่นยนต์ที่มีความคล่องตัว สามารถเคลื่อนที่ได้ทุกสภาพพื้นผิว จึงจะไม่ทำให้หุ่นยนต์เกิดปัญหาในการเคลื่อนที่ และควบคุมด้วยระบบควบคุมทั้งสองส่วนโดยจะแบ่งการออกแบ่งเป็นส่วน ๆ ดังนี้

3.1 การออกแบบโครงสร้างหุ่นยนต์

3.1.1 หลักการออกแบบตัวหุ่นยนต์

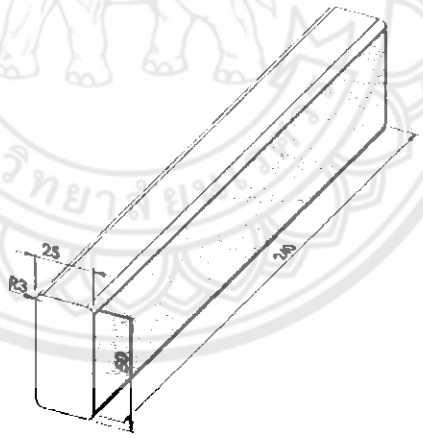
เนื่องจากโครงการนี้เป็นโครงการหุ่นยนต์กู้ภัย วัสดุที่เลือกใช้งานจึงต้องการที่มีความคงทนและแข็งแรง ส่วนประกอบหลักของหุ่นยนต์คือ ตัวฐานของหุ่นยนต์และล้อตีนตะขาบ

3.1.2 โครงสร้างของหุ่นยนต์

โครงสร้างของหุ่นส่วนใหญ่จะเป็นเหล็กหรืออลูมิเนียม เนื่องจากต้องการให้มีความแข็งแรง

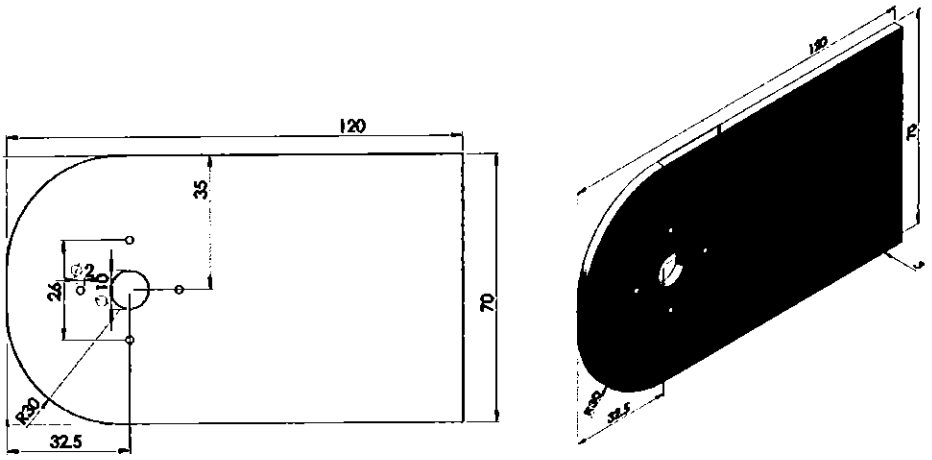
3.1.3 ออกแบบโครงล้อของหุ่นยนต์

โครงสร้างของล้อจะใช้เหล็กกระบอกสี่เหลี่ยมที่มีขนาด 2.5 ซม. × 5 ซม. มีขนาดยาว 25 ซม. ดังรูปที่ 3.2

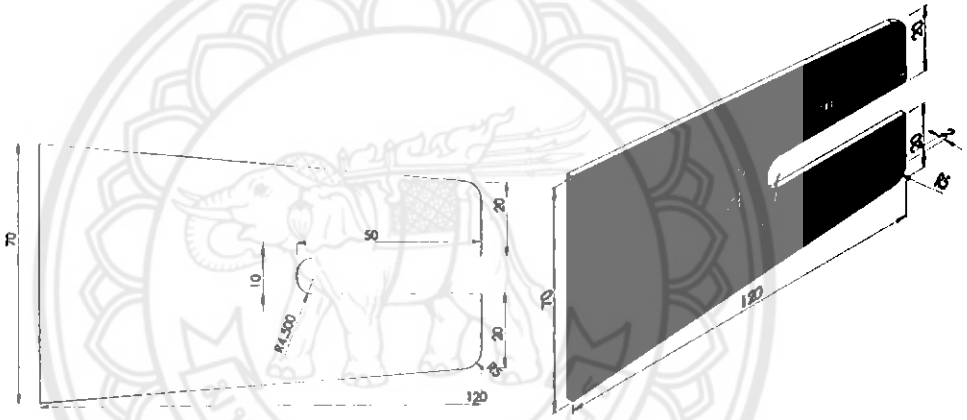


รูปที่ 3.2 เหล็กกระบอกสี่เหลี่ยมสำหรับทำโครงล้อ

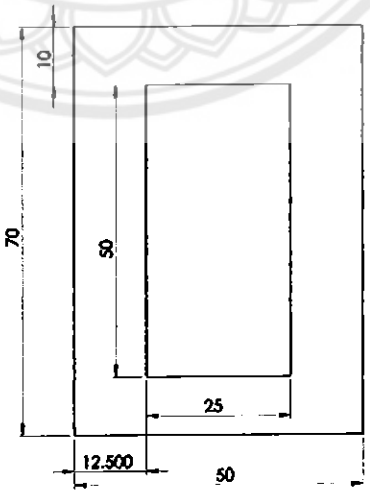
ตัวประกบล้อนั้นจะใช้แผ่นเหล็ก 2 อันมาประกบกันโดยตัดให้ได้ขนาดตามแบบดังรูป



รูปที่ 3.3 แผ่นเหล็กสำหรับทำที่ประกบล้อหน้า

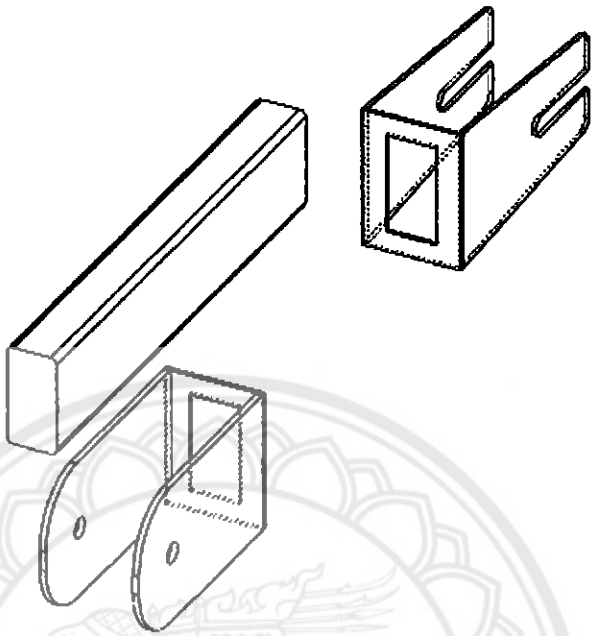


รูปที่ 3.4 แผ่นเหล็กสำหรับทำที่ประกบล้อหลัง

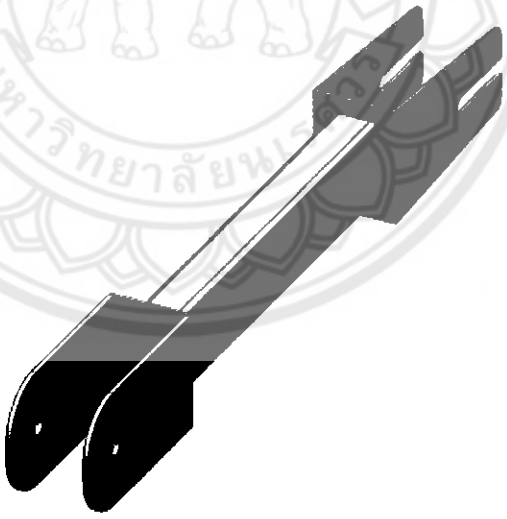


รูปที่ 3.5 แผ่นเหล็กไว้ยึดแผ่นประกบล้อ

เมื่อเราตัดแบบที่เราได้ออกแบบไว้เรียบร้อยแล้วก็จะนำมาประกอบกันโดยการเชื่อม

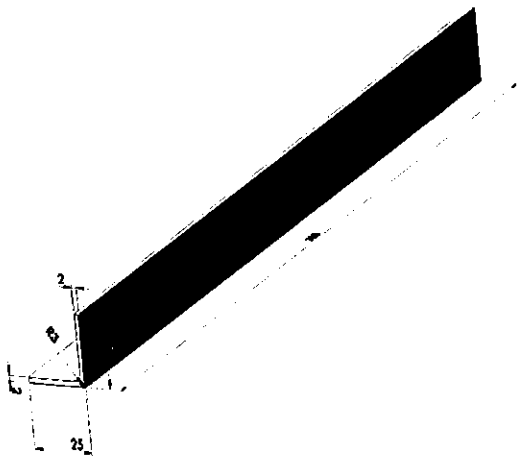


รูปที่ 3.6 การประกอบชิ้นส่วนของฐานล้อ

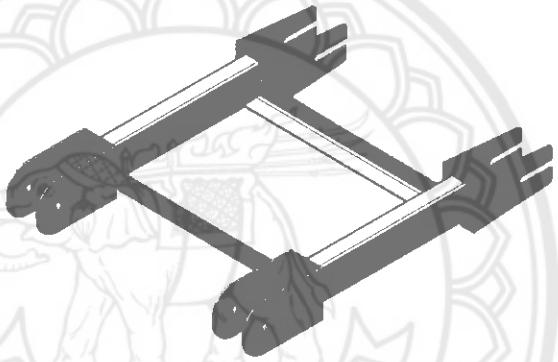


รูปที่ 3.7 รูปฐานล้อเมื่อประกอบแล้ว

เราจะทำชุดของฐานล้อจำนวน 2 ชุด จากนั้นนำเหล็กฉากมาทำเป็นตัวยึดระหว่างโครงล้อทั้ง 2 ข้างเข้าด้วยกันโดยตัดเหล็กฉากหนา 1 นิ้ว (25 มม.) ยาว 30 ซม.

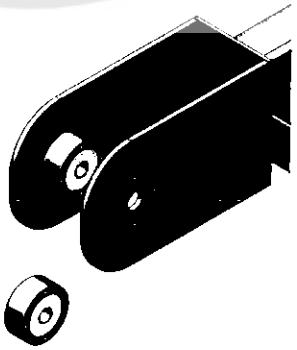


รูปที่ 3.8 เหล็กฉากที่จะนำไปยึดเป็นโครงของตัวหุ่นยนต์



รูปที่ 3.9 โครงสร้างฐานล้อของหุ่นยนต์

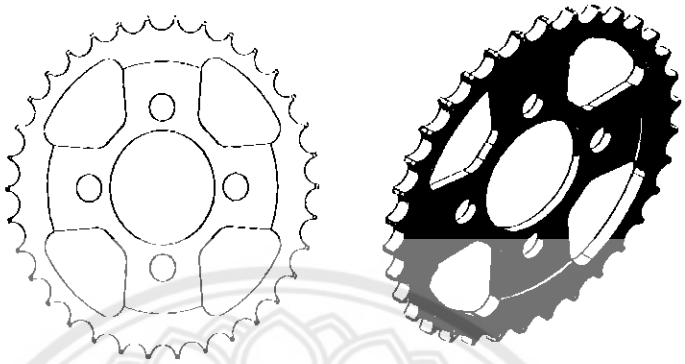
การติดตั้งบูทลูกปืนที่ล้อหน้า เพื่อลดแรงเสียดทานของล้อ จึงใช้บูทที่มี**ลูกปืนยึดเข้าไปกับแผ่นจับแขน**ของล้อ โดยใช้น๊อตเป็นตัวยึดระหว่างบูทกับแผ่นจับล้อ



รูปที่ 3.10 การติดตั้งชุดบูทกับแผ่นล้อหน้า

3.1.4 การออกบแบบล้อของหุ่นยนต์

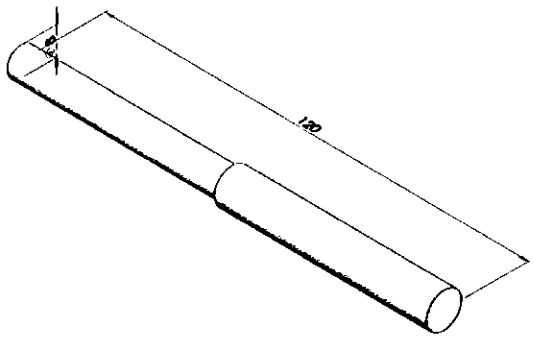
ล้อและแขนล้อของหุ่นยนต์จะใช้สเตอร์ของรถจักรยานยนต์ ขนาด 34 ฟัน (มีจำนวนฟัน 34 ฟัน) เบอร์ 428 ซึ่งจะต้องใช้กับโซ่เบอร์ 428 เช่นกัน ดังรูปที่ 3.11



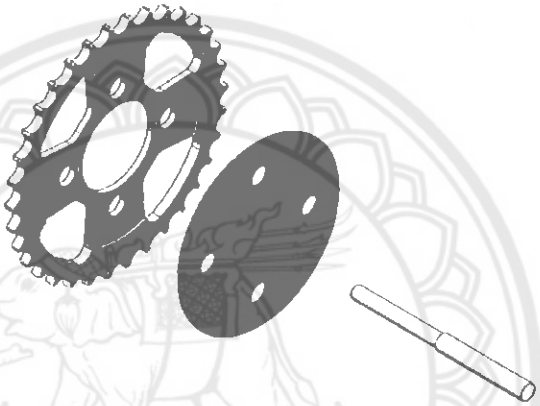
รูปที่ 3.11 สเตอร์รถจักรยานยนต์ขนาด 34 ฟัน

รูปที่ 3.12 รูปจริงของสเตอร์รถจักรยานยนต์ขนาด 34 ฟัน

ล้อหน้า การประกอบล้อหน้าจุมีคุมที่ยึดติดกับเพลา เพื่อที่จะใช้เพลาเป็นตัวขับเคลื่อน

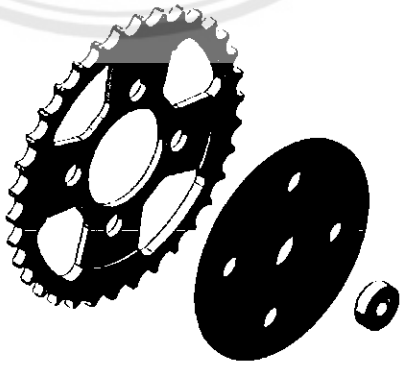


รูปที่ 3.13 เพลาล้อหน้า

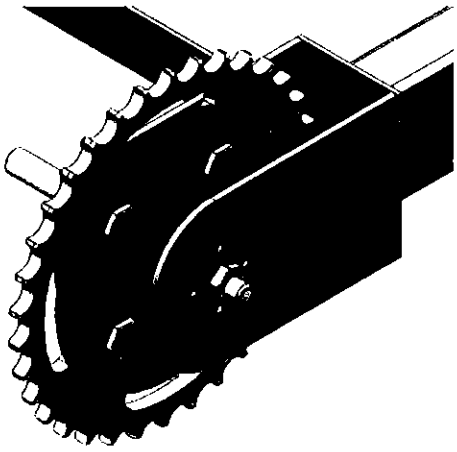


รูปที่ 3.14 การประกอบเพลาเข้ากับดุมและล้อ

ล้อหลัง การประกอบล้อหลังจะใช้ดุมมีประกบกับสเตอร์ ในดุมล้อนั้นจะมีลูกปืนอยู่ข้างใน เพื่อที่จะลดแรงเสียดทานของล้อกันเพลา

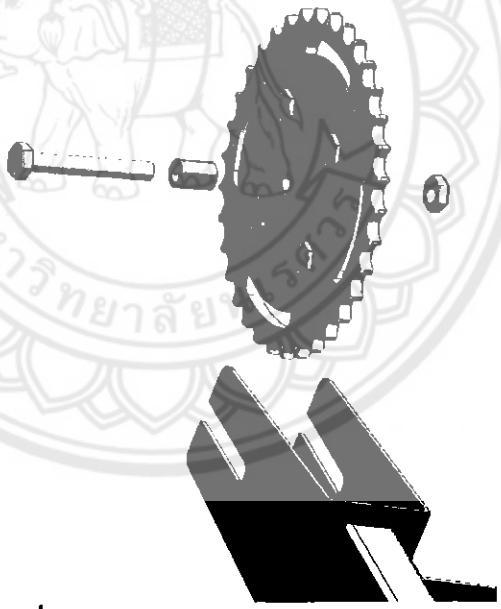


รูปที่ 3.15 ชุดส่วนประกอบของล้อหลัง



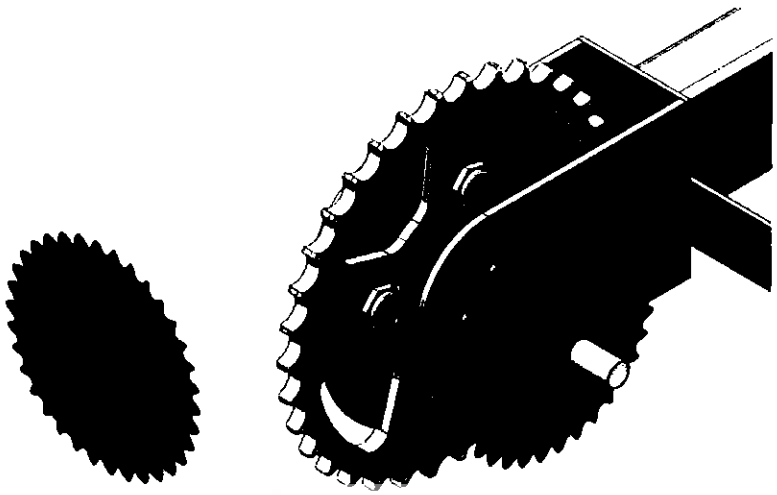
รูปที่ 3.16 การประกอบล้อหน้าเข้ากับโครงของหุ่นยนต์

การประกอบล้อหลังของหุ่นยนต์ จะใช้บูทที่มีรูขนาด 8 มิลลิเมตร เพื่อไม่ให้ล้อขยับไปขยับมาได้ และยังช่วยให้ล้อมีความแน่นเมื่อไขน็อต

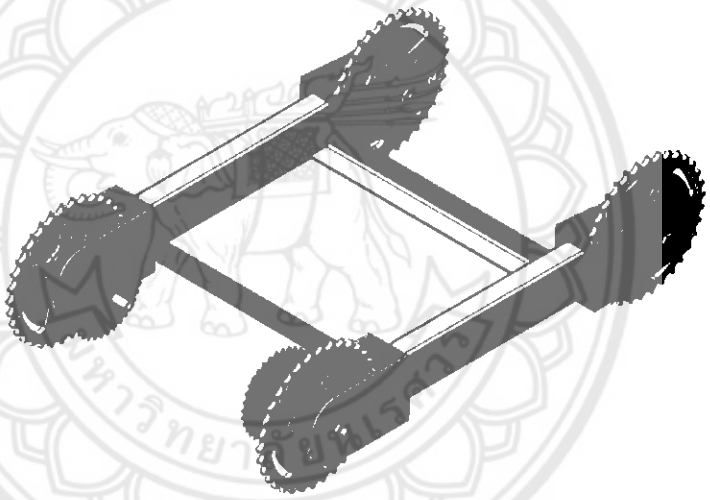


รูปที่ 3.17 การประกอบล้อหลังของหุ่นยนต์

สำหรับเฟืองโซ่ที่จะใช้ขับเคลื่อนนั้นจะเป็นเฟืองโซ่ขนาดเล็ก มีขนาด 27 ฟัน ซึ่งโว่ที่ใช้นั้นจะเป็นโซ่ขนาดเล็ก



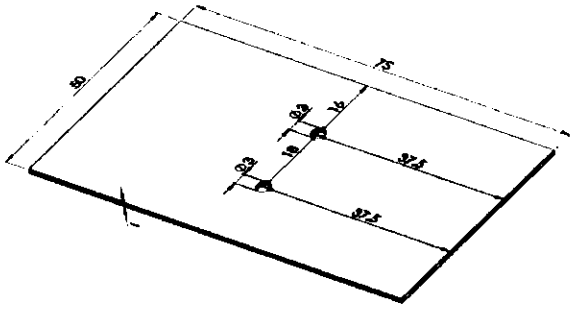
รูปที่ 3.18 เฟืองโซ่ขับเคลื่อนและการติดตั้งเฟืองโซ่



รูปที่ 3.19 โครงฐานล้อของหุ่นยนต์

3.1.5 การออกแบบล้อตีนตะขาบ

ล้อตีนตะขาบเป็นชิ้นส่วนในการขับเคลื่อนรถถัง เหมาะสำหรับพื้นผิวที่มีละกษณะตะปุ่มตะป่ำ ขรุขระ มีสิ่งกรีดขวาง หรือกรณีที่ล้อทั่วไปวิ่งได้ลำบาก และสามารถเปลี่ยนทิศทาง ณ จุดที่ต้องการได้โดยทันที แต่มีข้อเสียตรงที่เวลาในการประดิษฐ์ ถ้าประกอบไม่ดีอาจทำให้ไม่แข็งแรงและหักได้ ส่วนประกอบของล้อตีนตะขาบ ออกแบบสำหรับชิ้นส่วนแผ่นเหล็กและยางที่สำหรับใช้เป็นล้อมีขนาดกว้าง × ยาว เท่ากับ 5 ซม. × 7.5 ซม. ดังรูป



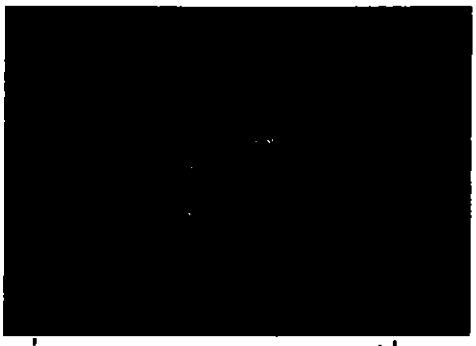
รูปที่ 3.20 ภาพแสดงการออกแบบส่วนประกอบของล้อตีนตะขาบ

โซ่ที่จะนำมาทำเป็นตีนตะขาบนั้นจะใช้โซ่เบอร์ 428 โดยจะนำโซ่มาเชื่อมกับแผ่นเหล็กสี่เหลี่ยมที่ได้ออกแบบไว้ข้างต้น

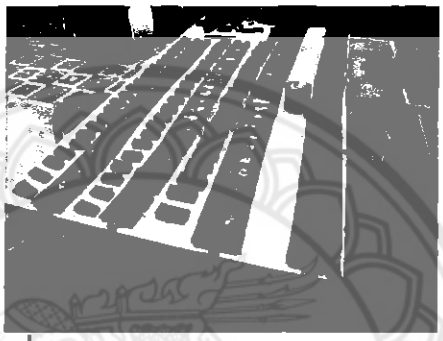
รูปที่ 3.21 แผ่นเหล็กที่เชื่อมกับโซ่

รูปที่ 3.22 ชิ้นส่วนของโซ่ตีนตะขาบ

จากนั้นจะนำแผ่นยางที่ตัดมาจากสายพานของรถจักรยานยนต์ มาติดกับแผ่นเหล็ก เพื่อเพิ่มแรงเสียดทานให้กับล้อของหุ่นยนต์ โซ่ที่ที่เชื่อเสร็จแล้วโดยใช้กาวยางติด



รูปที่ 3.23 สายพานรถจักรยานยนต์ที่ตัดแล้ว



รูปที่ 3.24 ส่วนประกอบของล้อตีนตะขาบ

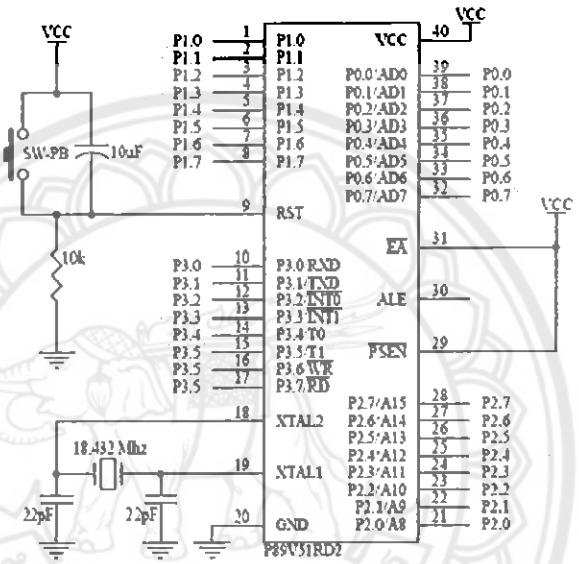
เมื่อประกอบโซ่ตีนตะขาบเรียบร้อยแล้วก็จำนำไปใส่กับล้อที่ได้ออกแบบไว้

รูปที่ 3.25 ล้อเมื่อใส่โซ่ตีนตะขาบ

3.2 การออกแบบระดับล่าง (Low Level)

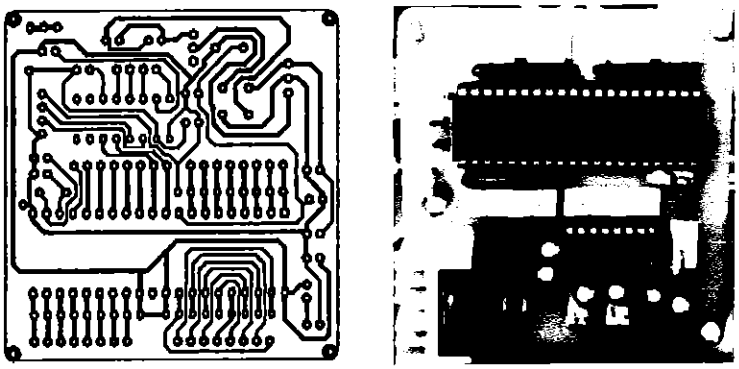
3.2.1 ชุดควบคุมการทำงานโดยไมโครคอนโทรลเลอร์

ชุดควบคุมโดยไมโครคอนโทรลเลอร์ ทำหน้าที่ในการรับคำสั่งจากเครื่องคอมพิวเตอร์เพื่อไปสั่งงานควบคุมตัวขับมอเตอร์ โดยจะใช้มอเตอร์ทั้งหมด 2 ตัว ซึ่งชุดควบคุมมอเตอร์สามารถที่จะควบคุมมอเตอร์ได้ 1 ตัวต่อหนึ่งพอร์ตควบคุม โดยชุดควบคุมมอเตอร์จะถูกสั่งงานจากพอร์ทไมโครคอนโทรลเลอร์ 1 พอร์ทต่อ 2 ชุดควบคุม



รูปที่ 3.26 วงจรไมโครคอนโทรลเลอร์แสดงพอร์ทต่าง ๆ

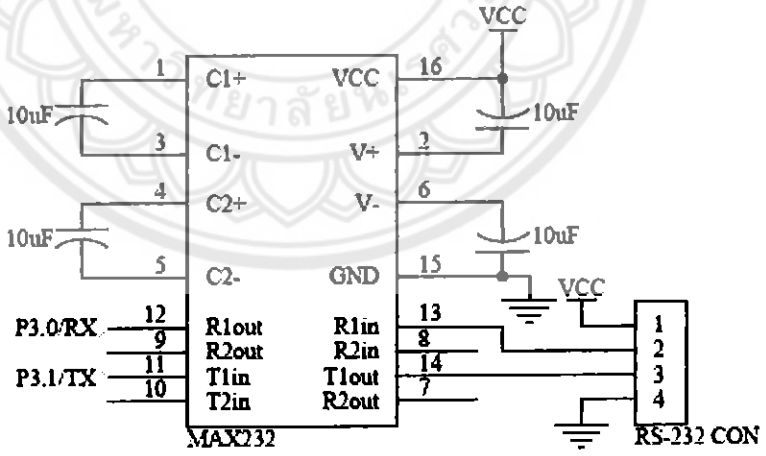
จากรูปไมโครคอนโทรลเลอร์เบอร์ MCS-51 P89V51RD2 จะมีพอร์ทอยู่ 4 พอร์ต ซึ่งพอร์ทที่ใช้ในการควบคุมมอเตอร์คือพอร์ท 0 และพอร์ท 2 ซึ่งเป็นพอร์ทเอาต์พุตของไมโครคอนโทรลเลอร์โดยจะต่อกับชุดควบคุมมอเตอร์ และพอร์ต 1 เป็นการติดต่อกับชุดเซนเซอร์ตรวจจับวัตถุ แบบ I^2C BUS โดยการใช้ I^2C BUS จะใช้ เพียง 2 บิต ของพอร์ต 1 รูปตัวอย่างบอร์ดไมโครคอนโทรลเลอร์ที่ประกอบเสร็จจากการออกแบบวงจรด้วยโปรแกรม ญrotel 99SE และกัดลายปริ้นด้วยวิธีการใช้กรดกัดปริ้นกัดในบริเวณที่ไม่มีหมึกของลายวงจรที่ได้ออกแบบไว้ ดังตัวอย่างในรูปที่ 3.27



รูปที่ 3.27 บอร์ดไมโครคอนโทรลเลอร์และอุปกรณ์ประกอบพร้อมลายปริ้นจากโปรแกรม Protel 99SE

3.2.2 การติดต่อพอร์ทอนุกรม

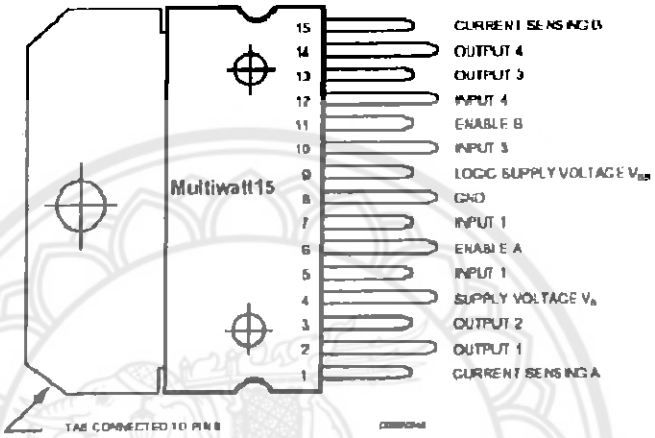
ดังที่ได้กล่าวไปแล้วว่าระบบ RS-232 (Computer Port) จะใช้ระดับแรงดันไฟฟ้าต่างกับระบบ TTL ดังนั้นในการนำข้อมูลจากคอมพิวเตอร์ไปประมวลผลในไมโครคอนโทรลเลอร์ จะต้องมีการแปลงระดับสัญญาณซึ่งในการแปลงระดับสัญญาณได้ใช้ไอซี MAX 232 ซึ่งไอซีจะรับสัญญาณ RS-232 และแปลงเป็น TTL และในขณะเดียวกันก็สามารถแปลงระบบ TTL เป็น RS-232 ได้เช่นกัน การจัดการขาและการใช้งานไอซีแสดงดังรูป



รูปที่ 3.28 วงจรติดต่อผ่านพอร์ทสื่อสารอนุกรม RS-232

3.2.3 ชุดควบคุมมอเตอร์แบบ H-Bridge

การเปลี่ยนทิศทางการหมุนของดีซีมอเตอร์แบบแม่เหล็กถาวร ทำได้โดยการสลับขั้วแหล่งจ่ายไฟฟ้าที่ต่อเข้ากับตัวมอเตอร์และในทางปฏิบัติจะใช้วงจรอิเล็กทรอนิกส์ที่เรียกว่า H Bridge เป็นตัวจัดการการทำงาน ซึ่งในปัจจุบันนี้ได้มีการผลิต IC ขับมอเตอร์แบบ H Bride ขึ้นมามากมาย ในส่วนของโครงงานนี้ได้ศึกษา DUAL FULL-BRIDGE DRIVER L298 ซึ่งมีตัวถังและวงจรภายในดังรูป

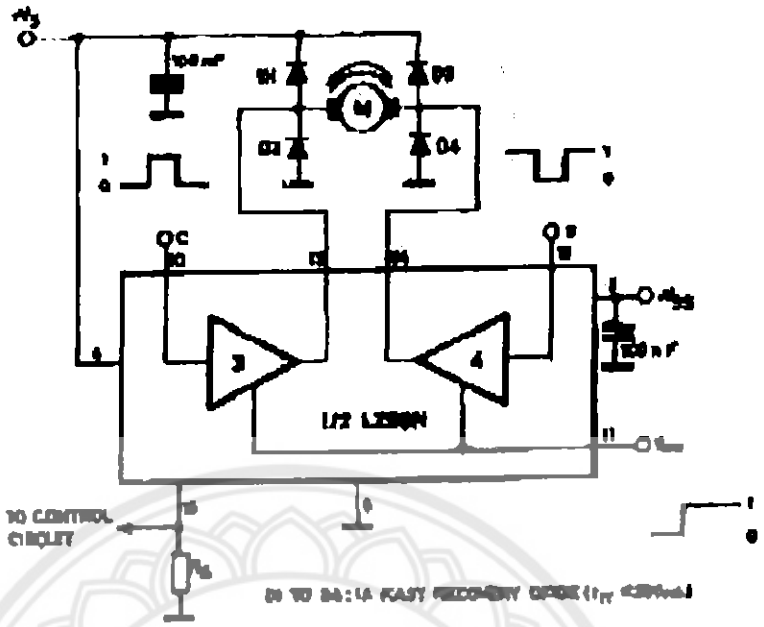


รูปที่ 3.29 การจัดการขาของไอซี L298

จากวงจรภายในจะเห็นว่า L298 สามารถขับโหลดได้ 2 ช่องและสามารถรับสัญญาณ ควบคุมแบบ TTL (Transistor Transistor Logic) เพื่อที่จะควบคุมทิศทางการไหลของกระแส นอกจากนี้ยังมีขา Enable เป็นตัวตัดสินว่าจะให้โหลดที่ต่อยู่ทำงานหรือไม่โดยไม่สนใจสัญญาณควบคุม

ขา Emitter ของ Transistor ทั้งสองข้างของแต่ละ bridge จะต่อกับตัวต้านทานภายนอกเพื่อที่จะใช้ในการกำหนดค่ากระแสไฟฟ้าที่ไหลได้โดยหากเกินกว่าที่วงจรและโหลดที่ต่ออยู่สามารถที่จะรับได้ก็อาจจะมีวงจรเพื่อที่ทำการ Disable การทำงานของโหลดได้

จะเห็นว่าวงจรภายในไม่มีการต่อไดโอดเพื่อให้กระแสไฟฟ้าไหลได้ เมื่อโหลดที่ใช้เป็นตัวเหนี่ยวนำ (มอเตอร์รีเลย์) เนื่องจากสำหรับโหลดที่เป็นตัวเหนี่ยวนำค่ากระแสไฟฟ้าจะไม่สามารถเปลี่ยนเป็นศูนย์ได้ในเวลาทันทีทันใด (คล้ายกับกรณีที่ตัวเก็บประจุไม่สามารถเปลี่ยนค่าความต่างศักย์ตกคร่อมได้อย่างทันทีทันใด) จึงต้องมีการต่อไดโอดเพื่อให้กระแสไฟฟ้าไหลในกรณีที่เราเปิดสวิตช์แต่มอเตอร์ยังคงหมุนอยู่โดยที่การต่อไดโอดภายนอกแสดงดังรูป

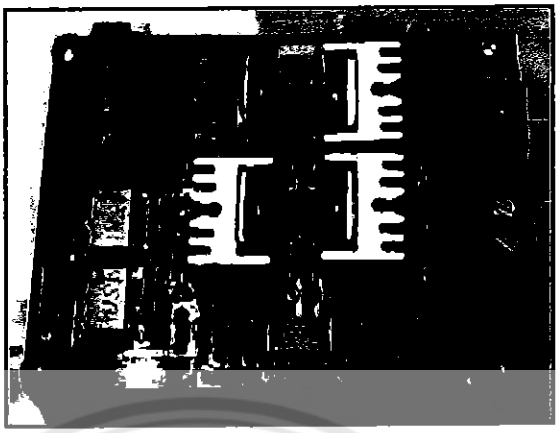


รูปที่ 3.30 การต่อวงจรใช้งานแบบ 1 ช่องของ L298

สำหรับโครงงานนี้ได้ทำการดัดแปลงวงจร H-Bridge โดยต้องการให้สามารถควบคุมทิศทางได้ โดยการใช้สายควบคุม 3 เส้น และใช้สัญญาณ Pulse Width Modulation ต่อเข้ากับขา Enable เพื่อให้ได้ระดับความเร็วตามต้องการดังรูป

รูปที่ 3.31 บอร์ดควบคุมมอเตอร์ H-Bridge

ชุดควบคุมทิศทางการหมุนของมอเตอร์กระแสตรง



รูปที่ 3.32 Module ควบคุมทิศทางการหมุนของมอเตอร์ขับเคลื่อน

เหตุผลที่เลือกใช้ H-Bridge รุ่น SE-HB40-1

1. หากมีการขับเกินกำลังจนมอสเฟทพัง ให้ตรวจสอบโดยการใช้มิเตอร์วัดการลัดวงจรของมอสเฟททั้ง 4 ตัว หากตัวใดเสียขั้วมอสเฟทจะลัดวงจร ให้เปลี่ยนมอสเฟทได้เลย วงจรส่วนอื่นปลอดภัย ไม่จำเป็นต้องตรวจสอบ

2. จากการทดลองไอซีควบคุมมอเตอร์สำเร็จรูป เบอร์ L298D ร่วมกับ Relay 10 A ไม่เหมาะสมกับการสวิตซ์ซึ่งจากการบังคับแบบไม่ต่อเนื่อง เช่น เมื่อกดบังคับให้หุ่นยนต์เดินหน้าและหยุดสลับกันอย่างรวดเร็วจะทำให้หน้าสัมพัสของ Relay เกิดอุณหภูมิสูงทำให้ส่วนหน้าสัมผัสโลหะติดกันไม่แยกจากกันทำให้ควบคุมหุ่นยนต์ไม่ได้จึงเลือกใช้โมดูล H-Bridge รุ่น SE-HB40-1 แทน

3. จากแผ่นปริ้นที่หาซื้อตามท้องตลาดเพื่อนำมาออกแบบวงจรเอง จะทำให้เกิดปัญหาเมื่อวงจรเกิดความร้อนและความถี่สูงระหว่าง ลายทองแดงทำให้ประสิทธิภาพของวงจรที่ออกแบบเองเกิดปัญหาความผิดพลาดของการส่งสัญญาณบ่อยครั้งจากการทดลอง และที่สำคัญคือลายทองแดง บางมากจะต้องใช้สายไฟเดินบริเวณที่มีกระแสไฟฟ้าไหลผ่านมากๆ

4. สามารถทนกระแสได้ถึง 40 A ซึ่งแตกต่างจากวงจรที่ออกแบบขึ้นเอง ที่ทนกระแสได้น้อย

โดยการออกแบบโปรแกรมการควบคุมจะเป็นไปตามตารางที่ 3.1 โดยโปรแกรมจะรับอินพุตจากส่วนควบคุมระดับบน ถ้าส่วนควบคุมระดับบนส่งข้อมูลมาสั่งงานส่วนควบคุมระดับล่าง (MCU) ก็จะส่งแรงดันเอาต์พุต ประมาณ 5 V ตามจำนวนบิตที่วงจรควบคุมทิศทางมอเตอร์ H-Bridge รุ่น SE-HB40-1 ดังนี้

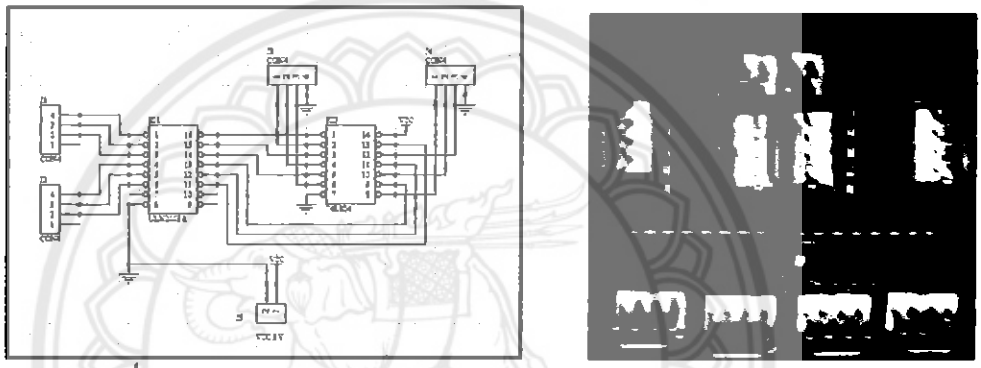
ตารางที่ 3.1 ค่าสถานะการเคลื่อนที่ของหุ่นยนต์

สถานะ	Wheel Left				Wheel Right				Hex Code
	IN1	IN2	EN	GND	IN1	IN2	EN	GND	
Port Microcontrollers (Bit)	7	6	5	4	3	2	1	0	
Forward Robot	0	1	1	0	0	1	1	0	0x66
Backward Robot	1	0	1	0	1	0	1	0	0xAA
Turn Left Robot	0	0	0	0	0	1	1	0	0x06
Turn Right Robot	0	1	1	0	0	0	1	0	0x60
Turn Short Left Robot	1	0	1	0	0	1	1	0	0xA6
Turn Short Right Robot	0	1	1	0	1	0	1	0	0x6A

จากตารางที่ 3.1 จะทำให้ทราบได้ว่าถ้าต้องการให้หุ่นยนต์เคลื่อนที่ไปในทิศทางใด ก็ต้องเขียนโปรแกรมควบคุมไมโครคอนโทรลเลอร์ให้ส่งแรงดันออกที่พอร์ตของไมโครคอนโทรลเลอร์ ซึ่งในที่นี้การควบคุมจะส่งเอาต์พุตออกที่พอร์ต 2 ของบอร์ดไมโครคอนโทรลเลอร์ตามบิตแต่ละบิตของพอร์ตไมโครคอนโทรลเลอร์ ที่ช่อง สถานะ IN1 และ IN2 คือ ขาควบคุมทิศทางการหมุนของมอเตอร์ และ EN คือ ขา Enable มีหน้าที่ในการเปิดการทำงานของวงจรควบคุมการหมุนของมอเตอร์และยังเป็นขาที่ใช้ปรับความเร็วของมอเตอร์ในการใช้วิธีการ Pulse Wide Modulation ตามที่ได้กล่าวแล้วในบทที่ 2 และขา GND คือ ขากราวด์ที่ต้องต่อร่วมกับบอร์ดไมโครคอนโทรเลอร์ด้วยเพื่อให้กระแสไหลได้ครบวงจรหรือมองเป็นกราวด์ร่วมนั้นเอง เมื่อวงจรควบคุมทิศทางการหมุนของมอเตอร์ ได้รับการส่งเอาท์พุตจากไมโครคอนโทรลเลอร์เรียบร้อยมอเตอร์จะหมุนโดยมีจุดต่อมอเตอร์ และรับแรงดันจากภายนอกเพื่อให้มอเตอร์ได้รับกระแส โดยแหล่งจ่ายแรงดันจากภายนอกนี้จะเลือกใช้แบตเตอรี่รถจักรยานยนต์ที่มีแรงดัน 12 VDC 3A/hr ซึ่งมีราคาถูกและสามารถประจุไฟเข้าได้อย่างสะดวก แต่มีน้ำหนักมากถ้าเลือกเป็นแบตเตอรี่แบบใช้น้ำกลั่น และในที่นี้เลือกแบบแบตเตอรี่แห้ง และการเลือกแรงดันที่จ่ายให้มอเตอร์นั้นพิจารณาโดยมอเตอร์เป็นมอเตอร์กระแสตรงที่ทนแรงดันสูงสุดไม่เกิน 12 V

เมื่อออกแบบรหัสควบคุมเพื่อเตรียมการเขียนโปรแกรมแล้วนั้น การออกแบบวงจรเชื่อมต่อระหว่างไมโครคอนโทรลเลอร์กับชุดควบคุมทิศทางการหมุนของมอเตอร์นั้น กระแสที่ออกจากพอร์ตของไมโครคอนโทรลเลอร์ไม่เพียงพอต่อการควบคุมชุดควบคุมทิศทางการหมุนของมอเตอร์เนื่องจากชุดควบคุมทิศทางการหมุนของมอเตอร์ต้องการกระแสเข้าที่ขา IN1, IN2, EN ที่กระแส สูงกว่า 8 mA แต่ไมโครคอนโทรลเลอร์จ่ายกระแสออกพอร์ตได้เพียง 1 mA เท่านั้น

จึงต้องออกแบบวงจรขยายกระแสให้กับบอร์ดควบคุมทิศทางการหมุนของมอเตอร์โดยใช้อุปกรณ์ดังต่อไปนี้ คือ IC ULN2003A และ IC Not Gate 7404 โดย IC ULN2003A โดยไอซีตัวนี้จะทำหน้าที่ขยายกระแสให้ ให้กับบอร์ดควบคุมทิศทางการหมุนของมอเตอร์และต่อกับ IC Not Gate 7404 เพื่อกลับบิตและส่งเข้าให้กับบอร์ดควบคุมทิศทางการหมุนของมอเตอร์ การทำงานของวงจรขยายกระแสมีลักษณะเหมือนกับวงจรบัพ-เฟอร์บางวงจรหรือ IC สำเร็จรูปบางตัวเช่น IC 74LS245 ทำหน้าที่ขยายกระแสจากพอร์ตของไมโครคอนโทรลเลอร์ หรืออาจจะใช้วิธีการ ต่อตัวต้านทานอนุกรมกับแหลงจ่ายแรงดัน 5 Volts เพื่อทำให้เกิดกระแสที่แต่ละบิตสูงขึ้นแต่กระแสก็ยังไม่เพียงพอ ตัวอย่างการต่อวงจร โดยใช้ IC ULN2003A และ IC Not Gate 7404 ดังรูปที่ 3.33

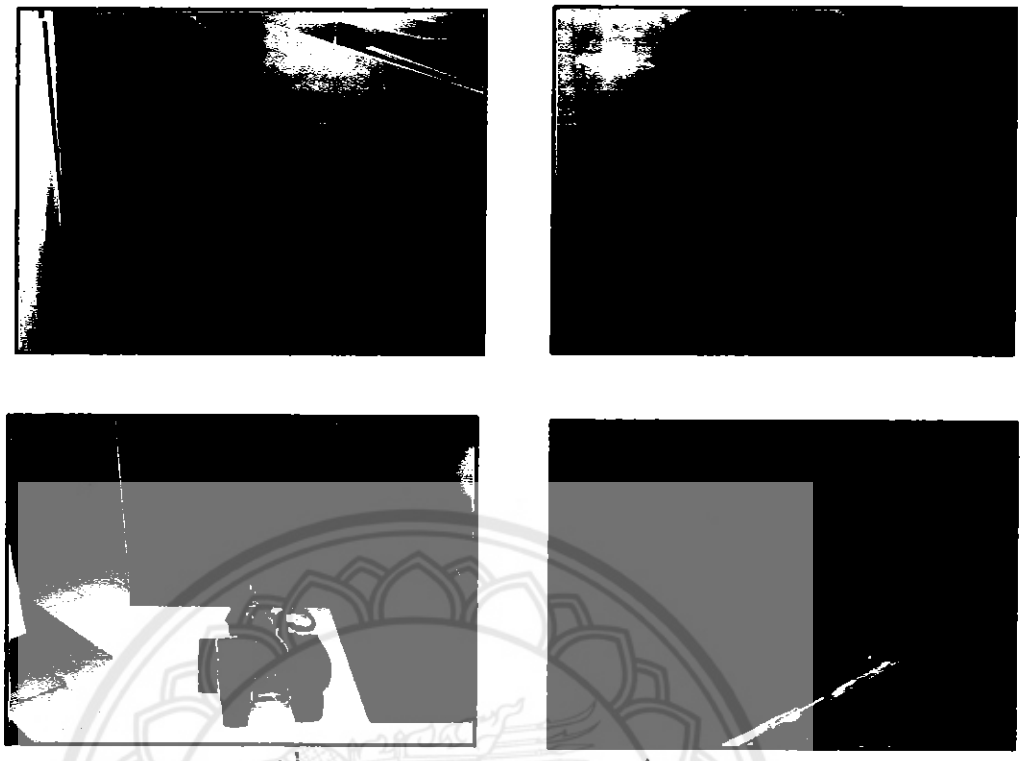


รูปที่ 3.33 วงจรขยายกระแสด้วย IC ULN2003A และ IC Not Gate 7404

เมื่อประกอบอุปกรณ์ดังแสดงในรูปที่ 3.33 เรียบร้อย ก็จะสามารถนำไปใช้งานในควบคุมการหมุนของมอเตอร์ได้โดยกระแสที่ได้จากวงจร Buffer ที่ออกแบบขึ้นให้กระแสสูงถึง 500 mA จากการแสดงคุณสมบัติจากเอกสารแสดงคุณสมบัติ

3.2.4 การเชื่อมต่ออุปกรณ์ตรวจจับวัตถุ

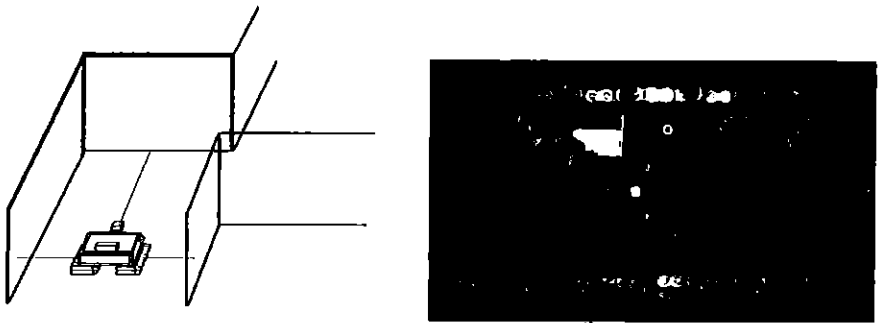
อุปกรณ์ตรวจจับวัตถุ (Sensor) เป็นอุปกรณ์ตรวจวัดระยะทางด้วยระบบ Infrared โดยจะมีตัวรับการสะท้อนของตัวส่งที่กระทบกับวัตถุ โดยเอาต์พุตจากตัว Sensor จะกลับมาเป็นแรงดันที่มีค่าแตกต่างกันตามระยะทางที่ตรวจจับได้ โดยการเลือกใช้ Sensor ที่มีการตรวจวัดระยะทางที่ 0 – 80 เซนติเมตร ซึ่งจากการจำลองสภาพสถานการณ์ขึ้นโดยจะมีการจำลองขึ้นดังนี้



รูปที่ 3.34 ภาพการจำลองเหตุการณ์เกิดเหตุภัยพิบัติ

จากสภาพสถานที่ที่จำลองในภาพ ก) จะเห็นได้ว่ามีการจำลองกำแพงขึ้นด้วยไม้กระดานโดยทั้งสองข้างจะมีความกว้าง 1 เมตร และมีทางลาดชัน ที่ทำมุมประมาณ 10 องศากับแนวระดับเพื่อทดสอบทางเดินของหุ่นยนต์อัตโนมัติ

ดังนั้นอุปกรณ์ตรวจจับนั้นจะมีความสำคัญในการใช้ตรวจสอบว่าหุ่นยนต์เคลื่อนที่เข้าใกล้ สิ่งกีดขวางมากน้อยเพียงใด เช่น ถ้าหุ่นยนต์เคลื่อนที่เข้าใกล้กำแพง ตัวเซนเซอร์จะตรวจจับระยะห่างจากตัวหุ่นยนต์กับสิ่งกีดขวางเท่าใด โดยการติดตั้งเซนเซอร์จะติดที่รอบตัวหุ่นยนต์ทั้งสามด้าน คือ ด้านหน้า ด้านซ้ายและด้านขวา ดังรูปที่ 3.35



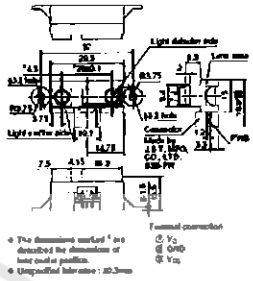
รูปที่ 3.35 การแสดงการตรวจจับสิ่งกีดขวาง และการติดตั้งอุปกรณ์ตรวจวัดระยะทาง

โมดูลตรวจวัดระยะทางด้วยแสงอินฟราเรด

คุณสมบัติ

- เป็นโมดูลตรวจจับระยะทางแบบอินฟราเรด ที่สามารถวัดระยะทาง ได้ถูกต้อง
- GP2Y0A21YK0F วัดระยะทางในช่วง 10 - 80 เซนติเมตร

(ก)



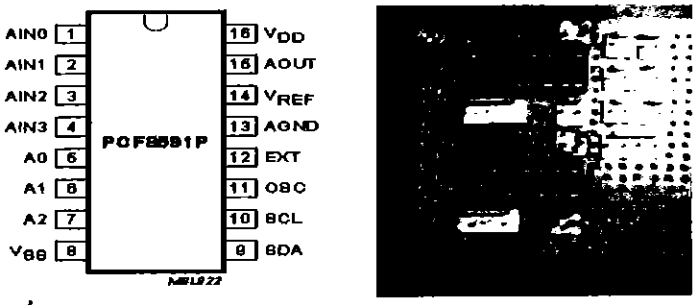
(ข)

รูปที่ 3.36 โมดูลตรวจวัดระยะทางด้วยแสงอินฟราเรด GP2Y0A21YK0F
(ก) ลักษณะภายนอกของโมดูลตรวจวัดระยะทาง
(ข) ขนาดของโมดูลตรวจวัดระยะทาง

โดยให้ผลการตรวจจับเป็นแรงดันไฟตรง ในย่าน 0.4 - 2.4V ประกอบด้วยตัวส่ง และตัวรับอินฟราเรดที่ติดตั้งภายใต้ตัวถังเดียวกันจะทำงานทันทีที่มีไฟเลี้ยง 5V จ่ายให้ โดยตัวส่งอินฟราเรด จะขับแสงอินฟราเรดจากตัวมันตลอดเวลา และเมื่อใดที่มีวัตถุมาขวางกั้น ทำให้เกิดการสะท้อนของแสงอินฟราเรดกลับไปยังตัวรับภายใน

โมดูล Analog to Digital เบอร์ PCF8591 8-bit A/D

โมดูลตรวจวัดระยะทางด้วยแสงอินฟราเรด GP2Y0A21YK0F ให้ผลการตรวจจับเป็นแรงดันไฟตรง ในย่าน 0.4 - 2.4V ดังนั้นจึงต้องมีการทำการแปลงแรงดันนี้จากอะนาล็อกเป็นดิจิตอล (Analog to Digital) โดยการแปลงจะมีอุปกรณ์ที่แปลงจากดิจิตอลเป็นอะนาล็อก(Digital to Analog) ซึ่งการแปลงนี้จะเป็นการแปลงแรงดันให้อยู่ในระดับ 0 และ 1 หรือ 0V และ 5V ดังนั้นการเลือกใช้อุปกรณ์ในการทำการ A/D นั้นจะเลือกใช้ เบอร์ PCF8591 8-bit A/D and D/A converter โดยเป็นการสื่อสารในรูปแบบ I^2C BUS การเชื่อมต่อระหว่างไมโครคอนโทรลเลอร์ ตามที่กล่าวมาข้างต้นจะใช้พอร์ตที่ 1 ในบิตที่ 1 และบิตที่ 2 ในการรับสัญญาณ SDA และ SCL ตามลำดับ และจากนี้จะเป็นการออกแบบวงจรในการสร้างโมดูล Analog to Digital เบอร์ PCF8591 8-bit A/D and D/A converter ดังแสดงในรูปที่ 3.37

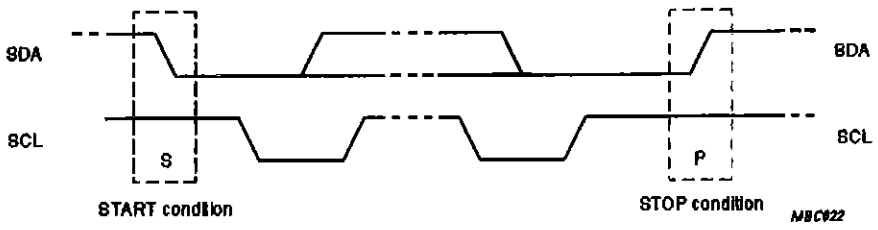


รูปที่ 3.37 โมดูล Analog to Digital เบอร์ PCF8591 8-bit A/D

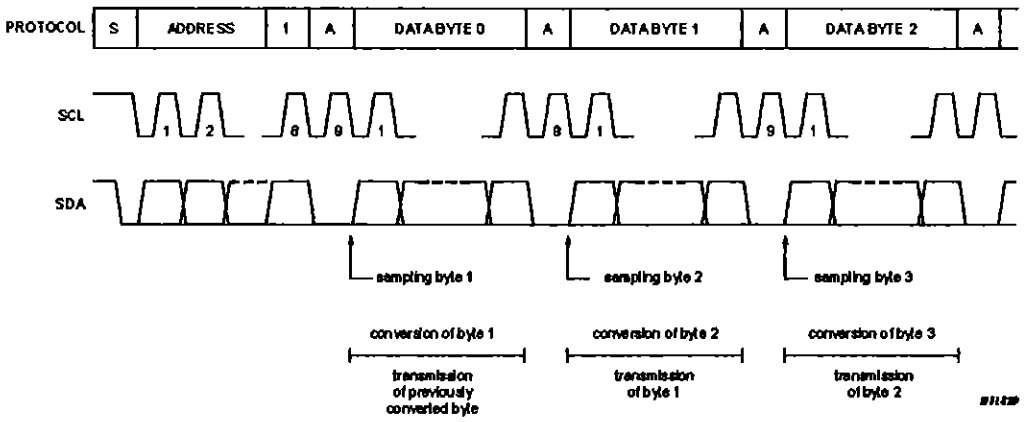
คุณสมบัติของ Analog to Digital เบอร์ PCF8591 8-bit A/D

- ใช้ไฟเลี้ยงไอซี 5V.
- สามารถทำงานได้ในระดับกระแสและแรงดันอินพุตต่ำ
- 4 อะนาล็อกอินพุต เพื่อใช้ในการเขียนโปรแกรมควบคุม แต่ละช่องสัญญาณอินพุต
- 8 bit A/D ที่ให้ความละเอียดของข้อมูล 0 -256 ค่า
- การสื่อสารเป็นแบบ I^2C BUS

การสื่อสารแบบ I^2C BUS ของโมดูล Analog to Digital เบอร์ PCF8591 8-bit A/D สามารถให้การทำงานได้โดย ใช้สายสัญญาณ 2 เส้น คือ SCL , SDA สำหรับติดกับอุปกรณ์แบบ 2 ทิศทาง โดยที่ขาสัญญาณทั้ง 2 จะต้องต่อกับตัวต้านทานแบบ pull up 2-10K เนื่องจากเอาต์พุตมีลักษณะเป็น แบบ Open Darin หรือเป็นแบบ Open Collector เพื่อให้เอาต์พุตเชื่อมต่อกันได้หลายตัว การรับและส่งข้อมูลด้วยการส่งสถานะเริ่มต้นเพื่อแสดงการร้องขอใช้บัส และตามด้วยรหัสควบคุมซึ่งประกอบด้วยรหัสของ Analog to Digital เบอร์ PCF8591 บอกตำแหน่งของอุปกรณ์และ โหมดในการอ่านหรือเขียนข้อมูล จากนั้นเมื่ออุปกรณ์ทราบว่า MCU ต้องการจะติดต่อด้วย ก็ต้องส่งสถานะการรับรู้ หรือแจ้งให้ MCU รับรู้ว่าข้อมูลที่ได้มามีความถูกต้อง และเมื่อสิ้นสุดการส่งข้อมูล MCU จะต้องส่ง สถานะสิ้นสุดเพื่อบอกอุปกรณ์ว่าสิ้นสุดการใช้บัส



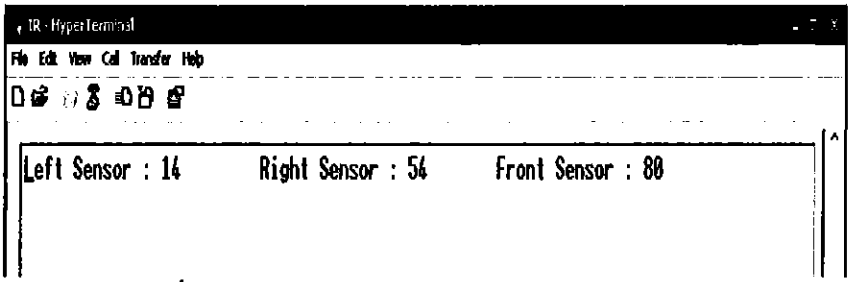
รูปที่ 3.38 การทำงานของ SDA และ SCL โมดูล Analog to Digital เบอร์ PCF8591 8-bit A/D



รูปที่ 3.39 ลำดับของข้อมูลของโมดูล Analog to Digital เบอร์ PCF8591 8-bit A/D

รูปที่ 3.40 แสดงการวัดสัญญาณการรับส่งข้อ SDA (เส้นบน) และ SCL (เส้นล่าง)

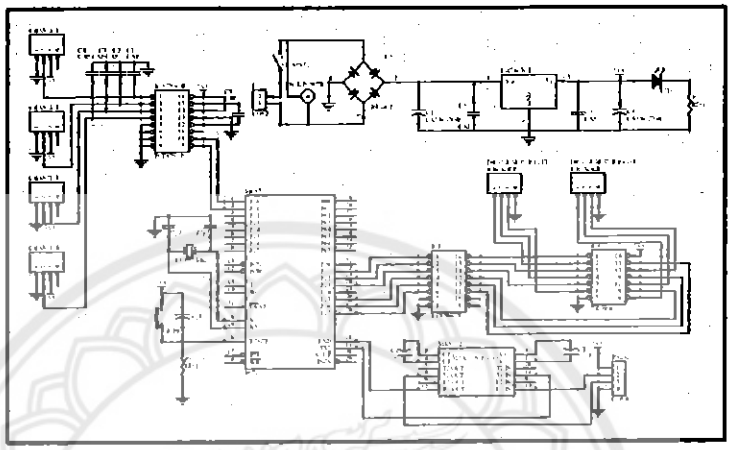
เมื่อทำการประกอบชุดอุปกรณ์จะเป็นการเขียนโปรแกรมภายใน MCU เพื่อติดต่อโมดูล A/D โดยภาพที่ 3.16 เป็นการแสดงการรับส่งข้อมูลระหว่างโมดูล A/D กับ MCU เพื่อรับระยะทางจากเซนเซอร์ตรวจจับระยะทางเพื่อทำการประมวลการเคลื่อนที่ โดยการทดสอบจะใช้ โปรแกรม HyperTerminal แสดงค่าการรับข้อมูลจากเซนเซอร์



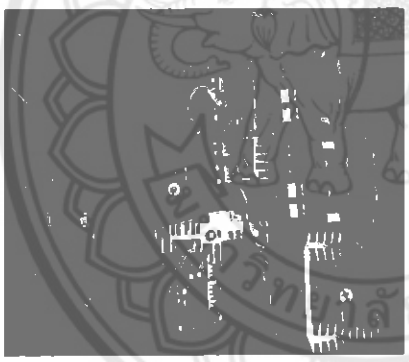
รูปที่ 3.41 แสดงการทดสอบการรับค่าจากเซนเซอร์

ชุดควบคุมหุ่นยนต์ระดับ Low Level

เมื่อสร้างวงจรและทดสอบเรียบร้อย ให้ทำการติดตั้งอุปกรณ์ต่างๆเข้าสู่กล่องอเนกประสงค์โดยเลือกขนาดที่พอเหมาะกับขนาดของอุปกรณ์ทั้งหมดและขนาดของหุ่นยนต์ โดยจะแสดงการประกอบชุดโมดูลต่างลงสู่กล่องอเนกประสงค์ โดยแสดงการเชื่อมต่อบัสต่าง ๆ ดังรูปที่ 3.42



(ก)



(ข)

(ค)

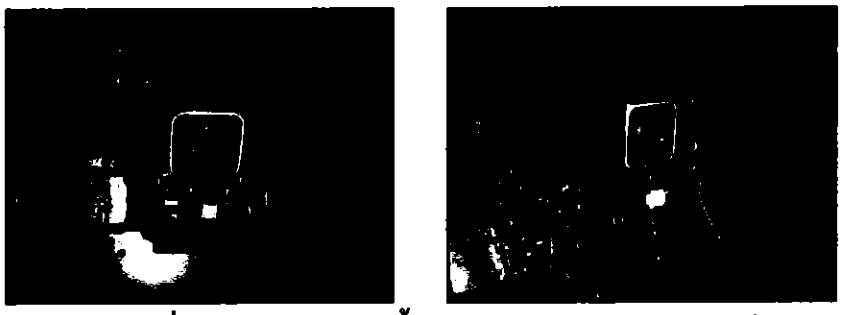
รูปที่ 3.42 แสดงการประกอบอุปกรณ์และเชื่อมต่อสายบัสต่าง

(ก) ภาพวงจรที่รวมวงจรทั้งหมดเข้าด้วยกัน

(ข) แสดงการเชื่อมต่อของ MCU และโมดูลต่าง ๆ จากมุมบน

(ค) แสดงเชื่อมต่อบัสต่างๆ เข้าสู่ตัวหุ่นยนต์

ประกอบและติดตั้งกล้อง Webcam เข้าสู่ตัวหุ่นยนต์ โดยกล้องจะติดตั้งอยู่ด้านหน้าของหุ่นยนต์เพื่อค้นหาผู้ประสบภัยหรือสิ่งที่สนใจดังรูปที่ 3.43



รูปที่ 3.43 แสดงการติดตั้งกล้อง Webcam กับตัวหุ่นยนต์

การติดตั้งหน่วยประมวลผลระดับบน (Note Book) เข้าสู่หุ่นยนต์ และหุ่นยนต์ประกอบเสร็จสมบูรณ์แสดงในรูปที่ 3.44

รูปที่ 3.44 แสดงหุ่นยนต์ประกอบเสร็จสมบรูณ์

3.3 การควบคุมระดับบน (High level control)

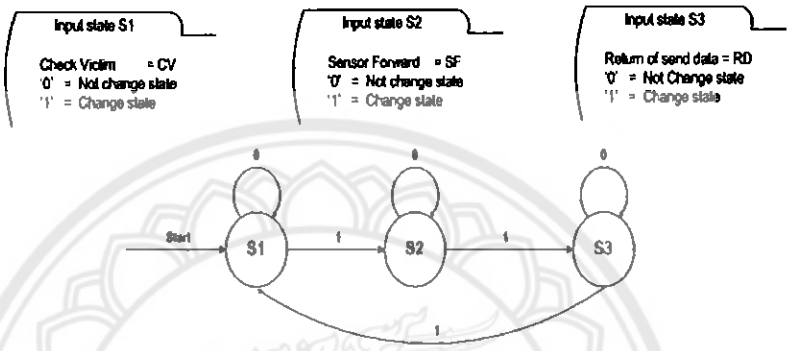
จุดมุ่งหมายหลักของการทำงานของโปรแกรมในส่วนของการควบคุมระดับบนนี้คือ การค้นหาผู้ประสบภัย นี่คือปัญหาที่ว่า How do I going? คือต้องไปหาผู้ประสบภัย ส่วนปัญหาต่อไปที่ต้องแก้ คือ How do I get there? จะไปที่นั่นอย่างไร ทั้งสองส่วนนี้เป็นปัญหาของการควบคุมระดับบนนี้ ซึ่งมีแนวคิดที่จะจัดการกับปัญหาดังนี้

3.3.1 แนวคิดการแก้ปัญหาจะต้องไปที่ไหน

จากเป้าหมายที่หุ่นยนต์กู้ภัยต้องการก็คือการค้นหาผู้ประสบภัยซึ่งอาจจะติดอยู่ในสถานที่ หรือสภาพแวดล้อมที่ไม่สามารถมองเห็นได้โดยง่าย ภาพที่หุ่นยนต์มองเห็นอาจจะไม่ชัดเจนมากนักหรือมีเพียงจุดสังเกตเพียงเล็กน้อยที่จะสื่อความหมายว่าเป็นผู้ประสบภัย เช่น สีของเครื่องแต่งกายของผู้ประสบภัย ความร้อนจากร่างกาย ก๊าซคาร์บอนไดออกไซด์จากการหายใจของมนุษย์ และเสียงจากการขอความช่วยเหลือ ซึ่งเหล่านี้จะเป็นสิ่งที่หุ่นยนต์สามารถใช้เป็นปัจจัยในการเข้าค้นหาเป้าหมายได้ ในโครงงานนี้จะจำลองสถานการณ์ที่ผู้ประสบภัยติดอยู่ในสถานที่เกิดเหตุดังรูปที่ 3.45 จากแบบจำลองสถานการณ์อ้างอิงจากการแข่งขันหุ่นยนต์กู้ภัยชิงแชมป์ประเทศไทยปี 2552 จึงได้ออกแบบสถานที่จำลองให้ใกล้เคียงกันดังนั้นสำหรับหุ่นยนต์กู้ภัยอัตโนมัติจะมีผนังและจุดจำลองตำแหน่งที่ผู้ประสบภัยติดอยู่ด้วย ตู้กดา, เครื่องให้ความร้อน, ลำโพง, น้ำแข็งแห้ง (dry ice) เป็นคาร์บอนไดออกไซด์ในสถานะของแข็ง

รูปที่ 3.45 ภาพการจำลองผู้ประสบภัยของการแข่งขันหุ่นยนต์กู้ภัยชิงแชมป์ประเทศไทย ประจำปี 2552

เมื่อสร้างสถานการณ์จำลองเพื่อความใกล้เคียงกับสถานการณ์จริงดังรูปที่ 3.45 โดยสิ่งสนใจเกี่ยวกับเป้าหมายคือ สีเครื่องแต่งกายของผู้ประสบภัย โดยจำลองสีขึ้นสามสีเพื่อการทดลองในขั้นตอนทดสอบ คือ สีแดง สีเขียว สีน้ำเงินหรือฟ้า แต่การเคลื่อนที่ของหุ่นยนต์ก่อนที่จะค้นหาเป้าหมายจะออกแบบให้หุ่นยนต์เคลื่อนที่โดยการจับสิ่งกีดขวางหรือกำแพงตามสถานที่จำลองในรูปที่ 3.34 เพื่อให้กล้องทำการตรวจสอบหาสีทดสอบที่ต้องการค้นหา โดยออกแบบการเคลื่อนที่ของหุ่นยนต์ตามแผนผังรูปที่ 3.46



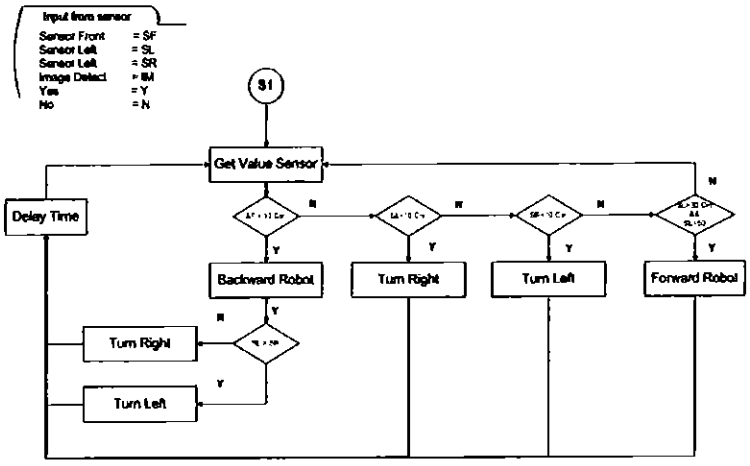
รูปที่ 3.46 แผนผังการเคลื่อนที่ของหุ่นยนต์ของแต่ละสถานะ

สำหรับวงกลมแต่ละวงกลมจะแสดงสถานะของหุ่นยนต์ที่กำลังทำงานอยู่ในสถานะต่าง ๆ โดยจะมีสัมพันธ์ระหว่างระบบ High Level และ Low Level ซึ่งจะอธิบายตามแผนผังดังนี้

1. ให้วงกลมแต่ละรูปแสดงสถานะ (State) ของหุ่นยนต์และมีชื่อตามรูปที่ 3.46
2. เส้นเชื่อมและหัวลูกศร จะแสดงทิศทางการไหลของข้อมูลจากอินพุตที่แต่ละสถานะต้องการเมื่อข้อมูลอินพุตมีการเปลี่ยนแปลงโดยให้ข้อมูลอินพุตเป็นแบบ True หรือ False ในที่นี้ True คือ logic 1 และ False คือ logic 0

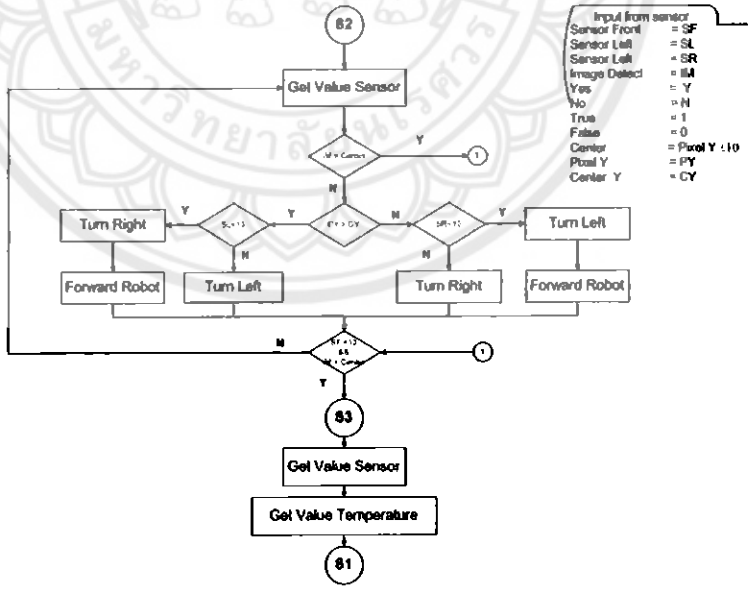
จากนี้จะเป็นการอธิบายการทำงานของสถานะแต่ละสถานะของแผนผังรูปที่ 3.46 โดยเริ่มต้นที่ State (S1) ดังนี้

เริ่มต้น เมื่อเปิดสวิตช์ควบคุมหุ่นยนต์ อินพุตที่ได้รับคือ start จะทำให้หุ่นยนต์เปลี่ยนไปอยู่ในสถานะ S1 คือสถานะที่ระบบควบคุมระดับ High Level จะใช้กล้องในการตรวจสอบภาพที่กล้องได้รับภาพเข้ามาว่ามีสีที่หุ่นยนต์ต้องการหรือไม่ ถ้าในเฟรมภาพในขณะนั้นยังไม่มีสีที่สนใจเกิดขึ้น อินพุตจากระดับ High Level จะส่งข้อมูล logic 0 ทำให้ไม่เกิดการเปลี่ยนสถานะไปที่สถานะ S2 โดยหน้าที่ของสถานะ S1 คือ การเคลื่อนที่ตามผนังข้างซ้ายของสถานการณ์จำลอง ซึ่งจะแสดงในรูปแบบของ Flow chart เพื่อแสดงหลักการเคลื่อนที่ของหุ่นยนต์ในรูปที่ 3.47



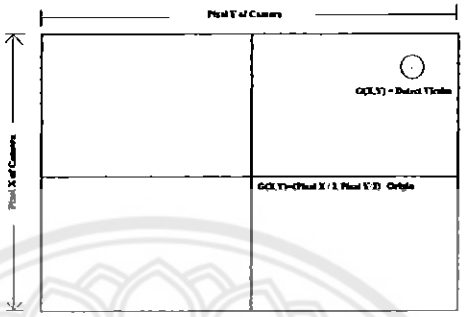
รูปที่ 3.47 แสดง Flow chart การเคลื่อนที่ของหุ่นยนต์ในสถานะ S1

จาก Flow chart แสดงให้เห็นว่าเมื่อหุ่นยนต์เคลื่อนที่เข้าใกล้กำแพงด้านใดด้านหนึ่งก็จะทำการเลี้ยวออกจากผนังด้านนั้น ถ้าหุ่นยนต์อยู่ในระยะที่เหมาะสมกับการเดินไปข้างหน้าหุ่นยนต์ก็จะเดินหน้าในขณะเดียวกันเมื่อกระทำกระบวนการทั้งหมดในสถานะ S1 เรียบร้อยก็กลับไปตรวจสอบว่าขณะนี้หุ่นยนต์พบเป้าหมายหรือไม่ ถ้าไม่ก็ยังคงอยู่สถานะ S1 ต่อไป แต่เมื่อไหร่ที่พบเป้าหมายหุ่นยนต์จะเปลี่ยนสถานะไปอยู่ที่ S2 โดยจะแสดงตาม Flow chart รูปที่ 3.48



รูปที่ 3.48 Flow chart การเคลื่อนที่ของหุ่นยนต์ S2 ไปยัง S3 ไปยัง S1

จาก Flow chart รูปที่ 3.23 จะทำให้หุ่นยนต์ในสถานะ S2 นี้เคลื่อนที่เข้าหาเป้าหมายได้ โดยเริ่มต้นจากการรับค่าจากเซนเซอร์ทั้งหมดเข้ามา และตรวจสอบว่าขณะนี้หุ่นยนต์เห็นเป้าหมายอยู่ที่ตำแหน่งได้ในจอภาพโดยลักษณะการตรวจสอบว่าเป้าหมายอยู่ระยะประมาณกลางจอหรือไม่จะมีหลักการทางภาพโดยจะอธิบายด้วยภาพจำลองตามรูปที่ 3.49



รูปที่ 3.49 ภาพจำลองการมองเห็นของหุ่นยนต์เมื่อพบผู้ประสบภัย

จากรูปที่ 3.49 สามารถอธิบายได้ว่าเมื่อหุ่นยนต์เคลื่อนที่อยู่ในสถานะ S2 แต่เมื่อกล้องได้ทำการตรวจพบผู้ประสบภัย ทางส่วน High Level จะทำการประมวลผลข้อมูลเพื่อให้ได้ตำแหน่งบนภาพว่าขณะนี้ผู้ประสบภัยอยู่ที่ตำแหน่งได้บนภาพ จากแบบจำลองจะแสดงให้เห็น Pixel X และ Pixel Y ที่พิกัด X, Y ถูกหาร 2 จะได้ จุดตรงกลางภาพ เมื่อ การประมวลผลข้อมูลได้ พิกัดที่ผู้ประสบภัยอยู่ก็เช่นเดียวกันจะทำให้ได้พิกัด X, Y ออกมา เราจึงนำตัวเลขของ Y ไปเปรียบเทียบกับ Y ที่กลางภาพจากภาพจำลองแสดงให้เห็นว่าขณะนี้ ผู้ประสบภัยอยู่ตำแหน่งขวามือของหุ่นยนต์ ดังนั้นหุ่นยนต์จะต้องเลี้ยวขวาเพื่อทำให้ตำแหน่งของผู้ประสบภัยย้ายมาอยู่ที่ตรงกลางจอภาพ จากนั้นเมื่อภาพอยู่ในระยะประมาณกึ่งกลางภาพก็ให้หุ่นยนต์เคลื่อนที่ เข้าหาเป้าหมาย แต่ในระยะการเลี้ยวซ้ายหรือขวาจะมีระบบเซนเซอร์ตรวจสอบว่าหุ่นยนต์อยู่ในระยะที่สามารถเคลื่อนที่ไปได้หรือไม่ เมื่อระยะห่างระหว่างหุ่นยนต์กับผู้ประสบภัยอยู่ใกล้เพียงพอที่หุ่นยนต์จะเก็บค่าสภาพแวดล้อมต่างๆ หุ่นยนต์ก็จะเปลี่ยนสถานะไปสู่สถานะ S3 เพื่อเก็บค่าต่างไปประมวลผลเมื่อเก็บค่าต่างเรียบร้อยหุ่นยนต์ก็กลับไปสู่สถานะ S1 เพื่อเคลื่อนที่หาเป้าหมายใหม่ต่อไป

3.3.2 การประมวลผลภาพค้นหาเป้าหมาย

จากขั้นตอนที่กล่าวมาข้างต้นเพื่อให้หุ่นยนต์เข้าหาเป้าหมายได้นั้น หุ่นยนต์ที่สามารถรู้จักหรือเข้าใจความหมายของภาพนั้นจำเป็นที่ต้องใช้หลักการประมวลผลภาพตามที่ได้กล่าวในบทที่ 2 เรื่อง หลักการประมวลผลภาพ โดยจำเป็นที่ต้องใช้หลักการต่าง ๆ และทฤษฎีต่างๆเพื่อทำให้หุ่นยนต์นำภาพที่ได้มาประมวลผลและตีความหมาย ดั้งนั้นจุดประสงค์หลักของการค้นหาเป้าหมายโดยใช้หลักการของสีเข้ามาช่วย คือ ค้นหาสีที่สนใจ และสีที่เห็นนี้เป็นสมมุติฐานสำคัญว่าเป็นสีของเครื่องแต่งกายของผู้ประสบภัย เมื่อหุ่นยนต์พบผู้ประสบภัยแล้วก็ต้องตีความหมายว่าขณะนี้อยู่ที่ตำแหน่งใด โดยจะต้องบอกว่าอยู่ที่ พิกัดใดในภาพ ซึ่งการประมวลผลภาพด้วยคอมพิวเตอร์มีหลายโปรแกรม ยกตัวอย่าง เช่น โปแกรม MATLAB, โปรแกรม Microsoft Visual Studio 2008, โปรแกรม Adobe Media Flash MX และอื่น ๆ แต่ละโปรแกรมจะมีลักษณะเด่นแต่ละด้าน จึงขอยกตัวอย่างโปรแกรมที่กล่าวมาข้างต้นตามลำดับดังนี้

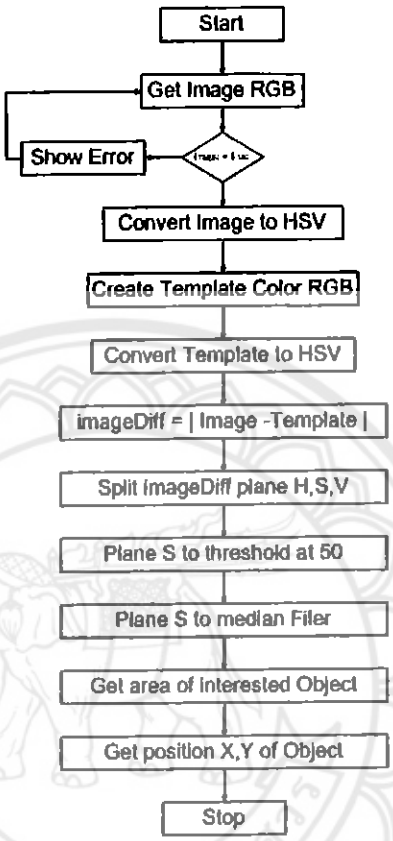
MATLAB เป็นโปรแกรมที่มี Function ต่างๆ ที่สะดวกให้ใช้งาน เป็นการทำงานในรูปแบบ Matrix หรือ Vector ทำให้สามารถเข้าถึงระดับ pixel ของภาพ สามารถใช้การ plot กราฟดูค่าข้อมูลต่างๆได้ง่าย สามารถเขียน เป็น Graphic User Interface ได้ ทำงานบนระบบปฏิบัติการ Windows

Microsoft Visual Studio 2008 C++ เป็นโปรแกรมที่มีการเขียนโปรแกรมเชิงวัตถุ OOP ซึ่งจะมีคลาสและไลบารี่ต่าง ๆ มากมาย และใช้ Microsoft Foundation Class Library (MFC) ในการเขียนโปรแกรม Graphic User Interface และมีไลบารี่ที่สำคัญคือ OpenCV ซึ่งเป็น open source ที่เปิดให้มีการพัฒนาต่อ การพัฒนาความสามารถของ OpenCV จะมีหลาย Version ตามการพัฒนา โดยลักษณะเด่นคือสามารถเชื่อมต่อกล้องแบบ Real Time ได้ 2 ตัว การเข้าถึงลักษณะ Array List ทำให้มีความรวดเร็วในการเข้าถึงเฟรมภาพแต่ละเฟรมจากการรับภาพจากกล้อง มี Class การทำงานต่างๆเกี่ยวกับการประมวลผลภาพ ยกตัวอย่างเช่น การทำ Threshold, การแปลงภาพ RGB เป็น HSV และอีกมากมาย รวมทั้งยังสามารถจัดทำระบบ AI กับข้อมูลภาพได้อีกด้วย

ดังนั้นในโครงงานนี้จึงเลือกใช้ Microsoft Visual Studio 2008 C++ ด้วย OpenCV เพื่อทำการประมวลผลภาพค้นหาเป้าหมายโดยจะมีขั้นตอนในการประมวลผลภาพดังต่อไปนี้

1. ติดตั้ง Microsoft Visual Studio 2008 C++ บน ระบบปฏิบัติการ Windows XP
2. ติดตั้ง Library OpenCV Version 1.0
3. ติดตั้งกล้อง webcam OKER OE192
4. ออกแบบขั้นตอนการประมวลผลภาพ

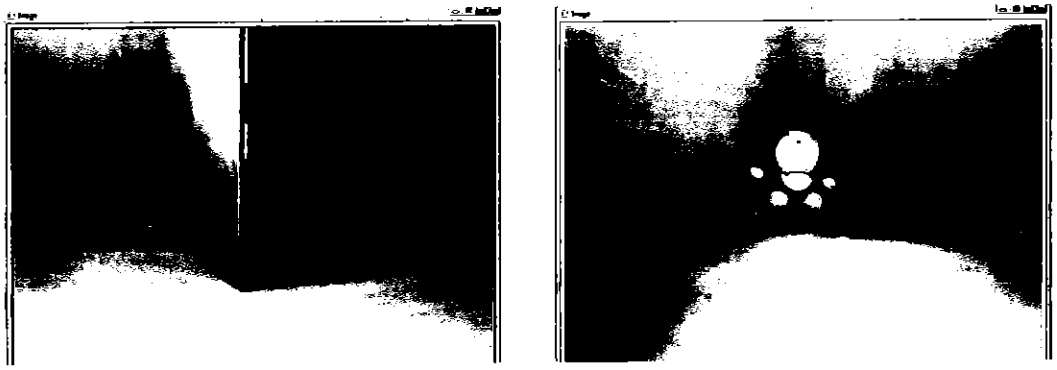
เมื่อทำการติดตั้งองค์ประกอบต่าง ๆ สำหรับการเขียนโปรแกรมเรียบร้อย ก็จะออกแบบการประมวลผลภาพดังรูปที่ 3.50



รูปที่ 3.50 แสดง Flow chart การค้นหาเป้าหมายโดยการประมวลผลภาพ

จากรูปที่ 3.25 แสดง Flow chart การประมวลผลภาพเพื่อค้นหาเป้าหมายหรือผู้ประสบภัยโดยใช้สีของเครื่องแต่งกายของผู้ประสบภัยเป็นจุดสนใจในการค้นหา ซึ่งจะอธิบายแต่ละขั้นตอนของการทำงานตั้งแต่เริ่มต้นโดยจำลองการทำงานด้วยการใช้ภาพนิ่งในการทดสอบก่อนใช้ภาพจากกล้องวิดีโอ

3.3.2.1 Get Image RGB ขั้นตอนนี้เป็นการเปิดภาพจากแหล่งเก็บข้อมูล (Hard disk) เพื่อนำไปประมวลผลภาพ และทดสอบโดยแสดงออกที่หน้าต่าง ดังรูปที่ 3.51



(ก) (ข)

รูปที่ 3.51 การนำภาพจากแหล่งเก็บข้อมูลเข้าประมวลผลภาพ

(ก) ภาพสถานการณ์จำลองแบบไม่มีผู้ประสบภัย

(ข) ภาพสถานการณ์จำลองแบบมีผู้ประสบภัย

3.3.2.2 **Convert Image to HSV model** คือการแปลงภาพจาก RGB Model เป็น HSV Model ก่อนทำการประมวลผล (Pre – processing) เนื่องจาก RGB Model จะมีปัญหาเกี่ยวกับเรื่องของแสงสว่างที่แตกต่างกันในช่วงเวลาต่างกันจึงทำให้แสงสว่างมีผลต่อ RGB Model สำหรับ HSV Model จะแทบไม่มีผลต่อแสงสว่างที่ต่างกัน ดังนั้นจึงต้องทำการแปลง Model Color ก่อนทำการประมวลผล และจะแสดงตัวอย่างการแปลงโดยแสดง HSV Model ใน RGB Model มีความแตกต่างกันดังรูปที่ 3.52

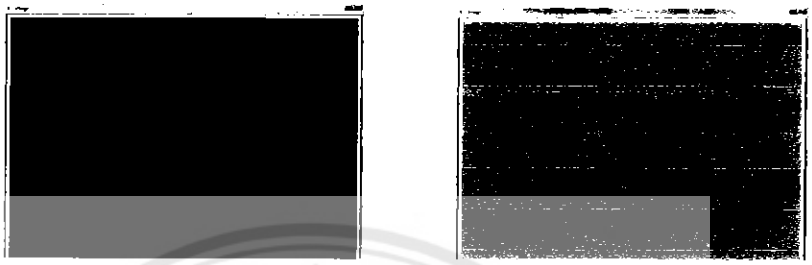
(ก) (ข)

รูปที่ 3.52 การแปลงภาพ HSV Model ใน RGB Model

(ก) ภาพการแปลงสถานการณ์จำลองแบบไม่มีผู้ประสบภัย

(ข) ภาพการแปลงสถานการณ์จำลองแบบมีผู้ประสบภัย

3.3.2.3 Create template color คือการสร้างภาพที่เป็นสีที่สนใจหรือสีเครื่องแต่งกายนั่นเอง และในแบบจำลองนี้จะใช้สีฟ้าเป็นสีในการทดลอง จึงต้องสร้างภาพสีฟ้าที่มีขนาด Pixel เท่ากับขนาดภาพจริง เพื่อในขั้นตอนต่อไปจะไม่เกิดปัญหาในการประมวลผลเนื่องด้วยขนาดภาพที่ไม่เท่ากัน ทำการแปลงภาพจาก RGB Model เป็น HSV Model ดังแสดงในรูปที่ 3.53



(ก) (ข)

รูปที่ 3.53 การแปลงภาพ HSV Model จาก RGB Model
(ก) ภาพแสดงสีจำลองเครื่องแต่งกายผู้ประสบภัย
(ข) ภาพการแปลงเป็น HSV Model แสดงใน RGB Model

3.3.2.4 Image Subtraction คือการลบภาพต้นแบบด้วยภาพที่เป็นสีจำลองของเครื่องแต่งกายผู้ประสบภัยโดยเมื่อลบกันแบบไม่คิดเครื่องหมาย (Absolute) ค่าใน Pixel นั้นๆของแต่ละภาพที่เท่ากันลบกันจะมีค่าใกล้เคียง 0 ภาพจากผลลัพธ์นี้คือจุดที่จะนำไปวิเคราะห์ว่าเป็นจุดสนใจหรือไม่ต่อไป

(ก) (ข) (ค)

รูปที่ 3.54 การแยกภาพจากการลบกันของภาพจริงกับภาพจำลองสีเครื่องแต่งกายผู้ประสบภัย
(ก) แสดงภาพ H Plane ของ HSV Model
(ข) แสดงภาพ S Plane ของ HSV Model
(ค) แสดงภาพ V Plane ของ HSV Model

3.3.2.5 Threshold คือการกำหนดค่าที่ต่ำกว่าค่าที่กำหนดให้เป็น 0 หรือ 1 โดยแบบ Binary inverse คือทำการกลับบิตนั้นเอง จึงทำให้ได้ภาพที่ได้หลังจากการลบกันดังแสดงในรูปที่ 3.55

(ก) (ข) (ค)

รูปที่ 3.55 การทำ Threshold ที่ค่าลบกันต่ำกว่า 50 ให้เป็น 0 แล้วกลับค่าด้วย Binary Inverse

(ก) แสดงการทำ Threshold ภาพ H Plane ของ HSV Model

(ข) แสดงการทำ Threshold ภาพ S Plane ของ HSV Model

(ค) แสดงการทำ Threshold ภาพ V Plane ของ HSV Model

จากรูปการทำ Threshold ด้วยค่าประมาณ 50 นี้จะให้ภาพที่แน่นอนสำหรับตัวอย่างภาพนี้และเป็นที่น่าสังเกตว่า ทั้งภาพทั้งสามมีภาพที่สามารถสื่อความหมายได้ดีที่สุดคือ S Plane สำหรับการทดลองจากภาพตัวอย่างนี้

3.3.2.6 Median Filter ภาพตัวอย่าง S Plane ในรูปที่ 3.55 ข) จากกระบวนการทำ Threshold ที่ระดับ 50 จะยังคงมี Noise ที่เป็น Salt noise เหลืออยู่จึงต้องทำการกรองส่วนที่ไม่ใช่บริเวณที่สนใจออก ก่อนที่จะทำการนับจำนวนพิกเซลต่อไป โดยผลการทำ Filter นี้จะแสดงผลในรูปที่ 3.56

(ก) (ข) (ค)

รูปที่ 3.56 การทำกรอง Noises ด้วย Median filter ด้วย Mask 3 x 3

(ก) แสดงการทำ Median filter ภาพ H Plane ของ HSV Model

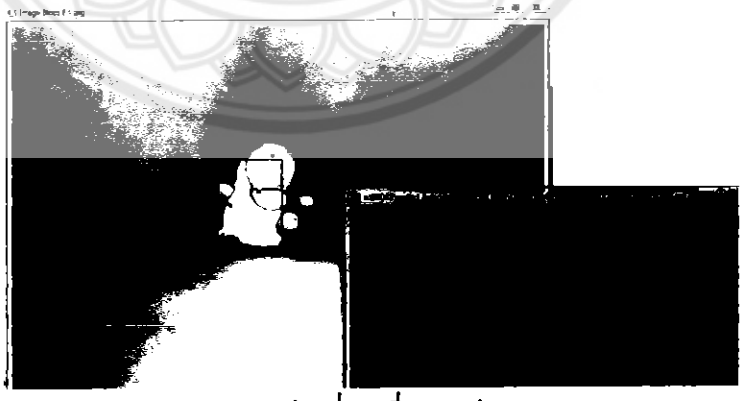
(ข) แสดงการทำ Median filter ภาพ S Plane ของ HSV Model

(ค) แสดงการทำ Median filter ภาพ V Plane ของ HSV Model

3.3.2.7 Area of interested คือการนำภาพ S Plane ที่ได้จากขั้นตอนที่ผ่านมาข้างต้นมาทำการนับจำนวนพิกเซลที่ต่อกันว่ามีจำนวนมากพอกับค่าที่ตั้งไว้หรือไม่ มากเพียงพอสำหรับการจะตัดสินใจได้ว่าสิ่งที่นับเป็นผู้ประสบภัย โดยจะสามารถตั้งเกณฑ์ในการตัดสินเบื้องต้นนี้ว่าถ้ามีมากกว่า 1000 พิกเซลที่ติดกันจะแสดงว่า ภาพสีขาวที่ได้นี้มีความน่าจะเป็นผู้ประสบภัยแต่เนื่องจากผู้ประสบภัยอาจจะมีขนาดที่แตกต่างกันดังนั้น ค่าเกณฑ์จึงควรมีการทดสอบในหลาย ๆ กรณีจากการทดลอง โดยต่อไปจะแสดงการนับจำนวนผู้ประสบภัยและแสดงตำแหน่งที่มีผู้ประสบภัยติดอยู่ด้วยวงกลม ดังรูปที่ 3.57

รูปที่ 3.57 การทำกำหนดขอบเขตของผู้ประสบภัยค้นหา

3.3.2.8 Get spatially (X, Y) คือ การกำหนดตำแหน่งที่ผู้ประสบภัยอยู่ ณ ตำแหน่งใดบนภาพที่หุ่นยนต์กำลังมองเห็นอยู่โดยการคำนวณจะได้จุดพิกัดมาจำนวนสองคู่ คือ X min, Y min และ X max, Y Max โดยจะแสดให้เห็นในรูปที่ 3.58



รูปที่ 3.58 แสดงการหาจุดพิกัดรอบกรอบสีเหลี่ยม ที่แสดงค่า X Min, Y Min และ X Max, Y Max

จากรูปที่ 3.58 จะแสดงค่าใน Console Application โดยค่าจะมีดังต่อไปนี้ Pixel X = 636, Pixel Y = 867 ขนาดภาพ 636 x 867 Pixel และพิกัด X Min = 336 , Y Min = 217 และ X max = 439, Y Max = 357 ซึ่งค่าที่ได้ทั้งหมดนี้สามารถนำไปคำนวณหาค่าจุดกึ่งกลางของรูปสี่เหลี่ยมได้ เพื่อนำค่าที่ได้จากการคำนวณนี้ไปเปรียบเทียบเพื่อหาระยะเป้าหมายต่อไป

3.3.2 ส่วนควบคุมติดต่อกับผู้ใช้ (Graphic User Interface, GUI)

การควบคุมหุ่นยนต์ก่อนที่จะให้หุ่นยนต์เคลื่อนที่เข้าทำภารกิจจะต้องทำการกำหนดค่าต่าง ๆ ให้กับหุ่นยนต์ เช่น กำหนดค่าการเชื่อมต่อระหว่างส่วนควบคุมระดับบนกับส่วนควบคุมระดับล่าง เมื่อส่วนควบคุม GUI ให้คำสั่งให้ส่วนควบคุมระดับบนทำงาน โดยลักษณะของการออกแบบจะเป็นไปในลักษณะดังรูปที่ 3.59

รูปที่ 3.59 ภาพแสดงส่วนของ Graphic User Interface

จากขั้นตอนที่ผ่านมาทั้งหมดในบทที่ 3 นี้เมื่อนำทุกส่วนประกอบเข้าด้วยกันเป็นตัวหุ่นยนต์เพื่อพร้อมที่จะทำการทดสอบหุ่นยนต์ในสถานการณ์ต่างๆ จากการจำลองสถานการณ์ซึ่งการทดสอบจะมีหลายกรณี เช่น ค้นหาในพื้นที่ที่มีทางแยก ค้นหาในพื้นที่ต่างระดับ ค้นหาในพื้นที่ที่ไม่มีผู้ประสบภัย ค้นหาในพื้นที่ที่มีผู้ประสบภัยอยู่หลายคน ซึ่งจากการทดสอบนี้จะแสดงผลการทดสอบในบทที่ 4 ซึ่งจะเป็นการแสดงผลการทดลองทั้งหมดต่อไป

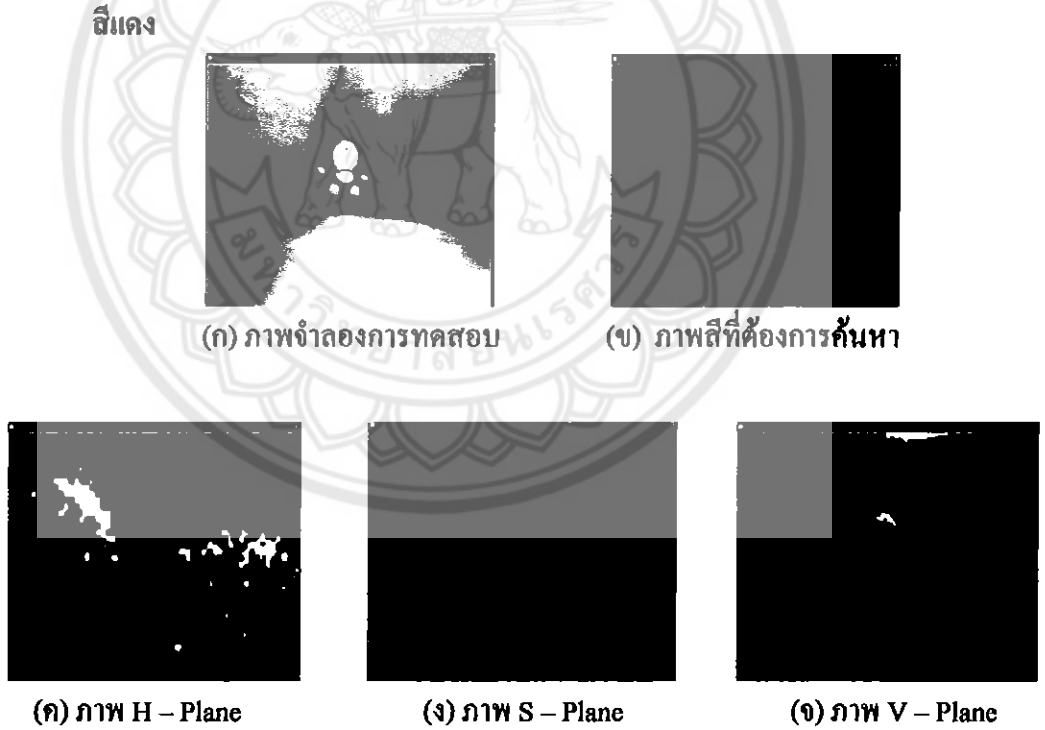
บทที่ 4

ผลการทดลอง

การออกแบบและการสร้างหุ่นยนต์กู้ภัยหลบหลีกสิ่งกีดขวางและค้นหาเป้าหมายอัตโนมัติ จากทฤษฎีและหลักการทำงานจากบทที่ผ่านมาสามารถสร้างหุ่นยนต์ที่สามารถเคลื่อนที่และค้นหาเป้าหมายหรือผู้ประสบภัยได้ และในบทนี้จะเป็นการนำหุ่นยนต์กู้ภัยเข้าสถานการณ์จำลองเพื่อทดสอบการทำงานของหุ่นยนต์โดยจะแบ่งการทดสอบออกเป็น 2 ส่วน คือ การค้นหาเป้าหมายอัตโนมัติและการเคลื่อนที่หลบหลีกสิ่งกีดขวางอัตโนมัติ ซึ่งสามารถอธิบายได้ดังต่อไปนี้

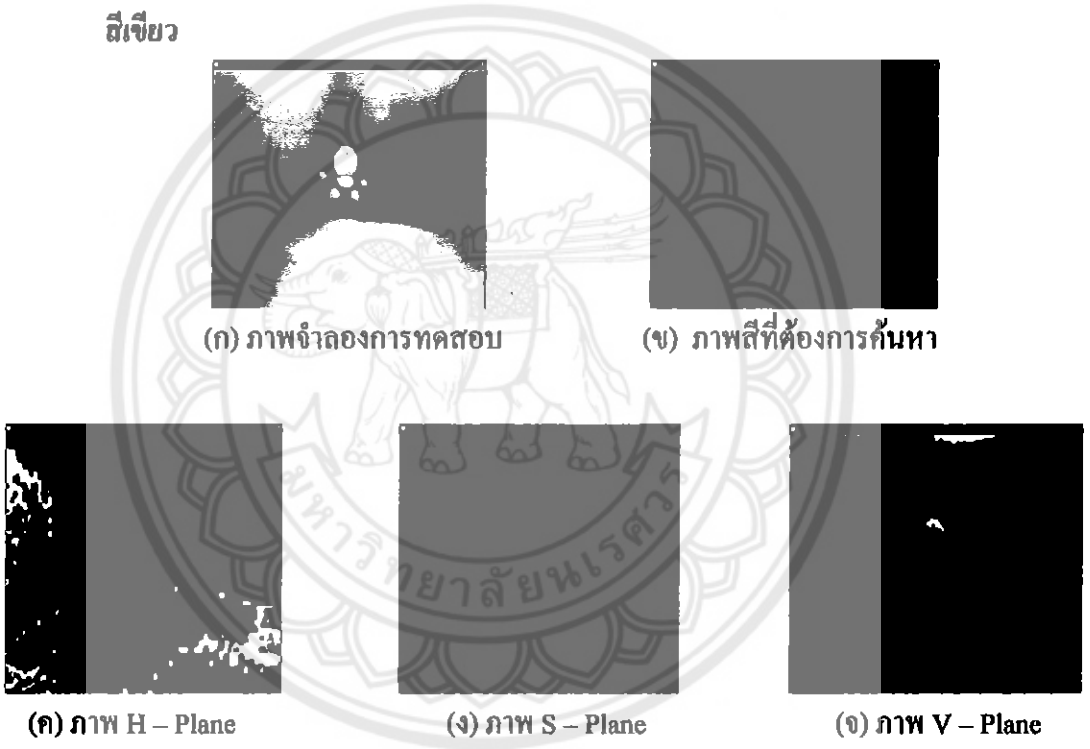
4.1 การค้นหาเป้าหมายอัตโนมัติ

4.1.1 การทดสอบค้นหาตำแหน่งของสีที่สนใจ โดยจำลอง 3 สี คือ สีแดง, สีเขียว, สีน้ำเงิน



รูปที่ 4.1 แสดงการหาตำแหน่งของสีที่สนใจด้วยการลบกันระหว่าง ภาพจำลอง กับสีแดงที่สนใจ

จากการทดลองการค้นหาสีแดงจากภาพสถานการณ์จริงโดยสีแดงคือสีโมเดล RGB (255, 0, 0) นั้นหมายถึงสีแดงเข้มที่สุดโดยไม่มีสีอื่นเจือปน จากภาพ ก) และ ภาพ ข) เมื่อนำมาลบกันแบบไม่คิดเครื่องหมายหรือการลบกันแล้วผลลัพธ์จะเป็นบวกเสมอ การทำการลบจะใช้คำสั่งใน OpenCV คือ คำสั่ง cvAbsDiff (img, find, dst) img เป็นตัวแปรของภาพสถานการณ์จริง find คือตัวแปรของสีที่ต้องการหา dst คือภาพผลลัพธ์ที่ได้จากการลบ ภาพ ค) ง) จ) คือภาพที่แสดงผลการลบของภาพโมเดล HSV ซึ่งทำการแยกแผ่นโดยใช้คำสั่ง cvCvtPixToPlane(dst, H, S, V, 0) โดยที่ H ,S,V คือภาพที่แยกแผ่นเรียบร้อยซึ่งเรียงตามลำดับในรูปที่ 4.1 ผลที่ได้คือ สีแดงจะค้นหาไม่พบในภาพจริงดังแสดงในภาพ S-Plane จะไม่มีสีขาวปรากฏแสดงว่าไม่มีเกิดการลบกันแล้วใกล้เคียง 0 เลย



รูปที่ 4.2 แสดงการหาตำแหน่งของสีที่สนใจด้วยการลบกันระหว่าง ภาพจำลอง กับสีเขียวที่สนใจ

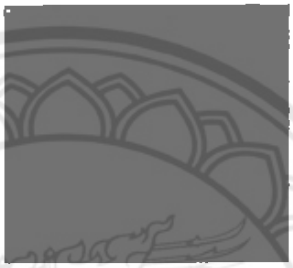
มาถึงการทดลองการค้นหาสีเขียวจากภาพสถานการณ์จริง คือ สีโมเดล RGB (0, 255, 0) นั้นหมายถึง สีเขียวที่เข้มสุดโดยไม่มีสีอื่นเจือปน จากภาพ ก) และ ภาพ ข) ส่วนภาพ ค) ง) จ) คือภาพที่แสดงผลการลบของภาพโมเดล HSV ซึ่งทำการแยกแผ่นที่ได้จากการลบเรียบร้อย ผลที่ได้คือ สีเขียวจะค้นหาไม่พบในภาพจริงดังแสดงในภาพ S-Plane จะไม่มีสีขาวปรากฏแสดงว่าไม่มีเกิดการลบกันแล้วเป็น 0 เลย

สีน้ำเงิน

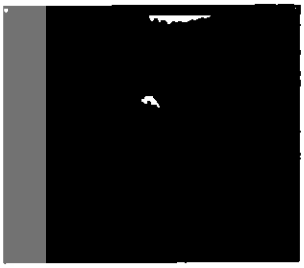
(ก) ภาพจำลองการทดสอบ

(ข) ภาพสีที่ต้องการค้นหา

(ค) ภาพ H – Plane



(ง) ภาพ S – Plane



(จ) ภาพ V – Plane

รูปที่ 4.3 แสดงการหาตำแหน่งของสีที่สนใจด้วยการลบกันระหว่าง ภาพจำลอง กับสีน้ำเงินที่สนใจ

การทดลองการค้นหาสีเขียวจากภาพสถานการณ์จริง คือ สีโมเดล RGB (0, 0, 255) นั้นหมายถึงสีเขียวที่เข้มสุดโดยไม่มีสีอื่นเจือปน จากภาพ ก) และ ภาพ ข) ภาพ ค) ง) จ) คือภาพที่แสดงผลการลบของภาพโมเดล HSV ซึ่งทำการแยกแผ่นที่ได้จากการลบเรียบร้อยแล้วผลที่ได้ คือ สีเขียวจะค้นหาไม่พบในภาพจริงดังแสดงในภาพ S-Plane จะไม่มีสีขาวปรากฏแสดงว่าไม่มีเกิดการลบกันแล้วเป็น 0 เลย

สีฟ้า (สีที่ใกล้เคียงกับผู้ประสบภัย)

(ก) ภาพจำลองการทดสอบ

(ข) ภาพสีที่ต้องการค้นหา

(ค) ภาพ H – Plane

(ง) ภาพ S – Plane

(จ) ภาพ V – Plane

รูปที่ 4.4 แสดงการหาตำแหน่งของสีที่สนใจด้วยการลบกันระหว่าง ภาพจำลอง กับสีฟ้าที่สนใจ

จากการทดลองค้นหาตำแหน่งด้วยสีที่สนใจทั้ง 3 สีนั้นสรุปได้ว่าภาพ S – Plane สามารถแสดงให้เห็นว่า ค่า Saturation จะมีความสามารถแยกระหว่างความเข้มของสีที่เปลี่ยนแปลงไปได้อย่างชัดเจน ซึ่งจากการที่เลือกใช้โมเดลสี HSV ก็เนื่องจากว่า งานวิจัยโมเดลสี RGB และ HSV[1] ซึ่งแสดงให้เห็นความแตกต่างที่ว่าเมื่อค่าของแสงเปลี่ยนไป ค่าสีของโมเดล RGB จะให้ค่าไม่เท่ากัน ยกตัวอย่างเช่น ภาพที่ได้ในช่วงเวลา เช้า กับ บ่าย เมื่อถ่ายภาพในสถานที่เดียวกันแล้วนำค่าสีแต่ละพิกเซลมาเปรียบเทียบกันจะได้ความแตกต่าง และอ้างอิงจากการทดลองในงานวิจัยที่กล่าวถึงเล่มนี้ได้แสดงการทดลองการใช้โมเดลสี HSV ซึ่งแสดงให้เห็นว่าแม้แสงจะเปลี่ยนแปลง แต่ภาพที่ได้ก็จะมีสีและความเข้มสีที่ไม่เปลี่ยนแปลงไป ดั้งนั้นในโครงงานนี้จึงเลือกใช้โมเดลสี HSV และภาพ S-Plane โดยการแปลงด้วย cvCvtColor(img, img, CV_BGR2HSV) CV_BGR2HSV คือการแปลงจากโมเดล RGB เป็นโมเดล HSV จากผลการทดลองนี้จะเห็นได้ว่า การค้นหาเป้าหมายสามารถทำได้ สำหรับการทดสอบการค้นหาเป้าหมายในขั้นตอนต่อไปคือ การเลือกค่า Threshold ที่เหมาะสม

4.1.2 การทดสอบเลือกค่าเส้นแบ่งที่เหมาะสม

จากการค้นหาเป้าหมายในหัวข้อที่ 4.1.1 ได้ข้อสรุปว่าการใช้ S-Plane ให้ผลลัพธ์ที่ใกล้เคียงแต่มีขั้นตอนในการทำค่าเส้นแบ่ง (Threshold) หลังจากการลบระหว่างภาพสถานการณ์จริงกับภาพเทมเพลท เราต้องการภาพลัพธ์ที่มีค่าเป็น Binary จึงต้องมีเส้นแบ่งเพื่อปรับค่าในพิกเซล โดยจะจำลองค่าของการลบกันระหว่างภาพที่มีค่าตั้งแต่ 0 – 255 ดังรูปที่ 4.5

255	253	128	210		128	128	128	128		127	125	0	82
255	253	170	223		128	128	128	128		127	125	42	95
255	253	160	210		128	128	128	128		127	125	32	82
223	223	146	223		128	128	128	128		95	95	18	95
255	223	122	210		128	128	128	128		127	95	6	82
255	223	150	210	—	128	128	128	128	=	127	95	22	82
255	223	190	233		128	128	128	128		127	95	62	105
255	223	140	223		128	128	128	128		127	95	12	95
255	223	140	200		128	128	128	128		127	95	12	72
255	223	140	200		128	128	128	128		127	95	12	72
255	223	140	200		128	128	128	128		127	95	12	72
Original					Template					Destinate			

รูปที่ 4.5 ภาพจำลองการลบกันระหว่างภาพสองภาพและภาพผลลัพธ์

จากรูปที่ 4.5 เมื่อสังเกตค่าที่อยู่ในหลักที่ 3 ของทั้งสองภาพคือ Original กับ Template มีค่าใกล้เคียงกัน นั้นหมายความว่า ภาพ Original เป็นสีที่มีใกล้เคียงกับสีที่ค้นหา ค่าที่ได้จากการลบจะมีค่าน้อยใกล้เคียงศูนย์ ดังนั้นการใช้ค่าเส้นแบ่งนี้จะเป็นตัวกำหนดว่าค่าใดบ้างเป็นสีที่ใกล้เคียงกับสีที่เราค้นหา ยกตัวอย่างเช่น ถ้าเราใช้ค่าเส้นแบ่ง ที่ 50 ก็หมายความว่าค่าที่อยู่ เหนือ 50 หรือ มากกว่า 50 จะเป็นสีที่ไม่ใกล้เคียงกับสีที่ค้นหา เราจึงให้ค่าสีเหล่านี้เป็น 0 ทั้งหมด สำหรับค่าที่อยู่ต่ำลงมาจะเป็น 255 ทั้งหมดก็คือสีที่ค้นหาอยู่นั้นเอง แต่ถ้าเราให้ค่าเส้นแบ่งนี้น้อยๆ ก็หมายถึงว่าโอกาสที่จะเกิดค่าสีที่สนใจมากที่สุดหรือใกล้ศูนย์มากที่สุดนั้นจะเป็นสีที่ถูกต้องมากที่สุด จะเห็นได้ว่าในหลักที่ 3 ของภาพ Destination ถ้าเรากำหนดค่าเส้นแบ่งเท่ากับ 5 จะมีเพียง 1 พิกเซลเท่านั้นที่ตำแหน่ง D(1,3) ที่เป็นสีใกล้เคียงแต่ก็มีค่าที่ใกล้เคียงค่าเส้นแบ่งก็ คือ 6 ใน D(5,3) ก็อาจมีความเป็นไปได้ที่จะเป็นสิ่งที่สนใจแต่เราก็ทำการตัดสินว่าไม่ใช่สีที่ต้องการดังนั้นจึงต้องมีการทดสอบหาค่าเส้นแบ่งที่เหมาะสม แต่จากทฤษฎีจะใช้การทำ Histograms เพื่อดูค่าระหว่างจุดเชื่อมต่อจาก Histogram และนำค่านั้นมาหาค่าเส้นแบ่ง แต่จากโครงงานนี้การใช้วิธีนี้ภาพที่มีลักษณะ Real Time นั้นขนาดของพิกเซลมาก ๆ จะต้องใช้เวลานานในการประมวลผลดังนั้นการประมาณค่าลงไป จะเป็นทางเลือกที่ดีกว่าการประมวลผลทุก ๆ เฟรมภาพตลอดเวลา จากนี้จะเป็นการแสดงการทดสอบค่าเส้นแบ่งที่แตกต่างกัน โดยจะทดสอบค่าเส้นแบ่งเป็นช่วงที่ 25 ,50 ,75 ,100 และ125

การกำหนดค่าจะใช้คำสั่ง cvThreshold(S, S, 50, 255, CV_THRESH_BINARY_INV) โดยค่าที่กำหนดเป็นพารามิเตอร์จะมีดังนี้

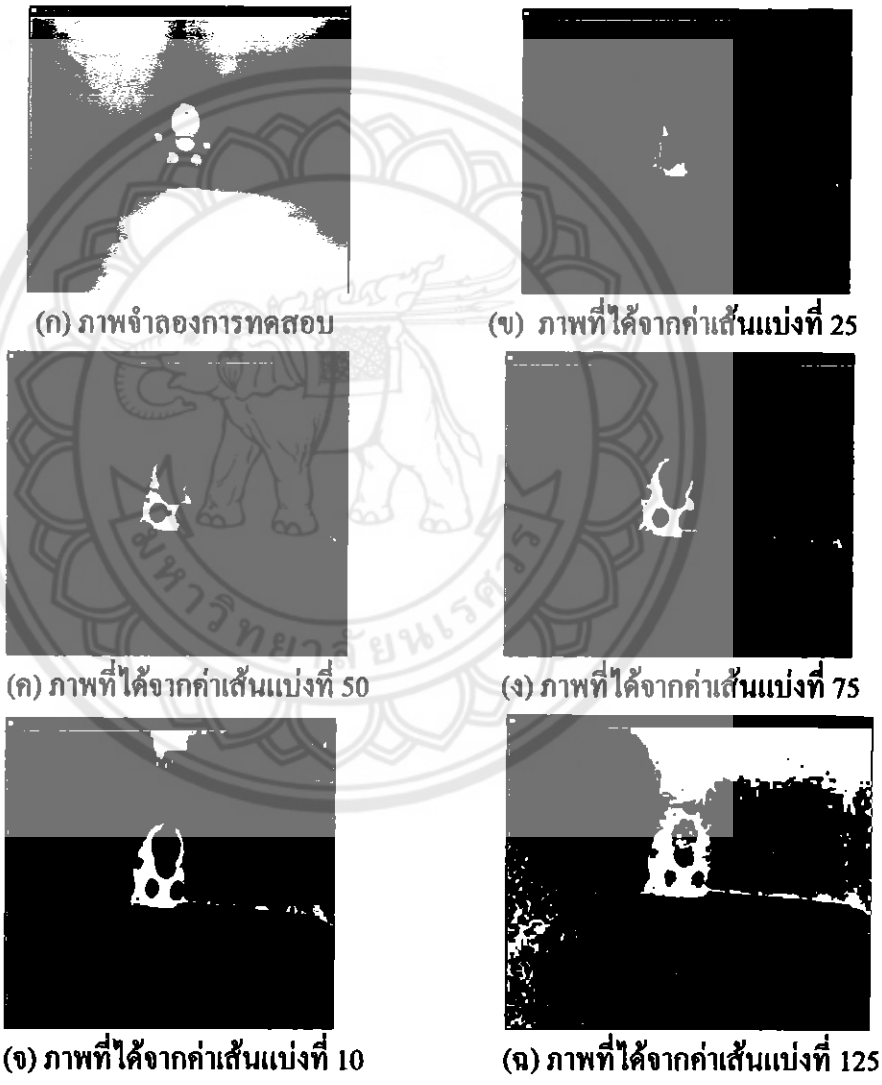
S คือ ภาพที่ได้จากการลบกันระหว่างภาพจริงกับภาพสีที่ต้องการค้นหา

50 คือ ค่าเส้นแบ่ง

255 คือ ค่าที่กำหนดให้ ค่าที่มากกว่าค่าเส้นแบ่ง 50 มีค่าเป็น 255

CV_THRESH_BINARY_INV คือการสลับค่ากันระหว่างค่ามากสุดและน้อยสุด

การกำหนดค่าเส้นแบ่ง



(ก) ภาพจำลองการทดสอบ (ข) ภาพที่ได้จากค่าเส้นแบ่งที่ 25

(ค) ภาพที่ได้จากค่าเส้นแบ่งที่ 50 (ง) ภาพที่ได้จากค่าเส้นแบ่งที่ 75

(จ) ภาพที่ได้จากค่าเส้นแบ่งที่ 10 (ฉ) ภาพที่ได้จากค่าเส้นแบ่งที่ 125

รูปที่ 4.6 ภาพแสดงการเลือกค่าเส้นขีดแบ่งที่เหมาะสม

จากการทดลองกำหนดค่าเส้นแบ่ง (Threshold) ที่แตกต่างกันจะให้ผลที่แตกต่างกันจากการอธิบายในรูปที่ 4.5 และการใช้ค่าเส้นแบ่งนี้ผลการทดลองจะแสดงให้เห็นว่าเมื่อค่าเส้นแบ่งมากขึ้นจะทำให้เกิดสัญญาณรบกวน (noise) เมื่อมีสัญญาณรบกวนเกิดขึ้นมาจนสัญญาณรบกวนติดกันจะเกิดพิกเซลที่เชื่อมต่อกันจนอาจทำให้หุ่นยนต์รวมว่าเป็นเป้าหมายที่สนใจได้ ซึ่งแท้จริงแล้วตรงตำแหน่งนั้นไม่มีสิ่งที่สนใจแต่อย่างใด จึงทำให้การประมวลผลผิดพลาดได้ ดังนั้นค่าระหว่าง 25 – 75 จะเป็นช่วงที่เหมาะสมสำหรับโครงงานนี้และสภาพจำลองนี้ แต่เมื่อมีสัญญาณรบกวนเกิดขึ้นมาจะต้องทำการกำจัดสัญญาณรบกวนออกไป ซึ่งมีหลายกระบวนการในการกำจัดสัญญาณรบกวน ซึ่งจะแสดงการทดลองในหัวข้อต่อไป

4.1.3 การทดสอบค้นหาตำแหน่งของสีที่สนใจและการกำจัดสัญญาณรบกวน

ใน OpenCV การกำจัดสัญญาณรบกวนจะมีหลายชนิดทั้งในรูปแบบ 1 มิติ และ 2 มิติ โดยฟังก์ชั่นในการกรองสัญญาณรบกวนนี้ จะเป็นการทดสอบการกรองของ CV_BLUR , CV_GAUSSIAN, CV_MEDIAN ซึ่งคำสั่งที่ใช้ คือ cvSmooth(N, S, CV_MEDIAN, 9, 0, 0, 0); โดยพารามิเตอร์ของฟังก์ชั่นจะมีดังต่อไปนี้

N คือ ภาพนำเข้าทำการกรองสัญญาณรบกวน

S คือ ภาพผลลัพธ์ที่ได้จากการกรอง

CV_MEDIAN คือ การกรองแบบ MEDIAN

9 คือ ขนาดของ Mask หรือ Kernel ที่ทำการ Convolution กับภาพ N หรือ 3 x 3 พิกเซล

จากนี้จะเป็นการทดสอบการกรองสัญญาณรบกวนที่ลักษณะเป็นฝุ่นและเม็ดพริกไทย ดังแสดงในรูปที่ 4.6

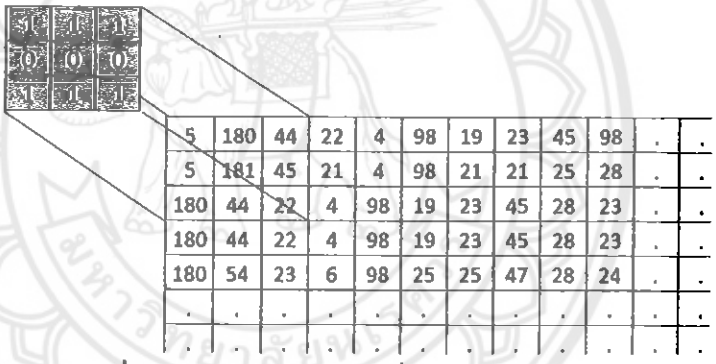
(ก) Filter CV_MEDIAN (ข) Filter CV_BLUR (ค) Filter CV_GAUSSIAN

รูปที่ 4.7 ภาพแสดงการกำจัดสัญญาณรบกวนด้วย filter แบบต่างๆ

จากผลการทดลองแสดงให้เห็นได้ว่าการกรองเพื่อกำจัดสัญญาณรบกวนทั้งสามแบบจะได้ผลที่คล้ายกันคือ สัญญาณรบกวนจะลดลงขอบของภาพผู้ประสบภัยจะมนลงตัดส่วนที่ไม่เชื่อมต่อกันกันออกไป ภาพ ก) แสดงรายละเอียดได้ชัดเจนมากที่สุดดังนั้นจึงเลือกใช้ CV_MEDIAN และส่วนที่ต้องพิจารณาอันดับต่อไปที่ต้องทำการทดลองก็คือ Mask หรือ Kernel ซึ่งเป็นกรอบที่ใช้ในการทำการ Convolution ไปบนภาพที่ต้องทำการกรองสัญญาณรบกวน ซึ่งจะแสดงการทดลองในหัวข้อต่อไป

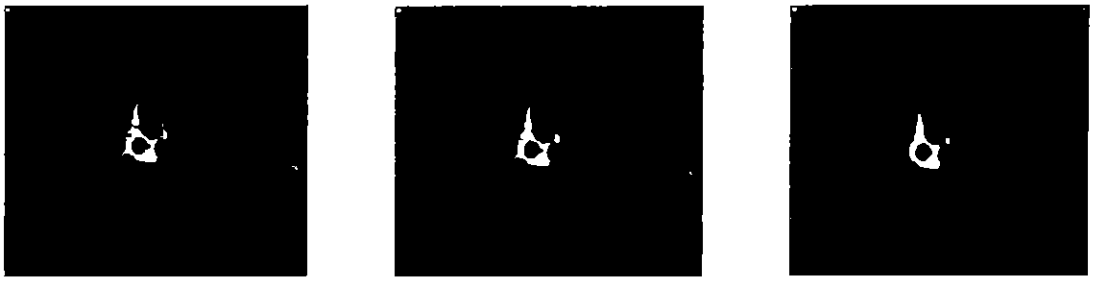
4.1.4 การกำจัดสัญญาณรบกวนและเลือกขนาด Kernel ที่เหมาะสม

จากหัวข้อที่ผ่านการทำการกรองสัญญาณรบกวนออก จะทำให้ผลลัพธ์ที่ได้นั้นมีความเรียบบริเวณขอบ ประโยชน์ที่เกิดขึ้นในโครงงานนี้คือ จะตัดจุดพิกเซลที่มีขนาดเล็กเกินกว่าที่จะเป็นจุดสนใจออกไปเพื่อลดเวลาในการประมวลผลภาพ เนื่องจากการพิจารณาว่าเป็นสิ่งที่ค้นหาหรือไม่จากการนับจุดที่ต่อเชื่อม จึงจำเป็นต้องกรองจุดเล็กๆเหล่านั้นออกไป แต่สำหรับการทดลองนี้ เมื่อเลือกวิธีการกรองได้แล้วนั้น การเลือกขนาดของ Kernel ที่เหมาะสมก็มีส่วนสำคัญ รูปที่ 4.8 จะแสดงตัวอย่างการใช้ Kernel 3x3



รูปที่ 4.8 แสดงตัวอย่างการเลื่อนของ Kernel 3x3

ซึ่งจะแสดงให้เห็นในตัวอย่างจากการทดลองดังรูปที่ 4.8 ค่าตรงกลางจะถูกเปลี่ยนไปจากค่าที่อยู่รอบข้าง ดังนั้นถ้ากรอบมีขนาดใหญ่ขึ้นการคูณก็จะเพิ่มมากขึ้นทำให้ภาพที่ได้เป็นการเฉลี่ยจากค่ารอบข้างมากขึ้น จึงใช้การทดสอบ Kernel ที่มีขนาดดังนี้ คือ 3 x 3, 7 x 7, 15 x 15 และแสดงตัวอย่างในรูปที่ 4.9



(ก) MEDIAN Kernel 3 x 3 (ข) MEDIAN Kernel 7 x 7 (ค) MEDIAN Kernel 15 x 15

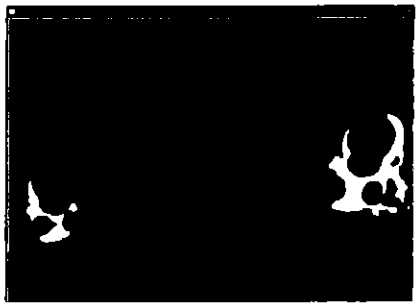
รูปที่ 4.9 แสดงตัวอย่าง MEDIAN Filter กับ Kernel ขนาดต่างๆ

จากผลการทดลองที่การใช้ Kernel ขนาดต่างๆ กันภาพผลลัพธ์ที่เกิดขึ้นจะเห็นว่าที่ Kernel 3x3 พิกเซลเล็กๆ จะกระจายออกจากกลุ่มของเป้าหมายที่ค้นหา ถัดมาภาพ ข) แสดงให้เห็นว่าเกิดการตัดจุดเล็กๆออกไป ทำให้ จุดพิกเซลเล็กที่กระจายออกจากกลุ่มของเป้าหมายที่ค้นหาลดลง และหายไป ส่วนภาพแสดง Kernel 15x15 สีแดงให้เห็นว่า จุดพิกเซลเล็กที่กระจายออกจากกลุ่มของเป้าหมายที่ค้นหาหายไปทั้งหมดจึงทำสรุปได้ว่าในภาพตัวอย่างที่ทำการทดสอบนี้ถ้า Kernel มากขึ้นขนาดของภาพเป้าหมายจะลดลงจุดพิกเซลเล็กๆ จะลดลง ซึ่งในการเลือกที่เหมาะสมนี้มีความสำคัญในการเลือกเพื่อให้จุดสำคัญของผู้ประสบภัยหรือรายละเอียดบางส่วนอาจไม่ครบถ้วน ซึ่งอาจเป็นผลกระทบในการประมวลผลภาพเพื่อวิเคราะห์หรือชี้ตำแหน่งของผู้ประสบภัยได้ ดั้งนั้นจึงเลือกที่ Kernel 9 x 9 ในการทำโครงงานนี้ และการวิเคราะห์จากจำนวนพิกเซลที่ต่อกันเพื่อกำหนดตำแหน่งของผู้ประสบภัยจะอธิบายในหัวข้อต่อไป

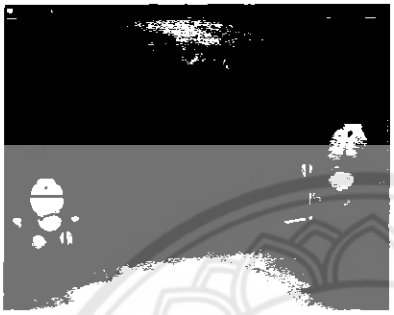
4.1.5 การทดสอบการกำหนดตำแหน่งของผู้ประสบภัย

จากขั้นตอนการทดลองที่ผ่านมาทั้งหมดจะได้ภาพที่มีการระบุว่าหุ่นยนต์สามารถค้นหาผู้ประสบภัยได้ แต่จากการทดลองที่ผ่านมาจะเห็นได้ว่าโปรแกรมจะนับจุดที่สนใจที่เรียงติดกันเพื่อหาว่ากลุ่มสีนั้นๆมากพอจะเป็นสิ่งที่เราสนใจหรือไม่ เพราะฉะนั้นหากภาพที่รับมาได้มองเห็นผู้ประสบภัยในระยะไกลจุดสีขาวในภาพก็จะมีจำนวนน้อย แต่เมื่อหุ่นยนต์เคลื่อนตัวเข้าไปใกล้ๆภาพที่ได้ก็จะใหญ่ขึ้นทำให้กลุ่มสีในภาพก็ใหญ่ขึ้นไปด้วย จึงเกิดคำถามที่ว่าระยะห่างที่เหมาะสมที่จะทำให้เกิดกลุ่มสีขาวที่จะตัดสินใจว่าเป็นเหยื่อนั้นควรเป็นเท่าใด ในการเริ่มวิเคราะห์หรือนับจำนวนพิกเซล ซึ่งถ้าวิเคราะห์ที่ระยะทางระหว่างผู้ประสบภัยมากเกินไป สิ่งที่ตรวจสอบได้อาจเป็นวัตถุที่ไม่ใช่เป้าหมายก็เป็นได้ดังนั้นจะต้องมีการทดสอบหาค่าจำนวนพิกเซลที่เหมาะสมในการนับ จากการทดสอบนี้จะทดลองที่ค่า 1000 และ 5000 พิกเซล ซึ่งผลการทดได้เป็นดังต่อไปนี้

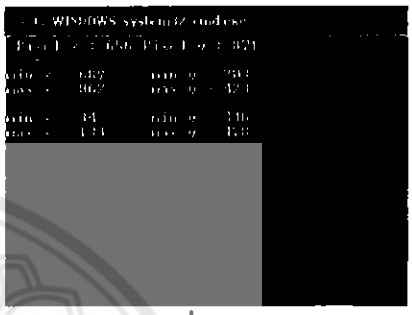
(ก) ภาพจำลองการทดสอบ



(ข) ภาพจากการค้นหา



(ค) แสดงการมาร์คตำแหน่ง



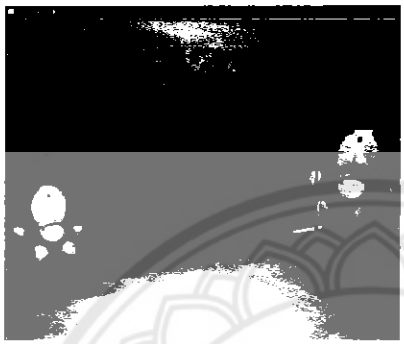
(ง) ตำแหน่งที่ได้จากเป้าหมาย

รูปที่ 4.10 แสดงตัวอย่างตรวจนับพิกเซลเป้าหมายที่ค้นหาพบที่ 1000 พิกเซล

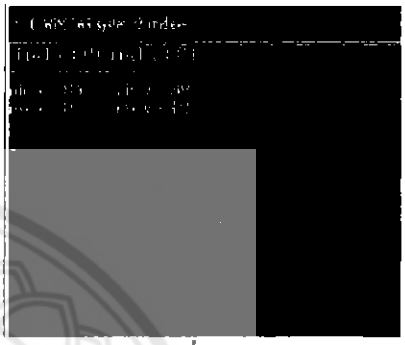
จากทดลองนี้ที่การนับพิกเซลที่หุ่นยนต์ค้นหาเป้าหมายพบ จะเห็นว่ามีเป้าหมายอยู่ 2 ตำแหน่ง ที่มีขนาดต่างกัน ซึ่งจะจำลองในสองแง่มุมคือในขณะที่หุ่นยนต์อยู่ใกล้เป้าหมายขนาดจะใหญ่กว่าขณะที่อยู่ห่างจากเป้าหมาย ภาพของเป้าหมายจะเล็กกว่าทำให้เกิดจำนวนพิกเซลไม่เท่ากัน ดังนั้นเป้าหมายที่มีขนาดเล็กกว่าจำนวนพิกเซลจะต้องน้อยกว่า ซึ่งในการทดลองนี้ทดลองที่ว่าถ้าพิกเซลมากกว่า 1000 พิกเซลแสดงว่าเป็นเป้าหมายจึงมีการกำหนดตำแหน่งของเป้าหมาย 2 ตำแหน่ง และถ้ามีการเพิ่มการนับเป็น 5000 พิกเซล การกำหนดตำแหน่งจะเป็นอย่างไร ซึ่งจะแสดงในการทดลองรูปที่ 4.11

(ก) ภาพจำลองการทดสอบ

(ข) ภาพจากการค้นหา



(ค) แสดงการมาร์คตำแหน่ง



(ง) ตำแหน่งที่ได้จากเป้าหมาย

รูปที่ 4.11 แสดงตัวอย่างตรวจนับพิกเซลเป้าหมายที่ค้นหาพบที่ 5000 พิกเซล

จากผลการทดลองจะแสดงให้เห็นได้ว่าเมื่อการนับพิกเซลที่ต่างกันขนาดของผู้ประสบภัยในระยะห่างที่ต่างกันก็จะมีผลในการค้นหาเป้าหมาย ซึ่งอาจทำให้เกิดความผิดพลาดในการระบุตำแหน่งของผู้ประสบภัยได้ ดังนั้นจำเป็นที่ต้องมีการทดสอบและวัดระยะทางจากหุ่นยนต์กับผู้ประสบภัยและหาอัตราความผิดพลาดซึ่งจะแสดงในส่วนต่อไป คือการให้หุ่นยนต์เคลื่อนที่และค้นหาเป้าหมายเพื่อเป็นการทดสอบระยะห่างและอัตราความผิดพลาด

4.2 การเคลื่อนที่หุ่นยนต์และค้นหาเป้าหมายอัตโนมัติ

4.2.1 การทดสอบหุ่นยนต์เคลื่อนที่หลบหลีกสิ่งกีดขวาง

จากการจำลองสถานการณ์ให้ใกล้เคียงกับเหตุการณ์ที่เกิดจริง ที่มีอุปสรรคในการเคลื่อนที่ของหุ่นยนต์เช่น กำแพงหรือก้อนหิน และอื่นๆ อีกมากมาย ซึ่งหุ่นยนต์จะต้องตรวจจับวัตถุด้วยระบบเซนเซอร์ตรวจจับระยะทางจากหุ่นยนต์ถึงวัตถุที่ตรวจพบ ตัวอย่างเช่นเมื่อหุ่นยนต์เคลื่อนที่เข้าใกล้กำแพงหุ่นยนต์จะรู้ได้ว่าขณะนี้หุ่นยนต์ตรวจจับวัตถุได้แล้วเมื่อเคลื่อนที่เข้าไประบบเซนเซอร์ทุกตัวจะทำการตรวจจับโดยที่รับสัญญาณจากเซนเซอร์แต่ละตัวเข้าประมวลผล เพื่อควบคุมให้หุ่นยนต์เคลื่อนที่หลบหลีกหรือเคลื่อนที่ตามวัตถุที่ตรวจพบ

จากนี้เป็นการทดสอบหุ่นยนต์ให้เคลื่อนที่ตามกำแพงโดยที่ยังไม่มีการค้นหาเหยื่อ เพื่อทดสอบว่าหุ่นยนต์จะเคลื่อนที่ได้ตามสถานการณ์จำลองได้ถูกต้องเพียงใด โดยที่การทดลองจะมีการจำลองสถานการณ์ที่แตกต่างกันและผลการทดลองจะแสดงในรูปแบบของตาราง

4.2.1.1 การทดสอบหุ่นยนต์เคลื่อนที่หลบหลีกสิ่งกีดขวาง สถานการณ์ที่ 1

รูปที่ 4.12 แสดงสถานการณ์จำลองการเคลื่อนที่แบบทางตรง และทางแยก มีผนังกีดขวาง

ตารางที่ 4.1 ผลการทดสอบหุ่นยนต์เคลื่อนที่ค้นหาเป้าหมายแบบมีผนังกีดขวางและทางแยก

ครั้งที่	ระยะทาง (เมตร)	เวลา (วินาที)	ผลการทดลอง
1	8	50	ผ่าน
2	8	51	ผ่าน
3	8	53	ผ่าน
4	8	53	ผ่าน
5	8	54	ผ่าน
6	8	53	ผ่าน
7	8	55	ผ่าน
8	8	57	ผ่าน
9	8	60	ผ่าน
10	8	58	ผ่าน

จากผลการทดลองสามารถทำให้เห็นได้ว่าการเคลื่อนที่ของหุ่นยนต์ทำได้ทุกครั้ง เวลาที่แสดงในตารางที่ 4.1 นี้หมายถึงระยะเวลาที่หุ่นยนต์เคลื่อนที่จากจุดเริ่มต้นไปจุดที่จุดสิ้นสุด เมื่อสังเกตในการทดสอบครั้งที่ 1 เวลาที่ใช้ในการเคลื่อนที่ของหุ่นยนต์จะน้อยที่สุด และเพิ่มมากขึ้นเรื่อย ๆ

4.2.1.2 การทดสอบหุ่นยนต์เคลื่อนที่หลบหลีกสิ่งกีดขวาง สถานการณ์ที่ 2

รูปที่ 4.13 แสดงสถานการณ์จำลองการเคลื่อนที่แบบทางตรง ทางแยก และทางตันขวาง

ตารางที่ 4.2 ผลการทดสอบหุ่นยนต์เคลื่อนที่ค้นหาเป้าหมายแบบมีผนังกีดขวาง ทางแยกและทางตันขวาง

ครั้งที่	ระยะทาง (เมตร)	เวลา (วินาที)	ผลการทดลอง
1	9	60	ผ่าน
2	9	61	ผ่าน
3	9	61	ผ่าน
4	9	63	ผ่าน
5	9	64	ผ่าน
6	9	65	ผ่าน
7	9	65	ผ่าน
8	9	68	ผ่าน
9	9	70	ผ่าน
10	4	-	ไม่ผ่าน

จากทดสอบการเคลื่อนที่ในสถานการณ์ที่ 2 นี้แสดงให้เห็นการเคลื่อนที่ด้วยความเร็ว และใช้ระยะเวลาในการเคลื่อนที่จากจุดเริ่มต้นทดสอบ จนถึงจุดสิ้นสุดของการทดสอบ มีการเพิ่มขึ้นเมื่อทดสอบหุ่นยนต์ไปหลาย ๆ รอบ ยกตัวอย่างผลการทดลองในครั้งที่ 10 จะเห็นว่าหุ่นยนต์ไม่สามารถเคลื่อนที่ออกจากทางตัน เนื่องจากขณะที่กลับตัวหุ่นยนต์นั้นความเร็วในการเลี้ยวลดลงอย่างมากจึงทำให้หลบสิ่งกีดขวางไม่พ้น การทดลองในครั้งที่ 10 จึงไม่ผ่านการทดสอบ แต่สำหรับการทดสอบครั้งก่อนหน้าตั้งแต่ครั้งที่ 1 – 9 ระยะเวลาในการเคลื่อนที่ของหุ่นยนต์มีความแตกต่างในลักษณะเดียวกันกับการทดลองในสถานการณ์ที่ 1 จึงเป็นข้อสันนิฐานเบื้องต้นว่าแหล่งพลังงานที่จ่ายให้กับหุ่นยนต์ที่ใช้ในการขับเคลื่อนจะลดลงไปเรื่อย ๆ ความเร็วในการหมุนของมอเตอร์ขับเคลื่อนจึงลดลง

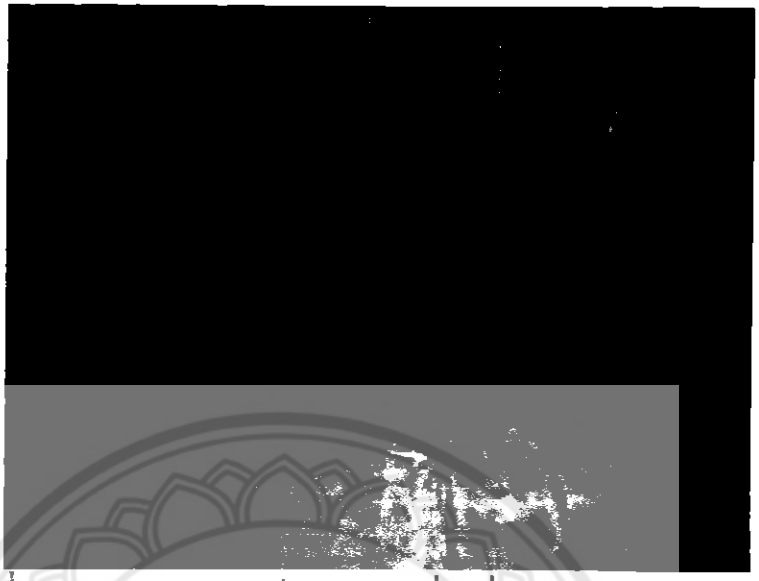
วิธีแก้ไขในเบื้องต้นคือ การนำแบตเตอรี่มาชาร์จพลังงานและทำการทดลองใหม่อีกในสถานการณ์จำลองเดิมอีก 5 ครั้ง

ตารางที่ 4.3 ผลการทดสอบหุ่นยนต์เคลื่อนที่แบบมีผนังกีดขวาง ทางแยกและทางตันขวาง รอบที่ 2 หลังจากการชาร์จแบตเตอรี่

ครั้งที่	ระยะทาง (เมตร)	เวลา (วินาที)	ผลการทดลอง
1	9	60	ผ่าน
2	9	61	ผ่าน
3	9	61	ผ่าน
4	9	61	ผ่าน
5	9	61	ผ่าน

จากการทำสอบในรอบที่ 2 ในสถานการณ์จำลองที่ 2 จำนวน 5 ครั้งจะเห็นได้ว่าหุ่นยนต์สามารถเคลื่อนที่ได้โดยที่เวลาไม่เพิ่มขึ้น และผ่านการทดสอบ อย่างไม่มีปัญหา ดังนั้นในการทดสอบนี้จะได้ประโยชน์และเห็นข้อบกพร่องของหุ่นยนต์เมื่อหุ่นยนต์มีพลังงานลดลงจากการทำงาน หุ่นยนต์จะเคลื่อนที่ผิดพลาดและอาจติดอยู่ในสถานการณ์จริงได้เมื่อเข้าช่วยเหลือผู้ประสบภัย และการทดสอบในสถานการณ์ต่อไปเมื่อหุ่นยนต์เคลื่อนที่ผ่านทางแยก

4.2.1.3 การทดสอบหุ่นยนต์เคลื่อนที่หลบหลีกสิ่งกีดขวาง สถานการณ์ที่ 3



รูปที่ 4.14 แสดงสถานการณ์จำลองการเคลื่อนที่แบบทางตรง ทางแยกซ้าย

ตารางที่ 4.4 ผลการทดสอบหุ่นยนต์เคลื่อนที่แบบมีผนังกีดขวาง ทางแยกซ้าย

ครั้งที่	ระยะทาง (เมตร)	เวลา (วินาที)	ผลการทดลอง
1	5	40	ผ่าน
2	5	41	ผ่าน
3	5	41	ผ่าน
4	5	43	ผ่าน
5	5	44	ผ่าน
6	5	44	ผ่าน
7	5	44	ผ่าน
8	5	44	ผ่าน
9	5	44	ผ่าน
10	5	45	ผ่าน

การทดสอบในการทดลองนี้ข้อกำหนดของการทดสอบคือ เมื่อหุ่นยนต์เคลื่อนที่พบทางแยกซ้ายมือหุ่นยนต์ต้องเลี้ยวซ้ายเพื่อเข้าสำรวจพื้นที่ทางบริเวณซ้ายมือก่อน

เมื่อพบทางตันหุ่นยนต์จะต้องเคลื่อนที่ออกจากเส้นทางนั้นและเคลื่อนที่เลี้ยวซ้ายเพื่อค้นหาในตำแหน่งอื่นต่อไป จากผลการทดลองหุ่นยนต์สามารถเคลื่อนที่ผ่านการทดสอบ และเมื่อระยะทางลดลงเวลาในการเคลื่อนที่ก็ลดลงด้วย แต่เมื่อสังเกตเวลาในการเคลื่อนที่จะไม่แตกต่างกันมาก

4.2.1.4 การทดสอบหุ่นยนต์เคลื่อนที่หลบหลีกสิ่งกีดขวาง สถานการณ์ที่ 4

รูปที่ 4.15 แสดงสถานการณ์จำลองการเคลื่อนที่แบบทางตรง ทางแยกขวา

ตารางที่ 4.5 ผลการทดสอบหุ่นยนต์เคลื่อนที่แบบมีผนังกีดขวาง ทางแยกขวา

ครั้งที่	ระยะทาง (เมตร)	เวลา (วินาที)	ผลการทดลอง
1	7	55	ผ่าน
2	7	55	ผ่าน
3	7	56	ผ่าน
4	5	-	ไม่ผ่าน
5	5	58	ผ่าน
6	7	57	ผ่าน
7	5	-	ไม่ผ่าน
8	7	58	ผ่าน
9	7	58	ผ่าน
10	7	58	ผ่าน

การทดสอบในการทดลองนี้ข้อกำหนดของการทดสอบคือ เมื่อหุ่นยนต์เคลื่อนที่พบทางแยกขวามือหุ่นยนต์ต้องเคลื่อนที่ผ่านไปโดยไม่เข้าสำรวจพื้นที่ทางบริเวณขวามือ เมื่อเคลื่อนที่เข้าไปแต่ไม่พบเป้าหมายใดหรือพบทางตันหุ่นยนต์จะต้องเคลื่อนที่ออกจากเส้นทางนั้นและเคลื่อนที่เลี้ยวซ้ายเพื่อค้นหาในบริเวณทางขวามือที่ได้เคลื่อนที่ผ่านมาที่ได้กล่าวมาแล้วในข้างต้น จากผลการทดลองหุ่นยนต์สามารถเคลื่อนที่ผ่านการทดสอบ และเมื่อระยะทางลดลงเวลาในการเคลื่อนที่ก็ลดลงด้วย แต่เมื่อสังเกตเวลาในการเคลื่อนที่จะไม่แตกต่างกันมาก

ปัญหา การทดลองนี้ในครั้งที่ 4 และการทดลองครั้งที่ 7 ไม่ผ่านการทดสอบเนื่องจากเกิดความผิดพลาดในช่วงกลับตัวเมื่อพบทางตัน จากการแก้ไขปัญหา จากการตรวจสอบระบบเซนเซอร์ทำให้ทราบว่าหุ่นยนต์ได้รับค่าที่ผิดพลาดทำให้เกิดการประมวลผลผิดพลาดตามไปด้วย เมื่อตรวจสอบจึงพบว่าการทำงานของระบบเซนเซอร์ไม่มีปัญหา แต่ปัญหาที่เกิดขึ้นคือการรับส่งข้อมูลระหว่างหน่วยประมวลผลระดับ High Level (Laptop) กับหน่วยประมวลผลระดับ Low Level (MCU) สื่อสารกันผิดพลาดจากการทำงานไม่เข้าจังหวะกันในการรับส่งข้อมูล ทำให้เกิดการรับค่าที่ไม่ถูกต้องจากเซนเซอร์ไปประมวลผลทำให้หุ่นยนต์เคลื่อนที่ผิดพลาด หลังจากทราบสาเหตุ สามารถแก้ปัญหาได้โดยเพิ่มความเร็วในการรับส่งข้อมูลการสื่อสารอนุกรมจากเดิมให้สื่อสารกันที่ 9600 bit / sec ปรับเป็น 19200 bit / sec ผลที่ได้จากการเปลี่ยนค่า Baud rate นี้จะให้เกิดการสื่อสารที่เร็วขึ้นจำนวนข้อมูล จะถูกส่งได้เร็วยิ่งขึ้น แต่จำเป็นที่ต้องเปลี่ยนแปลงค่าใน Register TH ของการสื่อสารอนุกรมด้วยสำหรับ MCU และสำหรับการตั้งค่าใน Laptop ก็ต้องปรับค่าใน Comport เป็น 19200 bit / sec ด้วย และจากการรับ – ส่งขอมูลไม่เข้าจังหวะกัน การแก้ไขคือการให้ Laptop ทำการหน่วงเวลาหรือรอให้ MCU ปฏิบัติงานให้เรียบร้อยจึงไปรับค่ามาประมวลผล ซึ่งจากการแก้ปัญหาด้วยวิธีการนี้ได้ผล สามารถแก้ปัญหาได้ จากการทดลองครั้งที่ 8,9,10 หุ่นยนต์สามารถเคลื่อนที่ได้อย่างไม่เกิดปัญหาใด

4.2.2 การทดสอบหุ่นยนต์เคลื่อนที่หลบหลีกสิ่งกีดขวางและค้นหาเป้าหมาย

จากหลักการและการทดลองที่ผ่านมาในขั้นตอนต่อไปนี้เป็นสิ่งที่สำคัญคือการรวมกันระหว่างการหลบหลีกสิ่งกีดขวางและค้นหาเป้าหมายในเวลาเดียวกัน จากหลักการออกแบบในบทที่ 3 ได้แสดงสถานการณ์ทำงานของหุ่นยนต์คือเมื่อหุ่นยนต์ไม่พบเป้าหมายหุ่นยนต์จะเคลื่อนไปตามเส้นทาง แต่เมื่อหุ่นยนต์ค้นหาเป้าหมายพบ การเคลื่อนที่ของหุ่นยนต์จะเปลี่ยนไปคือ เคลื่อนที่เข้าหาเป้าหมาย และเมื่ออยู่ในระยะที่เหมาะสม หุ่นยนต์จะทำการเก็บสภาวะแวดล้อม และตำแหน่งของผู้ประสบภัยไปประมวลผลตำแหน่งของผู้ประสบภัยต่อไป ดังนั้นการทดลองนี้จะเป็นการจำลองสถานการณ์เหมือนกับการทดสอบการเคลื่อนที่ในสถานการณ์ที่ 2 และ ในสถานการณ์ที่ 3

ผู้ประสบภัยจำลองใช้ตุ๊กตาที่มีสีต่างจากพื้นและกำแพงเพื่อให้หุ่นยนต์สามารถตรวจจับสีตามหลักการที่ได้กล่าวไว้ในบทที่ 2 ในเรื่องของการค้นหาสีที่น่าสนใจด้วยการประมวลผลจากภาพ ตัวอย่างของผู้ประสบภัยแสดงไว้ในรูปที่ 4.16

รูปที่ 4.16 ภาพหุ่นยนต์และผู้ประสบภัยจำลอง

4.2.2.1 การทดสอบหุ่นยนต์เคลื่อนที่หลบหลีกสิ่งกีดขวางและค้นหาเป้าหมาย สถานการณ์ที่ 5

รูปที่ 4.17 แสดงสถานการณ์จำลองการค้นหาผู้ประสบภัย 1 คน

ตารางที่ 4.6 ผลการทดสอบหุ่นยนต์เคลื่อนที่และค้นหาผู้ประสบภัยจำนวน 1 คน

ครั้งที่	ระยะทาง (เมตร)	เวลา (วินาที)	ผู้ประสบภัย ที่ 1	ผลการเคลื่อนที่
1	8	68	พบ	ผ่าน
2	8	68	พบ	ผ่าน
3	8	71	พบ	ผ่าน
4	4	-	พบ	ไม่ผ่าน
5	8	72	พบ	ผ่าน
6	8	73	พบ	ผ่าน
7	8	-	พบ	ผ่าน
8	8	74	พบ	ผ่าน
9	4	75	พบ	ไม่ผ่าน
10	8	75	พบ	ผ่าน

จากการทดลองในสถานการณ์ที่ 5 จะเห็นได้ว่า หุ่นยนต์สามารถค้นหาเป้าหมายได้อย่างไม่ผิดพลาดสำหรับผู้ประสบภัยจำนวน 1 คน แต่การเคลื่อนที่ยังคงมีปัญหาอยู่ดังนั้นในการทำงานประสานกันของทั้งการค้นหาเป้าหมายและการเคลื่อนที่ ทำงานเข้ากันได้เป็นอย่างดีก็จะทำให้หุ่นยนต์สามารถทำงานได้อย่างมีประสิทธิภาพมากยิ่งขึ้น ต่อไปจะเป็นการทดสอบด้วยการเพิ่มผู้ประสบภัยในสถานการณ์จำลองอีก 1 คน

4.2.2.2 การทดสอบหุ่นยนต์เคลื่อนที่หลบหลีกสิ่งกีดขวางและค้นหาเป้าหมาย สถานการณ์ที่ 6

รูปที่ 4.18 แสดงสถานการณ์จำลองการค้นหาผู้ประสบภัย 2 คน

ตารางที่ 4.7 ผลการทดสอบหุ่นยนต์เคลื่อนที่และค้นหาผู้ประสบภัยจำนวน 2 คน

ครั้งที่	ระยะทาง (เมตร)	เวลา (วินาที)	ผู้ประสบภัย ที่ 1	ผู้ประสบภัย ที่ 2	ผลการเคลื่อนที่
1	8	68	พบ	พบ	ผ่าน
2	8	68	พบ	พบ	ผ่าน
3	8	71	พบ	พบ	ผ่าน
4	4	-	พบ	ไม่พบ	ไม่ผ่าน
5	8	72	พบ	พบ	ผ่าน
6	8	73	พบ	พบ	ผ่าน
7	8	-	พบ	พบ	ผ่าน
8	8	74	พบ	พบ	ผ่าน
9	4	75	พบ	ไม่พบ	ไม่ผ่าน
10	8	75	พบ	พบ	ผ่าน

การทดลองนี้ได้ทำการเพิ่มผู้ประสบภัยเข้าไปในสถานการณ์จำลองโดยเงื่อนไขในการทดลองนี้คือ เมื่อหุ่นยนต์เคลื่อนที่พบเป้าที่ 1 และจะต้องเคลื่อนที่ต่อไปและค้นหาเป้าหมายใหม่ที่ไม่ซ้ำกับเป้าหมายเดิมที่เคยค้นหาพบแล้วอีก ดังนั้นเมื่อหุ่นยนต์ค้นหาเป้าหมายแรกสำเร็จแล้วหุ่นยนต์เคลื่อนที่ไปตามเส้นทาง หุ่นยนต์อาจยังเห็นเป้าหมายแรกอยู่ในกล้องอีกแล้วเข้าใจว่าเป็นผู้ประสบภัยก็จะกลับไปตรวจสอบอีก ดังนั้นจากปัญหาที่เกิดขึ้นนี้สามารถแก้ไขได้โดยไม่ให้หุ่นยนต์เคลื่อนที่กลับมาในเส้นทางที่เคยเคลื่อนที่ไปแล้ว โดยการเพิ่มการรู้จำนวนรอบการหมุนของมอเตอร์หรือล้อของหุ่นยนต์เพื่อนับและเขียนแผนผังการเคลื่อนที่ของหุ่นยนต์ได้ ซึ่งจะเป็นการพัฒนาเพิ่มต่อในโครงงานหุ่นยนต์กู้ภัยนี้ แต่สำหรับโครงการนี้จะให้หุ่นยนต์เคลื่อนที่หลบจากเป้าหมายเดิมเพื่อให้ภาพในกล้องไม่มีผู้ประสบภัยอยู่ภาพแล้วจึงทำการเคลื่อนที่ต่อไปซึ่งผลการทดลองได้แสดงในตารางที่ 4.7 จะเห็นได้ว่าการเคลื่อนที่ของหุ่นยนต์จะมีบางครั้งที่หุ่นยนต์เคลื่อนที่ผิดพลาด ยกตัวอย่างเช่นเมื่อหุ่นยนต์ทำการเลี้ยวแต่เลี้ยวไม่พ้นจากขอบกำแพงทำให้หุ่นยนต์ไม่สามารถเคลื่อนที่ต่อได้ สาเหตุที่เกิดขึ้นนั้นก็คือเมื่อแหล่งจ่ายพลังงาน (แบตเตอรี่) มีแรงดันลดลงทำให้แรงในการหมุนของล้อลดลงมีผลให้การเคลื่อนที่ไม่สมบูรณ์ในขั้นตอนการเลี้ยว ดังนั้นจากที่กล่าวมาข้างต้นเกี่ยวกับการติดตั้ง ตัวนับจำนวนรอบของล้อ (Encoder) จะสามารถนำมาช่วยการทำงานในลักษณะการควบคุมวนปิด (Close loop control) ดังแสดงหลักการในบทที่ 2 เรื่องหลักการทำงานของ Encoder สามารถพัฒนาในส่วนนี้เพิ่มเติมซึ่งจะทำให้หุ่นยนต์เคลื่อนที่ได้อย่างไม่ผิดพลาด

จากการทดลองทั้งหมดของบทที่ 4 นี้สามารถแสดงให้เห็นการทำงานของหุ่นยนต์ที่สามารถเคลื่อนที่และค้นหาเป้าหมายได้ ซึ่งได้อธิบายสาเหตุของปัญหาต่าง ๆ และวิธีการแก้ปัญหาไว้ท้ายการทดลองแต่ละการทดลอง โดยการทดลองเหล่านี้เป็นการจำลองสถานการณ์ขึ้น ซึ่งแต่ละการทดลองจะมีการบันทึกภาพการทดลองเพื่อให้ง่ายต่อความเข้าใจ สามารถดูได้จากแผ่นข้อมูลประกอบการนำเสนอโครงงานซึ่งจะแสดงสถานการณ์ต่างๆ ได้ชัดเจนมากยิ่งขึ้น และในบทที่ 5 จะอธิบายถึงการสรุปผลการทดลองและการทำงานภายในโครงงานนี้ต่อไป

บทที่ 5
สรุปผล

ในบทนี้จะกล่าวถึงสรุปการดำเนินงานที่ได้ทำไป ปัญหาที่พบและข้อเสนอแนะ แนวทางในการพัฒนาเพิ่มเติม เพื่อช่วยในการดำเนินงานดีขึ้น

5.1 สรุปผลของโครงงาน

จากการสร้างและทดสอบหุ่นยนต์กู้ภัยหลบหลีกสิ่งกีดขวางและค้นหาเป้าหมายอัตโนมัติ โดยใช้หลักการและทฤษฎีทางวิศวกรรม ทำให้โครงงานประสบความสำเร็จได้ตามเป้าหมาย จากการทดลองค้นหาสีที่สนใจเพื่อระบุตำแหน่งของสิ่งที่สนใจในภาพนั้นสามารถค้นหาสีได้ทุกครั้งแม้ว่าความสว่างของการค้นหาแต่ละครั้งจะต่างกัน จึงสามารถสรุปได้ว่าหลักการค้นหาสีที่สนใจด้วยวิธี Image Subtraction และใช้แบบจำลองสี HSV ที่พิจารณาเฉพาะแผ่น Saturation นั้นทำงานได้อย่างมีประสิทธิภาพ การทดลองต่อมาคือการทดลองให้หุ่นยนต์เคลื่อนที่ในสนามที่จำลองขึ้นตามมาตรฐานการแข่งขันหุ่นยนต์กู้ภัยชิงชนะเลิศระดับประเทศ ด้วยการใช้อุปกรณ์ตรวจวัดระยะทางที่ติดตั้งไว้รอบตัวหุ่นยนต์นั้น ในเงื่อนไขแรกเป็นการจำลองสนามในทางที่ไม่มีทางแยกโดยมีกำแพงขวางไม่ให้เดินได้อย่างสะดวก ซึ่งเป็นลักษณะพื้นฐานที่หุ่นยนต์ต้องเคลื่อนที่ไม่ให้ชนกำแพงรอบๆ ผลการทดลองหุ่นยนต์สามารถเดินจากจุดเริ่มต้นไปยังจุดหมายที่กำหนดได้ทุกครั้ง และเงื่อนไขที่สองเป็นการจำลองสนามที่มีทางแยกซ้าย หุ่นยนต์สามารถเดินไปตามทางได้อย่างถูกต้อง 9 ใน 10 ครั้ง คิดเป็น 90% ซึ่งสาเหตุเกิดมาจากพลังงานจากแบตเตอรี่ไม่เพียงพอ เมื่อชาร์จแบตเตอรี่แล้ว หุ่นยนต์สามารถเคลื่อนที่ตามเงื่อนไขที่กำหนดได้ 100% เงื่อนไขต่อมาเป็นทางเลี้ยวขวาและมีทางตัน ดังนั้นหุ่นยนต์จะต้องเคลื่อนที่กลับมาทางเดิม ในการทดลองนี้หุ่นยนต์สามารถทำได้ตามเงื่อนไขที่กำหนดได้ 8 ใน 10 ครั้ง คิดเป็น 80% สาเหตุที่ทำให้หุ่นยนต์ไม่สามารถเคลื่อนที่ตามต้องการนั้นคาดว่าเป็นเพราะการติดต่อกันระหว่างไมโครคอนโทรลเลอร์และโปรแกรมควบคุมหุ่นยนต์บนคอมพิวเตอร์ไม่เข้าจังหวะกัน

การทดลองอีกส่วนหนึ่งที่สำคัญคือ การทดลองให้หุ่นยนต์เคลื่อนที่ในสนามจำลองเช่นเดิมโดยที่ในสนามนั้นมีวัตถุที่จำลองเป็นผู้ประสบภัยด้วย และวัตถุนั้นจะมีสีตามที่ตั้งโปรแกรมไว้ให้ค้นหาคือสีแดงและสีเหลือง ในการทดลองแรกมีผู้ประสบภัยเพียงตำแหน่งเดียวหุ่นยนต์สามารถเคลื่อนที่ค้นหาเป้าหมายและเข้าไปตรวจสอบได้ตรงตามข้อกำหนด 8 ใน 10 ครั้ง คิดเป็น 80%

การทดลองสุดท้ายคือการจำลองสนามให้มีผู้ประสบภัย 2 ตำแหน่งต่างกัน แล้วให้หุ่นยนต์ค้นหา ผลปรากฏว่าหุ่นยนต์สามารถค้นหาผู้ประสบภัยได้ครบทั้ง 2 ตำแหน่งและเคลื่อนที่โดยที่ไม่ชนเข้ากับกำแพงหรือผู้ประสบภัยเลย 8 ใน 10 ครั้ง คิดเป็น 100% ในบางครั้งที่หุ่นยนต์ไม่สามารถเคลื่อนที่หรือประมวลผลจากภาพได้ตามที่ต้องการเนื่องจากกล้อง Webcam ที่ใช้มีการเปิด infrared ขณะที่ภาพมืด ทำให้สีในภาพที่กล้องรับได้เปลี่ยนแปลงไป มีผลทำให้การประมวลผลผิดพลาดได้ หุ่นยนต์จึงไม่สามารถระบุได้ว่าภาพที่เห็นมีสีที่สนใจอยู่ด้วยหรือไม่ ดังนั้นจึงจำเป็นที่ต้องมีการทดสอบหุ่นยนต์ก่อนที่จะให้หุ่นยนต์ออกปฏิบัติภารกิจ และหุ่นยนต์ที่เกิดขึ้นจากโครงงานนี้เป็นจุดเริ่มต้นในการพัฒนาหุ่นยนต์กู้ภัยอัตโนมัติ ซึ่งจะมีหลายสิ่งที่ต้องพัฒนาต่อเพื่อให้หุ่นยนต์กู้ภัยอัตโนมัติสามารถออกปฏิบัติภารกิจเพื่อช่วยเหลือผู้ประสบภัยได้ในสถานการณ์จริง และเพื่อเป็นประโยชน์ต่อสังคมต่อไป

5.2 ปัญหาที่พบ

5.2.1 ปัญหาทางด้านโครงสร้างหุ่นยนต์ (ตัวหุ่นยนต์)

1. ข้อจำกัดในเรื่องการแข่งขัน เกี่ยวกับน้ำหนัก ขนาด ของตัวหุ่นยนต์
2. การหาวัสดุอุปกรณ์ที่เหมาะสมในการสร้างหุ่นยนต์ ซึ่งหายากและไม่เหมาะสมกับหุ่นยนต์
3. การใช้เครื่องมือช่าง เช่น เครื่องเชื่อมไฟฟ้า เครื่องกลึง ยังขาดประสบการณ์และความชำนาญ
4. สถานที่ในการสร้างหุ่นยนต์ เวลาที่ใช้ในการสร้างหุ่นยนต์ค่อนข้างจำกัด
5. ประสบการณ์ในการออกแบบตัวหุ่นยนต์ เนื่องจากไม่ได้ศึกษาทางด้านระบบเครื่องกลจึงทำให้การออกแบบหุ่นยนต์ ไม่เป็นไปตามทฤษฎีต่าง ๆ ที่เกี่ยวข้อง
6. งบประมาณที่จำกัดในการสร้างหุ่นยนต์ ซึ่งอุปกรณ์แต่ละชิ้นส่วนมีราคาสูงจึงมีความเสี่ยงเกิดขึ้นถ้าหุ่นยนต์ไม่สามารถปฏิบัติงานได้จริง
7. ระยะเวลาที่จำกัดในการออกแบบเพื่อสร้างหุ่นยนต์
8. เมื่อผลิตชิ้นส่วนมาประกอบกันมักจะไม่พอดี เช่น รูเจาะ ขนาด
9. ยางที่ติดกับตีนตะขาบไม่เกาะกับพื้นเมื่อขึ้นทางลาดชัน

5.1.2 ปัญหาทางด้านระบบควบคุมระดับล่าง (Microcontroller)

1. อุปกรณ์อิเล็กทรอนิกส์แต่ละชนิดมีราคาสูงและเสียหายได้ง่าย
2. การสั่งซื้ออุปกรณ์แต่ละชนิดมักใช้เวลาหลายวันและค่าขนส่งสูงถ้าอุปกรณ์มีน้ำหนักมาก
3. อุปกรณ์ต่าง ๆ และไมโครคอนโทรลเลอร์มีขนาดใหญ่

4. การออกแบบชุดไดรเวอร์ควบคุมมอเตอร์และไมโครคอนโทรลเลอร์ด้วยการใช้น้ำยากัดแผ่นปริ้นทำให้ลายท้องแดงมีเส้นบางและขาดง่ายเมื่อประกอบอุปกรณ์
5. ระบบเซนเซอร์ที่ติดตั้งมีจำนวนน้อย และระยะทางที่ตรวจจับได้อยู่ที่ 0 – 80 Cm. เมื่อตรวจวัดค่าผิดพลาดหุ่นยนต์จะทำงานผิดพลาด
6. อุปกรณ์ในการบันทึกสภาพแวดล้อมผู้ประสบภัยมีราคาสูงมาก เช่น ตรวจวัดก๊าซคาร์บอนไดออกไซด์ อุปกรณ์ตรวจวัดอุณหภูมิ เอ็นโค๊ดเดอร์(นับจำนวนรอบล้อ)
7. แบตเตอรี่หมดเร็ว เนื่องจากหุ่นยนต์เคลื่อนที่ไม่ต่อเนื่องจึงทำให้หุ่นยนต์ใช้กระแสไฟมาก
8. ขณะหุ่นยนต์เคลื่อนที่ผ่านทางลาดชัน กล้องจะไม่อยู่ในระนาบขนานกับพื้นจึงทำให้กล้องมองภาพในมุมบนและล่าง ทำให้หุ่นยนต์อาจประมวลผลภาพที่ผิดพลาดได้

5.1.3 ปัญหาทางด้านระบบควบคุมระดับบน (Image processing)

1. โน้ตบุ๊กมีขนาดใหญ่และน้ำหนักมากและทำให้หุ่นยนต์ขนาดใหญ่ตามไปด้วย
2. การประมวลผลของโน้ตบุ๊ก กับ MCU มักทำงานไม่เข้าจังหวะกัน เช่นจะยกตัวอย่างขณะที่ระบบควบคุมระดับล่างกำลังทำงานอยู่ในสถานะเคลื่อนที่ค้นหาเป้าหมาย และหุ่นยนต์เคลื่อนที่เลี้ยวซ้าย ในขณะนี้กล้องอาจจับภาพเป้าหมายได้ในขณะเลี้ยว แต่หุ่นยนต์ก็ยังคงทำงานต่อไปในสถานะเดิมอยู่จึงทำให้เกิดความผิดพลาดได้
3. สามารถสื่อสารอนุกรมด้วยความเร็วต่ำเมื่อเลือกใช้อุปกรณ์กำเนิดความถี่ที่ความถี่ต่ำ แต่ถ้าใช้ที่อัตราความถี่สูงความผิดพลาดจะสูงขึ้น
4. การประมวลผลภาพมีปัญหาเรื่องของแสงของแต่ละ Model สี มีความผิดพลาดบ่อย ๆ จึงจำเป็นต้องมีเงื่อนไขสำรองหรือการเพิ่มกล้องเพื่อให้ช่วยประมวลผลภาพเพื่อป้องกันการประมวลผลภาพผิดพลาด
5. กล้อง Webcam มีการดีเลย์ภาพ กล้องบางตัวมีการขยายภาพอัตโนมัติ หรือการเปิดแสงอินฟาเรดอัตโนมัติเมื่อแสงสว่างน้อย ทำให้การประมวลผลผิดพลาดได้
6. การเชื่อมต่อสาย USB to Serial มีการดิสคอนเน็กบ่อย ๆ เมื่อเชื่อมต่อกับหุ่นยนต์ เนื่องจากอุปกรณ์ไม่มีมาตรฐาน
7. เมื่อหุ่นยนต์ตรวจพบเป้าหมายผิดพลาดจากการประมวลผลภาพ แต่ความจริงแล้วไม่มีเป้าหมายอยู่ที่ตำแหน่งนั้น ดังนั้นการแก้ปัญหาควรจะมีระบบ AI เข้ามาช่วยในการตัดสินใจในการระบุตำแหน่งของผู้ประสบภัย

8. เมื่อหุ่นยนต์พบเป้าหมายแต่อยู่ห่างมากเกินกว่าค่าที่กำหนด หุ่นยนต์จะตัดสินใจเข้าตรวจสอบที่ตำแหน่งอื่น ดังนั้นควรจะมีการค้นหาระยะทางที่สั้นที่สุดก่อนในการตัดสินใจค้นหาเป้าหมายตำแหน่งต่อไป
9. การรับภาพจากกล้องมาเก็บในลักษณะภาพนิ่งลงในโครงสร้างข้อมูล มักเกิดการนำข้อมูลที่ไม่เป็นปัจจุบันมาประมวลผล

5.3 แนวทางแก้ปัญหาและข้อเสนอแนะ

5.3.1 การแก้ปัญหาทางด้านโครงสร้างหุ่นยนต์ (ตัวหุ่นยนต์)

1. การออกแบบในขั้นตอนแรกและการหาอุปกรณ์มักจะไม่ตรงกันดังนั้นควรหาอุปกรณ์หลายแบบก่อนที่จะทำการออกแบบและลงมือทำ
2. เนื่องจากสถานที่ทำงาน ที่ห้องชมรมโรบอท การทำงานมักเกิดเสียงดังรบกวนการเรียนการสอนของผู้อื่น จึงต้องไปทำงานตอนกลางคืนที่ไม่มีการเรียนการสอน
3. อุปกรณ์ที่ผลิตออกมาแล้วประกอบไม่พอดี เวลาทำไม่ควรรีบร้อน ถ้ายังไม่พอดีก็ให้ทำใหม่ก่อนที่จะนำมาใช้จริง ๆ เพราะว่าถ้าประกอบแล้วจะแก้ไขยากกว่าตอนยังไม่ประกอบ
4. ควรเลือกชนิดของยางให้เหมาะสมแต่ยางมีราคาแพงและอาจต้องสั่งพิเศษจากโรงงานเพื่อให้ยึดเกาะพื้นที่ลื่นได้

5.3.2 การแก้ปัญหาปัญหาทางด้านระบบควบคุมระดับล่าง

1. เมื่อออกแบบระบบควบคุมระดับล่างเรียบร้อย การสั่งซื้อของจากต่างจังหวัดควรที่จะสั่งสินค้าให้ครบถ้วนในคราวเดียวเพื่อลดปัญหาค่าขนส่ง หรืออาจเดินทางไปซื้อเอง
2. การออกแบบแผงวงจรเองจะทำให้ได้ประสบการแต่อาจมีค่าใช้จ่ายสูงกว่าถ้าสั่งซื้อสำเร็จรูปอาจจะได้อุปกรณ์ที่มีเสถียรภาพมากกว่า
3. ติดเซนเซอร์ตรวจวัดระยะทางเพิ่มเพื่อป้องการผิดพลาดจากการที่ติดเซนเซอร์ตัวเดียว
4. อาจเลือกซื้อเซนเซอร์ที่มีการตรวจวัดระยะทางที่ไกลขึ้น แต่ราคาก็จะสูงมากขึ้น
5. แบตเตอรี่หมดเร็ว จากผลการทดลองจะเห็นได้ว่าเมื่อหุ่นยนต์ทำการทดสอบไปได้ประมาณ 10 -15 รอบ กำลังของหุ่นยนต์จะตกลง อาจแก้โดยใช้แบตเตอรี่ที่มีกระแสสูงขึ้นหรือเปลี่ยนให้มอเตอร์ขับเคลื่อนลดการทิฟโหลดมากขึ้น (มอเตอร์หมุนและหยุดหมุนสลับกันไปเรื่อย ๆ แทนที่สมควรจะหมุนแบบต่อเนื่อง) หรือลดน้ำหนักหุ่นยนต์ลง
6. เมื่อหุ่นยนต์เคลื่อนที่ผ่านทางลาดชัน จะต้องติดอุปกรณ์ตรวจวัดความลาดเอียงเพื่อเป็นอินพุตที่ทำให้กล้องหมุนปรับระดับให้กล้องขนานกับพื้น

5.3.3 การแก้ปัญหาปัญหาทางด้านระบบควบคุมระดับบน

1. เมื่อส่วนควบคุมทั้งสองส่วนทำงานไม่เข้าจังหวะกัน ในขณะที่ระดับล่างทำงานภาพในสถานะเคลื่อนที่จะไม่มีการส่งข้อมูลจากเซนเซอร์มาให้กับระบบควบคุมระดับบน ในขั้นตอนนี้มักจะเกิดปัญหา ขึ้นการแก้ปัญหาคือให้ส่วนควบคุมระดับบนหยุดรอโดยไม่รับค่าใด ๆ ถ้าส่วนควบคุมระดับล่างยังทำงานไม่เสร็จ
2. ความเร็วในการสื่อสารอนุกรมอาจเปลี่ยน Buarate ในการสื่อสารเพื่อให้เกิดความเร็วเพิ่มขึ้นเช่น จาก 9600 bit/sเป็น 19200 bit/s และเปลี่ยนแปลงค่าในรีจิสเตอร์ TH และ TL
3. เลือกกล้องให้เหมาะสมคือ มี Frame Rate ไม่ต่ำเกินไปคือ ที่ประมาณ 30 fps ขึ้นไป แต่ราคาอาจสูงขึ้นมากซึ่งตัวที่ใช้อยู่นี้ประมาณ 600 บาท
4. ถ้า Frame ของภาพที่ Query มาไม่ใช่เฟรมปัจจุบันเมื่อมีการเปลี่ยนสถานะ ดังนั้นเมื่อก่อนมีการเปลี่ยนสถานะใด ๆ ให้โปรแกรมของระบบควบคุมส่วนบนเรียกการ Query Frame ก่อนที่จะทำการประมวลผล ภาพเฟรมที่ได้จะเป็นเฟรมปัจจุบัน

5.4 แนวทางในการพัฒนาเพิ่มเติม

1. พัฒนาให้หุ่นยนต์เคลื่อนที่ได้ในทุกสภาพพื้นผิว
2. การติดตั้ง Encoder และ เข็มทิศอิเล็กทรอนิกส์ เพื่อแสดงเส้นทางและระบุตำแหน่งผู้ประสบภัย
3. การติดตั้งอุปกรณ์ตรวจจับสภาพแวดล้อมในบริเวณรอบตัวของผู้ประสบภัย เช่น ติดตั้งอุปกรณ์ตรวจวัดก๊าซคาร์บอนไดออกไซด์ อุปกรณ์ตรวจวัดอุณหภูมิ อุปกรณ์ตรวจจับความเคลื่อนไหวของผู้ประสบภัย อุปกรณ์ตรวจรับเสียงของผู้ประสบภัย เป็นต้น
4. การพัฒนาส่วนควบคุมติดต่อกับผู้ใช้ (GUI) เพื่อแสดงผลการทำงานกับผู้ปฏิบัติหน้าที่และให้ข้อมูลที่กล่าวมาในข้อที่ 2 ได้ เช่น ปริ้นตำแหน่งของผู้ประสบภัยได้
5. การใช้ ระบบ Artifact Intelligent ในการทำ Image processing ด้วย OpenCV[2] ในการเขียนโปรแกรมส่วนควบคุมระดับบนด้วยภาษา Visual Studio 2008 C++ เพื่อช่วยในการวิเคราะห์หาป้ายเตือนในสภาพแวดล้อมที่เสี่ยงต่ออันตราย เช่นการเคลื่อนที่เข้าบริเวณที่มีสารเคมีแพร่กระจาย หรือมีวัตถุระเบิด เป็นต้น ซึ่งจะมีป้ายเตือนเหล่านี้ในการแข่งขัน ซึ่งระบบ AI นี้ เช่น Neural Network , Neigh Bay การทำ Image Matching เพื่อระบุป้ายเตือนเพื่อบอกหน่วยกู้ภัยก่อนเข้าช่วยเหลือผู้ประสบภัย

6. การพัฒนาระบบควบคุมส่วนล่าง (MCU) เป็นประเภทอื่นหรือตระกูลอื่นที่มีความเร็วมากขึ้น เช่น ARM7 ,AVR, PIC
7. ระบบควบคุมป้อนกลับควบคุมการหมุนของล้อให้เท่ากัน
8. การตรวจสอบแบตเตอรี่และการสำรองแบตเตอรี่ เพื่อป้องกันไม่ให้หุ่นยนต์ติดอยู่ในภารกิจและสามารถเคลื่อนที่ออกจากภารกิจได้เมื่อแบตเตอรี่ใกล้หมด

เอกสารอ้างอิง

[1] Dr.Surachet Kanprachar and Settha Tangkawanit : **Performance of RGB and HSV Color Systems in Object Detection.** Naresuan University.2006

[2] Dr.Gary Rost Bradski and Dr.Adrian Keahler. : **Learning OpenCV Computer Vision with the OpenCV Library.**First Edition. O'Reilly Media .September 2008

[3] นิรุธ อำนวยศิลป์. **เขียนโปรแกรมบนวินโดวส์ด้วย Visual C++ และ MFC.** พิมพ์ครั้งที่ 1 กรุงเทพมหานคร. เคทีพี คอมพ์ แอนด์ คอนซัลท์. พ.ศ. 2548

[4] รศ.ธีรวัฒน์ ประกอบผล: **การประยุกต์ใช้งานไมโครคอนโทรลเลอร์.** พิมพ์ครั้งที่ 8. กรุงเทพ ฯ. บริษัท ดวงกมลสมัย จำกัด. พ.ศ. 2547

ภาคผนวก

โปรแกรมส่วนควบคุมระดับล่าง

```
#include <reg51.h>
#include <stdio.h>
#include <intrins.h>

#define  CH0   0x41
#define  CH1   0x42
#define  CH2   0x43
#define  CH3   0x44

// ********** port I2C **********//
sbit    XXSDA                   = P1^0;
sbit    XXSCL                   = P1^1;
sbit    IPSDA                   = P1^2;
sbit    IPSCL                   = P1^3;

unsigned char cmdRS,chk = 1;
unsigned int Tmp = 0;
unsigned char anlg_0,anlg_1,anlg_2,anlg_3;
unsigned int SL,SR,SF,SB;

void dmsec (unsigned int count) {     // mSec Delay
   unsigned int i;                 // Keil CA51 (x2)
   while (count) {
      i = 230; while (i>0) i--;
      count--;
   }
}
```

```
void fw(unsigned int dlay)
{
        P2 = 0x66;
        dmsec(dlay);
}
void ll(unsigned int dlay)
{
        P2 = 0x60;
        dmsec(dlay);
}
void lr(unsigned int dlay)
{
        P2 = 0x06;
        dmsec(dlay);
}
void shl(unsigned int dlay)
{
        P2 = 0x6A;
        dmsec(dlay);
}
void shr(unsigned int dlay)
{
        P2 = 0xA6;
        dmsec(dlay);
}

void ipdel (void)
{
  _nop_ ();
  _nop_ ();
  _nop_ ();
  _nop_ ();
```

```
    _nop_ ();
    _nop_ ();
    _nop_ ();
    _nop_ ();
}

void xxchigh (void)
{           // I2C clock high
    XXSCL = 1;
    ipdel ();
}

void xxclow (void)
{           // I2C clock low
    XXSCL = 0;
    ipdel ();
}

void xxstart (void)
{           // start condition
    XXSDA = 1;
    XXSCL = 1;
    XXSDA = 0;
    ipdel ();
    XXSCL = 0;
    XXSDA = 1;
}

void xxstop (void)
{           // stop condition
    XXSDA = 0;
    XXSCL = 1;
```

```
    ipdel ();
    XXSDA = 1;
}

void ipchigh (void) {          // I2C clock high
    IPSCL = 1;
    ipdel ();
}

void ipclow (void) {           // I2C clock low
    IPSCL = 0;
    ipdel ();
}

void ipstart (void) {          // start condition
    IPSDA = 1;
    IPSCL = 1;
    IPSDA = 0;
    ipdel ();
    IPSCL = 0;
    IPSDA = 1;
}

void ipstop (void) {           // stop condition
    IPSDA = 0;
    IPSCL = 1;
    ipdel ();
    IPSDA = 1;
}

bit xxwrbyte (unsigned dat)
{                                          // write one byte
```

```
    unsigned char i;                    // return 0 = ok
    bit outbit;                         // return 1 = error
    for (i=1;i<=8;i++)
            {
      outbit = dat & 0x80;
      XXSDA = outbit;
      dat = dat << 1;
      xxchigh ();
      xxclow ();
    }
    xxchigh ();
    outbit = XXSDA;
    xxclow ();
    return (outbit);
}
bit irwrbyte (unsigned dat) {          // write one byte for ds1307
    unsigned char i;                   // return 0 = ok
    bit outbit;                        // return 1 = error
    for (i=1;i<=8;i++) {
      outbit = dat & 0x80;
      IPSDA = outbit;
      dat = dat << 1;
      ipchigh ();
      ipclow ();
    }

    ipchigh ();
    ipclow ();
          outbit = IPSDA;
    return (outbit);
}
unsigned char xxrdbyte ()
```

```
{                                                    // read one byte
   unsigned char i,dat;              // return 0xff = error
   bit inbit;
   dat = 0;
   for (i=1;i<=8;i++)
         {
      xxchigh ();
      inbit = XXSDA;
      dat = dat << 1;
      dat = dat | inbit;
      xxclow ();
   }

   xxchigh ();
   inbit = XXSDA;
   xxclow ();
   if (~inbit) dat = 0xff;
   return (dat);
}
unsigned char irrdbytex () {          // read one byte for DS1307
   unsigned char i,dat;
   bit inbit;
   dat = 0;
   for (i=1;i<=8;i++) {
      ipchigh ();
      inbit = IPSDA;
      dat = dat << 1;
      dat = dat | inbit;
      ipclow ();
   }

   ipchigh ();
```

```
    ipclow ();
    return (dat);
}
unsigned char ADC(unsigned char channel)
{
        unsigned char temp;
        xxstart();
        while(xxwrbyte(0x90));
        while(xxwrbyte(0x40|channel));
        xxstop();

        xxstart();
        while(xxwrbyte(0x91));
        temp = xxrdbyte();
        xxstop();

        return(temp);
}

void getTmp()
{
        ipstart();
        irwrbyte(0xd0);
        irwrbyte(0x03);
        ipstop();
        ipstart();
        irwrbyte(0xd1);
        Tmp =  irrdbytex();
        ipstop();
}

void start232(void)
```

```
{
        SCON = 0x52;
   TMOD = 0x20;

        TH1 = 0xfd; PCON = 0x80;
        TR1 = 1;
        RI =.0;
}

void startinterupt(void)
{
        EA = 1;
        ES = 1;
}

void serial_ISR_Job(void)
{
        if(cmdRS=='a')
        {
        lr(200);
                fw(250);
                ll(25);
        }
  else if(cmdRS=='b')
        {
        ll(150);
                fw(250);
                lr(50);
        }
        else if(cmdRS=='c')
        {
        fw(250);
```

```
        lr(25);
}
else if(cmdRS=='d')
{
        shr(1200);
}
else if(cmdRS=='e')
{
        shl(500);
        fw(600);
        shl(500);
        fw(800);
}
else if(cmdRS=='f')
{
        ll(350);
fw(250);
}
else if(cmdRS=='g')
{
ll(250);
fw(250);
}
else if(cmdRS=='h')
{
lr(250);
fw(250);
}
else if(cmdRS=='i')
{
lr(350);
fw(250);
```

```
        }
        else if(cmdRS=='t')
        {
                Tmp = 1;
        }
        else
        {
                P2 = 0x00;
        }
}


void serial_ISR(void) interrupt 4
{
        if(RI)
        {
                ES = 0;
                cmdRS = SBUF;

                serial_ISR_Job();

                chk = 0;
                RI = 0;
                ES = 1;
        }
}

void main(void)
{
        start232();
```

```
startinterupt();
while(chk)
while(1)
{
        P2 = 0x00;

        anlg_0 = ADC(CH0);
        anlg_1 = ADC(CH1);
        anlg_2 = ADC(CH2);
        anlg_3 = ADC(CH3);

        SL = (1707.0/(anlg_1 - 3.0))-4.0;
        SR = (1707.0/(anlg_2 - 3.0))-4.0;
        SF = (1707.0/(anlg_3 - 3.0))-4.0;
        if(SL<10||SL>80||SR<10||SR>80||SF<10||SF>80)
        {
                if(SL<10)
                        SL = 10;
                if(SR<10)
                        SR = 10;
                if(SF<10)
                        SF = 10;
                if(SL>80)
                        SL = 80;
                if(SR>80)
                        SR = 80;
                if(SF>80)
                        SF = 80;
        }
        //printf("Left Sensor : %d\tRight Sensor : %d\tFront Sensor : %d\r",SL,SR,SF);
        printf("%d%d%d",SL,SR,SF);
        while(chk)
```

```
                if(Tmp){
                        //getTmp();
                        //printf("%d",Tmp);
                        P2 = 0x00;
                        dmsec(100);
                        Tmp = 0;
                }
                chk = 1;
        }
}
```

โปรแกรมส่วนควบคุมระดับบน

Function การเชื่อมต่อระบบการสื่อสารอนุกรม

```
#include "stdafx.h"
#include "String.h"
#include "TestMFC_RS232.h"

HANDLE hCompDrive;
DCB dcbDrive;
char *chCommPortDrive="COM5";
int valReturnDrive,ss = 1;
char chBuffDrive;
char check,cmd;
unsigned long nReadPortDrive;
unsigned long nWritePortDrive;



//-------------------------------------------------------------
//@return                initial serial wireless
```

```
//@description   initial serial wireless
//@contract              init_Serial: void -> void
//@example               init_Serial()== ""
//@example               init_Serial()== ""
//@example               init_Serial()== ""
void init_SerialDrive()
{

        hCompDrive = CreateFile( (CString)chCommPortDrive,
                        GENERIC_READ | GENERIC_WRITE,
                        0,
                        NULL,
                        OPEN_EXISTING, //
                        0,
                        NULL
                        );

 valReturnDrive = GetCommState(chCommPortDrive, &dcbDrive);

 dcbDrive.BaudRate = CBR_19200;
 dcbDrive.ByteSize = 8;
 dcbDrive.Parity = NOPARITY;
 dcbDrive.StopBits = ONESTOPBIT;

 valReturnDrive = SetCommState(hCompDrive, &dcbDrive);

}

void sendRS232CDrive(LPCVOID str_,int sizeByte)
{
        WriteFile(hCompDrive,LPCVOID(str_),sizeByte,&nWritePortDrive, NULL);
}
```

```
void sendRS232CReadTemp(LPVOID str)
{
        ReadFile(hCompDrive,str,1,&nReadPortDrive, NULL);
}

void sendRS232CReads(LPVOID str)
{
        ReadFile(hCompDrive,str,2,&nReadPortDrive, NULL);
}
```

Function การเคลื่อนที่ในสถานะ S2

```
char Move(int SL,int SR,int SF)
{
        if(SL > 70)
                cmd = 'e';          // มีทางเลี้ยวซ้าย
        else if(SF > 30)
        {
                if(SL < 30)
                        cmd = 'a';  // ชิดซ้ายเกิน
                else if(SL > 45 && SL <= 60)
                        cmd = 'b';  // ห่างซ้ายเกิน
                else
                        cmd = 'c';  // อยู่ในช่วง
        }else if(SF <= 30 && SL < 50)
                cmd = 'd';          // หน้า + ซ้าย ตัน

        else
                cmd = 'x';
```

```
        return cmd;
}
```

Function การเคลื่อนที่ในสถานะ S3

```
char Focus(int center,int cenObj,int SF)
{
        if(SF>45)
        {
                if( cenObj>=(5*center/3) )
                        cmd = 'i';
                else if( cenObj<=(center/3) )
                        cmd = 'f';
                else if( cenObj>(center/3) && cenObj<(2*center/3) )
                        cmd = 'g';
                else if( cenObj>(4*center/3) && cenObj<(5*center/3) )
                        cmd = 'h';
                else if( cenObj>=(2*center/3) && cenObj<=(4*center/3) )
                        cmd = 'c';
        }
        else
                cmd = 't';
        return cmd;
}
```

Function การประวลผลภาพค้นหาเป้าหมาย

```
// TestMFCDlg.cpp : implementation file
//

#include "stdafx.h"
```

```
#include "TestMFC.h"
#include "TestMFCDlg.h"
#include "TestMFC_RS232.h"
#include "string.h"
#include "stdio.h"
#include <fstream>
#include <iostream>
#include <cstring>
#include "BlobResult.h"
using namespace std;

#ifdef _DEBUG
#define new DEBUG_NEW
#endif

// ------- declaration variables -----------
char str[4],cmds[1];
IplImage *Org, *H, *S, *V, *find;
int cenX,cenY,i;
CBlobResult blobs;
CBlob *currentBlob;
int SL,SR,SF;

// CTestMFCDlg dialog
CTestMFCDlg::CTestMFCDlg(CWnd* pParent /*=NULL*/)
	: CDialog(CTestMFCDlg::IDD, pParent)
	, m_comport(_T(""))
{
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

void CTestMFCDlg::DoDataExchange(CDataExchange* pDX)
```

```
{
        CDialog::DoDataExchange(pDX);
        DDX_Text(pDX, IDC_Comport, m_comport);
}

BEGIN_MESSAGE_MAP(CTestMFCDlg, CDialog)
        ON_WM_PAINT()
        ON_WM_QUERYDRAGICON()
        //}}AFX_MSG_MAP
        ON_BN_CLICKED(IDOK, &CTestMFCDlg::OnBnClickedOk)
        ON_BN_CLICKED(IDC_BUTTONSTART,
&CTestMFCDlg::OnBnClickedButtonstart)
        ON_BN_CLICKED(IDC_INITRS232, &CTestMFCDlg::OnBnClickedInitrs232)
        ON_BN_CLICKED(IDC_BUTTON4, &CTestMFCDlg::OnBnClickedButton4)
END_MESSAGE_MAP()


// CTestMFCDlg message handlers

BOOL CTestMFCDlg::OnInitDialog()
{
        CDialog::OnInitDialog();

        // Set the icon for this dialog.  The framework does this automatically
        //  when the application's main window is not a dialog
        SetIcon(m_hIcon, TRUE);                 // Set big icon
        SetIcon(m_hIcon, FALSE);                // Set small icon

        // TODO: Add extra initialization here
  strWindowsImageName = "Show Camera";
        capture = 0;
        strFlg = false;
```

```
	// Initial serial port connect
	if(!serialFlg)
		init_SerialDrive();
		cmds[0] = 's';
		sendRS232CDrive(LPCVOID(cmds),1);

	return TRUE;  // return TRUE  unless you set the focus to a control
}

// If you add a minimize button to your dialog, you will need the code below
// to draw the icon.  For MFC applications using the document/view model,
// this is automatically done for you by the framework.

void CTestMFCDlg::OnPaint()
{
	if (IsIconic())
	{
		CPaintDC dc(this); // device context for painting

		SendMessage(WM_ICONERASEBKGND,
reinterpret_cast<WPARAM>(dc.GetSafeHdc()), 0);

		// Center icon in client rectangle
		int cxIcon = GetSystemMetrics(SM_CXICON);
		int cyIcon = GetSystemMetrics(SM_CYICON);
		CRect rect;
		GetClientRect(&rect);
		int x = (rect.Width() - cxIcon + 1) / 2;
		int y = (rect.Height() - cyIcon + 1) / 2;

		// Draw the icon
```

```
                dc.DrawIcon(x, y, m_hIcon);
        }
        else
        {
                CDialog::OnPaint();
        }
}

// The system calls this function to obtain the cursor to display while the user drags
// the minimized window.
HCURSOR CTestMFCDlg::OnQueryDragIcon()
{
        return static_cast<HCURSOR>(m_hIcon);
}

void CTestMFCDlg::OnBnClickedOk()
{
        UpdateData(1);
        MessageBox(m_comport);
}

void CTestMFCDlg::OnBnClickedButtonstart()
{
        strFlg = true;

        if( capture==0 )
  {
                capture = cvCreateCameraCapture(-1);
        }

  if( !capture )
  {
```

```
  this->MessageBoxW((CString)("Can not connect to the camera"),MB_OK);
  return;
}
int c;
      IplImage* frame = 0;

      while(strFlg)
{
            if(cmds[0] == 'e')
            {
                  for(int ii = 1;ii<=30000;ii++)
                        for(int jj = 1;jj<=30000;jj++);
            }
            // ------- Get value from sensors ------------------

            sendRS232CReads(LPVOID(str));
            this->SetDlgItemTextW(IDC_EDIT_READ, CString(str));
            SL = atoi(str);
            sendRS232CReads(LPVOID(str));
            this->SetDlgItemTextW(IDC_show, CString(str));
            SR = atoi(str);
            sendRS232CReads(LPVOID(str));
            this->SetDlgItemTextW(IDC_EDIT2, CString(str));
            SF = atoi(str);

  frame = cvQueryFrame( capture );
  if( !frame )
    break;
  if( !image )
  {
                  CvSize imgSize = cvGetSize(frame);
```

```
                image = cvCreateImage(imgSize,IPL_DEPTH_8U,3);      //create
destination variable
                cvResize(frame,image,1);
    }

        // ---------- Initial Create Image ----------
        if(!S)
        {
            Org = cvCreateImage( cvGetSize(frame), 8, 3);
            find = cvCreateImage( cvGetSize(frame), 8, 3);
            cvSet( find, CV_RGB(202, 85, 95), 0);
            H = cvCreateImage( cvGetSize(frame), 8, 1);
            S = cvCreateImage( cvGetSize(frame), 8, 1);
            V = cvCreateImage( cvGetSize(frame), 8, 1);
        }

        // ------- Converse Image ------
        cvCvtColor( frame, Org, CV_BGR2HSV );
        cvCvtColor( find, find, CV_BGR2HSV );

        // ------- Different Image ------
        cvAbsDiff( Org, find, find);

        // ------- Plane Image, Filter and Threshold ------
        cvSplit( find, H, S, V, 0);
        cvThreshold( S, S, 30, 255, CV_THRESH_BINARY_INV);
        cvThreshold( H, H, 10, 255, CV_THRESH_BINARY_INV);
        cvSmooth( S, S, CV_MEDIAN, 9, 0, 0, 0);
        cvSmooth( H, H, CV_MEDIAN, 9, 0, 0, 0);

        // ------- Blobs to cover interested object --------
        blobs = CBlobResult(S, NULL, 0 );
```

```
                blobs.Filter( blobs, B_EXCLUDE, CBlobGetArea(), B_LESS, 4000);



                if(blobs.GetNumBlobs()> 0)
                {
                        currentBlob = blobs.GetBlob(0);
                        cenY = (int)(((currentBlob->MaxY()-currentBlob-
>MinY())/2)+currentBlob->MinY());
                        cenX = (int)(((currentBlob->MaxX()-currentBlob-
>MinX())/2)+currentBlob->MinX());
                        cvCircle(S,cvPoint(cenX,cenY),(int)(currentBlob->MaxX()-
currentBlob->MinX())/2,CV_RGB(255,0,0),0,8,0);

                        serialFlg = true;
                        strcpy(str,"");
                        cmds[0] = Focus(frame->width/2,cenX,SF);
                        this->SetDlgItemTextW(IDC_EDIT2, CString(cmds));
                        sendRS232CDrive(LPCVOID(cmds),1);
                        if(cmds[0]=='z')
                        {
                                sendRS232CReadTemp(LPVOID(cmds));
                                this->SetDlgItemTextW(IDC_show, CString(str));
                                // ---- makes target disappear ----
                                /******



****/
                        }
                }
                else
                {
                        serialFlg = true;
```

```
                    strcpy(str,"");

                    cmds[0] = Move(SL,SR,SF);
                    sendRS232CDrive(LPCVOID(cmds),1);
                }

                cvFlip(S, S, 0);
                cvFlip(H, H, 0);
                //cvFlip(frame, frame, 0);
                cvNamedWindow( strWindowsImageName, CV_WINDOW_AUTOSIZE );
                cvNamedWindow( "Image S-Plane",  1 );
                cvShowImage( "Image S-Plane", S );
                cvNamedWindow( "Image H-Plane",  1 );
                cvShowImage( "Image H-Plane", H );
                cvShowImage( strWindowsImageName,frame );

        //ShowImageIpl(this,IDC_STATIC_IMAGEORG, frame ,24);
                //cvReleaseImage(&S);
                //cvReleaseImage(&image);

                c=cvWaitKey(5);
                if(cvGetWindowHandle(strWindowsImageName)==NULL)
                    break;
            }
    }
```