การควบคุมหุ่นยนต์หลายตัว

Group Behavior Control for Team Robot

โดย

นายกฤษณะ พรหมศรี

นายฉัตรชัย สุขศรีเมือง

ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมระบบควบคุม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2546

ปริญญานิพนธ์ปีการศึกษา 2546

ภาควิชาวิศวกรรมระบบควบคุม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การควบคุมหุ่นยนต์หลายตัว (Group Behavior Control for Team Robot)

ผู้จัดทำ

1. นาย กฤษณะ พรหมศรี รหัส 44015273

2. นาย ฉัตรชัย สุขศรีเมือง รหัส 44015278

..... อาจารย์ที่ปรึกษา

(รศ.ดร.จงกล งามวิวิทย์)

การควบคุมหุ่นยนต์หลายตัว

(Group Behavior Control for Team Robot)

นาย ฉัตรชัย สุขศรีเมือง
นาย กฤษณะ พรหมศรี
รศ.ดร. จงกล งามวิวิทย์
ปีการศึกษา 2546

บทคัดย่อ

ปัจจุบันมีการนำหุ่นยนต์มาใช้ในอุตสาหกรรมต่างๆมากมาย หุ่นยนต์แต่ละตัวถูกออกแบบมาเพื่อให้เหมาะสมกับงานแต่ละอย่าง ทำให้ต้นทุนการผลิตสูงขึ้น ดังนั้นเพื่อเป็นการลดต้นทุนการผลิต โครงงานนี้จึงนำเสนอการนำหุ่นยนต์หลายตัวมาทำงานร่วมกันเพื่อให้สามารถทำงานหลายอย่างได้โดยไม่ต้องมีการออกแบบหุ่นยนต์ใหม่ทั้งหมด และเพื่อให้หุ่นยนต์หลายตัวสามารถทำงานแทนหุ่นยนต์ตัวใหญ่ตัวเดียวได้ ข้อดีคือการประหยัดค่าใช้จ่ายในการบำรุงรักษา และราคาของหุ่นยนต์ขนาดใหญ่ที่มีราคาแพง การควบคุมการเคลื่อนที่ของหุ่นยนต์หลายตัวแบบเป็นกลุ่มจะใช้การควบคุมผ่านภาพที่รับมากล้องวีดีโอ โดยอาศัยการประมวลผลด้วยภาพ เพื่อคำนวณพิกัดของหุ่นยนต์แล้วใช้โปรแกรมสร้างรูปแบบ ในการควบคุมการเคลื่อนที่ของหุ่นยนต์หลายตัวในรูปแบบต่างๆ

Abstract

Nowadays, many factories have been adopting the robots to use in production line. Each robot is designed for each work so the cost of producing is higher than before. This project is implemented to reduce the cost of producing by using many robots to work together so it is no need to build the new robot for each new work and also replace the big robot by the smaller robots in the same work. The advantage is not only the cost reduction in production line but also the cost reduction of maintenance.

The motion control of team robot uses image processing. The image that is sent from video camera to the computer will be processed by software for evaluating and calculating the position of each robot. After that the software will create the pattern for controlling the movement of each robot.

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญภาพ

สารบัญภาพ (ต่อ)

สารบัญภาพ (ต่อ)

สารบัญภาพ (ต่อ)

สารบัญตาราง

บทที่ 1
บทนำ

1.1 ความเป็นมาของโครงงาน

ปัจจุบันเทคโนโลยีทางด้านอิเล็กทรอนิกส์มีความก้าวหน้าอย่างรวดเร็ว รวมถึงการพัฒนาทางด้านโปรแกรมต่างๆ ซึ่งปัจจุบันมีการประยุกต์ใช้กล้องวีดีโอในหลายๆด้านกับงานทางด้านวิศวกรรม เมื่อนำสัญญาณภาพที่ได้จากกล้องวีดีโอมาใช้งานร่วมกับโปรแกรม ทำให้เราสามารถวิเคราะห์สัญญาณภาพแล้วนำไปใช้กับงานควบคุมได้ในหลายรูปแบบ

คณะผู้จัดทำ จึงคิดทำโครงงานการควบคุมการเคลื่อนที่ของหุ่นยนต์หลายตัวแบบเป็นกลุ่มหรือกล่าวอีกนัยหนึ่งคือการควบคุมพฤติกรรมการเคลื่อนที่ของหุ่นยนต์ โดยใช้การควบคุมหุ่นยนต์หลายตัวผ่านภาพที่รับมากล้องวีดีโอ โดยอาศัยการประมวลผลด้วยภาพ (Image Processing) เพื่อคำนวณพิกัดของหุ่นยนต์แล้วใช้โปรแกรมสร้างรูปแบบ (Pattern) ในการควบคุมการเคลื่อนที่ของหุ่นยนต์หลายตัวในรูปแบบต่างๆ

1.2 วัตถุประสงค์ในการดำเนินงาน

1. เพื่อศึกษาการประมวลผลภาพ และนำไปประยุกต์ใช้ในการควบคุมระบบต่าง ๆ
2. เพื่อศึกษาและออกแบบหุ่นยนต์ขนาดเล็กโดยใช้อุปกรณ์อิเล็กทรอนิกส์ขนาดเล็ก(SMD)เพื่อใช้หุ่นยนต์กับงานที่มีพื้นที่ขนาดเล็ก
3. เพื่อศึกษาการเขียนโปรแกรม VISUAL BASIC กับการเชื่อมต่อกับอุปกรณ์ภายนอก
4. เพื่อศึกษาและออกแบบโปรแกรมเพื่อสร้างรูปแบบในการควบคุมการเคลื่อนที่ของหุ่นยนต์หลายตัวในรูปแบบต่างๆ
5. เพื่อสร้างต้นแบบเพื่อใช้ในการพัฒนาปรับปรุงต่อไป

1.3 ขั้นตอนการศึกษาและจัดทำโครงงาน

การศึกษาและจัดทำโครงงานนี้เริ่มต้นจาก การออกแบบและสร้างหุ่นยนต์ขนาดเล็กโดยใช้อุปกรณ์(SMD) จากนั้นทำการเขียนโปรแกรมควบคุมหุ่นยนต์โดยการควบคุมจากคอมพิวเตอร์และโปรแกรมประมวลภาพโดยสร้างรูปแบบในการควบคุมการเคลื่อนที่ของหุ่นยนต์หลายตัวในรูปแบบต่างๆ ผ่านกล้องวีดีโอ

โครงงานชิ้นนี้จะต้องอาศัยความรู้เกี่ยวกับการประมวลภาพ สำหรับใช้วิเคราะห์ภาพที่ได้จากกล้องวีดีโอ เพื่อให้คอมพิวเตอร์หรือตัวควบคุม รู้ว่าพิกัดของหุ่นแต่ละตัวอยู่ตำแหน่งใดและทิศทางด้าน

หน้าทำมุมอยู่กี่องศา และยังต้องมีความรู้ในการติดต่อกับหุ่นยนต์ผ่านคอมพิวเตอร์และการแยกรหัสเพื่อแยกข้อมูลให้หุ่นยนต์แต่ละตัวโดยใช้คลื่นแสงอินฟราเรดเป็นตัวเชื่อมต่อข้อมูลระหว่างหุ่นยนต์แต่ละตัวกับคอมพิวเตอร์

ในโครงงานชิ้นนี้ได้ทำการเขียนโปรแกรมรับภาพมาจากกล้องโดยตรง ทำให้เราสามารถคำนวณได้อย่างรวดเร็ว โดยไม่ต้องเก็บภาพและดึงมาประมวลผลทำให้ผลตอบสนองที่ช้ากว่า สุดท้ายทำการควบคุม โดยใช้ตัวควบคุมแบบ ON-OFF ตามเงื่อนไข พิกัด มุม ระยะทาง จากการประมวลผลภาพ

1.4 เนื้อหาที่จะกล่าวในปริญญานิพนธ์

เนื้อหาที่จะกล่าวในปริญญานิพนธ์ฉบับนี้คือ ในบทที่ 2 จะแสดงหลักการและทฤษฎีที่เกี่ยวข้องในการออกแบบและนำเอาความรู้ไปประยุกต์ใช้ในการจัดทำโครงงานและการประมวลผลภาพ รวมถึงหลักการออกแบบหุ่นยนต์ ในบทที่ 3 จะอธิบายเกี่ยวกับการออกแบบระบบจริงและส่วนประกอบทางด้านฮาร์ดแวร์ของระบบ รวมถึงแนวความคิดในการออกแบบ บทที่ 4 เป็นส่วนของการวิเคราะห์ผลและ การทดลอง ในบทที่ 5 จะเป็นส่วนวิจารณ์และสรุปผลการดำเนินงานและปัญหาที่ประสบ และแนวทางการปรับปรุงพัฒนาโครงงานชิ้นนี้ต่อไป

บทที่ 2
ทฤษฎีและความรู้ที่เกี่ยวข้อง

ในการจัดทำโครงงานชิ้นนี้เราจำเป็นจะต้องศึกษาเกี่ยวกับทฤษฎีของการประมวลผลภาพ การเขียนโปรแกรมติดต่อกับอุปกรณ์ภายนอก ความรู้เกี่ยวกับไมโครคอนโทรลเลอร์ การส่งข้อมูลผ่านคลื่นแสงอินฟราเรด และความเข้าใจในการออกแบบวงจรขับมอเตอร์เพื่อที่จะนำไปใช้งานได้อย่างถูกต้องและเหมาะสมในการออกแบบ ส่วนทฤษฎีที่กล่าวถึงมีหัวข้อดังต่อไปนี้

2.1 ทฤษฎีการประมวลภาพ

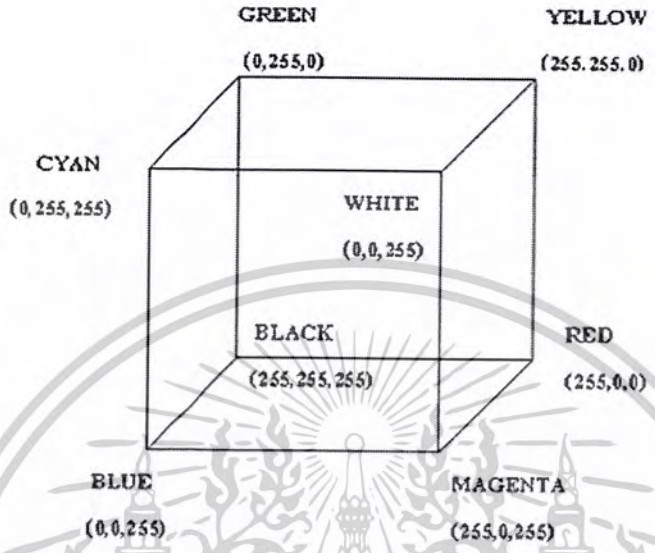
ทฤษฎีในการประมวลภาพนี้จะกล่าวถึงที่มาของสี ว่าสีที่ใช้งานในคอมพิวเตอร์กราฟฟิกมีรูปแบบและการนำไปใช้งานอย่างไร

2.1.1 การกำหนดสี

ในคอมพิวเตอร์สีทุกสีจะประกอบด้วย 3 สีพื้นฐาน คือ แดง เขียว และน้ำเงิน (RGB) ที่สามารถจะออกแบบสีได้เกือบทุกสี โดยการผสมสัดส่วนของสีพื้นฐาน การออกแบบสีตั้งอยู่บนพื้นฐานของความเข้มข้นขององค์ประกอบของ RGB ซึ่งเรียกว่า RGB โมเดล และที่กล่าวมาข้างต้นก็เป็นแนวความคิดพื้นฐานของ คอมพิวเตอร์ กราฟฟิก

ทุก ๆ สีซึ่งสามารถจินตนาการได้สามารถสร้างขึ้นมาได้ โดยการผสมสัดส่วนที่เหมาะสมของสามสีพื้นฐาน ดังนั้นแต่ละสีถูกนำเสนอโดยสัดส่วน (Red,Green,Blue) ซึ่งสีพื้นฐานทั้งสามในแต่ละสีถูกนำเสนอด้วยทั้งสามไบต์ค่าที่เล็กที่สุด 0 ชี้ให้เป็นถึงการไม่มีสีเลย ค่าที่ใหญ่ที่สุด 255 บอกถึงความเข้มข้นสูงสุด สัดส่วน (0,0,0) จะเป็นสีดำ เพราะสีทุกสีไม่มีความเข้มข้นเลย และสัดส่วน(255,255,255) เป็นสีขาว สีอื่นๆ ซึ่งมีการรวมกันได้มากมายเช่น (255,0,0) เป็นสีแดงบริสุทธิ์ (0,255,255) เป็นสีน้ำเงินเข้มและ (0,128,128) เป็นสีน้ำเงินอ่อนๆการรวมกันที่เป็นไปได้ของสีพื้นฐานเป็น 256x256x256 หรือ 16,77,216 สี

กระบวนการของการสร้างสีซึ่งมีส่วนประกอบพื้นฐาน 3 ส่วนประกอบ ตั้งอยู่บนพื้นฐานของ RGB จากกล่องสี่เหลี่ยมสี ดังรูปที่ 2.1



รูปที่ 2.1 กล่องสี RGB

กล่องสีสามมิติจะมีสีที่สอดคล้องกับสามสีพื้นฐานซึ่งมุมของสี่เหลี่ยมแทนด้วยสีต่างๆดังรูป สีพื้นฐานอยู่ที่แกน X, Y และ Z ตามลำดับ สีที่เป็นส่วนกลับ (Complementary Colors) สามารถคำนวณได้อย่างง่ายๆ โดยการลบค่าของสีจาก 255 ยกตัวอย่าง สี (0,0,255) เป็นสีน้ำเงินบริสุทธิ์ ดังนั้นสีน้ำเงินและเหลืองเป็นสีในส่วนกลับกัน และทั้งสองอยู่ที่มุมตรงกันข้ามสี่เหลี่ยมสี ส่วนมุมอื่น ๆ ก็เช่นเดียวกัน

องค์ประกอบของสีที่มุมของสี่เหลี่ยมสีจะมีความเข้มข้นสูงที่สุดหรือ ไม่มีความเข้มข้นเลยเพียงเท่านั้น เมื่อเคลื่อนที่จากมุมหนึ่งไปยังอีกมุมหนึ่งตามขอบที่เหมือนกันจะมีเพียงแค่องค์ประกอบเดียวเท่านั้นที่เปลี่ยนแปลง ยกตัวอย่างเช่น เมื่อเราทำการเคลื่อนที่จากมุมสีเขียว ไปยังมุมสีเหลือง องค์ประกอบของสีแดงจะเปลี่ยนแปลงจาก 0 ถึง 255 เมื่อเราเคลื่อนที่วัตถุ ออกจากมุมของสีเหล่านี้เราจะได้รับทุกโทนสี จากเขียวไปเหลือง

2.1.2 องค์ประกอบของสี

ในการประมวลภาพจะต้องการทราบองค์ประกอบของสีเหล่านี้ นั่นคือต้องอ่านค่าพิกเซลแยกองค์ประกอบของ RGBจากนั้นนำไปใช้งานแยกกัน ดังนั้นจึงจำเป็นจะต้องเขียนฟังก์ชั่น ซึ่งทำหน้าที่แยกองค์ประกอบของสี เพื่อที่จะทำเช่นนี้ เราจำเป็นที่จะต้องพิจรณาว่าค่าองค์ประกอบของสีทั้งสามถูกเก็บไว้อย่างไร ในตัวแปร LONG INTERGER ซึ่งตัวแปร ดังกล่าว ประกอบด้วย 4 ไบต์ โดยไบต์

แรก (THE MOST SIGNIFICANT BYTE) เก็บ 0 ต่อมาเป็นองค์ประกอบของสีน้ำเงิน เขียว และแดง ตามลำดับค่า LONG INTERGER ซึ่งสอดคล้องกับสัดส่วน (64, 32, 192) เป็น12591168 จำนวนนี้ไม่มีความเหมือนกับจำนวนเดิมเลย ถ้าเรานำเสนอองค์ประกอบของสีด้วยเลขฐานสิบหก แล้ว RGB สัดส่วนเป็น (40, 20, C0) ถ้าเราทำการวางจำนวนนี้สลับซึ่งกันและกัน (C0, 20, 40) ผลจะเป็นเลขฐานสิบหกของ LONG INTERGER สองหลักสุดท้ายสอดคล้องกับองค์ประกอบสีแดง สองหลักถัดมาสอดคล้องกับองค์ประกอบสีเขียว และหลักที่มีความสำคัญสูงสุดสอดคล้องกับค่าสีน้ำเงิน และนี่ก็เป็นเหตุผลว่าทำไมเลขฐานสิบหก จึงถูกใช้มากในการกำหนดค่าสี

STATEMENT ด้านล่างเป็นการแยกค่าขององค์ประกอบของสีจากค่าของสีที่ถูกเก็บในตัวแปร PIXEL

```
PIXEL  = FORM PICTURE 1 POINT( j.i)
RED    = PIXEL MOD 256
GREEN = ( ( PIXEL and &HFF00FF)/256)
BLUE   = (Pixel and &HFF0000/65535)
```

ตัวแปร i และ j เป็น Coordinate ของจุดซึ่งเราทำการตรวจสอบว่าค่าเนื่องจาก ค่าของสีแดงเก็บไว้ในไบต์สุดท้าย มันจึงเป็นเศษของการหารด้วย 256

เพื่อที่จะแยกไบต์ถัดไป เราจำเป็นจะต้องทำให้ไบต์ด้านหน้าและไบต์ด้านหลัง ของมันเป็นศูนย์ ซึ่งทำได้โดยการใช้ตัวดำเนินการ AND กับเลขฐานสิบหก FF00FF00 ค่าของสีเขียวถูกเก็บไว้ในไบต์ที่สอง ดังนั้นเพื่อที่จะลดให้มันอยู่ที่ ไบต์ที่หนึ่งนำค่าที่ได้หารด้วย 256

สุดท้ายเราทำเช่นเดียวกับไบต์ที่ 3 โดยดำเนินการ AND ค่าสีด้วย FF0000 และหลังจากนั้นหารผลที่ได้โดย 65536 ซึ่งจะเท่ากับค่า (256x256) หรืออาจใช้อีกโปรแกรมหนึ่งซึ่งทำงานได้เช่นเดียวกัน

```
Function color RGB ( color1  aslong , R,G,B  as interger )
R= color1  mod256
B= color 1\65536
B=(color1\256)mod256
End  function
```

2.1.3 การควบคุมกราฟฟิก (Manipulating Graphics)

ส่วนนี้อธิบายถึงเทคนิคบางอย่างสำหรับการควบคุม การแสดงภาพโดยอาศัย API Graphics function ของ Visual Basic

• Bitblt ( ) function

ฟังก์ชั่น Bitblt ย่อมาจากคำว่า Bit Block Transfer อันหมายถึงการถ่ายเทข้อมูลระหว่างหน่วยความจำหนึ่ง ณ ตำแหน่งหนึ่งไปยังหน่วยความจำตำแหน่งอื่นๆ ฟังก์ชั่นนี้ถูกใช้เพื่อที่จะคัดลอกภาพจากต้นกำเนิด (Source) ไปยังจุดหมายปลายทาง (Dentination) ฟังก์ชั่นจะทำการส่งผ่าน (transfer) พิกเซลจากต้นกำเนิดไปยังอุปกรณ์ปลายทาง เราสามารถประกาศ (declare) BitBlt () function เป็นดังนี้

Declare Function BitBlt Lib " gdi32 "Alias "Bitblt"
Byval hDestDC AS Long, Byval x as Long ,Byval y As Long , Byval n Width As Long , Byval n Hight as long ,Byval h scrcdc AsLong ,Byval x src as Long Byval ysrc As Long ,Byval dwrop as Long) AsLong

ในการประกาศฟังก์ชั่น BitBlt()ซึ่งเป็นฟังก์ชั่นจากไดนามิกลิงก์ไลบรารีชื่อ gdi32.dll นั้นค่าอาร์กิวเมนต์ที่ฟังก์ชั่นนี้ต้องการคือ

Byval hDestDC **AS Long**	ค่า hDC ของพื้นที่เป้าหมาย
Byval x **As Long**	ค่า x เพื่อกำหนดจุดมุมบนซ้ายของพื้นที่เป้าหมายที่จะให้ทำการ Blitting
Byval y **As Long**	ค่า y เพื่อกำหนดจุดมุมบนซ้ายของพื้นที่เป้าหมายที่จะให้ทำการ Blitting
Byval n Width **As Long**	ค่าความกว้างของพื้นที่เป้าหมายที่จะให้ทำการ Blitting
Byval n Hight **As Long**	ค่าความสูงของพื้นที่เป้าหมายที่จะให้ทำการ Blitting
Byval h scrcdc **As Long**	ค่า hDC ของพื้นที่อันเป็นต้นฉบับ
Byval x src **As Long**	ค่า x เพื่อกำหนดจุดมุมบนซ้ายของพื้นที่ต้นฉบับที่จะถูกใช้เป็นต้นฉบับ
Byval ysrc **As Long**	ค่า y เพื่อกำหนดจุดมุมบนซ้ายของพื้นที่ต้นฉบับที่จะถูกใช้เป็นต้นฉบับ
Byval dwrop **As Long**	ค่าคงที่ Raster Operation กำหนดรูปแบบการจัดการระดับบิต

x และ y เป็น coordinate และถ้าเราจำเป็นจะต้องให้ ความสูง และความกว้างพร้อมทั้งจุดเริ่มต้นของภาพต้นแบบ

ตัวอย่าง

BitBlt Picture2.hdc, 0, 0, 205, 157, Picture1.hdc, 0, 0, vbSrcCopy

Picture2 เป็นจุดหมายปลายทาง เริ่มต้นที่ x = 0 และ y= 0 โดยมีความกว้าง 205 และความสูง 157 ซึ่ง คัดลอกภาพจาก Picture 1 (ต้นกำเนิด) เริ่มคัดลอกที่ x = 0, y= 0 โดยคัดลอกหมดทั้งภาพ

รูปที่ 2.2 แสดงการใช้ฟังก์ชั่น Bitblt Copy ภาพจากต้นกำเนิด ไปยังจุดหมายปลายทาง

• StretchBlt ( ) function

ฟังก์ชั่น StretchBlt คล้ายกับฟังชั่น Bitblt ใช้ในกรณีที่เราต้องการให้ภาพที่ปรากฏบนพื้นที่เป้าหมายมีขนาดไม่เท่ากับภาพต้นฉบับ โดยการเพิ่มค่าอาร์การ์กิวเมนต์ของภาพต้นฉบับเพิ่มขึ้นอีก 2 ตัว เพื่อทำภาพหดหรือยืดภาพตามแต่สถานการณ์

เราสามารถประกาศ (declare) StretchBlt () function เป็นดังนี้

Public Declare Function StretchBlt Lib "gdi32" (ByVal hdc As Long, ByVal X As Long, ByVal Y As Long, ByVal nWidth As Long, ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal xSrc As Long, ByVal ySrc As Long, ByVal nSrcWidth As Long, ByVal nSrcHeight As Long, ByVal dwRop As Long) As Long

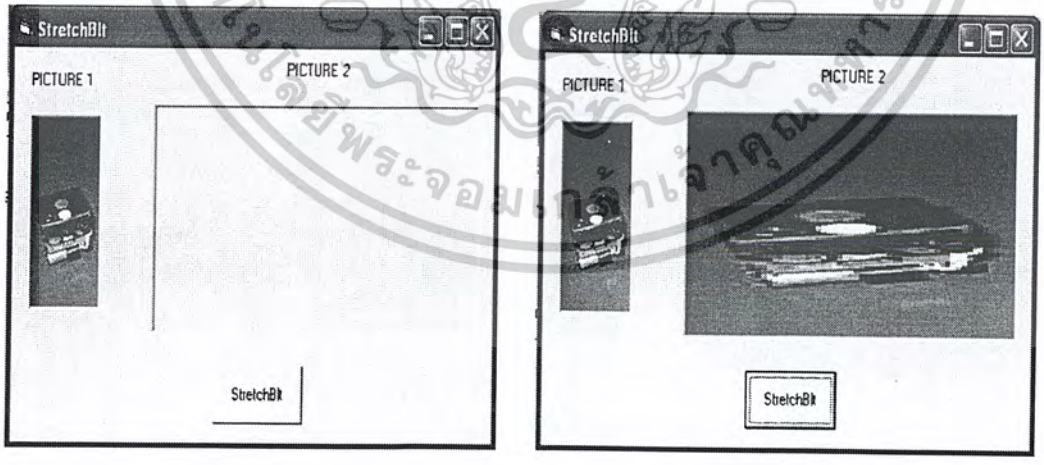
ในการประกาศฟังก์ชั่น StretchBlt () ซึ่งเป็นฟังก์ชั่นจากไดนามิกลิงก์ไลบรารีชื่อ gdi32.dll นั้น ค่าอาร์กิวเมนต์ที่ฟังก์ชั่นเหมือนกับฟังชั่น BitBlt แต่มีค่าอาร์การ์กิวเมนต์ของภาพต้นฉบับเพิ่มขึ้นอีก 2 ตัวคือ

Byval nSrcWidth **As Long** ค่าความกว้างของภาพต้นฉบับที่จะถูกนำไปเป็นเป็นภาพต้นฉบับ

Byval nSrcHight **As Long** ค่าความสูงของภาพต้นฉบับที่จะถูกนำไปเป็นเป็นภาพต้นฉบับ

ตัวอย่าง

StretchBlt Picture2.hdc, 0, 0, Picture2.ScaleWidth, Picture2.ScaleHeight, Picture1.hdc, 0, 0, Picture1.ScaleWidth, Picture1.ScaleHeight , vbSrcCopy



รูปที่ 2.3 แสดงการใช้ฟังก์ชั่น StretchBlt คัดลอกภาพจากต้นกำเนิดไปยังจุดหมายปลายทาง

• GetPixel V() function

ฟังชั่นจะคืนค่าของ pixel ที่ coordinate (x,y) ซึ่งสามารถประกาศได้โดย

Public Declare Function GetPixel Lib " gdi32 " Alias "GetPixel

(Byval hdc As long, Byval x as Long , Byval y asLong) as Long

Argument ตัวแรกของ GetPixel () Function คือ hdc คุณสมบัติของ PictureBox

• Setpixel () Function

ทำหน้าที่กำหนดค่าของแต่ละ pixel หรือ pixel ที่กำหนด สามารถประกาศได้ดังนี้

Public Declane Function setpixel Lib "gdi32" Alias "setpixel"

(Byval hdc As Long ,Byval x as long ,Byval y As long ,byval crcolor as long) as Long

หน้าที่ของแต่ละ Argument คือ

Hdc Picturebox สำหรับการกำหนดค่า

x และ y เป็น coordinate color ค่าของสีที่ ต้องการกำหนดไปยัง pixel

ตัวอย่าง

```
For i= 0 to 320
For j = 0 to 240
Color1 = GetPixel ( Piture1.hdc,I,j)
Setpixel picture2.hdc,I,j,color1
Next i
Next j
```

จะเป็นการ Copy ภาพจาก picture1 ไปยัง picture2 ครั้งละ 1 pixel

2.2 ความรู้เบื้องต้นของพอร์ตขนาน

พอร์ตขนาน (Parallel Port) สาเหตุที่มีชื่อนี้ เนื่องจากการถ่ายทอดข้อมูลของพอร์ตนี้เป็นแบบขนาน สำหรับชื่อเรียกของพอร์ตขนานคือ พอร์ตเครื่องพิมพ์ (Printer Port) เนื่องจากพอร์ตนี้ใช้สำหรับต่อเครื่องพิมพ์นั่นเอง

ด้วยการถ่ายทอดข้อมูลแบบขนานนี้เอง ทำให้พอร์ตขนานมีอัตราการถ่ายทอดข้อมูลสูงกว่าการถ่ายทอดข้อมูลแบบอนุกรมประมาณ 8-10 เท่า และการประมวลผลข้อมูลส่วนใหญ่จะมีขนาด 8 บิต ดังนั้นพอร์ตขนานจึงสามารถรองรับการถ่ายทอดข้อมูล 8 บิตได้โดยไม่ต้องต่อส่วนเพิ่มเติมใด ๆ

2.2.1 ลักษณะทางกายภาพของพอร์ตขนาน

เพื่อให้เข้าใจถึงการนำเอาพอร์ตขนานไปใช้งาน ก่อนอื่นต้องมาทำความเข้าใจก่อนว่า ปกตินั้นการสั่งพิมพ์งานจากคอมพิวเตอร์ไปยังพอร์ตขนานนั้นมีรูปแบบการทำงานภายในอย่างไร ในภาพรูปที่ 2.4 แสดงไดอะแกรมเวลาของติดต่อระหว่างพอร์ตขนานกับเครื่องพิมพ์ ซึ่งจะเห็นได้ว่ามีสัญญาณที่ใช้งานจริง ๆ มีไม่มาก เริ่มจากสัญญาณพอร์ตข้อมูล ถูกส่งออกไปยังเครื่องพิมพ์ พร้อมทั้งส่ง สัญญาณ Strobe ออกไปด้วยเพื่อให้เครื่องพิมพ์รับรู้ว่ามีการส่งข้อมูลใหม่มาที่ขา Data แล้ว จากนั้นคอมพิวเตอร์จะต้องรอการตอบกลับจากเครื่องพิมพ์ นั่นคือเครื่องพิมพ์จะสร้างสัญญาณ Busy หรือเพื่อบอกว่าเครื่องพิมพ์ยังไม่พร้อมที่จะรับข้อมูลใหม่ จนกระทั่งเมื่อเครื่องพิมพ์พร้อม เครื่องพิมพ์จะสร้างสัญญาณ ACK ส่งไปยังคอมพิวเตอร์เพื่อแจ้งว่า พร้อมที่จะรับข้อมูลใหม่แล้ว

รูปที่ 2.4 แสดงไดอาแกรมเวลาของการส่งข้อมูลไปยังเครื่องพิมพ์

สัญญาณข้อมูลขนาด 8 บิต, สัญญาณ Strobe และสัญญาณ ACK (acknowledge) เป็นสัญญาณที่สำคัญในการส่งข้อมูลจากคอมพิวเตอร์ไปยังเครื่องพิมพ์ นอกจากสัญญาณทั้งสามแล้วส่วน

ใหญ่การติดต่อกับเครื่องพิมพ์ยังต้องมีสัญญาณอื่น ๆ ร่วมด้วย เนื่องจากเครื่องพิมพ์ต้องทำหน้าที่ถึง 3 อย่างด้วยกันคือ รับข้อมูลจากคอมพิวเตอร์ พิมพ์ข้อมูลที่รับเข้ามา และตอบสนองต่อการใช้งานของผู้ใช้ เช่น การเปลี่ยนฟอนต์ เป็นต้น บางครั้งอาจเกิดเหตุการณ์ที่ไม่ปกติ เช่น บัฟเฟอร์สำหรับรับข้อมูล (เนื่องจากเครื่องพิมพ์เป็นอุปกรณ์ที่ทำงานทางกลย่อมทำงานได้ช้ากว่าการส่งข้อมูลของคอมพิวเตอร์) เครื่องพิมพ์จะต้องแจ้งไปยังคอมพิวเตอร์ว่าให้หยุดส่งข้อมูลชั่วคราว เนื่องจากสามารถรับข้อมูลมากกว่านี้ได้แล้ว สัญญาณที่ส่งจากเครื่องพิมพ์ไปยังคอมพิวเตอร์คือสัญญาณ Busy และเมื่อเครื่องพิมพ์เกิดข้อผิดพลาด เช่น กระดาษติด เครื่องพิมพ์จะต้องแจ้งไปยังคอมพิวเตอร์เช่นกัน โดยสัญญาณที่แจ้งไปยังคอมพิวเตอร์เรียกว่าสัญญาณ Error นอกจากนี้เมื่อคอมพิวเตอร์ต้องการรีเซตเครื่องพิมพ์ คอมพิวเตอร์จะต้องส่งสัญญาณ Reset ไปยังเครื่องพิมพ์เพื่อรีเซตเครื่องพิมพ์ด้วย สามารถสรุปขาสัญญาณที่จำเป็นสำหรับการติดต่อดังในตารางที่ 2.1

จากตาราง 2.1 จะเห็นได้ว่าพอร์ตขนานของคอมพิวเตอร์ยังแยกย่อยออกเป็นอีก 3 พอร์ต ได้แก่ พอร์ตเอาท์พุตที่ทำหน้าที่ส่งข้อมูลจากคอมพิวเตอร์ไปยังเครื่องพิมพ์ พอร์ตเอาท์พุตสำหรับสัญญาณ Strobe และ Reset พอร์ตอินพุตสำหรับการอ่านค่าสัญญาณ Acknowledge, Busy และ Error จากเครื่องพิมพ์

สัญญาณ	หน้าที่การทำงาน	ทิศทาง
ข้อมูล 8 บิต	ข้อมูลที่ส่งจากคอมพิวเตอร์ไปยังเครื่องพิมพ์	คอมพิวเตอร์
Strobe	แจ้งเครื่องพิมพ์ถึงข้อมูลที่ส่งมาใหม่	คอมพิวเตอร์
Acknowledge	เครื่องพิมพ์แจ้งมายังคอมพิวเตอร์ว่าได้รับข้อมูลแล้ว	เครื่องพิมพ์
Busy	แจ้งสถานะว่าเครื่องพิมพ์ไม่ว่างที่จะรับข้อมูลใหม่	เครื่องพิมพ์
Error	แจ้งสถานะว่าเครื่องพิมพ์เกิดข้อผิดพลาด	เครื่องพิมพ์
Reset	รีเซตเครื่องพิมพ์	คอมพิวเตอร์

ตาราง 2.1 สัญญาณสำคัญ ๆ ของพอร์ตขนานที่ใช้ติดต่อกับเครื่องพิมพ์

โดยปกติพอร์ตขนานออกแบบมาให้มีสายสัญญาณอยู่ทั้งหมด 17 เส้น สายสัญญาณเหล่านั้นจะมีรีจีสเตอร์ 3 ตัวควบคุมการทำงาน ดังนี้

1. พอร์ตเอาท์พุตสำหรับสัญญาณข้อมูล 8 เส้น มีรีจีสเตอร์ Data ควบคุม
2. พอร์ตอินพุตสำหรับการอ่านค่าสถานะต่าง ๆ จากภายนอกมีอยู่ด้วยกัน 5 เส้น ใช้รีจีสเตอร์ Status ในการควบคุม
3. พอร์ตเอาท์พุตสำหรับส่งสัญญาณควบคุมไปยังอุปกรณ์ภายนอก มีอยู่ด้วยกัน 4 เส้น ใช้รีจีสเตอร์ Control ในการควบคุม

บล๊อคไดอะแกรมในภาพประกอบในรูปที่ 2.5 แสดงระบบบัสของคอมพิวเตอร์สำหรับการติดต่อกับพอร์ตขนาน สัญญาณเอาท์พุต จากพอร์ตขนานจะถูกส่งไปยังคอนเน็กเตอร์แบบ DB-25 สำหรับคอมพิวเตอร์ส่วนใหญ่ในปัจจุบันพอ์ตขนานจะมีมาพร้อมกับเมนบอร์ดไม่จำเป็นต้องใช้การ์ดเสียบเพิ่มเติมเหมือนในอดีต พร้อมทั้งมีฟังก์ชันการทำงานที่ซับซ้อนขึ้น แต่ยังคงสนับสนุนการทำงานของพอร์ตขนานในรูปแบบมาตราฐาน (Standard Parallel Port : SPP) อยู่

เมื่อดูจากภาพประกอบ เทียบการทำงานโดยทั่วไปกับการเชื่อมต่อผ่านการ์ดเสียบลงในสล๊อตของคอมพิวเตอร์แล้ว พอร์ตขนานจะมีลักษณะใกล้เคียงกัน โดยการติดต่อกับพอร์ตขนานจะต้องมีการอ้างแอดเดรสตำแหน่งแอดแดรสที่ใช้อ้างถึงจะเป็นตำแหน่ง AO-A9 และใช้ขา IOR และ IOW สำหรับเป็นตัวเลือกว่าต้องการอ่านหรือเขียนรีจีสเตอร์ตัวใด จากการ์ดดีโค้ดแอสเดรส AO-A9 นี้เองทำให้ได้สัญญาณออกมาเพื่อไปควบคุมหรืออีนาเบิลวงจรบัฟเฟอร์ต่าง ๆ ดังนี้

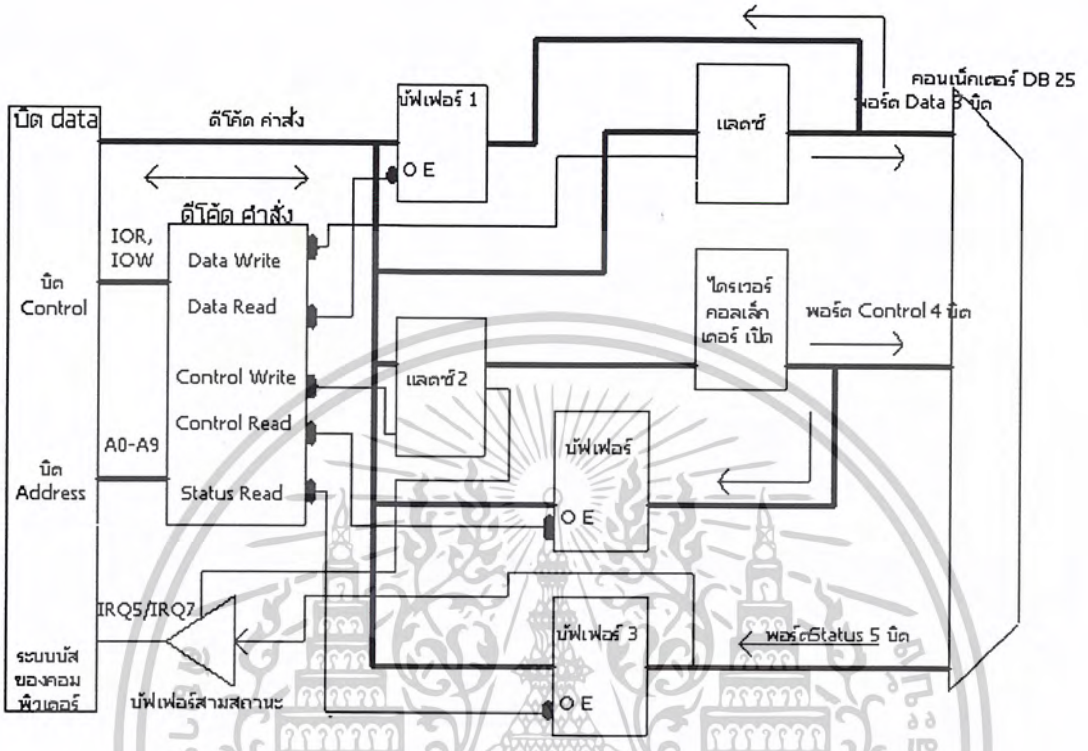
Data Write สัญญาณอีนาเบิลสำหรับนำข้อมูลที่อยู่ในบัส Data ไปออกที่ขา Data ของพอร์ตขนาน

Data Read สัญญาณอีนาเบิลสำหรับอ่านข้อมูลจากขา Data ของพอร์ตขนานมาเก็บไว้ในบัส Data

Control Write สัญญาณอีนาเบิลสำหรับนำข้อมูลที่อยู่ในบัส Data ไปออกที่ขา Control ของพอร์ตขนานสำหรับพอร์ตนี้นอกจากจะส่งข้อมูลไปยังพอร์ตขนานแล้ว ยังทำหน้าที่อีนาเบิลการอินเตอร์รัปต์ของการเปลี่ยนแปลงสัญญาณที่พอร์ตสถานะอีกด้วย

Control Read สัญญาณอีนาเบิลสำหรับอ่านค่าข้อมูลจากขา Control มาเก็บไว้ในบัส Data

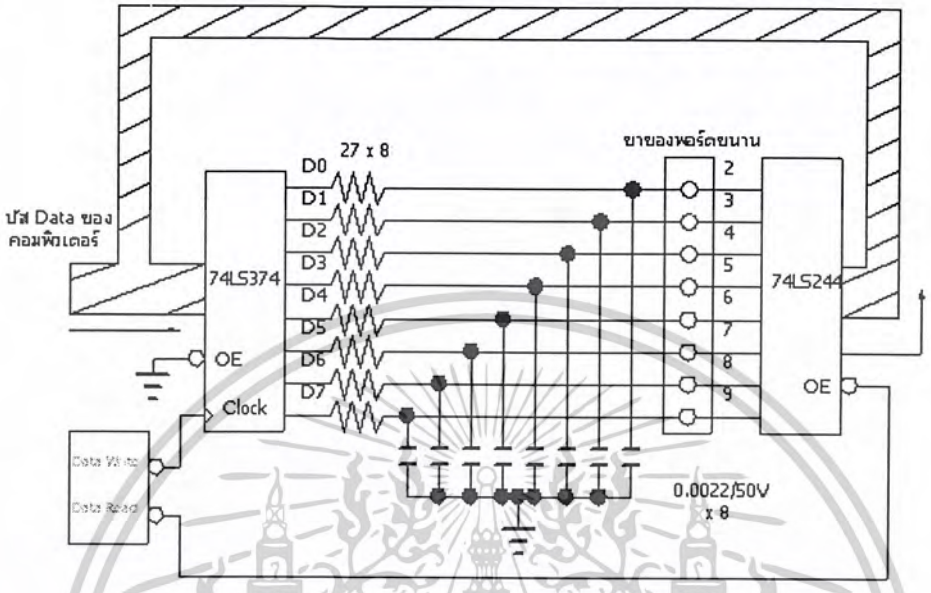
Status Read สัญญาณอีนาเบิลสำหรับอ่านค่าข้อมูลจากขาพอร์ต Status มาเก็บไว้ใน บัส Data



รูปที่ 2.5 แสดงระบบบัสภายในของพอร์ตขนาน

2.2.2 พอร์ตข้อมูล (Data Port)

จากรูปที่ 2.6 แสดงให้เห็นว่าพอร์ตข้อมูลจะประกอบไปด้วยบัฟเฟอร์ 1 ตัวและไอซีแลตช์อีก 1 ตัว เมื่อคอมพิวเตอร์ต้องการส่งข้อมูลไปยังเครื่องพิมพ์ คอมพิวเตอร์จะเขียนข้อมูลไปยังไอซีแลตช์ทั้ง 8 บิต เอาท์พุตของไอซีแลตช์ 1 คือ DO-D7 ซึ่งเอาท์พุตนี้จะไปปรากฏอยู่ที่พอร์ตขนานในตำแหน่งขา 2 ถึงขา 9 และที่ขาเอาท์พุตนี้สัญญาณ Data จะส่งกลับไปเป็นอินพุตของบัฟเฟอร์ 1 ด้วย ทำให้คอมพิวเตอร์สามารถอ่านค่าสถานะปัจจุบันที่เกิดขึ้นกับพอร์ตข้อมูลได้



รูปที่ 2.6 วงจรภายในของพอร์ตข้อมูล

เมื่อคอมพิวเตอร์ส่งข้อมูล ข้อมูลจะถูกส่งมาจากบัสข้อมูลของคอมพิวเตอร์ผ่านไปให้กับไอซี 74L374 ซึ่งเป็นไอซีแลตช์ข้อมูล และเมื่อต้องการให้ข้อมูลปรากฏที่เอาท์พุต คอมพิวเตอร์จะส่งสัญญาณ Data Write ออกไปที่ขา CLK ของ 74LS374 เอาท์พุตจาก 74LS374 จะถูกกรองด้วยวงจร RC ซึ่งประกอบด้วยตัวต้านทานค่า 27 โอห์ม และตัวเก็บประจุ 0.0022 uF เพื่อให้ช่วงเวลาที่เปลี่ยนจากลอจิก “0” เป็นลอจิก “1” หรือจากลอจิก “1” เป็นลอจิก “0” เป็นไปอย่างช้า ๆ เนื่องจากการเปลี่ยนแรงดันที่รวดเร็วทำให้ให้เกิดสัญญาณรบกวนเหนี่ยวนำข้ามไปยังข้อมูลบิตอื่น ๆ ได้ ทำให้ข้อมูลที่ส่งออกไปมีข้อผิดพลาด จากค่าตัวต้านทานและตัวเก็บประจุในวงจรทำให้เกิดการหน่วงเวลาไปประมาณ 60 นาโนวินาที จากวงจรในภาพประกอบ 10 ทำให้เอาท์พุตของพอร์ตข้อมูล มีคุณสมบัติดังนี้

- กระแสซิงก์สูงสุด 24 มิลิแอมป์
- กระแสซอร์สสูงสุด 206 มิลิแอมป์
- ระดับแรงดันของลอจิก “1” ต่ำสุดเท่ากับ 4 โวลต์
- ระดับแรงดันสูงสุดสำหรับลอจิก “0” เท่ากับ 0.5 โวลต์

สำหรับบัฟเฟอร์สำหรับการอ่านข้อมูลกลับได้แก่เบอร์ 74LS244 ซึ่งเมื่อต้องการอ่านค่าคอมพิวเตอร์จะส่งสัญญาณ Data Read ออกมาเพื่ออีนาเบิลไอซี 74LS244 สำหรับพอร์ตขนานแบบมาตรฐาน (Standard Parallel Port : SPP) พอร์ตข้อมูล จะต้องใช้เพื่อการส่งค่าออก เอาท์พุตเท่านั้น

DB-25	รีจิสเตอร์	ทิศทาง	ตำแหน่งบิต	ชื่อขาสัญญาณ	หน้าที่การทำงาน
1	Control	Out	$\overline{C0}$	$\overline{STROBE}$	แอกทีฟ "0" ส่งค่าออกไปเพื่อบอกว่าที่ราดค้ามีข้อมูลแล้ว
2-9	Data	Out	D1-D8	DATA1-DATA8	สำหรับพอร์ตขนานมาตรฐานเดิมขานี้ทำหน้าที่เป็นขาส่งข้อมูลเอาต์พุตเท่านั้น สำหรับในปัจจุบันขานี้รับข้อมูลอินพุตได้ด้วย
10	Status	In	S6	nACK	เป็นพัลส์ลอจิก "0" ที่ส่งมาจากเครื่องพิมพ์เพื่อบอกว่าได้รับข้อมูลที่ส่งไปแล้ว
11	Status	In	$\overline{S7}$	$\overline{BUSY}$	เป็นสัญญาณแจ้งมาจากเครื่องพิมพ์ว่ายังไม่พร้อมรับข้อมูล
12	Status	In	S5	PE	แจ้งกระดาษหมด
13	Status	In	S4	SELECT	แจ้งว่าเครื่องพิมพ์ต่ออยู่
14	Control	Out	$\overline{C1}$	$\overline{AUTO\ FEED}$	สั่งเครื่องพิมพ์ให้เลื่อนบันทัด
15	Status	In	S3	$\overline{ERROR}$	สัญญาณจากเครื่องพิมพ์มายังคอมพิวเตอร์เพื่อแสดงข้อผิดพลาดจากการพิมพ์
16	Control	Out	C2	$\overline{INIT}$	รีเซตเครื่องพิมพ์โดยให้ลอจิก "0"
17	Control	Out	$\overline{C3}$	$\overline{SELECT\text{-}IN}$	ส่งสัญญาณไปยังเครื่องพิมพ์เพื่อแจ้งว่าต้องการเลือกเครื่องพิมพ์เครื่องนี้
18-25				GND	กราวด์

ตารางที่ 2.2 แสดงสัญญาณทั้งหมดที่อยู่บนพอร์ตขนาน

แต่สำหรับพอร์ตขนานที่มีการสื่อสารสองทิศทาง (Bi-directional Parallel Port) สามารถอ่านค่าจากพอร์ตข้อมูล ได้ด้วยแต่ก่อนที่จะอ่านค่าต้องจำไว้เสมอว่าจะต้องป้อนค่าเอาท์พุตให้มีค่าลอจิก "1" ทั้งหมดก่อน

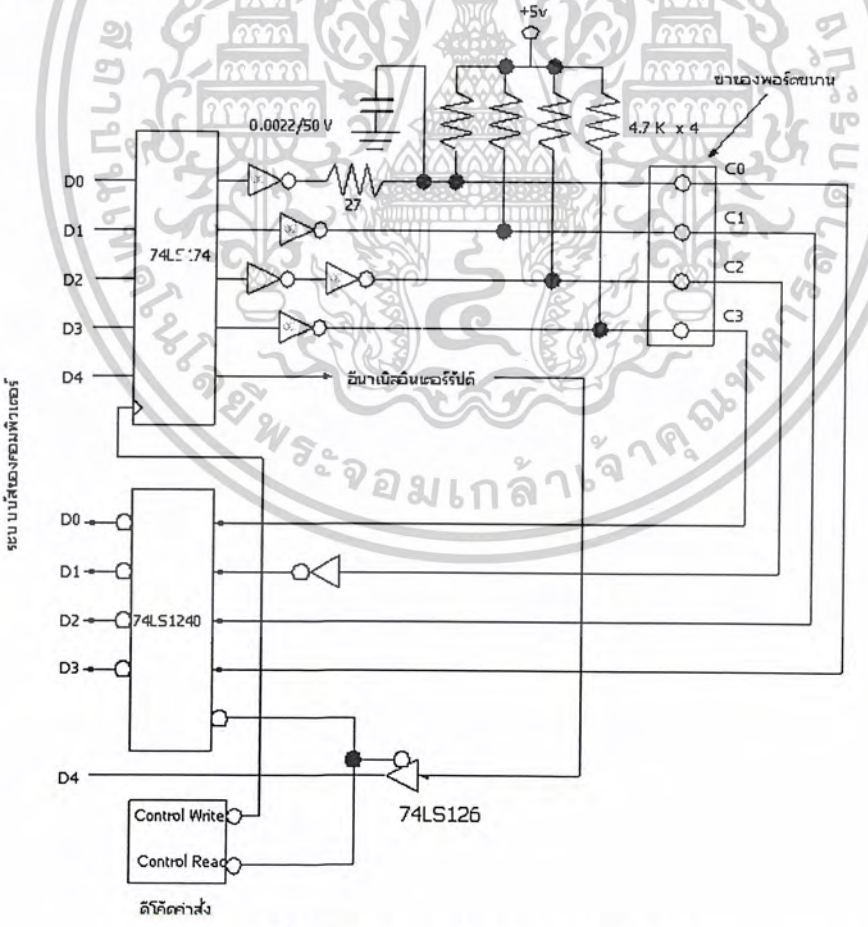
2.2.3 พอร์ตควบคุม (Control Port)

พอร์ตควบคุม ใช้สำหรับคอมพิวเตอร์ควบคุมเครื่องพิมพ์ ตารางที่ 2.2 จะเห็นว่าพอร์ตควบคุม ประกอบไปด้วยบิตเอาท์พุต 4 บิตที่ต่อออกไปยังเครื่องพิมพ์ ส่วนบิตอินาเบิลอินเตอร์รัปต์

ไม่ได้ถูกต่อออกไป รูปที่ 2.7 แสดงวงจรภายในของพอร์ตควบคุม เอาท์พุตของพอร์ตควบคุม มีอินเวอร์เตอร์แบบคอลเล็กเตอร์เปิดต่อร่วมอยู่ โดยเอาท์พุตเหล่านี้จะถูกพูลอัปไว้ด้วยตัวต้านทานค่า 4.7 กิโลโอมท์ สำหรับบิต C2 จะผ่านอินเวอร์เตอร์ถึงสองตัวทำให้ที่เอาท์พุตของบิต C2 ไม่มีการกลับสถานะลอจิก

สถานะของพอร์ตควบคุม สามารถอ่านกลับได้โดยการใช้บัฟเฟอร์เบอร์ 74LS240 ซึ่งเอาท์พุตของ 74LS240 มีอินเวอร์เตอร์อยู่ภายใน ทำให้ค่าที่อ่านได้ตรงกับค่าที่ส่งออกไป การควบคุมการอ่านและเขียนข้อมูลกับพอร์ตควบคุม คอมพิวเตอร์จะส่งข้อมูลมาที่ขา Control Write และ Control Read

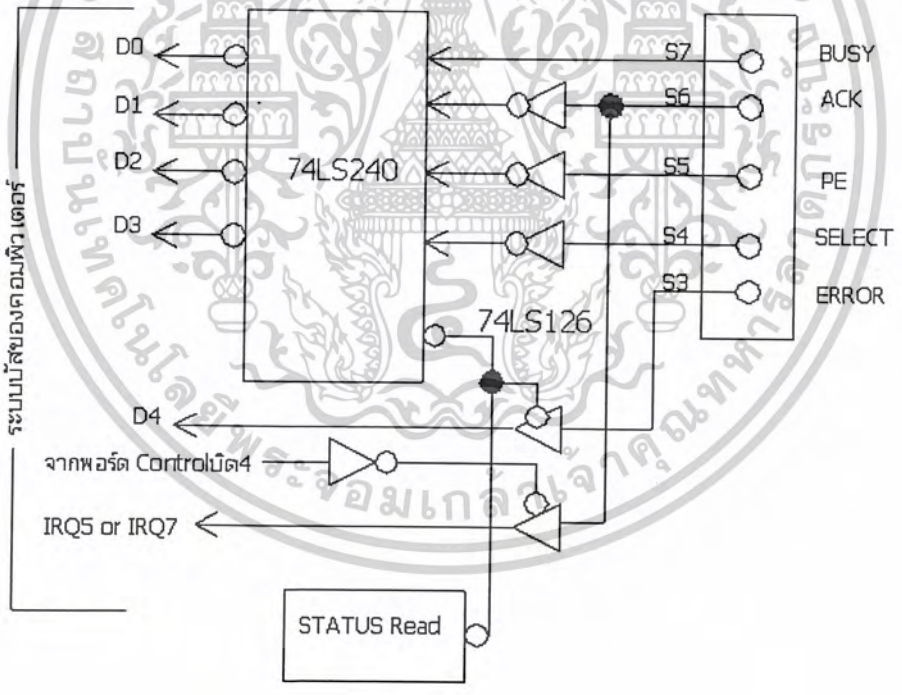
เนื่องจากเอาท์พุตของพอร์ตควบคุม เป็นแบบคอลเล็กเตอร์เปิด ดังนั้นผู้ใช้งานสามารถใช้พอร์ตนี้ในการอ่านค่าสัญญาณอินพุตจากภายนอกได้ โดยก่อนที่จะอ่านค่าจะต้องทำให้ขาพอร์ตที่ต้องการอ่านค่าลอจิก “1” เสียก่อน



รูปที่ 2.7 วงจรภายในของพอร์ตควบคุม

2.2.4 พอร์ตแสดงสถานะ (Status Port)

พอร์ตสถานะเป็นพอร์ตที่คอมพิวเตอร์ใช้สำหรับการอ่านค่าสถานะจากเครื่องพิมพ์ รูปที่ 2.8 แสดงรายละเอียดภายในของพอร์ตสถานะจะสังเกตุได้ว่ามีขาสัญญาณอยู่ทั้งหมด 5 สัญญาณด้วยกัน และจะเรียกชื่อเป็น S3, S4, S5, S6 และ S7 ซึ่งตัวเลขนั้นหมายถึงตำแหน่งบิตของขาเหล่านี้ภายในรีจีสเตอร์ Status นั่นเอง สำหรับ S7 จะมีข้อแตกต่างจากบิตอื่น ๆ ที่เมื่อสัญญาณจากภายนอกส่งเข้ามาแล้วจะไม่ผ่านอินเวอร์เตอร์ ในขณะที่ขาอื่น ๆ ผ่านอินเวอร์เตอร์ทั้งหมด ดังนั้นเมื่อข้อมูลจากขาอินพุตไปยัง 74LS240 ซึ่งเอาท์พุตมีการกลับสถานะทำให้บิต S7 เป็นบิตเดียวที่มีการกลับสถานะ นอกจากนี้ในการใช้งานถ้าต้องการให้มีการสร้างสัญญาณอินเตอร์รัปต์จากขอบขาขึ้นของขา S6 สามารถกำหนดค่าได้จากพอร์ตควบคุม บิต 4



รูปที่ 2.8 วงจรภายในของพอร์ตสถานะ

2.2.5 การนำพอร์ตขนานไปใช้งาน

สำหรับพอร์ตขนานแบบมาตรฐานผู้ใช้งานสามาถนำพอร์ตอินพุต 5 บิต (พอร์ต Status) พอร์ตเอาท์พุต 4 บิต (พอร์ต Control) และพอร์ตเอาท์พุตอีก 8 บิต (พอร์ตข้อมูล) ไปใช้งานได้โดยตรง โดยที่ 4 บิตของพอร์ตเอาท์พุตหรือพอร์ตควบคุม นั้นสามารถดัดแปลงให้ใช้งานเป็นพอร์ตอินพุตขนาด 4 บิตได้ด้วยดังนั้นผู้ใช้งานจึงสามารถนำสัญญาณจากพอร์ตขนานที่มีมากถึง 17 เส้นไปใช้งานในการควบคุมโดยใช้ระดับสัญญาณ TTL

• การเขียนโปรแกรมติดต่อกับพอร์ตขนาน

พอร์ตขนานของคอมพิวเตอร์จะมีลักษระเช่นเดียวกับอุปกรณ์อินพุตเอาท์พุตตัวอื่น ๆ คือ เมื่อต้องการติดต่อจะต้องกำหนดแอสเดรสที่ต้องการติดต่อด้วย ตาราง 3 แสดงแอสเดรสของพอร์ตขนาน โดยแบ่งออกเป็น 3 ตำแหน่งคือ แอดแดรสของรีจีสเตอร์ Data, รีจีสเตอร์ Status, และรีจีสเตอร์ Control โดยแอดแดรสนี้จะมีอยู่ 3 ชุด สำหรับพอร์ตขนาน 3 ชุดคือ LPT1,LPT2 และ LPT3

ชื่อพอร์ต	LPT1:		LPT2:		LPT3:	
	ฐากลิบ	ฐานสิบหก	ฐานสิบ	ฐานสิบหก	ฐานสิบ	ฐานสิบหก
DATA	888	378H	956	3BCH	632	278H
STATUS	889	379H	957	3BDH	633	279H
CONTROL	890	37AH	958	3BEH	634	27AH

ตารางที่ 2.3 แสดงแอดเดรสของพอร์ตขนาน

เมื่อต้องการต้องการติดต่อกับพอร์ตขนานในตำแหน่งใด ก็ให้ส่งค่าข้อมูลออกไปที่พอร์ตขนานในตำแหน่งนั้น ๆ ยกตัวอย่างการเขียนโปรแกรมด้วย QBASIC เพื่อส่งค่าลอจิก “1” ออกไปทุกบิตของพอร์ตดาต้า ของ LPT1 ดังนั้นจะต้องเขียนโปรแกรมดังนี้

OUT &H378,&HFF โดยที่เครื่องหมาย &H ที่แสดงนั้นหมายถึงตัวเลขฐานสิบหก

คำสั่ง OUT เป็นการส่งค่าข้อมูลออกเอาท์พุตของพอร์ตอินพุตเอาท์พุต

คำสั่ง 378 เป็นแอดแดรสของรีจีสเตอร์ Data สำหรับ LPT1

ค่าข้อมูล FF เป็นข้อมูลเลขฐานสิบหก ซึ่งหมายถึงการให้บิตทุกบิตของรีจีสเตอร์ Data มีลอจิกเป็น "1" นั่นเอง

ส่วนการอ่านค่าจากพอร์ตขนานมายังคอมพิวเตอร์ผ่านทางพอร์ตสถานะของ LPT1 สามารถเขียนโปรแกรมด้วย QBASIC ได้ดังนี้

Temp = INP (&H379) โดยที่

คำสั่ง INP(_) เป็นคำสั่งสำหรับการอ่านค่าข้อมูล

ค่า 379 เป็นตำแหน่งแอดแดรสของรีจีสเตอร์ Status สำหรับ LPT1 ในตัวเลขฐานสิบหก

ตัวแปร Temp เป็นตัวแปรที่ใช้เก็บข้อมูลที่อ่านได้จากพอร์ตขนาน

สำหรับโปรแกรมอื่น ๆ เช่น แอสเซมบลี้,เทอร์โบปาสคาล หรือเทอร์โบซี จะมีรูปแบบการเขียนโปรแกรมที่แตกต่างกันบ้างอย่างไรก็ตามโปรแกรมทุกตัวต่างก็ใช้วิธีการเดียวกันคือ กำหนดแอสเดรสที่ทำการติดต่อจากนั้นจึงติดต่อกับแอสเดรสเหล่านั้นด้วยคำสั่งสำหรับการอ่านหรือเขียน

• การเขียนโปรแกรมติดต่อกับพอร์ตขนานด้วย Visual Basic

การเขียนโปรแกรมด้วย Visual Basic ชุดคำสั่งส่วนใหญ่จะมีรูปแบบใกล้เคียงกับ QBASIC แต่ Visual Basic จะไม่มีคำสั่งสำหรับการติดต่อกับพอร์ตโดยตรงคือ คำสั่ง Inp() และคำสั่ง OUT เหมือนกับ QBASIC ดังนั้นเพื่อให้สามารถติดต่อกับพอร์ตขนานได้จึงจำเป็นต้องเพิ่มโปรแกรมบางตัวเข้าไป โดยโปรแกรมที่เพิ่มเข้าไปนี้จะอยู่ในรูปของ DLL (Dynamic Linked Library)

ไฟล์ DLL นี้จะมันอยู่ 2 ไฟล์คือ inpout.dll และ inpout32.dll โดยที่ inpout.dll นั้นใช้สำหรับระบบปฏิบัติการ 16 บิตหรือบนวินโดวส์ 3.1 นั่นเอง ส่วน inpout32.dll จะใช้สำหรับระบบปฏิบัติการที่เป็น 32 บิตซึ่งก็คือวินโดวส์95หรือวินโดวส์ 98

สำหรับตำแหน่งที่ใช้เก็บไฟล์ inpout32.dll นั้นจะต้องเก็บไว้ที่ไดเรกตอรี่ SYSTEM ซึ่งอยู่ภายในไดเรกตอรี่ที่เก็บโปรแกรมวินโดวส์ โดยส่วนใหญ่จะมีชื่อเป็น Windows

• การกำหนดค่าในโปรแกรมเพื่อเรียกใช้งานไฟล์ DLL มีรูปแบบการกำหนดค่าดังนี้

• สำหรับระบบปฏิบัติการ 16 บิต

```
Declare Function Inp% Lip "Input .DLL" Alias "Inp16" (ByVal PortAddress%)
Declare Sub Out Lip "Input .DLL" Alias "Out 16 " (ByVal PortAddress%, ByVal BtVal
ByteToWrite)
```

- สำหรับระบบปฏิบัติการ 32 บิต

```
Public Declare Function Inp Lip “Inpout 32.dll”
Alias “Inp32” (ByVal PortAddress As Integer) As Integer
Public Declare Sub Out Lip “inpout32.dll”-
Alias “Out32” (ByVal PortAddress As Integer,ByVal Value AS Integer)
```

ในกรณีที่มีการใช้งานโปรแกรมทั้ง 2 ระบบปฏิบัติการ สามารถเพิ่มคำสั่ง IF เข้าไปเพื่อตรวจสอบระบบปฏิบัติการก่อนที่จะเลือกใช้งาน DLL ตัวที่ต้องการ โดยสามารถเขียนโปรแกรมได้ดังนี้

```
#If Win32 Then
‘Declare Function Inp Lip “inpout32.dll”
Alias “Inp32” ” (ByVal PortAddress As Integer) As Integer
Public Declare Sub Out Lip “inpout32.dll”-
Alias “Out32” (ByVal PortAddress As Integer,ByVal Value AS Integer)
#Else
Declare Function Inp Lip “input .dll” (ByVa. Port%) AS Integer
Declare Sub Out Lip “Input .DLL(ByVal Value%)
#End If
```

แต่การกำหนดในสองรูปแบบหลังนี้จะสามารถใช้งานได้กับ VISUAL BASIC ตั้งแต่เวอร์ชั่น 4 ขึ้นไปเท่านั้น

เมื่อมาถึงตรงนี้ อาจเกิดคำถามว่าขึ้นว่า ทำไมไมโครซอฟต์ผู้พัฒนาซอฟต์แวร์ VISUAL BASIC จึงไม่รวมคำสั่ง Inp และคำสั่ง Out ไว้ในโปรแกรม VISUAL BASIC เนื่องจากว่าการเขียนและอ่านข้อมูลไปยังพอร์ตหรือหน่วยความจำ โดยตรงนั้นอาจทำให้เกิดปัญหาแฮงค์หรือทำงานผิดพลาดได้ และ VISUAL BASIC เป็นระบบปฏิบัติการ ที่ทำงานบนวินโดว์ซึ่งมีการทำงานแบบมัลติทาร์สกิ้ง (multitasking) มีโปรแกรมหลายๆตัวทำงานอยู่พร้อมกัน ดังนั้นเมื่อเกิดความเสียหายกับโปรแกรมตัวหนึ่งอาจส่งผลให้โปรแกรมที่ทำงานอยู่ทั้งหมดเสียหายได้ นอกจากนี้การเขียนข้อมูลโดยตรงไปยัง

พอร์ต อาจจะไปทับซ้อนกับโปรแกรมอื่นๆ ที่มีการเขียนข้อมูลไปยังพอร์ตเดียวกัน ส่งผลให้โปรแกรมทำงานผิดพลาด

สำหรับวินโดว์ 95 นอกจากสามาถใช้งาน DLL ในการติดต่อพอร์ตโดยตรงแล้ว ยังสามารถใช้งานในประเภท Visual Device driver (VXD) ในการติดต่อกับอุปกรณ์อินพุตเอาท์พุต โดย VXD จะตัดปัญหาเรื่องการเข้าถึงพอร์ตพร้อมกันของโปรแกรมหลายๆตัวได้ แต่สำหรับโปรแกรมสั้นๆ เช่น โปรแกรมอ่านค่าอุณหภูมิ โปรแกรมควบคุมอุปกรณ์อินพุตเอาท์พุตปกติ ซึ่งไม่มีการติดต่อกับพอร์ตอยู่ตลอดเวลา คำสั่ง Inp และ Out ใน DLL ก็ยังทำงานได้ดีและมีรูปแบบการใช้งานที่ง่ายกว่า

ฟังชั่นที่มีอยู่ใน io.dll

PortOut	ใช้ส่งข้อมูลขนาด 1 ไบต์ ไปยังพอร์ตที่กำหนด
PortWordOut	ใช้ส่งข้อมูล 1 เวิร์ด (16 บิต) ไปยังพอร์ตที่กำหนด
PortDWordOut	ใช้ส่งข้อมูลดับเบิ้ลเวิร์ด (32บิต) ไปยังพอร์ตที่กำหนด
PortIn	ใช้อ่านข้อมูลขนาด 1 ไบต์ จากยังพอร์ตที่กำหนด
PortWordIn	ใช้อ่านข้อมูล 1 เวิร์ด (16 บิต) จากยังพอร์ตที่กำหนด
PortDWordIn	ใช้ส่งข้อมูลดับเบิ้ลเวิร์ด (32บิต) จากยังพอร์ตที่กำหนด
SetPortBit	ใช้เซตข้อมูลในระดับบิตของพอร์ตที่กำหนด
ClrPortBit	ใช้เคลียร์ข้อมูลในระดับของพอร์ตที่กำหนด
NotPortBit	ใช้กลับข้อมูลในระดับของพอร์ตที่กำหนด
GetPortBit	ใช้อ่านสถานะของบิตที่กำหนด
RightPortShift	ใช้เลื่อนข้อมูลของพอร์ตไปทางขวา จะได้ค่าของบิต LSBกลับมา

ไฟล์ DLL นี้มีประโยชน์อย่างมากสำหรับ การติดต่อกับพอร์ตอินพุตเอาท์พุตของคอมพิวเตอร์ โดยเฉพาะอย่างยิ่งกับพอร์ตขนานของระบบปฏิบัติการวินโดว์ ซึ่งในปัจจุบันการติดต่อกับพอร์ตขนานทำได้ยากขึ้น แต่เมื่อใช้ไฟล์ DLL ตัวนี้จะช่วยให้สามารถเข้าถึงและควบคุมได้ง่ายและมีประสิทธิภาพ io.dll สามารถนำไปใช้กับระบบปฏิบัติการวินโดว์ 95/98/2000/NT หรือกระทั่ง XP จึงช่วยลดเวลาและขั้นตอนในการพัฒนาโปรแกรมสำหรับติดต่อพอร์ต ที่รันบนวินโดว์ได้อย่างมาก

2.3 ความรู้พื้นฐานเกี่ยวกับไมโครคอนโทรลเลอร์

PIC คือ microcontroller อีกตระกูลหนึ่ง ย่อมาจากคำว่า Peripheral Interface Controller ซึ่งแนวคิดของ microcontroller ตระกูลนี้ก็คือ พยายามรวมเอาทุกอย่างเอาไว้ในตัวของเองไม่ว่าจะเป็น PROGRAM MEMROY, RAM, EEPROM, SERIAL, I2C, PWM, A/D ฯลฯ โดยไม่จำเป็นต้องต่ออุปกรณ์เสริมจากภายนอก ในตัวของ PIC จะมีฟังก์ชันที่ใช้ในการประมวลผล รวมทั้งหน่วยความจำ ซึ่งทำให้มันเหมือนกัน CPU ตัวนึง

- **ความเร็วของ PIC**

ภาคของความถี่สัญญาณนาฬิกา ปัจจุบันสามารถทำสัญญาณนาฬิกาได้ที่ 20 MHz ซึ่งทำให้หนึ่งคำสั่งของ PIC ใช้เวลาเพียง 0.25 uSec

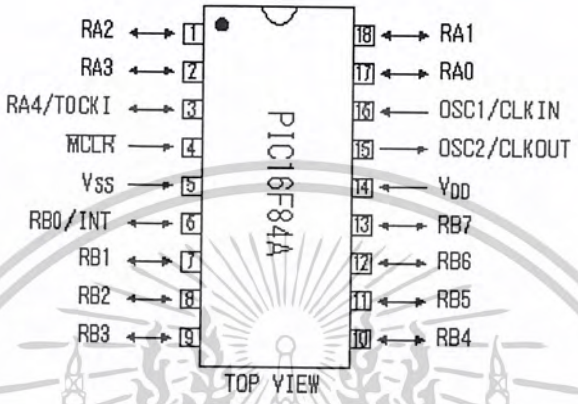
- **หน่วยความจำของ PIC**

ในอดีตหน่วยความจำของ PIC จะค่อนข้างน้อย คืออยู่ระหว่าง 512 words ถึง 4K words แต่ในปัจจุบัน บริษัท microchip ซึ่งเป็นผู้ผลิต PIC ได้พัฒนาจนทำให้ memory ของ PIC มีขนาดเป็นหลายสิบกิโลไบต์ และมีทีท่าว่าจะขยายได้ใหญ่ขึ้นเรื่อยๆ ในเรื่องของการนับขนาดของหน่วยความจำของ PIC จะนับไม่เหมือนปกติ โดยที่ หนึ่งคำสั่งของ PIC จะมีขนาด 14 bits ดังนั้นเราจะเรียกว่า 1 word ของ PIC จะมีขนาด 14 bits เช่น PIC16F84A ระบุว่ามีหน่วยความจำ 1 K (ซึ่งหมายถึง 1 Kword ถ้าคำนวณให้เป็นแบบ 1 byte = 8 bit จะได้ว่า 1 x 1,024 x 14 = 14,336 bits ดังนั้นก็คือ 14,336/(8 x 1,024) = 1.75K bytes นั่นเอง

- **Hardware เบื้องต้นของ PIC16F84A**

PIC16F84A จะมีโครงสร้างแบบ RISC(Reduced Instruction Set Computer) PIC ในตระกูล 16xxx จะมีคำสั่งให้ใช้ทั้งหมด 35 คำสั่ง

• **โครงสร้างขาของ PIC16F84A**



รูปที่ 2.9 การจัดขาของไมโครคอนโทรลเลอร์ PIC16F84A

OSC1/CLKIN : Oscillator crystal input /

External clock source input.

OSC2/CLKOUT : Oscillator crystal output.

ทั้งสองขานี้จะต่อกับ crystal หรือ resonator ในกรณีที่อยู่ใน crystal oscillator mode(ใช้สัญญาณนาฬิกาจากภายนอก)

MCLR(inv) : Master clear(reset)input /

Programming voltage input. เมื่อขานี้เป็น LOW แล้ว MCU จะถูก reset อีกหน้าที่หนึ่งของขานี้ก็คือจะเป็น input ของ voltage programming ขณะที่ทำการ program ตัว MCU

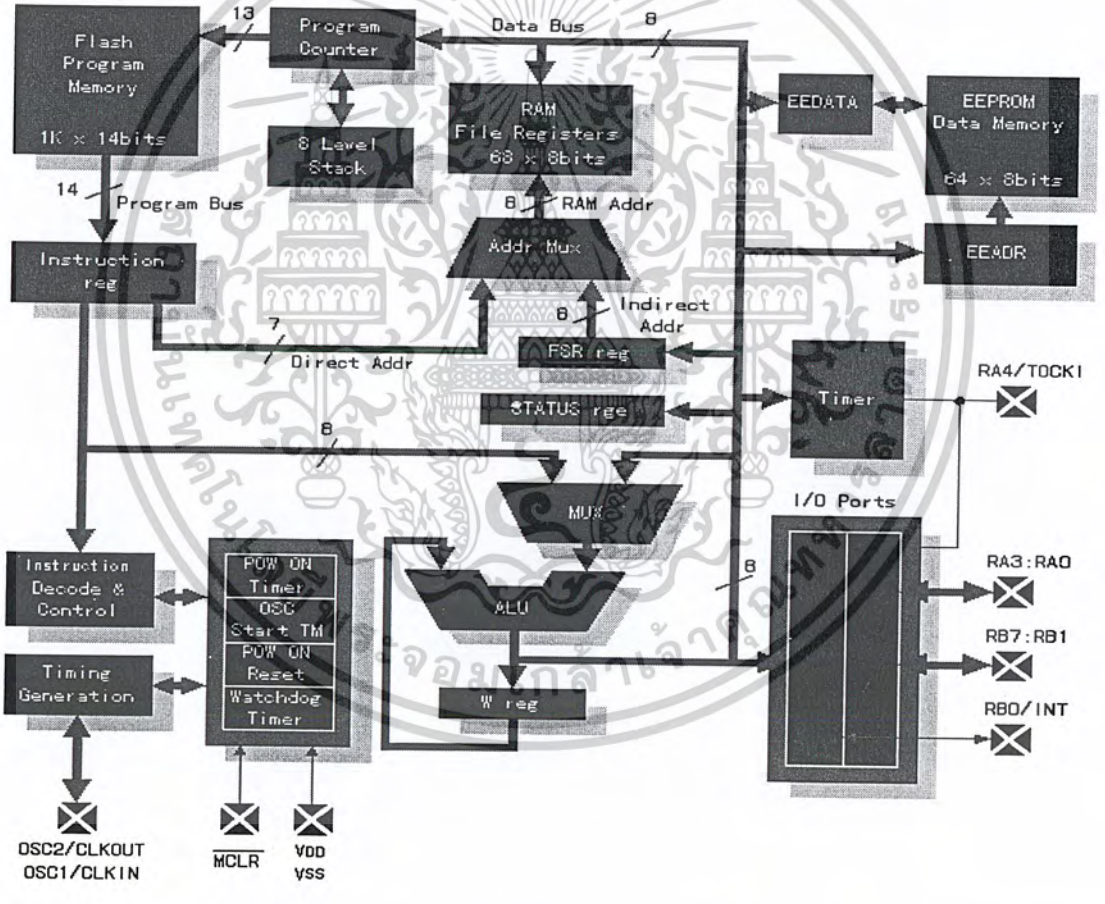
RA0 - RA3 : Bi-directional I/O port. เป็นพอร์ตแบบ สองทิศทาง คือเลือกให้เป็น INPUT หรือ OUTPUT ก็ได้อย่างใดอย่างหนึ่ง

RA4/T0CKI : Bi-directional I/O port. เป็นพอร์ตแบบสองทิศทาง คือเลือกให้เป็น INPUT หรือ **OUTPUT** ก็ได้อย่างใดอย่างหนึ่ง อีกหน้าที่หนึ่งก็คือ Clock input to the TMR0 timer/counter. เป็น input ของ สัญญาณนาฬิกาเพื่อป้อนให้กับ Timer 0 ซึ่งอยู่ภายใน MCU ในกรณีที่เลือกว่าแหล่งของสัญญาณนาฬิกาที่ป้อนให้กับ Timer 0 ให้ใช้จากภายนอก

RB0/INT : Bi-directional I/O port. เป็นพอร์ตแบบสองทิศทาง คือเลือกให้เป็น INPUT หรือ OUTPUT ก็ได้ อย่างใดอย่างหนึ่ง อีกหน้าที่หนึ่งก็คือ External interrupt pinรับสัญญาณ interrupt เมื่อเกิดการเปลี่ยนแปลงของสัญญาณที่ขานี้

RB1 - RB7 : Bi-directional I/O port. เป็นพอร์ตแบบสองทิศทาง คือเลือกให้เป็น INPUT หรือ OUTPUT ก็ได้ อย่างใดอย่างหนึ่ง

VSS : Ground



รูปที่ 2.10 สถาปัตยกรรมของไมโครคอนโทรลเลอร์ PIC16F84A

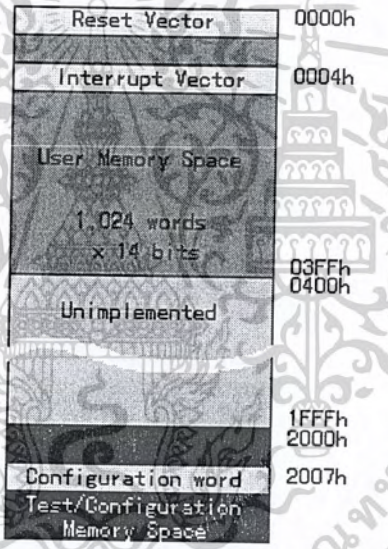
• **สถาปัตยกรรมของไมโครคอนโทรลเลอร์**

สถาปัตยกรรมของไมโครคอนโทรลเลอร์ภายในประกอบไปด้วยส่วนต่างๆหลายอย่างดังนี้

Flash Program Memory

Flash memory เป็นพื้นที่หน่วยความจำสำหรับเก็บ program ที่เราเขียนขึ้น ซึ่งมีขนาด1,024 words ถึงแม้ว่าจะไม่มีไฟฟ้าจ่ายให้กับ MCU ข้อมูลที่เก็บอยู่ใน flash memory ก็จะไม่หายไป จุดเด่นของ Flash memory ก็คือสามารถเขียนทับเข้าไปใหม่ได้หลายๆ ครั้ง ซึ่งจำนวนครั้งจะอยู่ที่ประมาณ 1000 ครั้ง

ภายใน program memory อาจสามารถแบ่งเป็นส่วนๆ ได้ดังนี้



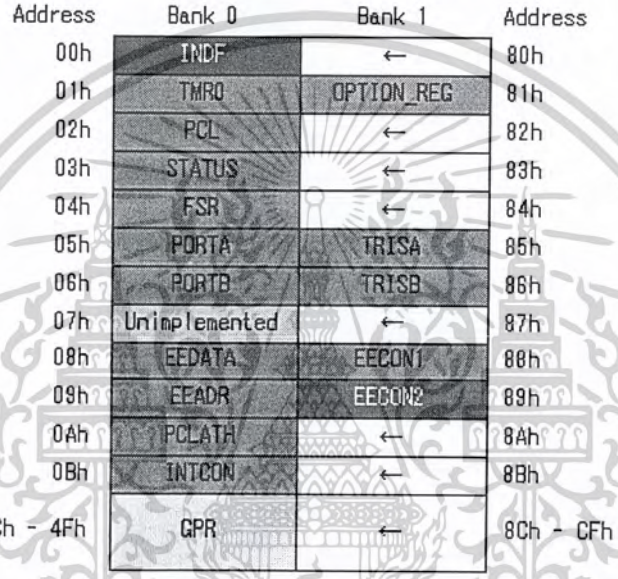
รูปที่ 2.11 การจัดสรรหน่วยความจำโปรแกรมใน PIC16F84A

Reset Vector (0000h) เมื่อการ reset เกิดขึ้นเนื่องจากมีการป้อนไฟเข้า MCU เป็นครั้งแรก หรือเกิด WDT(Watchdog Timer) time-out หรือในกรณีอื่นๆ โปรแกรมจะเริ่มต้นหลังจาก reset ณ ตำแหน่งนี้

Peripheral Interrupt Vector (0004h) เมื่อเกิดการ interrupt ขึ้น program memory pointer จะชี้มายัง ณ ตำแหน่งนี้

Configuration word (2007h) การทำงานเบื้องต้นของ PIC จะถูกกำหนดที่หน่วยความจำตรงนี้ ไม่ว่าจะเป็น Enable/Disable Power-up timer, Enable/Disable Watchdog timer, Oscillator Selection bits(กำหนดที่มาของสัญญาณนาฬิกา) หน่วยความจำที่ตำแหน่งนี้ไม่สามารถเขียนได้ด้วยการเขียนโปรแกรม จะต้องกำหนดในขณะทำการเขียนโปรแกรมลงสู่ Flash memory ของ MCU

RAM(Random Access Memory) File Registers



รูปที่ 2.12 การจัดสรรหน่วยความจำข้อมูลใน PIC16F84

หน่วยความจำของ RAM ภายใน PIC16F84(A) จะมีโครงสร้างเป็นแบ่งเป็น bank โดยจะมีขนาด 80 bytes(00h-4Fh) ต่อ bank ในกรณีของ PIC16F84A จะมี 2 banks Memory ในส่วนนี้จะถูกแบ่งออกเป็น 2 ส่วนคือ

ส่วนแรก 12 bytes(00h-0Bh) ของแต่ละ bank ซึ่งจะถูกเรียกว่า SFR(Special Function Registers) จะใช้สำหรับบันทึกสถานะการทำงานของ PIC, เงื่อนไขของการกำหนดสถานะของ port ว่าเป็น input/output ports และเงื่อนไขอื่นๆ ส่วนที่สองจะมีขนาด 68 bytes(0Ch-4Fh) ซึ่งเริ่มตั้งแต่ ไบต์ที่ 13 จะถูกเรียกว่า GPR(General Purpose Registers) ซึ่งสามารถใช้เป็นหน่วยความจำที่จะเก็บผลลัพธ์และเงื่อนไขต่างๆ เพื่อใช้ในการประมวลผลโปรแกรมขณะโปรแกรมทำงาน ถึงแม้ว่า PIC16F84A จะมี 2 bank แต่ SFR จะมีทั้งหมดเพียง 16 ชนิดไม่ใช่ 24 ชนิด โดย SFR บางตัวจะมีอยู่ทั้ง 2 bank ส่วน GPR

ถึงแม้จะเปลี่ยน bank ก็ยังคงชี้ไปยังตำแหน่งเดิมเพราะ 16F84A มี GPR อยู่เพียง bank เดียวเท่านั้น และเมื่อไม่มีไฟฟ้าจ่ายให้กับ PIC ค่าใน memory ส่วนนี้จะหายไปหมด อย่างไรก็ตามพื้นที่ใน GPR สามารถเขียนใหม่กี่ครั้งก็ได้ ไม่มีข้อจำกัดจำนวนครั้ง

EEPROM(Electrically Erasable Programmable Read Only Memory)

หน่วยความจำในส่วนนี้เมื่อไม่มีไฟฟ้าจ่ายให้กับ MCU แล้วข้อมูลที่อยู่ภายในยังคงอยู่จะไม่หายไป และสามารถเขียนด้วยคำสั่งของ program จะมีขนาด 64 bytes. อย่างไรก็ตามการเขียนซ้ำก็มีข้อจำกัดโดยสามารถเขียนทับใหม่ได้ประมาณ 1 ล้านครั้ง ดังนั้นหน่วยความในส่วนนี้จะใช้เก็บข้อมูลที่ไม่ค่อยจะเปลี่ยนแปลงบ่อยนัก หน่วยความจำในส่วนนี้สามารถเก็บข้อมูลได้นาน 40 ปี

SFR Registers

SFR(Special Function Registers) มีอยู่ 16 ชนิดด้วยกัน ซึ่งสามารถอ้างถึงด้วยการ เปลี่ยนตำแหน่งของ bank ที่จะอ้าง จากรูปข้างล่างแสดงถึงโครงสร้างของ File Registers. ทั้งหมดจะมีขนาด 160 bytes แต่ในข้อเท็จจริงแล้วจะไม่สามารถอ้างได้ครบทั้งหมด ในส่วนที่เป็นสีขาวแล้วมีลูกศรชี้จะหมายถึงว่า ถ้าเราอ้าง RAM ที่ตำแหน่งนี้ใน BANK 1 ค่าที่ได้ก็คือค่าทางซ้ายมือที่อยู่ใน BANK 0 นั่นเอง เช่นเมื่อเราอ้างที่ address 84h ใน Bank 1 ค่าที่ได้ก็จะเป็น ค่า FSR ที่อยู่ใน BANK 0 สำหรับในส่วนที่เป็นสีเทาก็ คือ memory ที่ไม่สามารถอ้างถึงได้

ตัวแปรใน SFR มีดังนี้

INDF : จะเก็บค่าของ Data memory ที่ถูกชี้แบบ indirect addressing

TMR0 : เป็นTimer counter ของ Timer 0

PCL : เก็บค่า 8 bits ล่างของ program counter

STATUS : จะเก็บค่า Flag ของผลลัพธ์ที่เกิดจากการคำนวณ

FSR : เป็น pointer ใช้สำหรับอ้างอิง data memory แบบ indirect

PORTA : เก็บค่าสถานะของ PORTA

PORTB : เก็บค่าสถานะของ PORTB

EEDATA : เก็บค่าของ Data ที่ EEPADR ชี้อยู่

EEADR : ตำแหน่งของ EEPROM ที่ต้องการอ้างถึง

PCLATH : เป็น 5 bits บนของ program counter

INTCON : ใช้คำคุมการเกิด Interruption

OPTION_REG : ใช้สำหรับกำหนด Mode การทำงานของ MCU

TRISA : ใช้กำหนด Mode ของ PORTA ว่าเป็น INPUT หรือ OUTPUT

TRISB : ใช้กำหนด Mode ของ PORTB ว่าเป็น INPUT หรือ OUTPUT

EECON1 : เป็น register ที่ใช้ควบคุม EEPROM

EECON2 : เป็น register ที่ใช้ป้องกันการเขียน EEPROM

Program Counter

เป็น counter ที่แสดงถึงตำแหน่ง address ของ program ที่เขียนเข้าไปไว้ใน flash memory ที่กำลังทำการประมวลผล ซึ่งจะเป็น counter ขนาด 13 bits โดยทั่วไปแล้ว counter ตัวนี้จะเพิ่มขึ้น 1 ทุกๆ ครั้งเมื่อมีการประมวลผลคำสั่งเกิดขึ้น 1 ครั้ง ซึ่งค่าที่แสดงก็คือตำแหน่งของคำสั่งต่อไปที่จะทำการประมวลผล แต่เมื่อประมวลคำสั่ง JUMP ตัว counter จะมีค่าเท่ากับตำแหน่งที่คำสั่ง JUMP นั้นอ้างถึง

8 Level Stack

Stack คือ memory ซึ่งจะเก็บค่า return address ของ program ตัวอย่างเช่น เมื่อต้องทำการประมวลผลอย่างหนึ่งหลายๆ ครั้ง ซึ่ง program ในส่วนนี้ถูกสร้างเป็น subroutine ไว้ ในตอนสุดท้ายของ subroutine ก็จะมีคำสั่ง RETURN ทุกครั้ง ในการเรียกใช้เราจะใช้คำสั่ง CALL ในการเรียก subroutine ตำแหน่ง program address ที่ถัดจากคำสั่ง CALL ก็จะถูกเก็บลงสู่ stack (กระบวนการนี้บางครั้งจะเรียกว่า PUSH) หลังจากได้ประมวลผลคำสั่งใน subroutine แล้ว ในตอนสุดท้ายเมื่อมาเจอคำสั่ง RETURN มันก็จะทำการกระโดดไปยังตำแหน่งที่เก็บไว้ใน stack (กระบวนการนี้บางครั้งจะเรียกว่า POP) แต่เนื่องจากว่า stack มีขนาดเพียง 8 เท่านั้น นั่นก็หมายความว่าเราสามารถเรียก คำสั่ง CALL ได้เก้าครั้งติดต่อกันเท่านั้น ซึ่งถ้าใช้คำสั่ง CALL ไปมากกว่านั้นโดยไม่ RETURN กลับ ค่า stack จะถูกทับเป็นผลทำให้ เมื่อใช้คำสั่ง RETURN ก็จะไม่สามารถกลับไปยังตำแหน่งเดิมได้

Instruction Register

คำสั่งต่างๆ ของโปรแกรม ที่ถูกชี้โดย program counter จะถูกอ่านข้าไปยัง register ตัวนี้ โดยกระบวนการนี้จะถูกเรียกว่า FETCH.

Instruction Decode & Control

คำสั่งที่ถูก FETCH ไว้ใน instruction register จะถูกแปลความหมายและทำงานตามคำสั่งนั้น

Multiplexer and Arithmetic Logic Unit

โดยการแปลความหมายและทำงานตามคำสั่งจะถูกกระทำโดย Multiplexer และ the Arithmetic Logic Unit(ALU)

W Register

ย่อมาจาก work register มันจะมีหน้าที่สำหรับเก็บผลของการคำนวณที่เกิดจาก ALU เอาไว้ชั่วคราว เพื่อที่จะนำมาคำนวณต่อไป ตัวของมันจะทำหน้าที่เป็นตัวกลางในการคำนวณต่างๆ และมันยังทำหน้าที่ส่งผ่านสถานะoutput ของ input-output port อีกด้วย

STATUS Register

เป็น register ซึ่งจะเก็บค่าผลของ ALU(เช่น ผลลัพธ์ของการ บวก ลบ ของ register เป็น 0, บวก, ลบ), เงื่อนไขการเกิด timeout , เป็นตัวกำหนดว่าขณะนี้ PIC อ้าง register ที่ bank ไหน

FSR Register

FSR(File Select Register) ใช้สาหรับอ้างตำแหน่งของ RAM ในรูปแบบ indirect address การอ้างแบบ direct address ก็คือรูปแบบที่อ้างถึง Address นั้นโดยเฉพาะเจาะจงเลย เช่น movfw h'20' ซึ่งหมายความว่าทำการอ่านค่า ที่ address 20 มาเก็บไว้ที่ w register ในกรณีนี้สามารถอ้างตำแหน่งได้ตั้งแต่ 0 ถึง 127 หรืออ้างได้เพียง 7 bit ซึ่งจะอยู่ในขอบเขต 1 bank ในการที่จะเปลี่ยน bank จำเป็นที่จะต้องเกี่ยวข้องกับ RP0 bits ของ STATUS register การอ้างแบบ indirect address โดยใช้ FSR register จะนิยมใช้ในการอ้าง address ที่อยู่ติดๆ กันด้วยการอาศัยคำสั่ง inc FSR เพื่อเลื่อนไปยังตำแหน่ง memory ถัดไป

Address Multiplexer

ใช้เป็นตัวแบ่งแยกว่าเป็น indirect addressing หรือ the direct address. ซึ่งหมายความว่าทำการอ่านค่า ที่ address 20 มาเก็บไว้ที่ w register ในกรณีนี้สามารถอ้างตำแหน่งได้ตั้งแต่ 0 ถึง 127 หรืออ้างได้เพียง 7 bit ซึ่งจะอยู่ในขอบเขต 1 bank ในการที่จะเปลี่ยน bank จำเป็นที่จะต้องเกี่ยวข้องกับ RP0 bits ของ STATUS register การอ้างแบบ indirect address โดยใช้ FSR register จะนิยมใช้ในการอ้าง address ที่อยู่ติดๆ กันด้วยการอาศัยคำสั่ง inc FSR เพื่อเลื่อนไปยังตำแหน่ง memory ถัดไป

Address Multiplexer

ใช้เป็นตัวแบ่งแยกว่าเป็น indirect addressing หรือ the direct address.

EEDATA

เป็น register ที่จะใช้เมื่อมีการอ่านหรือเขียนข้อมูล EEPROM

EEADR

เป็น register ที่ใช้สำหรับอ้าง address ของ EEPROM. ซึ่งใน PIC16F84A จะมีหน่วยความจำ EEPROM อยู่ทั้งหมด 64 bytes เมื่อจะทำการเขียน EEPROM จะต้อง เขียนข้อมูล 55h และ AAh ไปยัง EECON2 เสียก่อนจึงจะเริ่มต้นการเขียน EEPROM ได้

Timer

PIC16F84A จะมี timer เพียงแค่ตัวเดียว (TMR0) มีขนาด 8 bits การทำงานของมันก็คือมันจะทำการนับไปเรื่อยๆ และจะเกิดการ time-out เมื่อการนับมาถึง 256 ซึ่งจะทำให้ T0IF bit ของ INTCON register ซึ่งเป็น SFR กลายเป็น "1" ซึ่งมีผลทำให้เกิดการ interrupt เกิดขึ้นเมื่อเกิดการ time-out. สำหรับการที่จะกำหนดว่าจะให้มีการ interrupt ของ TMR0 เกิดขึ้นได้หรือไม่นั้นกำหนดได้จาก the GIF bit และ T0IE bit ของ INTCON register ซึ่งเป็น SFR โดยถ้าเป็น "1" ก็หมายความว่ากำหนดให้มีการ interrupt เกิดขึ้นได้

I/O Ports

PIC16F84A จะมี I/O Ports ทั้งหมด 13 ขา ซึ่งการกำหนดว่าจะให้ขานั้นเป็น INPUT หรือ OUTPUT นั้นสามารถกำหนดได้ด้วยโปรแกรม ทั้ง 13 ขานั้นสามารถแบ่งออกได้เป็น 2 กลุ่มด้วยกันก็คือ 5 ขาเป็น A port และอีก 8 ขาที่เหลือเรียกว่า B port

Timing Generation

PIC จะมีวงจรภายในที่จะสร้างสัญญาณนาฬิกาในการกำหนดความจังหวะของการทำงานของตัวมัน โดยสัญญาณนาฬิกานี้จะมีแหล่งกำเนิดมาจาก crystal หรือ ceramic oscillator จากภายนอก เมื่อสัญญาณนาฬิกาต้องการความแม่นยำสูงเราจะต้องเลือกใช้ crystal แต่โดยปกติทั่วๆไปแล้วจะใช้ ceramic resonator ต่อเข้ากับ capacitors เป็น module อยู่ภายนอก PIC16F84A จะ execute 1 คำสั่ง(1 cycle) จะใช้สัญญาณนาฬิกา 4 pulses โดยจะใช้ pipeline architecture แต่ในกรณีของคำสั่ง JUMP จะใช้ 2 cycle สำหรับเวลาที่ใช้ในการ execute นั้นโดยปกติแล้วจะใช้เวลา 200 nanoseconds ถ้าใช้ crystal ที่ความถี่ 20MHz, 1/(20MHz) = 50 nanoseconds. หมายความว่าสามารถ execute คำสั่ง 5,000,000 instructions ภายในเวลา 1 วินาที

Initialization circuits

ภายใน PIC16F84A จะต้องมีการกำหนดคุณสมบัติการทำงานให้แต่ตัวมันหลายอย่างดังนี้

POWER ON Timer

เป็น Timer ที่ใช้สำหรับสร้างช่วงระยะเวลาก่อนที่จะใช้ MCU ทำงานเพื่อรอจนกระทั่งแรงดันไฟฟ้าที่ป้อนให้กับ MCU จะนิ่งในกรณีเมื่อมีการป้อนไฟเข้าไปใหม่

OSC StartTimer

เป็น Timer ที่ใช้สำหรับสร้างช่วงระยะเวลาก่อนที่จะใช้ MCU ทำงานเพื่อรอจนกระทั่งสัญญาณนาฬิกาที่ป้อนให้กับ MCU จะนิ่งในกรณีเมื่อมีการป้อนไฟเข้าไปใหม่

POW ON Reset

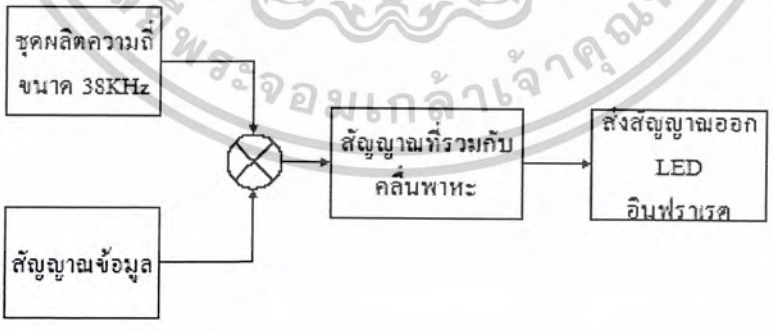
จะทำการ RESET วงจรภายใน PIC ใหม่เมื่อมีการป้อนไฟเข้าไปใหม่

Watchdog Timer

เป็น Timer สำหรับจับเวลาว่าโปรแกรมบางช่วงใน PIC ทำงานนานเกินไปหรือไม่ เพื่อป้องกันอาการที่เรียกว่า Dead Lock ซึ่ง Timer ตัวนี้จะต้องทำการ clear ด้วยคำสั่งทาง software เมื่อ timer ตัวนี้นับจนกระทั่ง time-out เกิดขึ้น PIC จะกลับไปอยู่ในสถานะเหมือนกับสภาพที่มีการป้อนไฟเข้าไปใหม่

2.4 การส่งข้อมูลผ่านอินฟราเรด

การส่งข้อมูลโดยผ่าน LED อินฟราเรดเพื่อเชื่อมต่อกับอุปกรณ์



รูปที่ 2.13 บล๊อกไดอาแกรมการทำงานของภาคส่ง

จากบล็อกไดอาแกรมที่ 2.13 การทำงานของภาคส่งจะเริ่มจากชุดผลิตความถี่ขนาด 38KHz จะทำการกำเนิดความถี่ขนาด 38KHz ขึ้นมา เพื่อทำหน้าที่เป็นสัญญาณพาหะ (Carrier) ให้กับสัญญาณข้อ

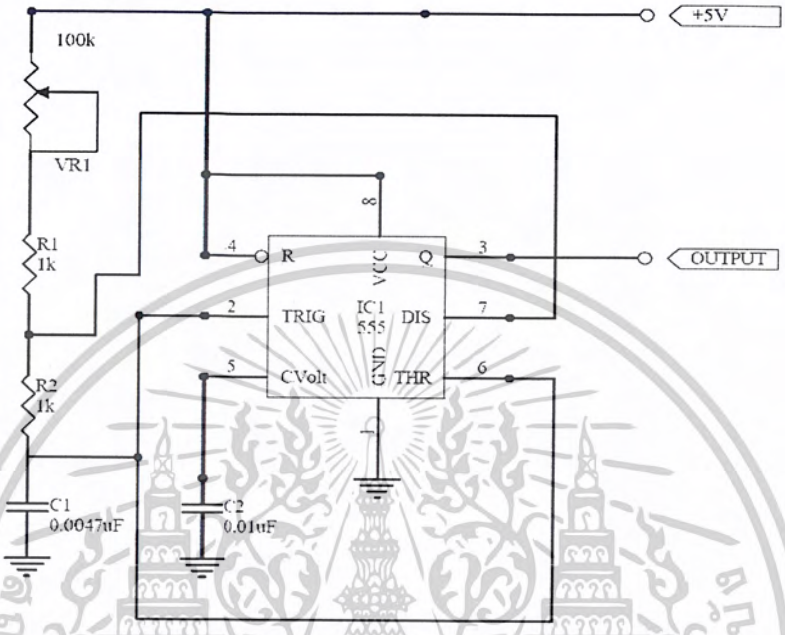
มูลที่เราจะส่งข้อมูลออกไปโดยความถี่ที่ได้นี้จะถูกส่งไปยังภาคมอดดูเลต เพื่อรอการผสมสัญญาณระหว่างสัญญาณพาหะกับสัญญาณข้อมูล สัญญาณข้อมูลที่ใช้ส่งนี้จะถูกส่งมาเป็นสัญญาณอนุกรม (Serial Data) ในชุดนี้อาจจะรับมาจากไมโครคอนโทรลเลอร์หรือจากคอมพิวเตอร์ เมื่อมีสัญญาณเข้ามาภาคมอดดูเลตจะทำการรวมสัญญาณข้อมูลกับสัญญาณความถี่พาหะ สัญญาณที่ถูกรวมแล้วจะถูกส่งไปให้กับชุดสัญญาณอินฟราเรดดังรูปที่ 2.14

รูปที่ 2.14 สัญญาณที่ถูกส่งออกจากอินฟราเรดไปยังตัวรับ

ในส่วนของภาครับ เมื่อรับสัญญาณที่ถูกส่งมาจากภาคส่งได้แล้วโดยผ่านชุด Photo Detector สัญญาณที่ได้จะถูกส่งไปเข้าภาค Demodulation เพื่อทำการกรองเอาสัญญาณ Carrier ขนาด 38KHz ออกมา สัญญาณที่ถูกกรองแล้วจะเหลือเพียงแต่สัญญาณข้อมูลเท่านั้น สัญญาณที่ได้นี้จะถูกส่งไปเข้าชุดรับ Serial Data ของไมโครคอนโทรลเลอร์ และนำข้อมูลไปใช้ตามที่เราต้องการ

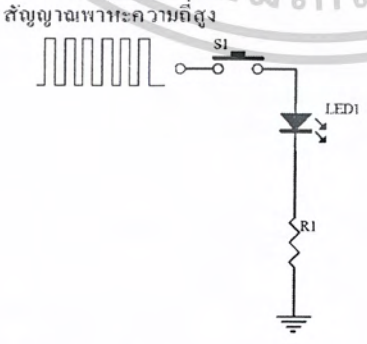
- **สัญญาณพาหะ** (Carrier)

สัญญาณพาหะเป็นสัญญาณความถี่สูง วงจรกำเนิดสัญญาณอย่างง่ายๆ โดยใช้ไอซียอดนิยมเบอร์ 555 ต่อเป็นวงจรชนิด อะสเตเบิลมัลติไวเบรเตอร์ ดังรูปที่ 2.15 การปรับค่าความถี่ของสัญญาณทำได้โดยการปรับค่าของ VR_1 ตัวอย่างนี้ ช่วงการเปลี่ยนแปลงค่าความถี่อยู่ระหว่าง 38 – 40 กิโลเฮิรตซ์ ซึ่งกว้างพอที่จะเลือกค่าความถี่สำหรับรีโมตคอนโทรลได้มากมายหลายชุด

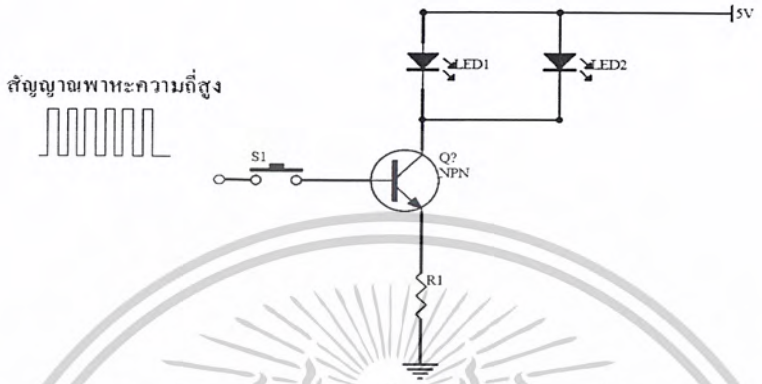


รูปที่ 2.15 วงจรกำเนิดความถี่สูงโดยใช้ ไอซี 555

สัญญาณเอาท์พุตความถี่สูงจากไอซี 555 สามารถนำไปขับ **LED** ชนิดอินฟราเรด เพื่อส่งเป็นสัญญาณควบคุมได้เลย หากต้องการส่งเพียงบิตเดียวออกได้เลย โดยต่อกับวงจรในรูปที่ 2.16 หรือหากต้องการเพิ่มกำลังส่งให้ไปได้ไกลขึ้นอาจใช้ทรานซิเตอร์เป็นตัวขับสัญญาณให้กับ **LED** 2 ตัว ดังรูปที่ 2.17



รูปที่ 2.16 วงจรส่งสัญญาณควบคุมโดยตรงกับ LED อินฟราเรต

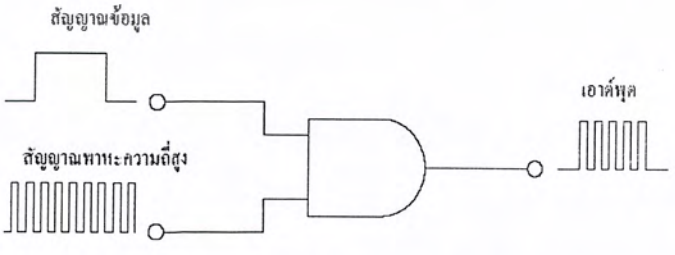


รูปที่ 2.17 วงจรส่งสัญญาณควบคุมโดยใช้ ทรานซิสเตอร์เป็นตัวขับอินฟราเรด

นอกเหนือจากนี้อาจใช้การกำเนิดสัญญาณรูปแบบอื่นนอกเหนือจากไอซี 555 ได้ เช่นอาจใช้ไมโครคอนโทรลเลอร์กำเนิดสัญญาณ

• **การมอดดูเลต**

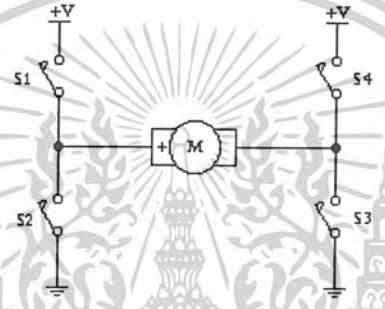
ในกรณีที่สัญญาณควบคุมเป็นลักษณะอนุกรมที่มีการจัดรหัสควบคุม ดังที่ได้กล่าวไว้แล้วในเรื่องสัญญาณอนุกรมพัลส์ การส่งสัญญาณแบบโทนเบิรตส์สามารถทำได้โดยการรวมสัญญาณอนุกรมพัลส์ความถี่สูง (จากเอาท์พุตไอซี 555) เข้าด้วยกัน ด้วยวงจรแอนด์เกตหรือวงจรแนนด์เกต ดังแสดงในในรูปที่ 2.18 แล้วส่งไปขับ **LED** อินฟราเรด



รูปที่ 2.18 วงจรผสมสัญญาณโดยใช้แอนด์เกต

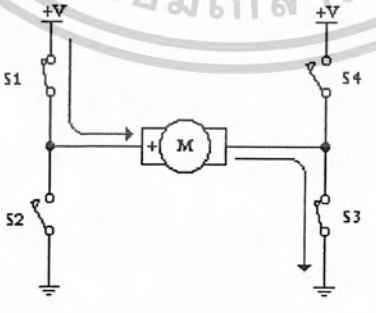
2.5 การขับมอเตอร์

เมื่อพูดถึงเรื่องของการสร้างหุ่นยนต์แล้ว คงจะหนีไม่พ้นเรื่องของ การควบคุม ทิศทางการหมุนของ DC-Motor อย่างแน่นอน เพราะควบคุมทิศทางของหุ่นยนต์จะต้องมีการกลับทิศทางการหมุนของมอเตอร์ หลักการทำงานพื้นฐานในการขับมอเตอร์แบบง่ายๆ ของวงจร H-Bridge จะเป็นหลักการของสวิทซ์ จึงเรียกวงจรแบบนี้ว่า H-Bridge Switching ดูจากรูปที่ รูปที่ 2.19



รูปที่ 2.19 วงจร H-Bridge Switching

เริ่มจากหลักการของวงจรนั้น จะประกอบไปด้วย สวิทช์ 4 ตัว นั่นก็คือ S1, S2, S3 และ S4 นั่นเอง ซึ่งในรูปตัวอย่าง จะใช้ DC-Motor เป็น Load ของวงจรในสภาวะเริ่มต้น สวิทช์ ทุกตัว Off อยู่ ก็จะไม่มีอะไรเกิดขึ้นทั้งสิ้น เพราะไม่มีกระแสไฟฟ้าไหลเข้าสู่ มอเตอร์



รูปที่ 2.20 การทำงานของวงจรในทิศทางเดินหน้า (Forward)

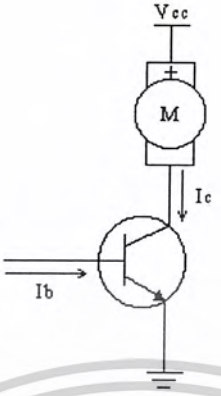
และเมื่อเราทำการ On สวิทช์ S1 และ S3 พร้อมกัน ดังรูปที่ 2.20 จะเป็นการเชื่อมวงจร ทำให้มีกระแสไฟฟ้า ไหลผ่านมอเตอร์ จากขั้วบวกของมอเตอร์ ไปยังขั้วลบของมอเตอร์ จึงทำให้มอเตอร์สามารถหมุนได้ ในทิศทางเดินหน้า (Forward) (จะหมุนแบบตามเข็มนาฬิกา หรือทวนเข็มนาฬิกานั้นขึ้นอยู่กับลักษณะของ การพันขดลวดภายในมอเตอร์)

รูปที่ 2.21 การทำงานของวงจรในทิศทางถอยหลัง (Reverse)

ในทางกลับกัน ถ้าหากเราทำการ On สวิทช์ S2 และ S4 พร้อมกัน ดังรูปที่ 2.21 ก็จะเป็นการเชื่อมวงจร และทำให้เกิดกระแสไฟฟ้า ไหลผ่านมอเตอร์ จากขั้วลบของมอเตอร์ ไปยังขั้วบวกของมอเตอร์ จึงทำให้มอเตอร์สามารถหมุนได้ และเป็นการหมุนในทิศทางถอยหลัง (Reverse) (กลับทิศทางกับกรณีแรก)

ดังนั้นวงจร H-Bridge Switching นี้จะอาศัยสวิทช์ 4 ตัว เพื่อบังคับทิศทางการไหล ของกระแสไฟฟ้า ที่ไหลผ่านมอเตอร์ เพื่อควบคุมให้มอเตอร์หมุนตามทิศทางที่เราต้องการ โดยการผลัดกัน On และ Off สวิทช์พร้อมกัน 2 ตัว

จากหลักการของวงจร H-Bridge Switching เวลาเราใช้งานจริงกลับหุ่นยนต์ในส่วนของสวิทช์พื้นฐานที่ได้กล่าวมาเราจะเปลี่ยนเป็นทรานซิสเตอร์ซึ่งทรานซิสเตอร์ เป็นอุปกรณ์สารกึ่งตัวนำ (Semiconductor device) ที่เราสามารถนำคุณสมบัติของการ Cutoff และการ Saturation มาประยุกต์ใช้งานเป็นสวิทช์ได้ และที่สำคัญ มันเป็นสวิทช์อิเล็กทรอนิกส์ ที่เราสามารถควบคุมการปิดและเปิด ได้

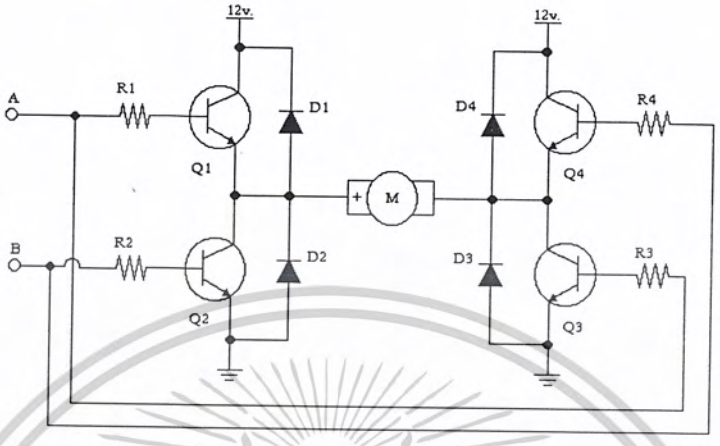


รูปที่ 2.22 การนำทรานซิสเตอร์ มาเป็นสวิทช์ควบคุมมอเตอร์

จากรูปที่2.22 เป็นวงจรสาธิตแบบง่าย ๆ โดยการนำทรานซิสเตอร์ มาเป็นสวิทช์ควบคุมมอเตอร์ หลักการคิดง่าย ๆ ก็คือ เมื่อเราป้อนกระแส Ib ด้วยปริมาณที่มากพอ ก็จะทำให้ทรานซิสเตอร์ทำงาน (On) จะทำให้กระแส Ic ไหล แปลว่ามีกระแสไหลผ่านมอเตอร์ได้ (กระแส Ib จะต้องมากเพียงพอที่จะทำให้ทรานซิสเตอร์อยู่ในสภาวะ "อิ่มตัว" ได้) ในสภาวะ อิ่มตัว (Saturation mode) นี้ ทรานซิสเตอร์จะทำงานเหมือนกับสวิทช์ปิดวงจร ค่าความต้านทานระหว่างขา C และขา E จะมีค่าเข้าใกล้ศูนย์ กระแส Ic ที่ไหลจะมีค่าเข้าใกล้ Ic(max)

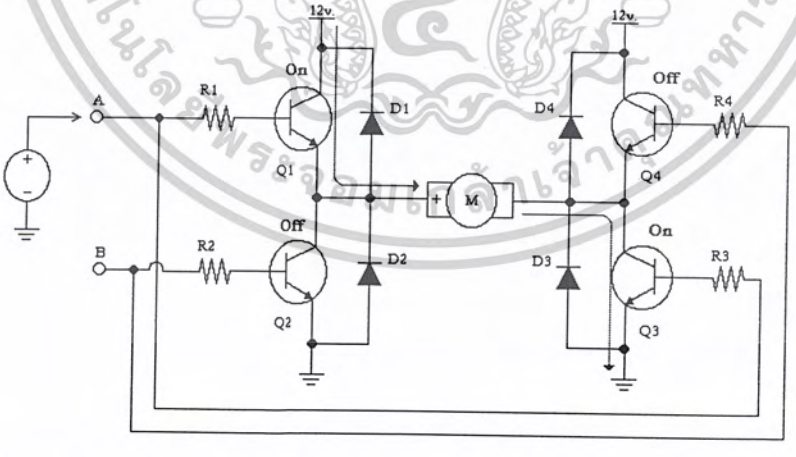
ในสภาวะ คัตออฟ (Cutoff mode) นี้ จะเกิดขึ้นเมื่อเราหยุดจ่ายกระแส Ib (Ib = 0) ทรานซิสเตอร์จะทำงานเหมือนกับสวิทช์เปิดวงจร ค่าความต้านทานระหว่างขา C และขา E จะมีค่าเป็นอนันต์ กระแส Ic จะมีค่าเข้าใกล้ศูนย์

ข้อดีของการนำทรานซิสเตอร์ มาประยุกต์ใช้งานเป็นสวิทช์นั้น น่าจะเป็นส่วนของการควบคุมที่สามารถตอบสนองจังหวะของการเปิด/ปิด สวิทช์ได้นับล้านครั้งต่อวินาที (ความเร็วในการตอบสนองมีหน่วยเป็น nS) และที่สำคัญ คือไม่ทำให้เกิดปัญหารบกวนจากสนามแม่เหล็ก อย่างวงจรรีเลย์



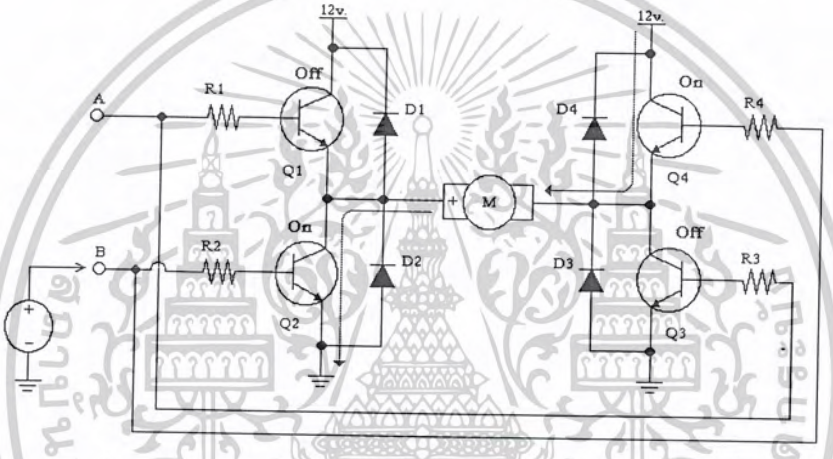
รูปที่ 2.23 วงจร H-Bridge Switching จาก Transistor

วงจร H-Bridge Switching จาก Transistor ดังรูปที่ 2.23 จะประกอบไปด้วย ทรานซิสเตอร์ Q1, Q2, Q3 และQ4 ทุกตัวใช้เบอร์ ส่วน R1, R2, R3 และR4 ทำหน้าที่จำกัดกระแส Ib ส่วนไดโอด D1, D2, D3 และD4 ทำหน้าที่ป้องกันกระแสไหลย้อนกลับจากมอเตอร์ ในขณะทรานซิสเตอร์หยุดทำงานหลัก การทำงานกรณีที่ Q1 และ Q3 ทำงานดังรูปที่ 2.24



รูปที่ 2.24 การทำงานของวงจร H-Bridge จาก Transistor ในทิศทางเดินหน้า (Forward)

เมื่อมีการจ่ายแรงดัน เข้าที่จุด A ทำให้มีกระแสไหลผ่าน R1 เข้าสู่ขา base ของ Q1 และมีกระแสไหลผ่าน R3 เข้าสู่ขา base ของ Q3 ทำให้ Q1 และ Q3 ทำงาน (On) เปรียบเสมือนสวิทช์ปิดวงจร ส่งผลให้มีกระแสไหลจากแหล่งจ่าย (12v.) ผ่านขา Collecter และEmitter ของ Q1 ผ่านเข้าสู่ขั้วบวก (+) ของมอเตอร์ ผ่านไปยังขา Collecter และEmitter ของ Q3 ทำให้มีกระแสไหลผ่านมอเตอร์ในทิศทางบวก และครบวงจร จึงทำให้มอเตอร์สามารถหมุน ในทิศทาง Forward ได้ กรณีที่ Q2 และ Q4 ทำงานดังรูปที่ 2.25



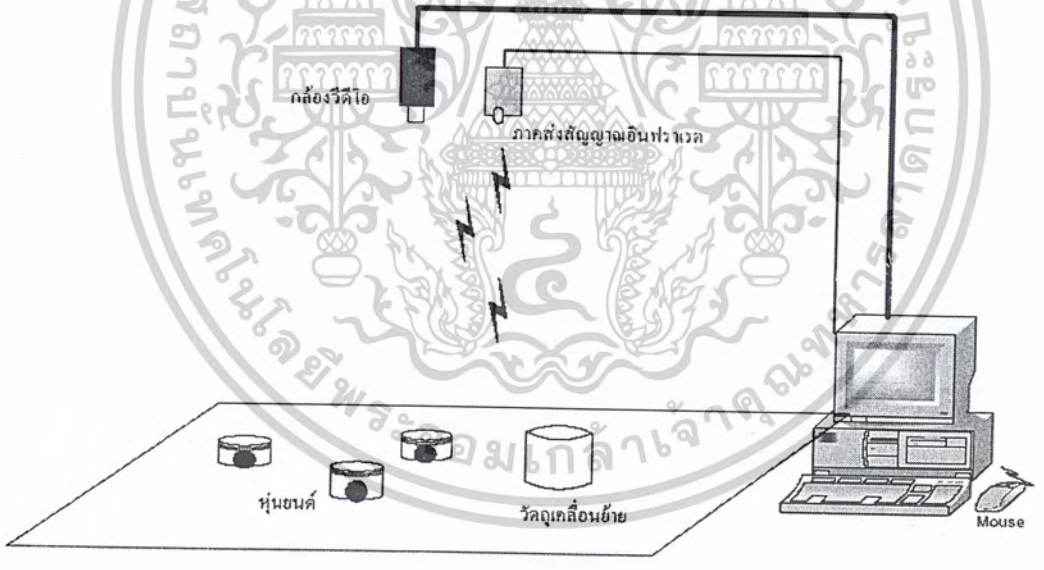
รูปที่ 2.25 การทำงานของวงจร H-Bridge จาก Transistor ในทิศทางถอยหลัง (Reverse)

เมื่อมีการจ่ายแรงดัน เข้าที่จุด B ทำให้มีกระแสไหลผ่าน R2 เข้าสู่ขา base ของ Q2 และมีกระแสไหลผ่าน R4 เข้าสู่ขา base ของ Q4 ทำให้ Q2 และ Q4 ทำงาน (On) เปรียบเสมือนสวิทช์ปิดวงจร ส่งผลให้มีกระแสไหลจากแหล่งจ่าย ผ่านขา Collecter และEmitter ของ Q4 ผ่านเข้าสู่ขั้วลบ (-) ของมอเตอร์ ผ่านไปยังขา Collecter และEmitter ของ Q2 ทำให้มีกระแสไหลผ่านมอเตอร์ในทิศทางลบ และครบวงจร จึงทำให้มอเตอร์สามารถหมุน ในทิศทาง Reward ได้

ข้อดีของวงจร H-Bridge Switching จาก Transistor คือเราสามารถที่จะควบคุมทิศทางของมอเตอร์ได้จากแหล่งจ่ายไฟเพียงแหล่งเดียวถึงแม้ว่าการสร้างวงจร H-Bridge switching จาก Transistor นั้นจะไม่มีปัญหา การรบกวนจากอำนาจสนามแม่เหล็ก และยังสามารถตอบสนองการทำงานได้เร็วมาก เร็วกว่ารีเลย์มาก

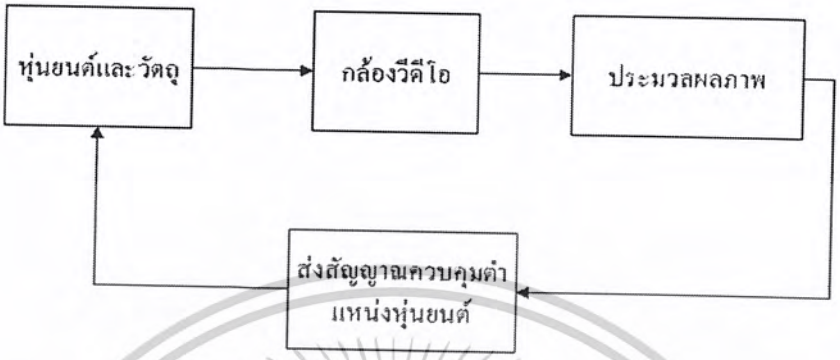
บทที่ 3
หลักการออกแบบ

ภาพของหุ่นยนต์แต่ละตัวและวัตถุถูกรับภาพเข้ามาผ่านกล้องวิดีโอ ภาพที่ได้มานั้นจะมองเห็นในมุมมองส่วนด้านบน (Top view) ของหุ่นยนต์และวัตถุ เมื่อทำการประมวลภาพ (Image Processing) โดยคอมพิวเตอร์เมื่อหาตำแหน่งพิกัดและมุมของหุ่นยนต์ได้แล้วจากนั้นนำพิกัดที่ได้มาทำการสร้างอัลกอลิทึม ในการควบคุมการเคลื่อนที่ของหุ่นยนต์ในรูปแบบต่างๆในการควบคุมพฤติกรรมมเชิงกลุ่ม เช่น การเดินเป็นกลุ่ม การเดินหน้าขนาน การเดินหน้ากระดาน และให้หุ่นยนต์ผลักสิ่งของไปยังตำแหน่งต้องการ โดยโครงสร้างโดยรวมของระบบแสดงดังรูปที่ 3.1



รูปที่ 3.1 รูปแบบโดยรวมของระบบ

จากโครงสร้างโดยรวมของระบบจะเห็นได้ถึงส่วนประกอบต่างๆของของโครงงานนี้ซึ่งมีหน้าที่แตกต่างกันไป ซึ่งถูกส่วนจะถูกเชื่อมโยงเข้าหากัน โดยมีแนวความคิดในการทำงานของระบบโดยรวมดังรูปที่ 3.2



รูปที่ 3.2 แสดงบล็อกไดอาแกรมแสดงการทำงานของระบบ

3.1 ส่วนประกอบของโครงงานนี้แบ่งออกเป็นส่วนต่างๆดังนี้

1. ส่วนประกอบทางด้านฮาร์ดแวร์ เป็นในส่วนของรูปแบบและหลักการทำงานของหุ่นยนต์ อีกส่วนหนึ่งเป็นส่วนของภาคส่งสัญญาณอินฟราเรด

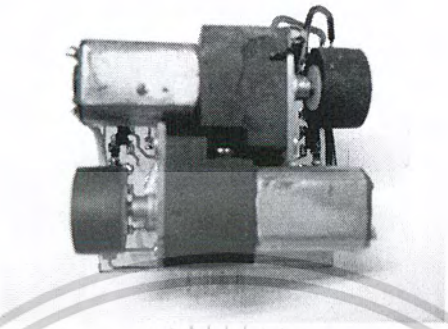
2. ส่วนประกอบทางด้านซอฟแวร์ เป็นในส่วนของการประมวลผลภาพและอัลกอลิทึมในการควบคุมหุ่นยนต์หลายตัวแบบเป็นกลุ่มในรูปแบบต่างๆ

3.2 ส่วนประกอบทางด้านฮาร์ดแวร์

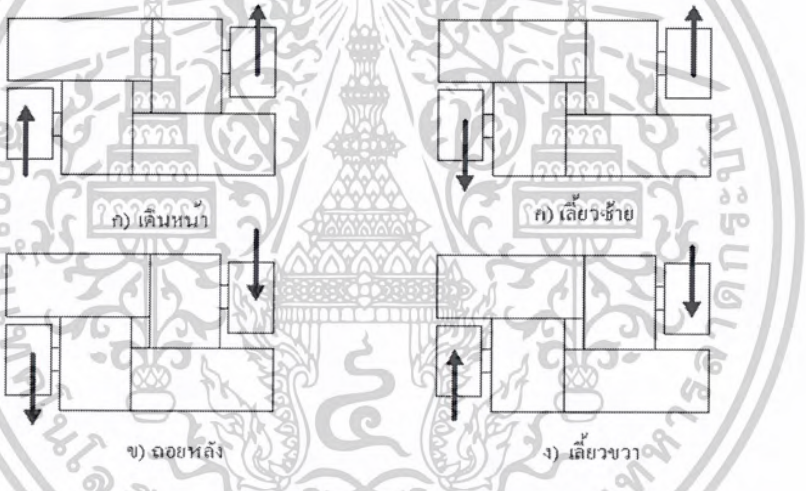
3.2.1 ส่วนของหุ่นยนต์

ส่วนของหุ่นยนต์ที่ใช้ในการทดลองนี้เป็นหุ่นยนต์ที่ออกแบบให้มีขนาดเล็ก อุปกรณ์ที่ใช้ในการออกแบบจึงเลือกใช้อุปกรณ์อิเล็กทรอนิกส์ขนาดเล็ก (SMD) ในส่วนของตัวประมวลผลเลือกใช้ไมโครคอนโทรลเลอร์ตระกูล PIC เบอร์16F84A และส่วนของตัวรับคลื่นอินฟราเรดเป็นโมดูลสำเร็จรูป ตอบสนองความถี่ช่วง 38 KHz โดยจะรับสัญญาณควบคุมจากคอมพิวเตอร์เป็นตัวสั่งงาน

ในส่วนของโครงสร้างของหุ่นยนต์ได้ออกแบบเป็นมอเตอร์ 2 ล้อต่อขนานกันเพื่อเป็นการลดขนาดของหุ่นยนต์ให้มีขนาดเล็กดังรูปที่ 3.3 ส่วนการควบคุมทิศทางการเคลื่อนที่ของมอเตอร์สามารถดูการควบคุมการเคลื่อนที่ได้ดังรูปที่ 3.4

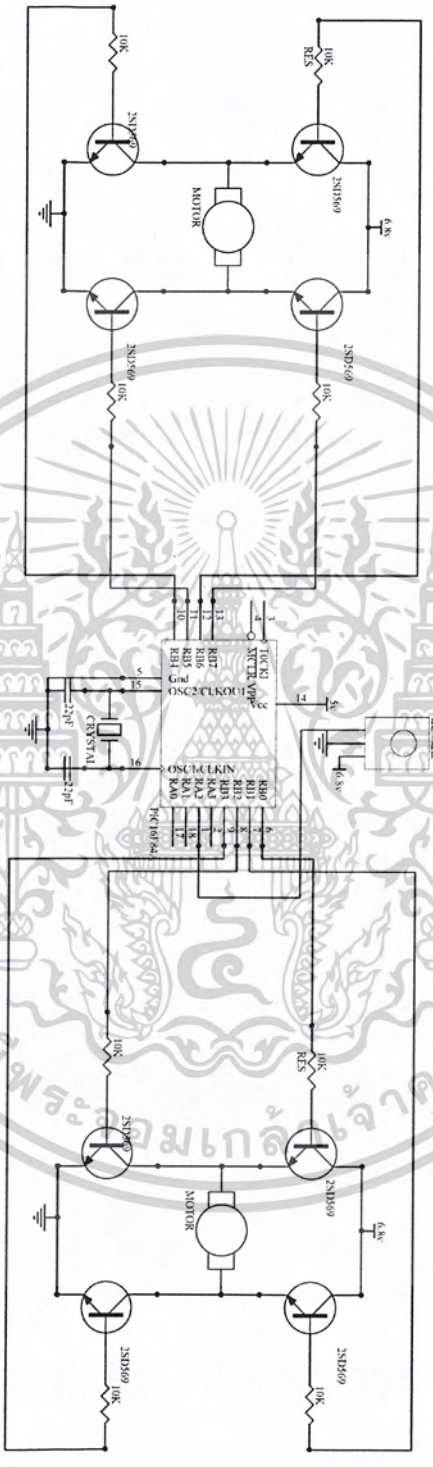


รูปที่ 3.3 แสดงการติดตั้งมอเตอร์

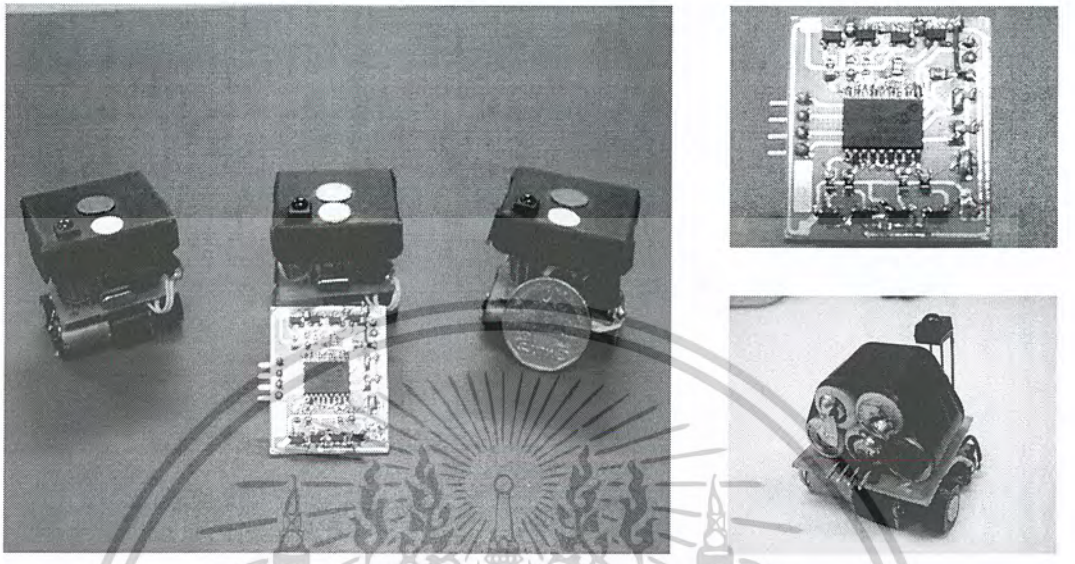


รูปที่ 3.4 การควบคุมทิศทางการเคลื่อนที่ของหุ่นยนต์

วงจรของหุ่นยนต์แสดงอยู่ในรูปที่ 3.5 การทำงานมีดังนี้คือเมื่อมีการส่งสัญญาณออกมาจากคอมพิวเตอร์มายังหุ่นยนต์ ตัวรับอินฟราเรดโมดูลจะทำหน้าที่แยกสัญญาณพาหะแล้วส่งข้อมูลอนุกรมต่อให้กับไมโครคอนโทรลเลอร์จากนั้นทำการเช็คเงื่อนไขแล้วส่งสถานะไปทริกวงจร H-Bridge ให้กลับทิศทางของมอเตอร์ให้เคลื่อนที่ตามรูปแบบคำสั่งที่ป้อนเข้ามาจากคอมพิวเตอร์รูปหุ่นยนต์ที่ได้ออกแบบเสร็จแสดงในรูปที่ 3.6



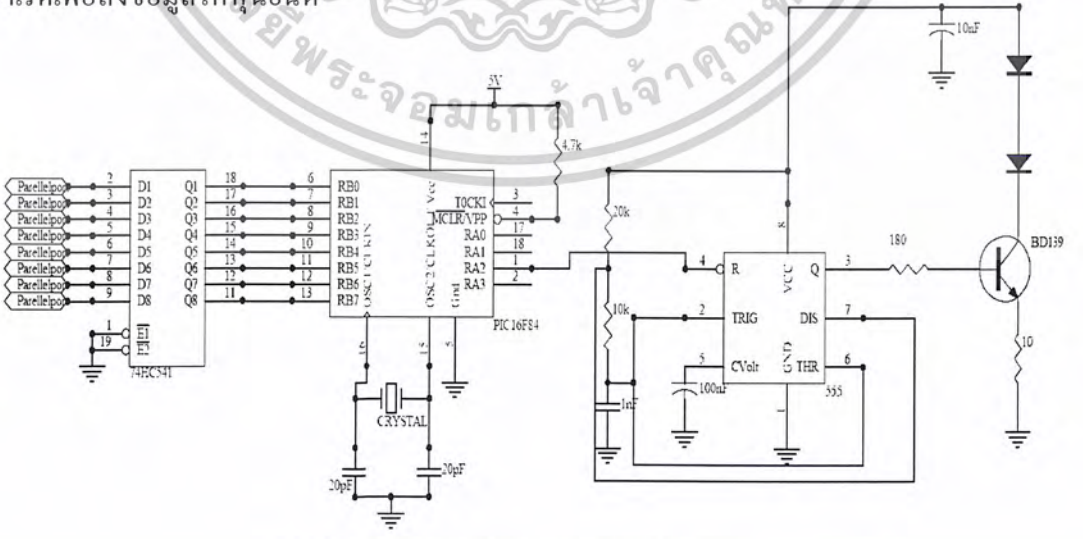
รูปที่ 3.5 วงจรของตัวหุ่นยนต์



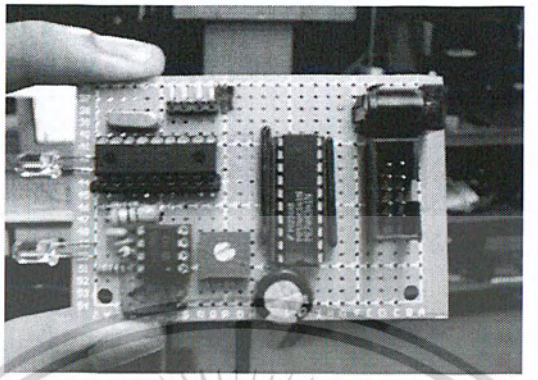
รูปที่ 3.6 แสดงรูปของหุ่นยนต์ที่ใช้ในการทดลอง

3.2.2 ส่วนของวงจรภาคส่งสัญญาณอินฟราเรด

ภาคส่งอินฟราเรตเป็นตัวรับข้อมูลจากคอมพิวเตอร์ผ่านทางพอร์ทขนานแล้วส่งเข้าตัวไมโครคอนโทรลเลอร์จากนั้นทำการแปลงเป็นสัญญาณอนุกรมส่งต่อให้กับไอซี 555 ซึ่งทำหน้าที่เอาสัญญาณอนุกรมผสมกับสัญญาณพาหะความถี่สูงที่ไอซี 555 สร้างขึ้นมา และส่งต่อให้ทรานซิสเตอร์ขับ LED อินฟราเรดเพื่อส่งข้อมูลให้หุ่นยนต์



รูปที่ 3.7 วงจรภาคส่งสัญญาณอินฟราเรด



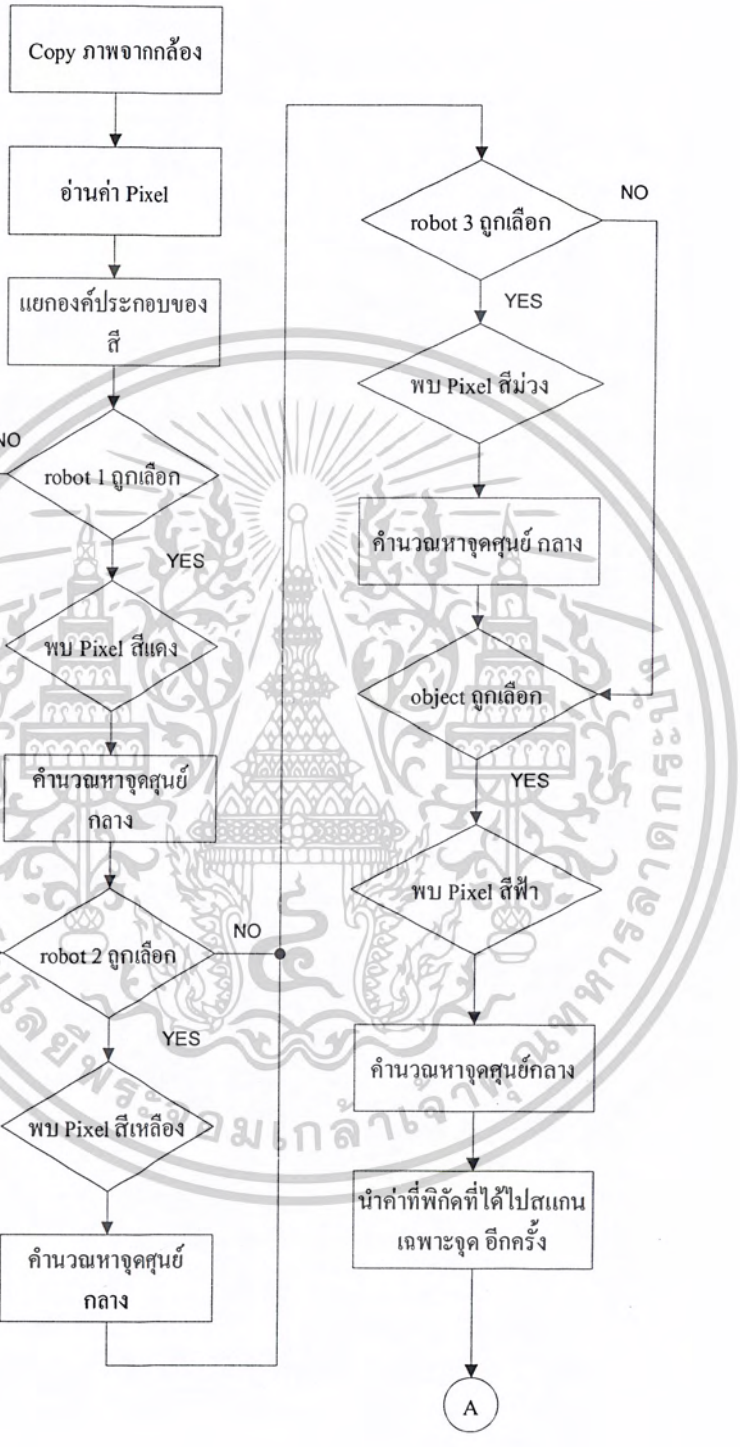
รูป 3.8 รูปวงจรภาคส่งสัญญาณอินฟราเรด

3.3 ส่วนประกอบทางด้านซอฟแวร์

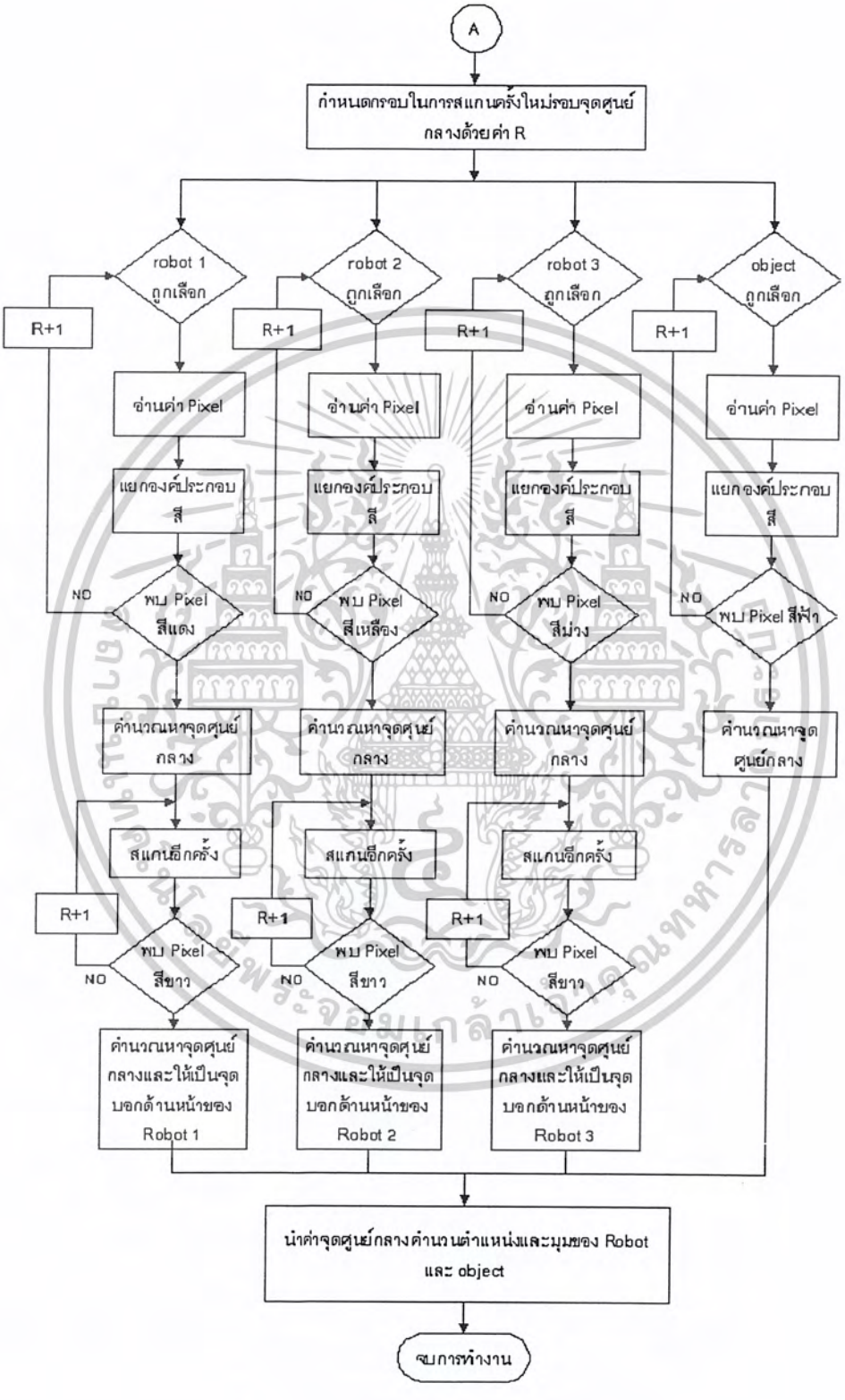
ในส่วนประกอบทางด้านด้านซอฟแวร์แบ่งออกเป็น 2 ส่วนดังนี้

3.3.1 ส่วนประมวลภาพ

ส่วนนี่จะทำหน้าที่รับภาพจากกล้องวีดิโอเข้ามาจากนั้นทำการคำนาณหาพิกัดต่างๆของ หุ่นยนต์แต่ละตัวและวัตถุซึ่งค่าตัวแปลเหล่านี้จะถูกส่งให้ส่วนควบคุมต่อไป หลักการทำงานในส่วนนี้ จะถูกแสดงแผนผัง (Flow chart) ดังนี้

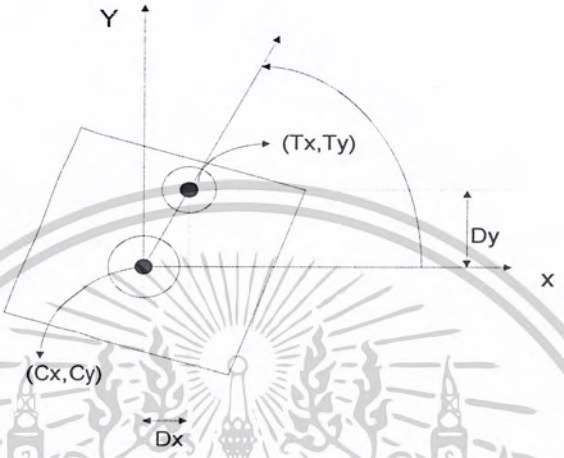


รูปที่ 3.9 แผนผังแสดงการทำงานประมวลภาพ



รูปที่ 3.10 แผนผังแสดงการทำงานประมวลภาพ (ต่อ)

การคำนวณหาค่ามุมและระยะทางของหุ่นยนต์สามารถคำนวณได้จาก



รูปที่ 3.11 แสดงการคำนวณหาค่ามุมของหุ่นยนต์

$Dx=Tx-Cx$

$Dy=Ty-Cx$

$\theta r = \tan^{-1}(Dy/Dx)$

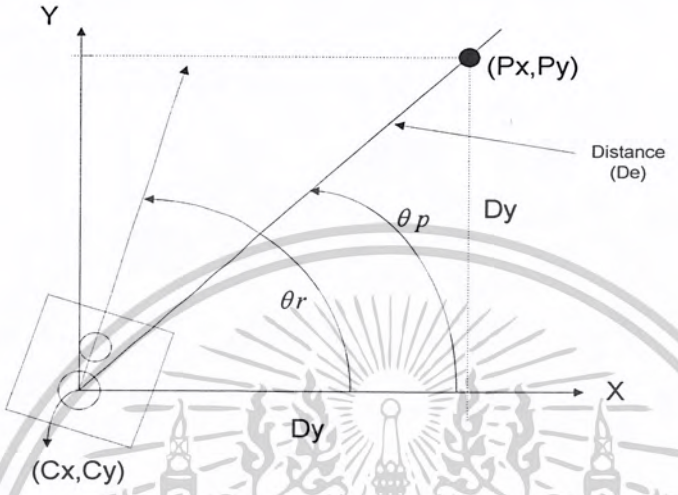
ค่า θr เป็นมุมที่หาได้มีค่ามุมตั้งแต่ 0-90° ซึ่งมีค่าเพียง Quadrant เดียวเท่านั้น จึงต้องมีการคำนวนหาค่ามุมของหุ่นยนต์ เพื่อให้ได้มุมครบตั้งแต่ 0-360° โดยการเช็คเงื่อนไข Dx และ Dy

การหาค่ามุม 0 - 360° ของหุ่นยนต์

จากสมการด้านบนนำค่ามุมที่หาได้มาทำการเช็คเงื่อนไขดังนี้

1. ถ้า $Dx \geq 0$ และ $Dy \leq 0$ ดังนั้นให้ $\theta r = \theta r \times (-1)$
2. ถ้า $Dx \leq 0$ และ $Dy \leq 0$ ดังนั้นให้ $\theta r = (90° - \theta r) + 90°$
3. ถ้า $Dx \leq 0$ และ $Dy \geq 0$ ดังนั้นให้ $\theta r = (\theta r \times (-1)) + 180°$
4. ถ้า $Dx \geq 0$ และ $Dy \geq 0$ ดังนั้นให้ $\theta r = 360° - \theta r$

จากสมการที่ผ่านมาเราสามารถที่จะหาค่ามุมและระยะทางระหว่างจุดที่พิกัดที่เราต้องการได้จาก



รูปที่ 3.12 แสดงการคำนวณหาค่ามุมระยะทางของหุ่นยนต์กับจุดอ้างอิง

$Dx=Px-Cx$

$Dy=Py-Cx$

$\theta p = \tan^{-1}(Dy/Dx)$

$De=\sqrt{Dx^2+Dy^2}$

ส่วนค่ามุม θp ที่หาได้ก็นำไปเทียบเงื่อนไขเหมือนกับการหามุมของหุ่นยนต์เพื่อให้ได้ค่ามุม 360° จากนั้น เมื่อเราได้ค่ามุมของหุ่นยนต์และมุมกับระยะทางของจุดที่เราต้องการเราสามารถนำค่ามุมทั้งสองมาเปรียบเทียบกันแล้วทำให้เราสามารถสร้างเงื่อนไขในการเคลื่อนที่ของหุ่นยนต์ได้ดังนี้คือ

1.เมื่อค่า De มีค่าที่น้อยกว่าค่า ระยะทางที่ยอมให้หุ่นยนต์หยุดได้ (Ds) ในที่นี้เราให้ค่า Ds มีค่า 10 pixel ถ้ายังมีค่ามากกว่าทำการเช็คเงื่อนไขข้อ 2 และข้อ 3 ต่อไป

2. ถ้าค่า θp มากกว่า θr

2.1 เมื่อ $\theta p - \theta r$ แล้วน้อยกว่า 8° ให้หุ่นยนต์เคลื่อนที่ไปข้างหน้า

2.2 เมื่อ $\theta p - \theta r$ แล้วน้อยกว่า 90° ให้หุ่นยนต์เลี้ยวซ้ายอย่างช้าๆ

2.3 เมื่อ $\theta p - \theta r$ แล้วน้อยกว่า 180° ให้หุ่นยนต์เลี้ยวซ้ายอย่างรวดเร็ว

2.4 เมื่อ $\theta_p - \theta_r$ แล้วน้อยกว่า 270° ให้หุ่นยนต์เลี้ยวขวาอย่างรวดเร็ว

2.5 เมื่อ $\theta_p - \theta_r$ แล้วมากกว่า 270° ให้หุ่นยนต์เลี้ยวขวาอย่างช้าๆ

2.6 เมื่อ $\theta_p - \theta_r$ แล้วมากกว่า 352° ให้หุ่นยนต์เคลื่อนที่ไปข้างหน้า

3. ถ้าค่า θ_p น้อยกว่า θ_r

3.1 เมื่อค่าสัมประสิทธิ์ของ$(\theta_p - \theta_r)$ มีค่าน้อยกว่า 8° ให้หุ่นยนต์เคลื่อนที่ไปข้างหน้า

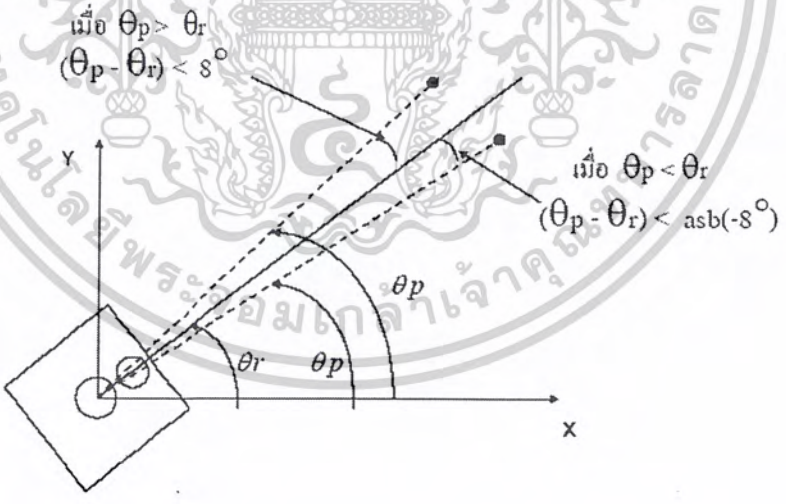
3.2 เมื่อค่าสัมประสิทธิ์ของ$(\theta_p - \theta_r)$ มีค่าน้อยกว่า 90° ให้หุ่นยนต์เลี้ยวขวาอย่างช้าๆ

3.3 เมื่อค่าสัมประสิทธิ์ของ$(\theta_p - \theta_r)$ มีค่าน้อยกว่า 180° ให้หุ่นยนต์เลี้ยวขวาอย่างเร็ว

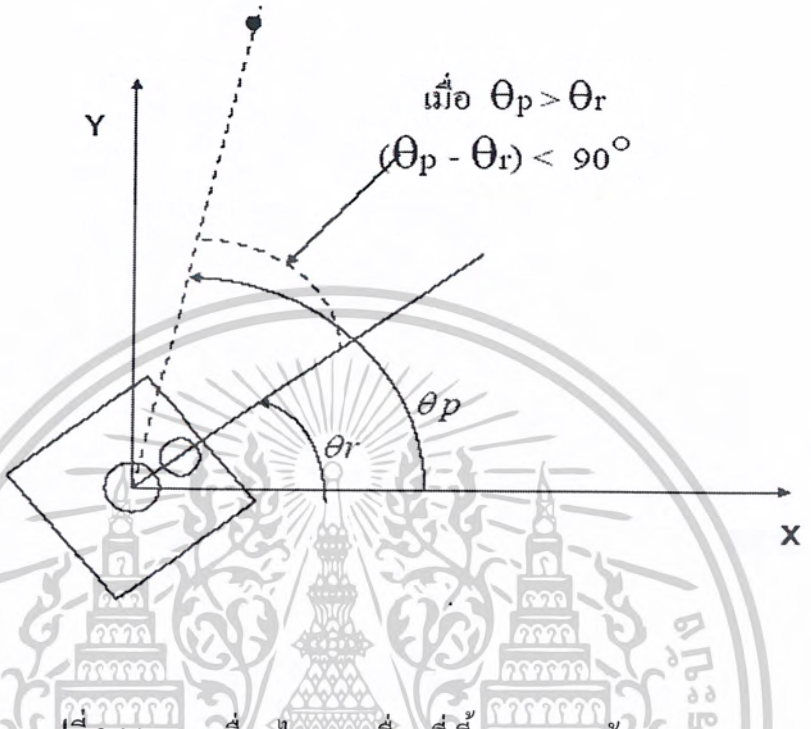
3.4 เมื่อค่าสัมประสิทธิ์ของ$(\theta_p - \theta_r)$ แล้วน้อยกว่า 270° ให้หุ่นยนต์เลี้ยวซ้ายอย่างช้าๆ

3.5 เมื่อค่าสัมประสิทธิ์ของ$(\theta_p - \theta_r)$ แล้วมากกว่า 270° ให้หุ่นยนต์เลี้ยวซ้ายอย่างเร็ว

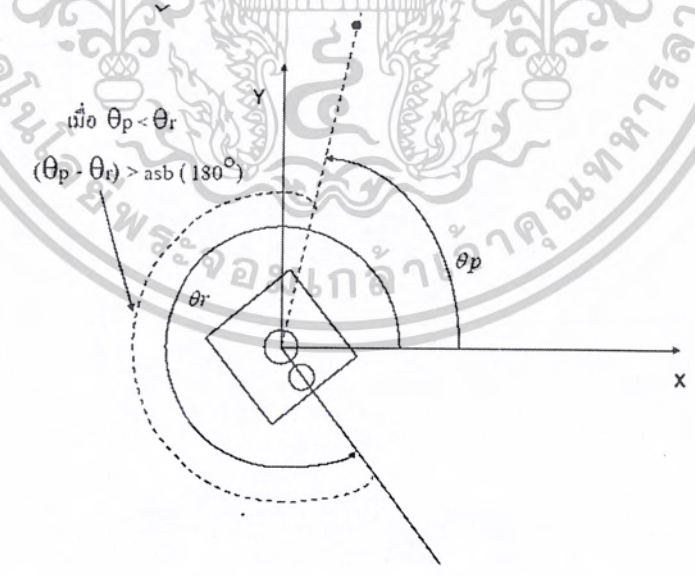
3.6 เมื่อค่าสัมประสิทธิ์ของ$(\theta_p - \theta_r)$ มีค่ามากกว่า 356° ให้หุ่นยนต์เดินที่ข้างหน้า



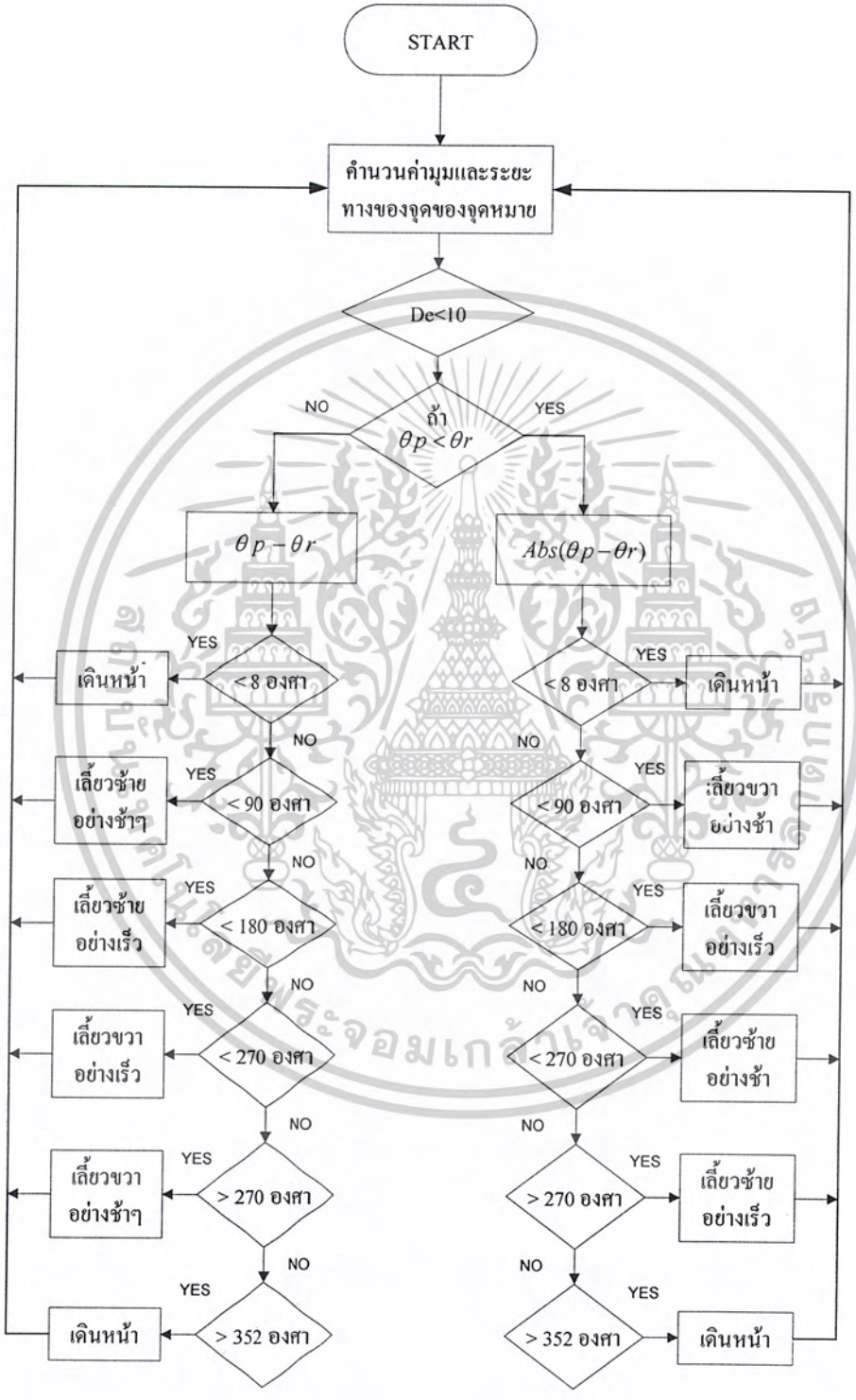
รูปที่ 3.13 แสดงเงื่อนไขการเคลื่อนที่แบบเดินหน้า



รูปที่ 3.14 แสดงเงื่อนไขการเคลื่อนที่เลี้ยวขวาแบบช้า



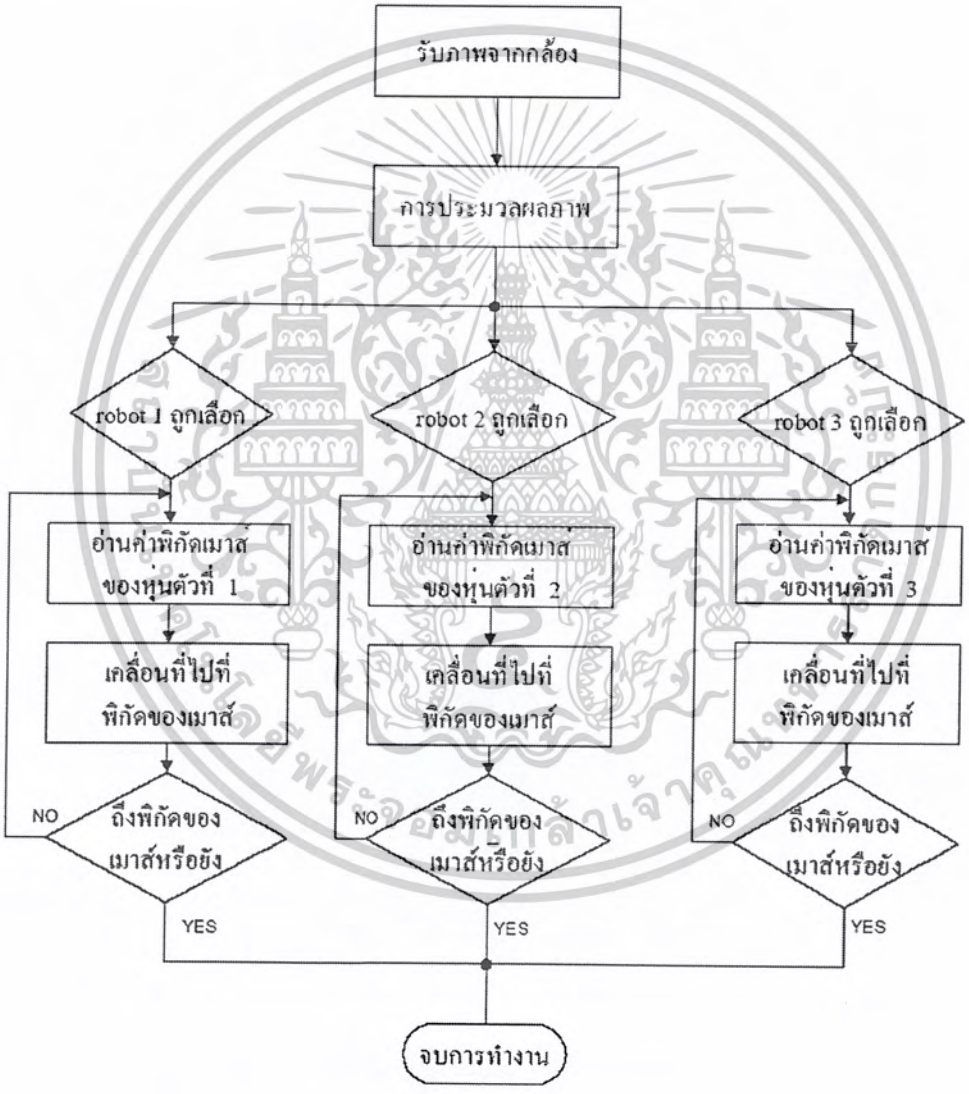
รูปที่ 3.15 แสดงเงื่อนไขการเคลื่อนที่เลี้ยวซ้ายแบบเร็ว



รูปที่ 3.16 แสดงแผนผังการควบคุมการเคลื่อนที่ไปยังพิกัดอ้างอิง

3.3.2 การควบคุมแบบอิสระ

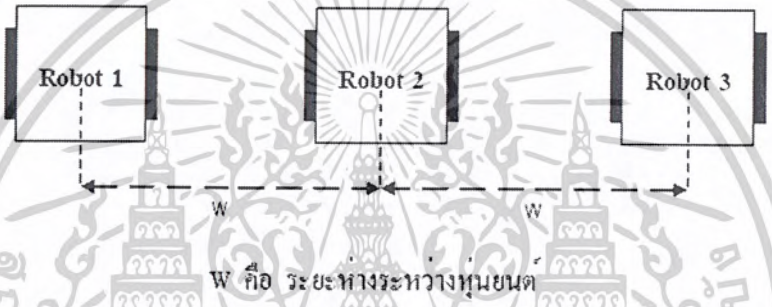
ในการควบคุมแบบอิสระ ในส่วนแรกจะใช้อัลกอลิทึมการประมวลภาพและใช้การควบคุมการเคลื่อนที่ไปยังจุดอ้างอิงรูปแบบเดิม เพียงการเปลี่ยนค่าตัวแปล 2 ตัว คือ พิกัดของหุ่นยนต์และพิกัดของจุดอ้างอิง โดยจุดอ้างอิงนี้เปลี่ยนเป็นตำแหน่งเมาส์ในการชี้ตำแหน่งอ้างอิงให้กับหุ่นยนต์แต่ละตัว การทำงานของโปรแกรมแสดงในแผนผังในรูปที่ 3.17



รูปที่ 3.17 แสดงแผนผังการควบคุมแบบอิสระ

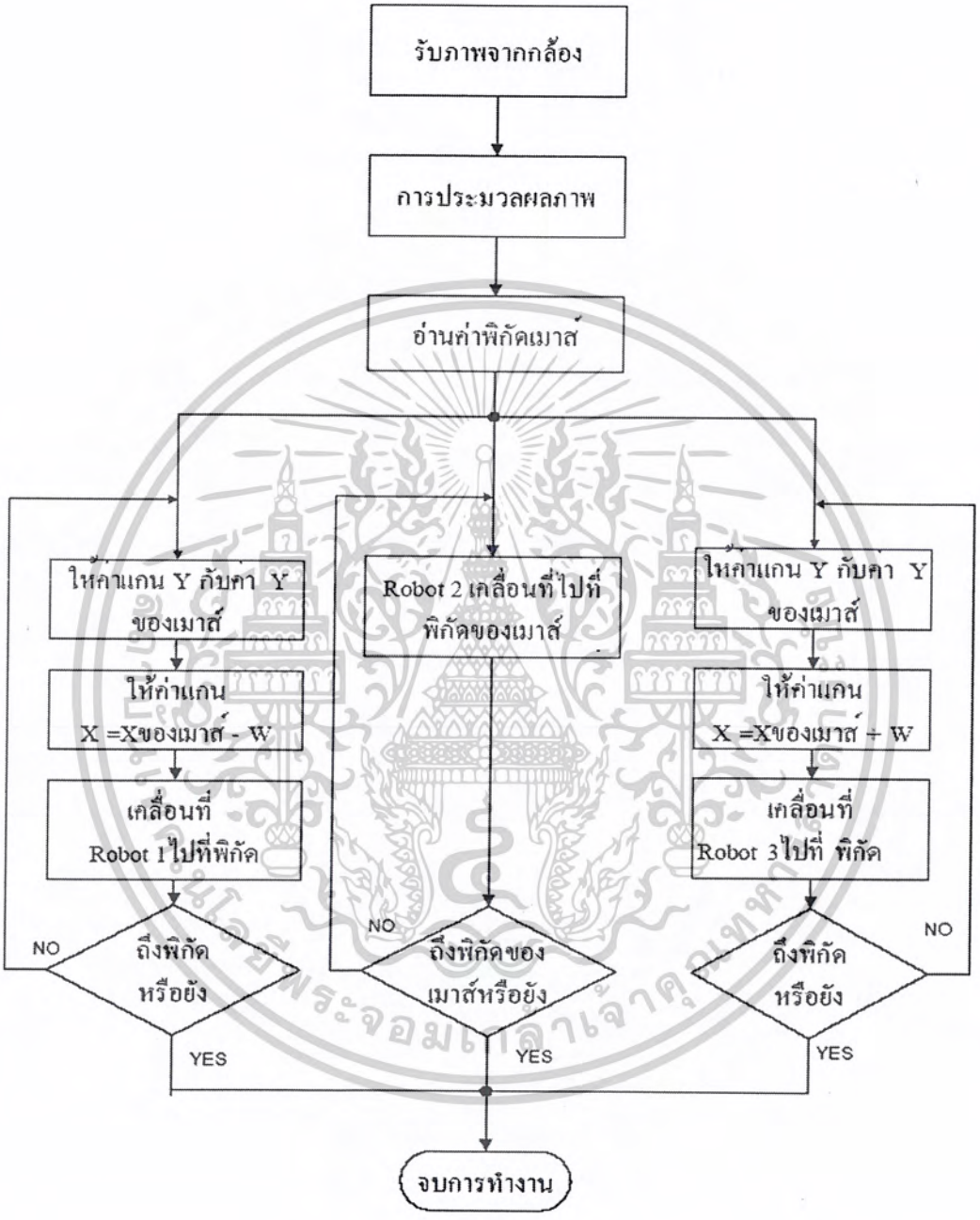
3.3.3 การควบคุมแบบหน้ากระดาน

ในการควบคุมแบบหน้ากระดาน ในส่วนแรกจะใช้ อัลกอลิทึมการประมวลภาพและใช้ การควบคุมการเคลื่อนที่ไปยังจุดอ้างอิงตัวเดิม เพียงการเปลี่ยนค่าตัวแปล 2 ตัว คือ พิกัดของหุ่นยนต์และพิกัดของจุดอ้างอิง โดยจุดอ้างอิงนี้เปลี่ยนเป็นตำแหน่งเมาส์ในการชี้ตำแหน่งอ้างอิงให้กับหุ่นยนต์ตัวที่สอง และใช้หุ่นยนต์ตัวที่หนึ่งและสามอ้างอิงตำแหน่งกับหุ่นยนต์ตัวที่สองดังรูปที่ 3.18



รูปที่ 3.18 แสดงตำแหน่งอ้างอิงระหว่างหุ่นยนต์แต่ละตัวในรูปแบบหน้ากระดาน

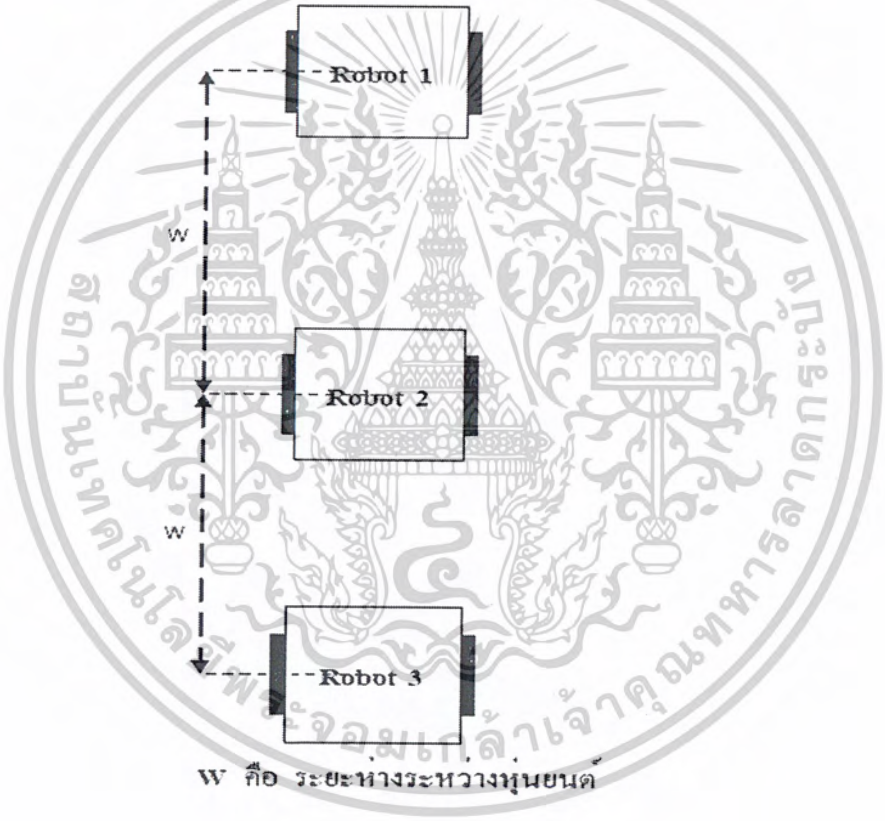
วิธีการข้างต้นจะให้หุ่นยนต์ตัวที่สองเป็นตัวหลักในการเคลื่อนที่อ้างอิงกับตำแหน่งพิกัดของเมาส์ส่วนหุ่นตัวทีเหลือจะเคลื่อนที่ตามหุ่นยนต์ตัวที่สองโดยการรักษาระนาบพิกัดและระยะห่างระหว่างกันทำให้หุ่นยนต์สามารถที่จะเคลื่อนที่ไปพร้อมกันตามรูปแบบที่เราสามารถกำหนดได้



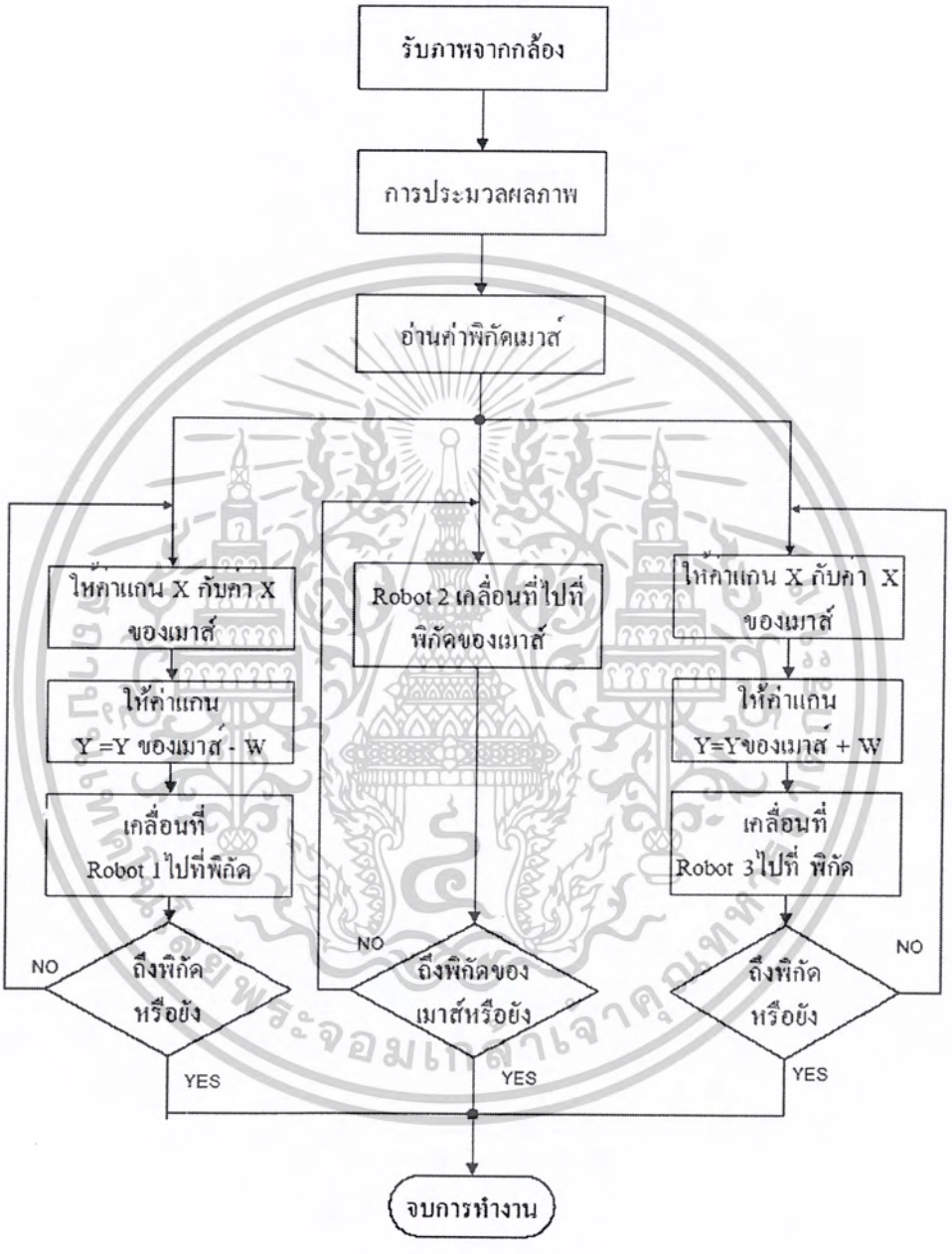
รูปที่ 3.19 แสดงแผนผังตำแหน่งอ้างอิงระหว่างหุ่นยนต์แต่ละตัวในรูปแบบหน้ากระดาน

3.3.4 การควบคุมแบบหน้าตอน

ในการควบคุมแบบหน้ากระดาน ในส่วนแรกจะใช้ อัลกอลิทึมการประมวลภาพและใช้ การควบคุมการเคลื่อนที่ไปยังจุดอ้างอิงตัวเดิม เพียงการเปลี่ยนค่าตัวแปล 2 ตัว คือ พิกัดของหุ่นยนต์และพิกัดของจุดอ้างอิง โดยจุดอ้างอิงนี้เปลี่ยนเป็นตำแหน่งเมาส์ในการชี้ตำแหน่งอ้างอิงให้กับหุ่นยนต์ตัวที่หนึ่ง และใช้หุ่นยนต์ตัวที่สองและสามอ้างอิงตำแหน่งกับหุ่นยนต์ตัวที่สองดังรูปที่ 3.20



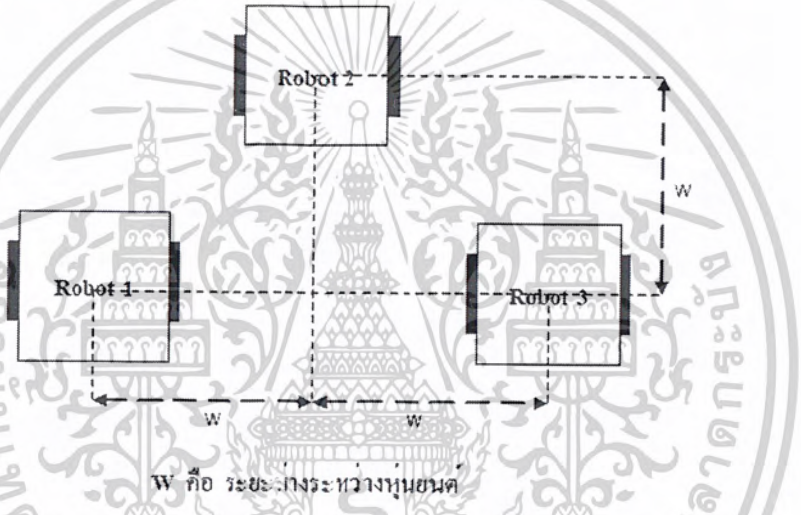
รูปที่ 3.20 แสดงตำแหน่งอ้างอิงระหว่างหุ่นยนต์แต่ละตัวในรูปแบบแถวตอน



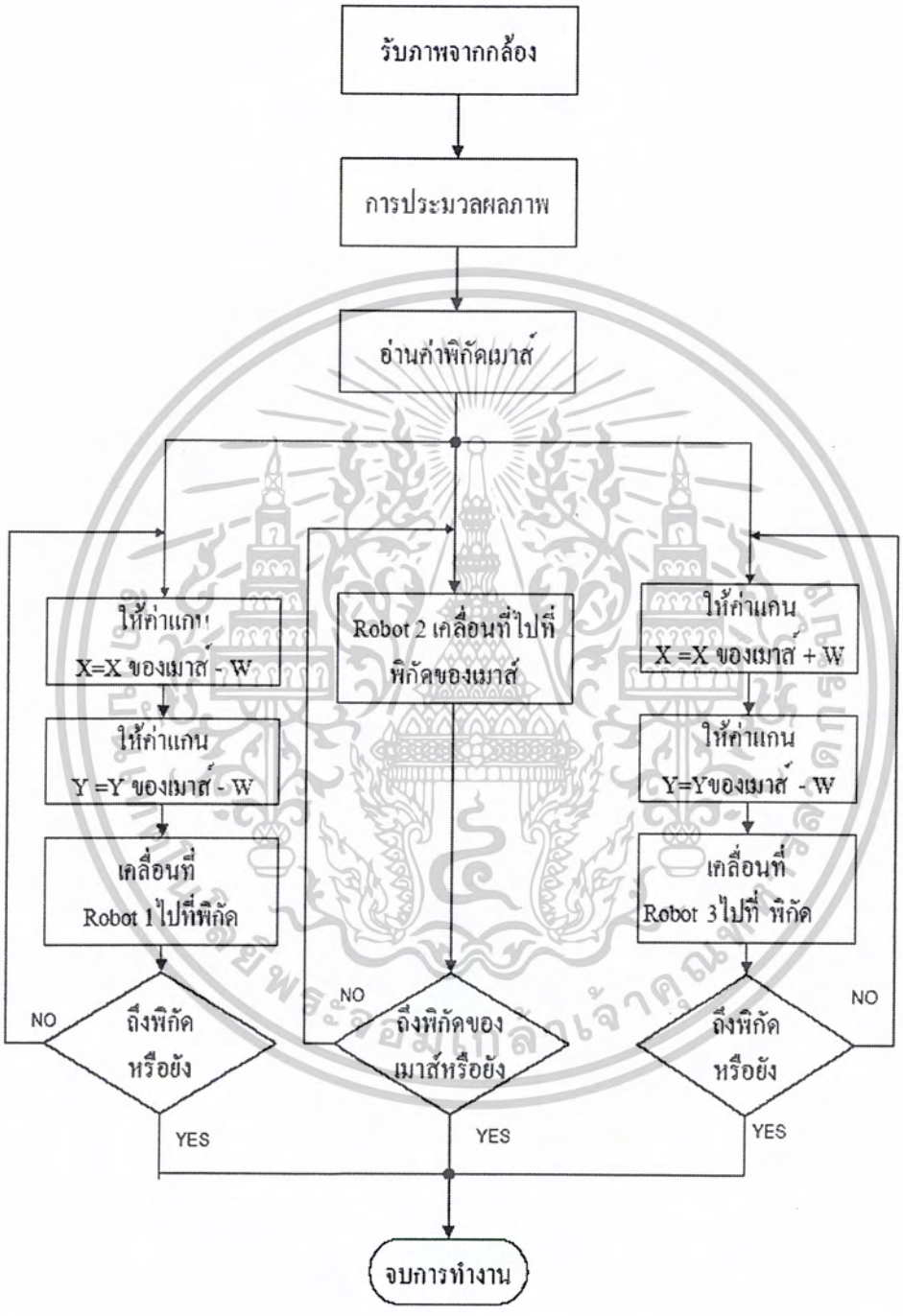
รูปที่ 3.21 แสดงแผนผังตำแหน่งอ้างอิงระหว่างหุ่นยนต์แต่ละตัวในรูปแบบแถวตอน

3.3.5 การควบคุมแบบเป็นกลุ่ม

ในการควบคุมแบบหน้ากระดาน ในส่วนแรกจะใช้ อัลกอลิทึมการประมวลภาพและใช้ การควบคุมการเคลื่อนที่ไปยังจุดอ้างอิงตัวเดิม เพียงการเปลี่ยนค่าตัวแปล 2 ตัว คือ พิกัดของหุ่นยนต์และพิกัดของจุดอ้างอิง โดยใช้ตำแหน่งเมาส์ในการชี้ตำแหน่งอ้างอิงให้กับหุ่นยนต์ตัวที่สอง และใช้หุ่นยนต์ตัวที่หนึ่งและสามอ้างอิงตำแหน่งกับหุ่นยนต์ตัวที่สองดังรูปที่ 3.22



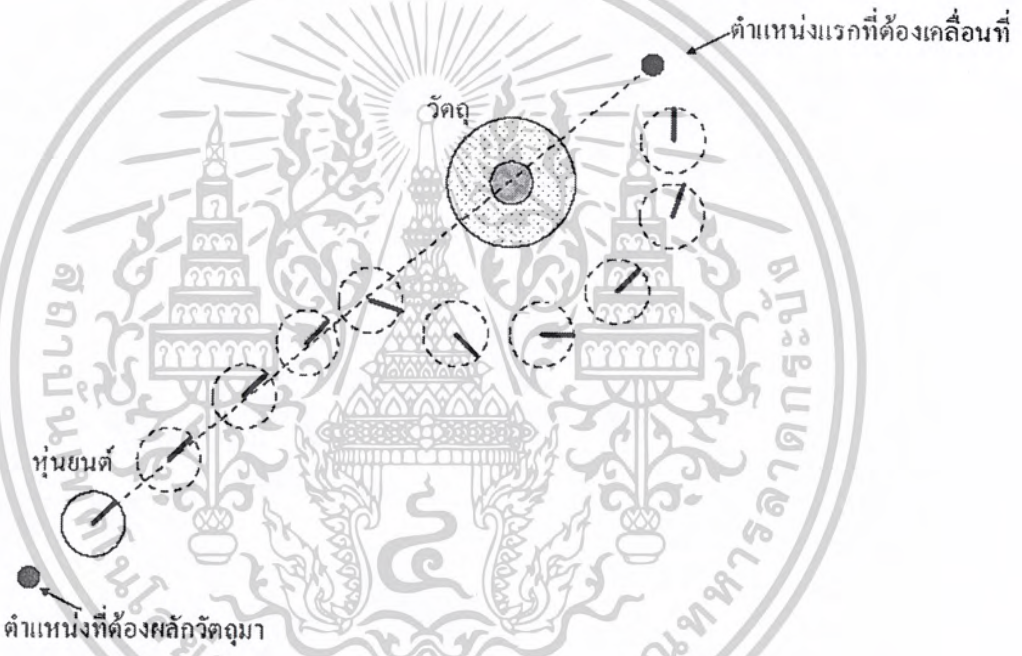
รูปที่ 3.22 แสดงตำแหน่งอ้างอิงระหว่างหุ่นยนต์แต่ละตัวในรูปแบบเป็นกลุ่ม



รูปที่ 3.23 แสดงแผนผังตำแหน่งอ้างอิงระหว่างหุ่นยนต์แต่ละตัวในรูปแบบเป็นกลุ่ม

3.3.6 การเคลื่อนย้ายวัตถุ

หลักการของการเคลื่อนย้ายวัตถุคือเมื่อประมวลผลได้แล้วจากนั้นทำการคำนวณหาตำแหน่งที่จะต้องเคลื่อนที่ไปก่อนซึ่งตำแหน่งใหม่นี้จะทำมุมตรงกลับตำแหน่งที่จะทำการเคลื่อนย้ายวัตถุทำให้สามารถเคลื่อนย้ายวัตถุได้อย่างง่าย ถ้าวัตถุเริ่มเกิดการเบี่ยงเบนออกจากทางก็ทำการชดเชยมุมจากหุ่นยนต์เพื่อให้กลับสู่เส้นทางเดิม ดังตัวอย่างในรูปที่ 3.24



รูปที่ 3.24 รูปแสดงการเคลื่อนที่ไปหาตำแหน่งใหม่ที่เหมาะสมในการเคลื่อนย้ายวัตถุ

- ขั้นแรกต้องคำนวนมุมจากจุดสูญกลางของวัตถุกับตำแหน่งที่ต้องการเคลื่อนย้ายวัตถุ
- ขั้นที่สองทำการเช็คเงื่อนไขมุม

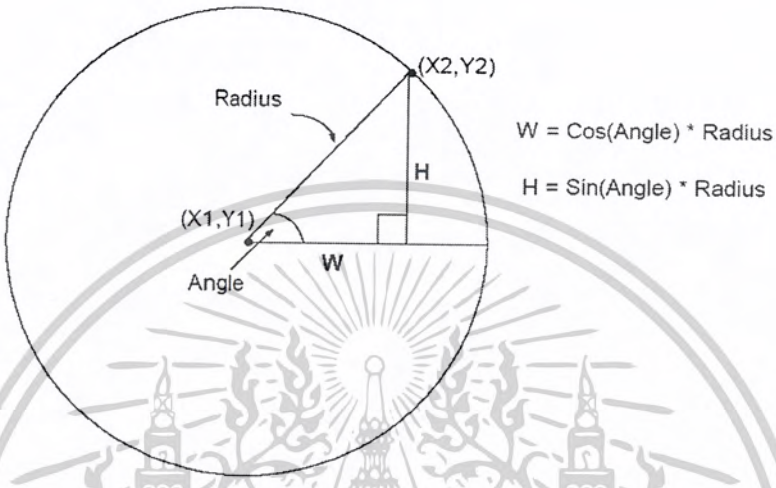
1.ถ้ามุม >= 0 และ มุม<= 180 ดังนั้นให้

มุม= มุม + 180

2.ถ้ามุม> 180 และ มุม<= 360 ดังนั้นให้

มุม = มุม – 180

• จากนั้นทำการนำค่าพิกัดใหม่จากค่ามุมโดยต้องหาตัวแปล W ,H จากหลักการคณิตศาสตร์ จากรูปที่ 3.25



รูปที่ 3.25 แสดงการหาพิกัด

W = cos (มุมใหม่ * 3.1428) * (รัศมี)

H = sin (มุมใหม่ * 3.1428) * (รัศมี)

ถ้าเพิ่มค่ารัศมีจะทำให้จุดพิกัดอยู่ห่างออกไปตามค่าของรัศมีด้วย

• ทำการเช็คเงื่อนไขมุมอีกครั้งหนึ่งเพื่อหาพิกัด

1. ถ้า Angle >= 0 และ Angle <= 90 ดังนั้นให้

   X = แกน X ของวัตถุ + W

   Y = แกน Y ของวัตถุ - H

2. ถ้าAngle >= 91 และ Angle <= 180 ดังนั้นให้

   X= แกน X ของวัตถุ - W

   Y = แกน Y ของวัตถุ - H

3. ถ้าAngle >= 181 และ Angle <= 270 ดังนั้นให้

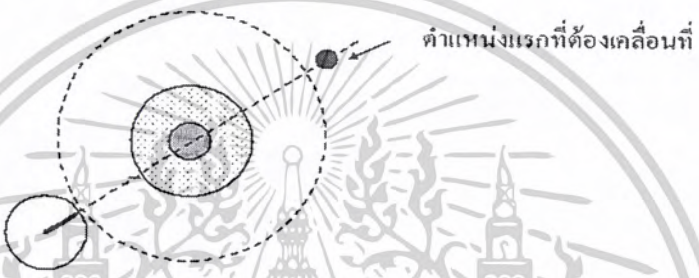
   X = แกน X ของวัตถุ - W

   Y = แกน Y ของวัตถุ + H

4. Angle >= 271 และ Angle <= 360 ดังนั้นให้

X = แกน X ของวัตถุ + W

Y = แกน Y ของวัตถุ + H

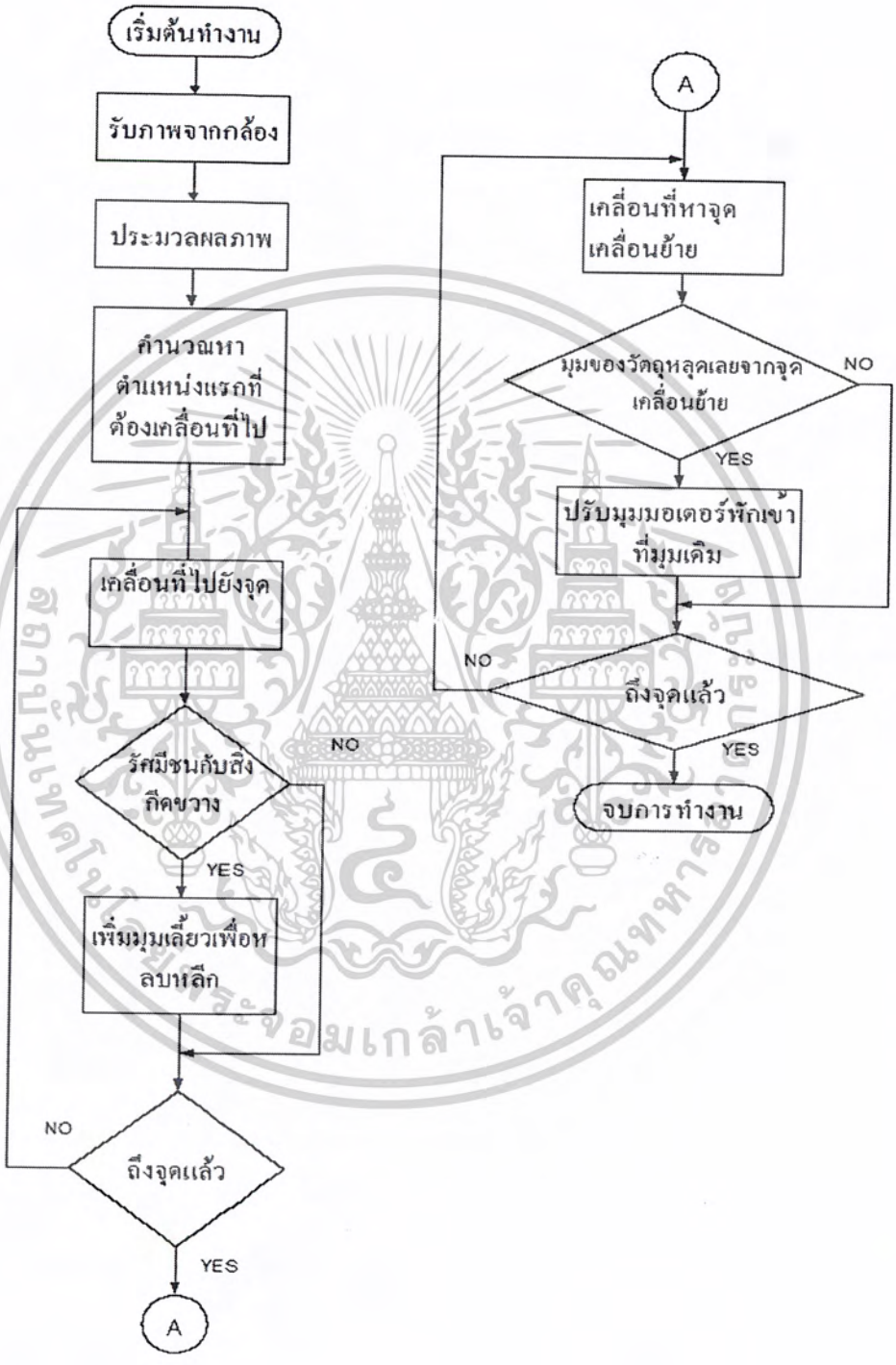


รูปที่ 3.26 แสดงการกำหนดรัศมีเพื่อให้หุ่นยนต์หลบหลีกสิ่งกีดขวาง

ในบางครั้งการเคลื่อนที่ไปยังตำแหน่งที่จะทำการเคลื่อนย้ายอาจเจอวัตถุที่เราต้องการเคลื่อนย้ายนั้นเป็นสิ่งกีดขวางเองดังนั้น จากรูปที่ 3.26 วิธีที่ใช้ในการหลบหลีกสิ่งกีดขวางจึงใช้วิธีของการเช็ครัศมี ดังนั้นถ้าเราทำการคำนวณระยะทางระหว่างตัวหุ่นยนต์กับตัววัตถุ ก็จะทำให้เราทราบระยะทางที่เราต้องทำการหลบหลีกได้ คือถ้าหุ่นยนต์เข้ามาถึงระยะรัศมีที่ถูกกำหนดก็ให้ทำการเลี้ยวไปทางใดทางหนึ่ง หุ่นยนต์ก็จะทำการวิ่งตามรัศมีได้โดยไม่โดยกับวัตถุ จนมาถึงตำแหน่งที่ต้องได้

จากนั้นพอมาถึงจุดแล้วทำการเคลื่อนที่โดยการรักษาระดับของมุมให้ตรงกับจุดหมายที่ต้องการให้เคลื่อนที่ไป และถ้าวัตถุเริ่มเบี่ยงเบนออกไปก็ทำการชดเชยมุมของหุ่นยนต์ตาม จนกว่าจะถึงจุดที่ต้องการให้หยุด

ตามที่กล่าวมานี้สามารถแสดงการทำงานในรูปแบบของ Flow chart ดังรูปที่ 3.27

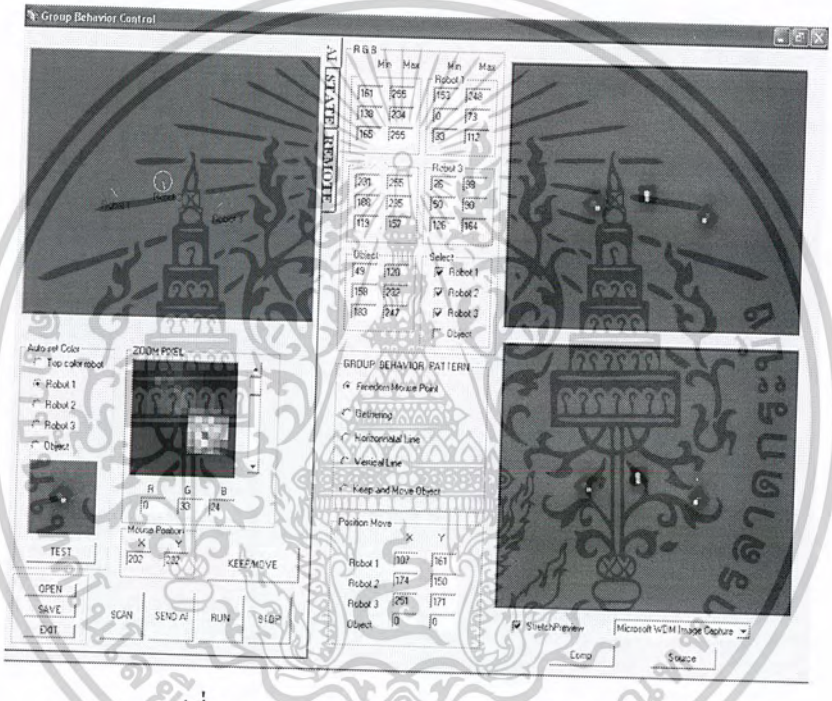


รูปที่ 3.27 แสดงแผนผังการทำงานของการเคลื่อนย้ายวัตถุ

บทที่4

การทดลองและผลการทดลอง

เมื่อเราได้ทำการติดตั้งฮาร์ดแวร์ และทำการเชื่อมต่อคอมพิวเตอร์จากนั้นทำการการรันซอร์ฟแวร์ที่เขียนขึ้นมาแล้วทำการทดลองและแสดงผลการทดลองที่ Control Form ของโปรแกรม



รูปที่ 4.1 ส่วนประกอบต่างๆของโปรแกรม

4.1 การรับสัญญาณภาพจากกล้องดิจิตอลวีดีโอ

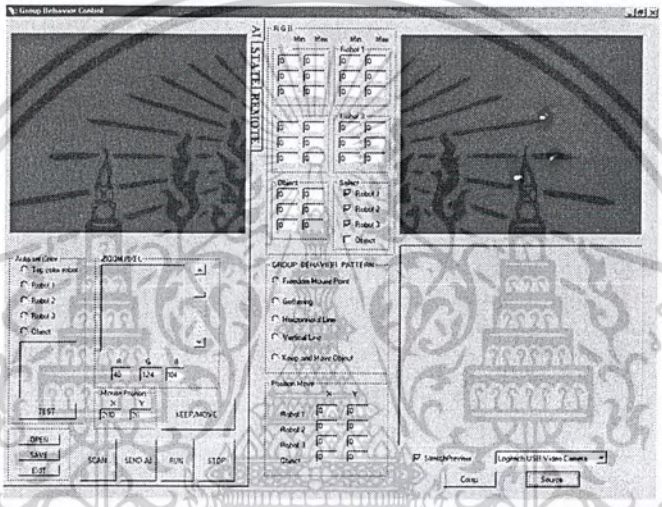
การดึงสัญญาณภาพจากกล้องวีดีโอ เข้ามาประมวลผลในโปรแกรม VISUAL BASIC และแสดงผลออกทางหน้าจอของคอมพิวเตอร์

การทดลอง

ทำการต่อกล้องวีดีโอเข้ากับคอมพิวเตอร์เข้าทางพอร์ต USB ของคอมพิวเตอร์ แล้วแล้วถ่ายภาพวัตถุให้นำภาพเข้ามาประมวลผลในโปรแกรมที่ได้เขียนไว้โดยตรง

ผลการทดลอง

สามารถรับภาพ และแสดงผลทางหน้าต่างจอคอมพิวเตอร์ ที่หน้าต่างภาพมุมทางด้านบนซ้ายของหน้าต่างโปรแกรม



รูปที่ 4.2 การดึงภาพเข้ามาประมวลผล

4.2 การปรับแต่ง (Calibration) แม่สี RGBบนตัวของหุ่นยนต์

การปรับแต่งแม่สี RGB บนตัวของหุ่นยนต์นี้เราได้ทำการเขียนโปรแกรมในส่วนนี้มาโดยเฉพาะเพื่อความสะดวกและความแม่นยำในการปรับแต่ง แม่สี RGB

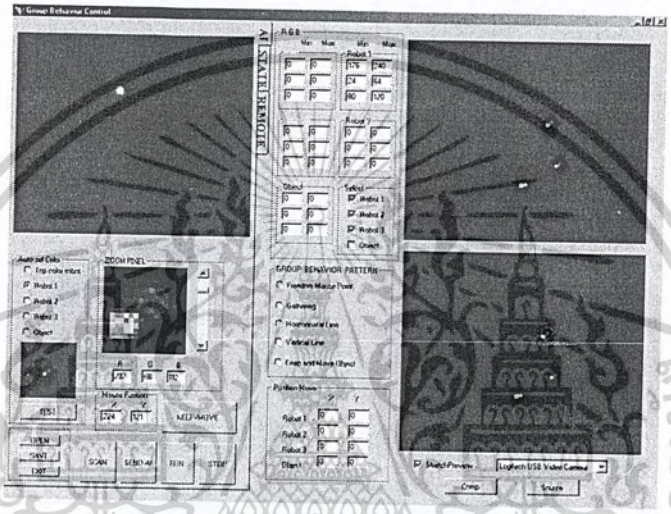
4.2.1 การขยายดูภาพเฉพาะบริเวณ

การทดลอง

ทำการคัดลอกภาพภาพจากกล้องวิดีโอมาประมวลผลอีกหน้าต่างหนึ่งจากนั้นทำการคลิกเมาส์เพื่อขยายภาพเฉพาะบริเวณ จากนั้นทำการปรับสกอรบาร์เพื่อขยายภาพเข้าออกเพื่อดูภาพเฉพาะจุด

การทดลอง

จากผลการทดลองที่ได้ในรูปที่ 4.3 ทำการคัดลอกภาพจากกล้องวีดีโอมาที่หน้าต่างต่าง ประมวลผลด้านล่างจากนั้นลากเมาส์ไปยังตำแหน่งที่ต้องการจะซูมแล้วทำการคลิก ภาพบริเวณที่ทำการคลิกจะไปปรากฏอยู่ที่หน้าจอเล็กๆในส่วนของปรับแต่ง แม่สี RGB ถ้าภาพที่ได้ยังเล็กสามารถทำการขยายหรือลดภาพได้จากสกอรบาร์ด้านข้าง



รูปที่ 4.3 การดึงภาพมาดูเฉพาะส่วน

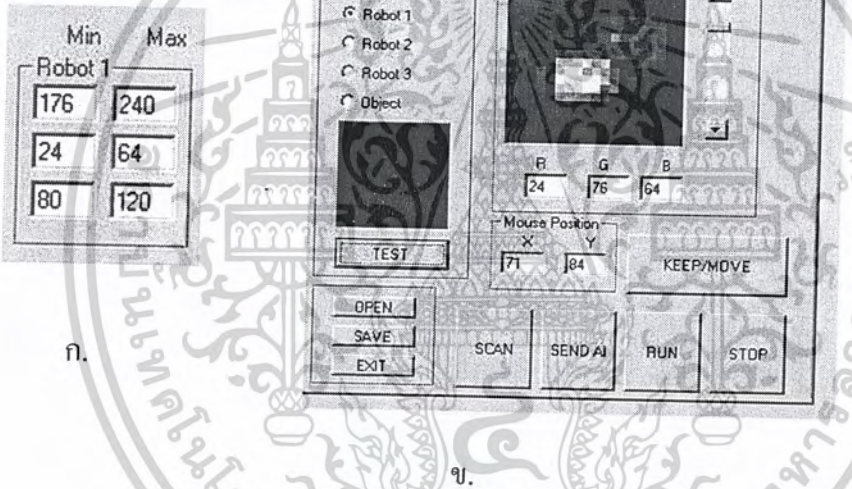
4.2.2 การปรับแต่งแม่สี RGB และการทดสอบ

การทดลอง

นำภาพที่ได้จากการซูมภาพเข้ามาแล้วการทำการปรับแต่งแม่สี RGB บนตัวหุ่นยนต์ เพื่อหาค่าแม่สี RGB ในย่านต่ำสุดและสูงสุดของสีในแต่ละย่าน จากนั้นทำการทดสอบแม่สีในย่านที่เราต้องการ

ผลการทดลอง

เมื่อทำการซูมภาพเข้ามาดูเสร็จเรียบร้อยแล้วการทำการปรับแต่งแม่สี RGB ทำโดยการดับเบิ้ลคลิกที่หน้าต่างรูปภาพแล้วกดเมาส์ข้างไว้แล้วลากตามสีที่เราต้องการในส่วนนี้โปรแกรมจะทำการปรับจูนแม่สี RGB โดยอัตโนมัติโดยจะมีค่าต่ำสุดและสูงสุดของของแม่สี RGB โดยสามรถดูได้จากหน้าต่างที่บอกค่าที่ถูกปรับแต่งแม่สี RGB แล้ว ดังรูปที่ 4.4 ก. เมื่อได้ค่าแม่สีแล้วทำการทดสอบโดยการกดปุ่มทดสอบ (Test) สังเกตดูที่สีของหน้าต่างทดสอบ สีที่ไม่ใช้สีในจากการปรับแต่งจะเป็นสีดำ ดังรูปที่ 4.4 ข.



ก.

ข.

รูปที่ 4.4 ก. ย่านของแม่สี RGB ที่ทำการปรับแต่งได้
ข. การซูมภาพมาปรับแต่งแม่สี RGB

4.3 การประมวลผลภาพของวัตถุ

การทดลอง

ทำการปรับแต่ง สีของหุ่นยนต์แต่ละตัว แล้วทำการสแกนภาพหาตำแหน่งแล้วส่งไปยังหน้าต่างจำลองตำแหน่งจริงของหุ่นยนต์

ผลการทดลอง

เมื่อทำการปรับแต่งสีของหุ่นยนต์แต่ละตัวเสร็จ แล้วทำการสแกนภาพคำนวณหาตำแหน่งและส่งไปยังหน้าต่างจำลองตำแหน่งของหุ่นยนต์โดยภาพที่แสดงเป็นรูปวงกลมมีขีดสีขาวเป็นตัวบอกมุมของหุ่นยนต์ว่าหันหน้าไปทางใหนและหน้าต่างนี้ยังเป็นหน้าต่างควบคุมซึ่งพิกัดที่หน้าต่างนี้มีขนาดพิกัดเท่ากับพิกัดหน้าต่างที่รับภาพเข้ามาประมวลดังรูปที่ 4.5 ข. ซึ่งหน้าต่างนี้จะใช้เป็นหน้าต่างที่ใช้ในการควบคุมการเคลื่อนที่ของหุ่นยนต์

ก. ข.

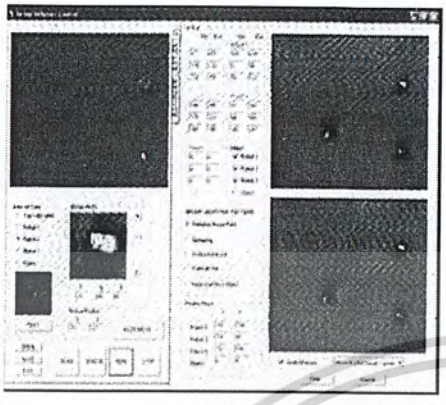
รูปที่ 4.5 ก. รูปที่รับมาจากกล้องวิดีโอ
ข. รูปจำลองตำแหน่งพิกัดที่ได้จากการประมวลผลจากคอมพิวเตอร์

4.4 การทดลองโปรแกรมการควบคุมหุ่นยนต์หลายตัว

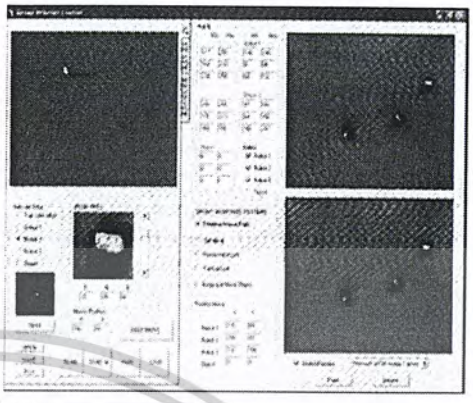
4.5.1 การควบคุมหุ่นยนต์แบบอิสระ

การทดลอง

การทดลองนี้เป็นการควบคุมหุ่นยนต์ให้เคลื่อนที่แบบอิสระหลายๆตัว โดยการคลิกเมาส์เลือกหุ่นยนต์ให้เคลื่อนที่ไปยังตำแหน่งที่เราต้องการ

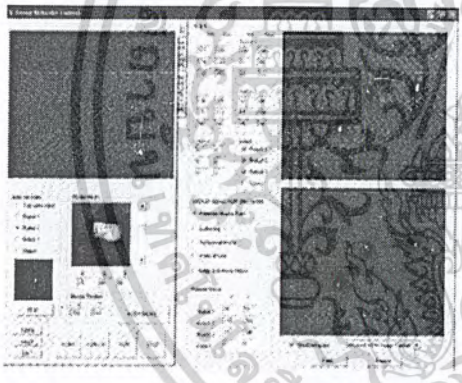


รูปที่ 4.6 ก.

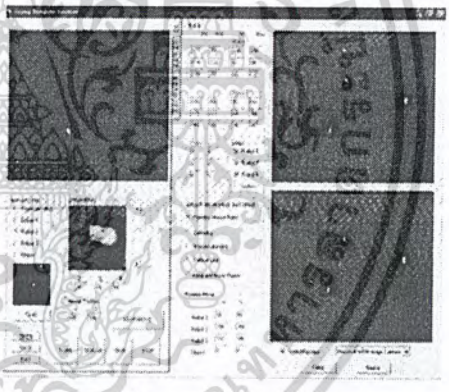


รูปที่ 4.6 ข.

เมื่อทำการรันโปรแกรมจากนั้นทำการคลิกที่ตัวหุ่นยนต์จากหน้าต่างจำลองพิกัดจริง ดังรูปที่ 4.6 ก. จากนั้นทำการลากเมาส์ไปยังตำแหน่งที่เราต้องการดังรูปที่ 4.6 ข. จากนั้นหุ่นยนต์ตัวแรกจะเคลื่อนที่ไปยังตำแหน่งที่เราต้องการ

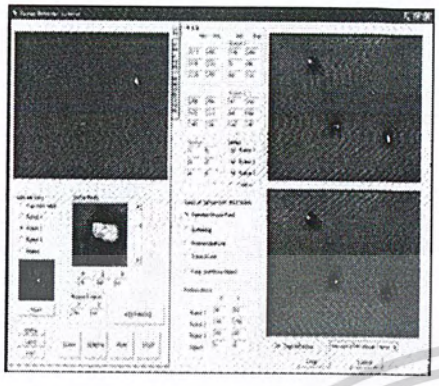


รูปที่ 4.6 ค.

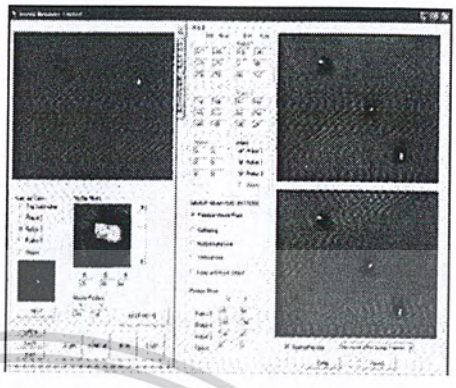


รูปที่ 4.6 ง.

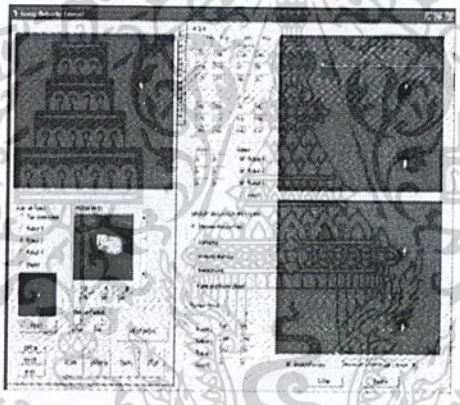
ในขณะที่หุ่นยนต์ตัวแรกยังเคลื่อนที่อยู่ทำการคลิกลากหุ่นยนต์ตัวที่สอง ให้เคลื่อนที่ไปอีกทิศทางหนึ่งดังรูปที่ 4.6 ค. หุ่นยนต์ตัวที่สองจะทำการเคลื่อนที่อยู่ในขณะที่หุ่นยนต์ตัวแรกยังเคลื่อนที่ไปในตำแหน่งเดิม และสุดท้ายทำการเคลื่อนที่หุ่นยนต์ตัวที่สามไปอีกทิศทางหนึ่งดังรูปที่ 4.6 ง. และ รูปที่ 4.6 จ.



รูปที่ 4.6 จ.

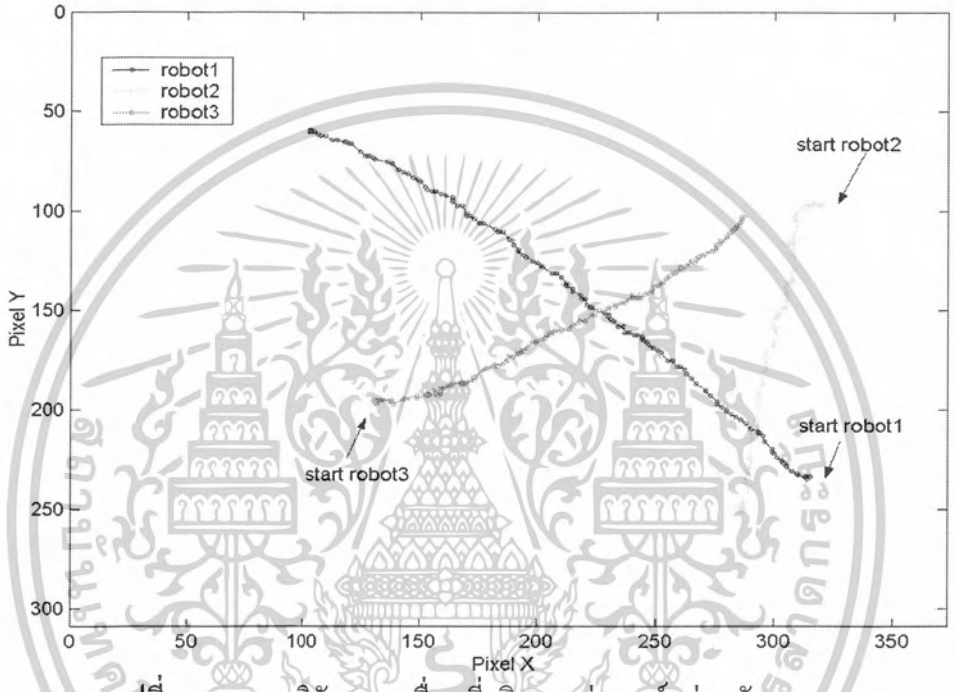


รูปที่ 4.6 ฉ.



รูปที่ 4.6 ช.

ผลที่ได้จากการทดลองทำให้เห็นว่าเราสามารถที่จะควบคุมการเคลื่อนที่ของหุ่นยนต์แต่ ได้อย่างอิสระโดยไม่เกี่ยวข้องกัน จากการคลิกตำแหน่งจากเมาส์ และจากการทดลองนี้เราสามารถแสดงพิกัดการเคลื่อนที่ของหุ่นยนต์ในรูปแบบของกราฟเทียบกับพิกัดพิกเซลแกน X และ Y ในรูปที่4.7 ดังนั้นตำแหน่งที่ได้คือพิกัดการเคลื่อนที่จริงของหุ่นยนต์แต่ละตัว



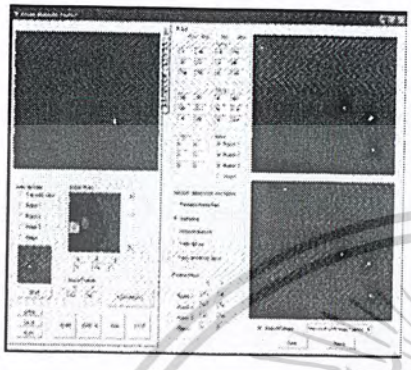
รูปที่ 4.7 แสดงพิกัดการเคลื่อนที่จริงของหุ่นยนต์แต่ละตัว

4.5.2 การควบคุมการเคลื่อนที่เป็นกลุ่ม

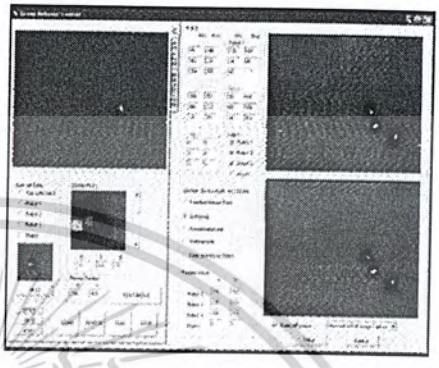
การทดลอง

ทำการควบคุมหุ่นยนต์ให้เคลื่อนเป็นกลุ่มโดยการคลิกเมาส์เลือกตำแหน่งการเคลื่อนที่แบบเป็นกลุ่มไปยังตำแหน่งที่ต้องการ

ผลการทดลอง

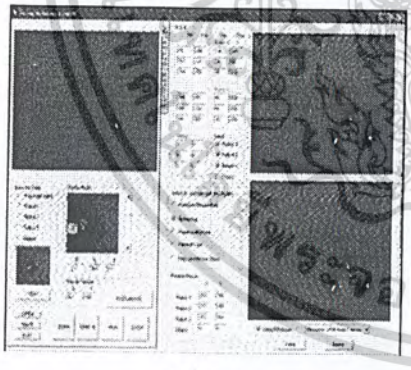


รูปที่ 4.8 ก.

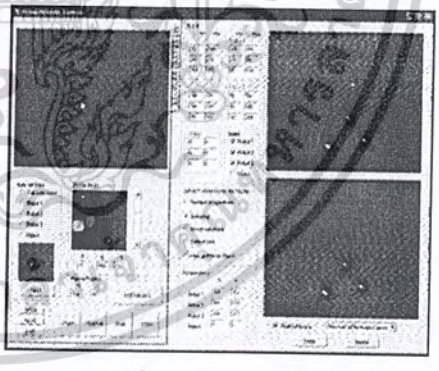


รูปที่ 4.8 ข.

จากการทดลองเมื่อทำการรันโปรแกรมแล้วทำการคลิกตำแหน่งที่ให้หุ่นยนต์เคลื่อนที่จากตำแหน่งเดิมดังรูปที่ 4.8 ก. จากนั้นหุ่นยนต์จะทำการเคลื่อนที่ ดังรูปที่ 4.8 ข. แล้วทำการจัดกลุ่มดังในรูปที่ 4.8 ค.

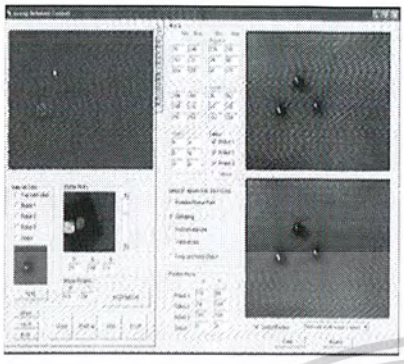


รูปที่ 4.8 ค.

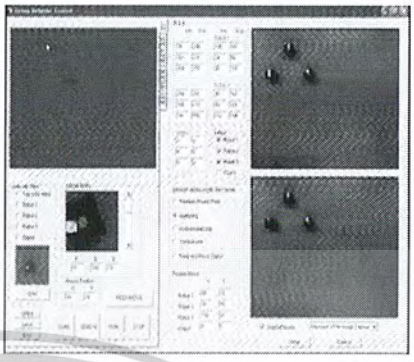


รูปที่ 4.8 ง.

เมื่อหุ่นยนต์จัดเป็นกลุ่มแล้วทำการคลิกเมาส์ลากให้หุ่นยนต์เคลื่อนที่ไปยังตำแหน่งใหม่ในรูปที่ 4.8 ง. จากนั้นหุ่นยนต์จะเคลื่อนที่แบบเป็นกลุ่มไปยังตำแหน่งที่เมาส์อยู่ดังรูปที่ 4.8 จ. และ รูปที่ 4.8 ฉ.

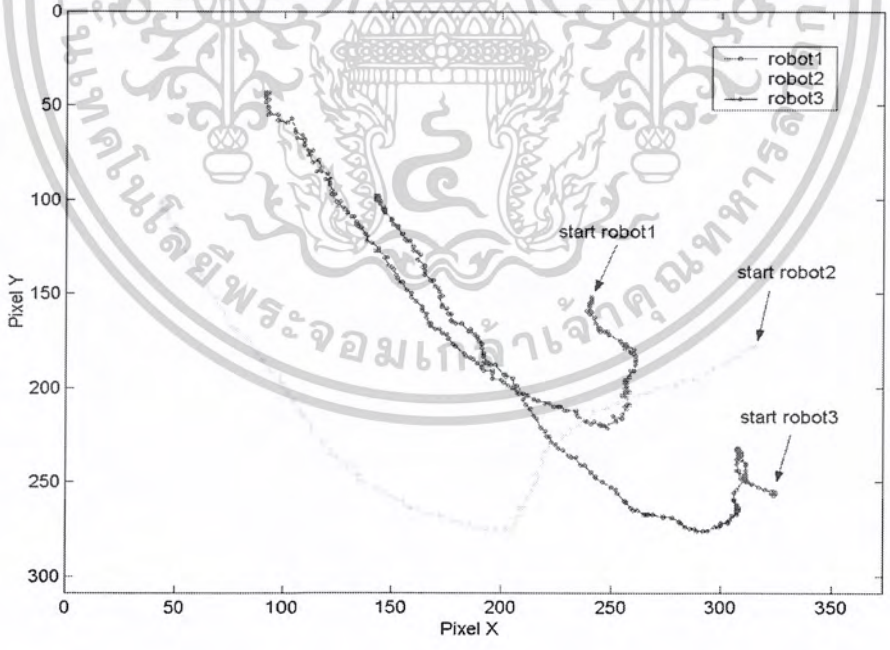


รูปที่ 4.8 จ.



รูปที่ 4.8 ฉ.

ผลจากการทดลองนี้เราสามารถแสดงพิกัดการเคลื่อนที่ของหุ่นยนต์ในรูปแบบของกราฟเทียบกับพิกัดพิกเซลแกน X และ Y ในรูปที่ 4.9 ดังนั้นตำแหน่งที่ได้คือพิกัดการเคลื่อนที่จริงของหุ่นยนต์แต่ละตัว



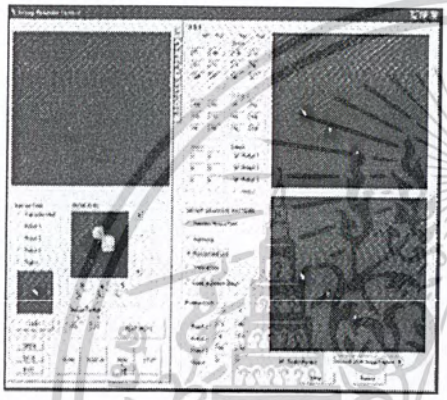
รูปที่ 4.9 แสดงพิกัดการเคลื่อนที่จริงของหุ่นยนต์แบบเป็นกลุ่ม

4.5.3 การควบคุมเคลื่อนที่หน้ากระดาน

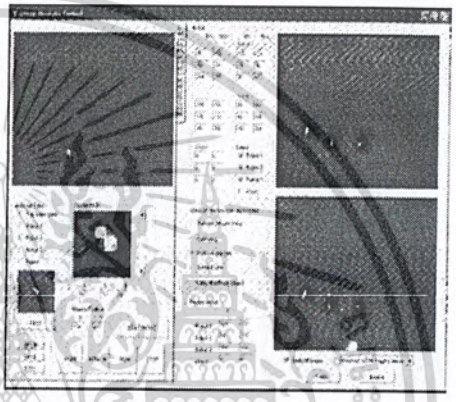
การทดลอง

ทำการควบคุมหุ่นยนต์ให้เคลื่อนเป็นกลุ่มแบบหน้ากระดาน โดยการคลิกเมาส์เลือกตำแหน่งการเคลื่อนที่แบบเป็นกลุ่มไปยังตำแหน่งที่ต้องการ

ผลการทดลอง

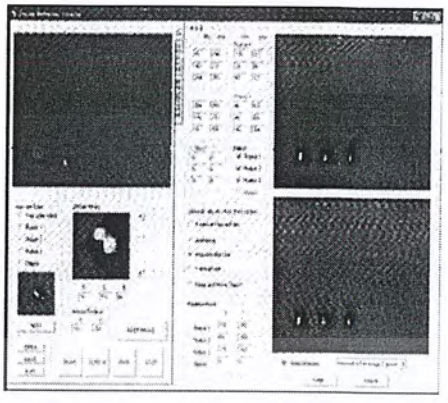


รูปที่ 4.10 ก.

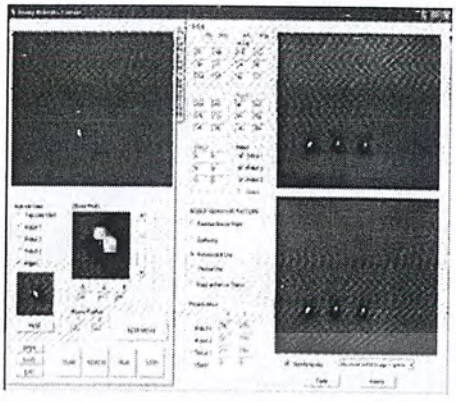


รูปที่ 4.10 ข.

จากการทดลองเมื่อทำการรันโปรแกรมแล้วทำการคลิกตำแหน่งที่ให้หุ่นยนต์เคลื่อนที่จากตำแหน่งเดิมดังรูปที่ 4.10 ก. จากนั้นหุ่นยนต์จะทำการเคลื่อนที่ ดังรูปที่ 4.10 ข. แล้วทำการจัดกลุ่มดังในรูปที่ 4.10 ค.

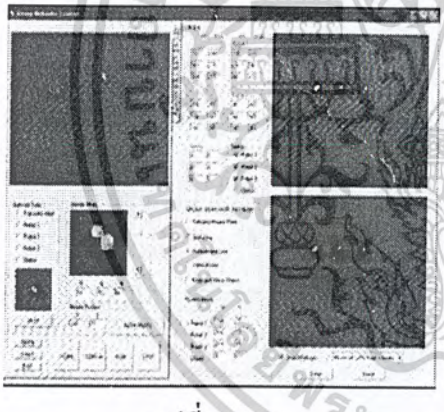


รูปที่ 4.10 ค.

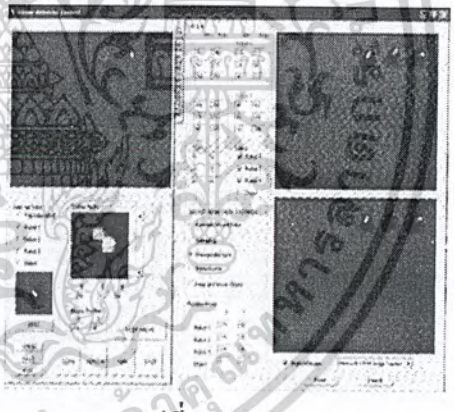


รูปที่ 4.10 ง.

เมื่อหุ่นยนต์จัดเป็นกลุ่มแบบหน้ากระดานได้แล้วทำการคลิกเมาส์ลากให้หุ่นยนต์เคลื่อนที่ไปยังตำแหน่งใหม่ในรูปที่ 4.10 ง. จากนั้นหุ่นยนต์จะเคลื่อนที่แบบเป็นกลุ่มไปยังตำแหน่งที่เมาส์อยู่ดังรูปที่ 4.10 จ. และ รูปที่ 4.10 ฉ.

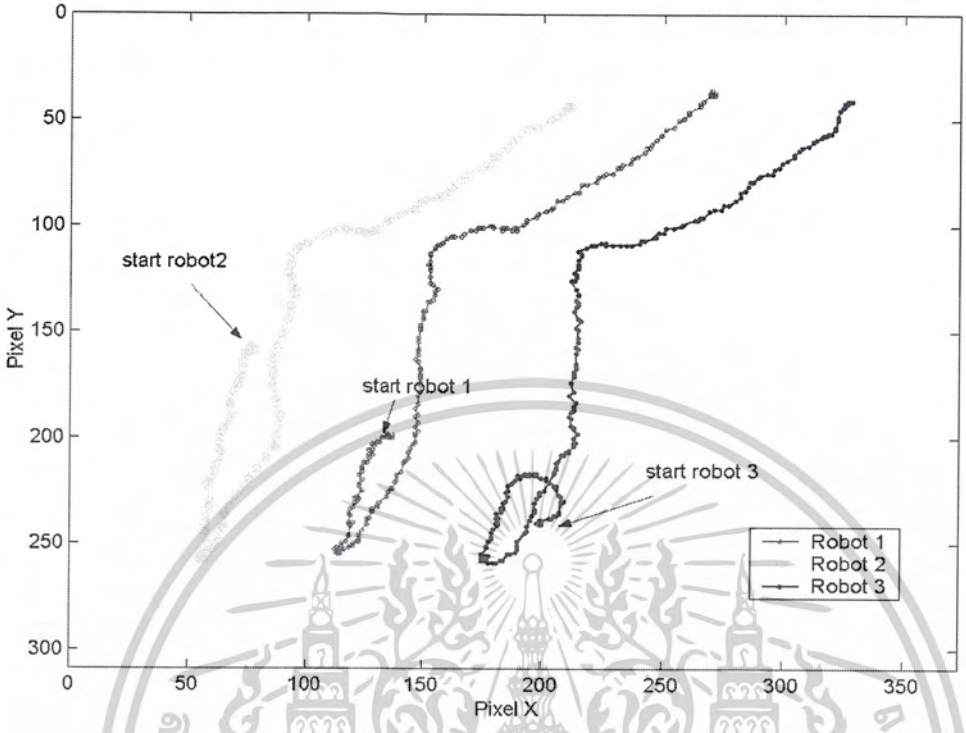


รูปที่ 4.8 จ.



รูปที่ 4.8 ฉ.

ผลจากการทดลองนี้เราสามารถแสดงพิกัดการเคลื่อนที่ของหุ่นยนต์ในรูปแบบของกราฟเทียบกับพิกัดพิกเซลแกน X และ Y ในรูปที่ 4.11 ดังนั้นตำแหน่งที่ได้คือพิกัดการเคลื่อนที่จริงของหุ่นยนต์แต่ละตัว



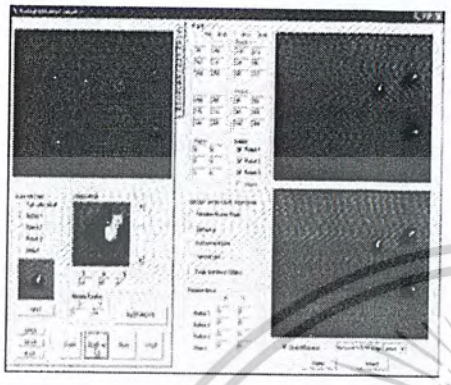
รูปที่ 4.11 แสดงพิกัดการเคลื่อนที่จริงของหุ่นยนต์แบบหน้ากระดาน

4.5.4 การควบคุมเคลื่อนที่แถวตอน

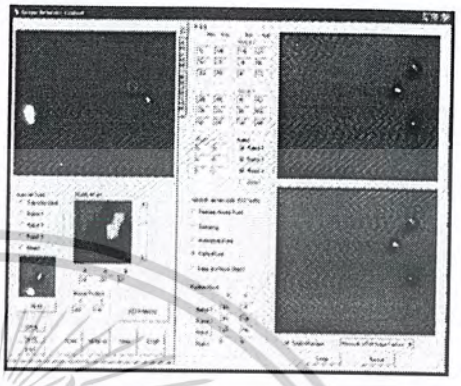
การทดลอง

ทำการควบคุมหุ่นยนต์ให้เคลื่อนเป็นกลุ่มแบบหน้ากระดาน โดยการคลิกเมาส์เลือกตำแหน่งการเคลื่อนที่แบบเป็นกลุ่มไปยังตำแหน่งที่ต้องการ

ผลการทดลอง

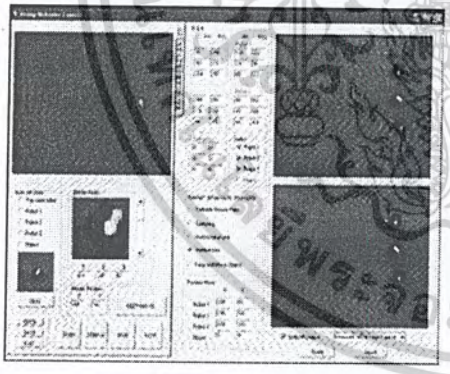


รูปที่ 4.12 ก.

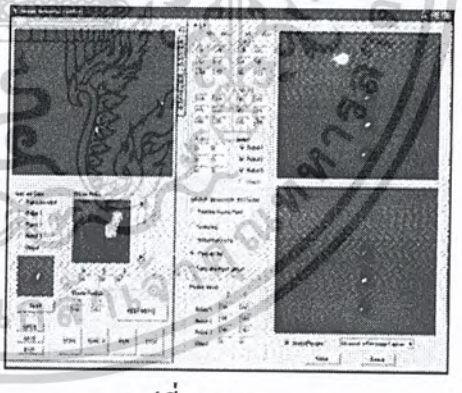


รูปที่ 4.12 ข.

จากการทดลองเมื่อทำการรันโปรแกรมแล้วทำการคลิกตำแหน่งที่ให้หุ่นยนต์เคลื่อนที่จากตำแหน่งเดิมดังรูปที่ 4.12 ก. จากนั้นหุ่นยนต์จะทำการเคลื่อนที่ ดังรูปที่ 4.12 ข. แล้วทำการจัดกลุ่มแบบหน้าตอนดังในรูปที่ 4.12 ค.

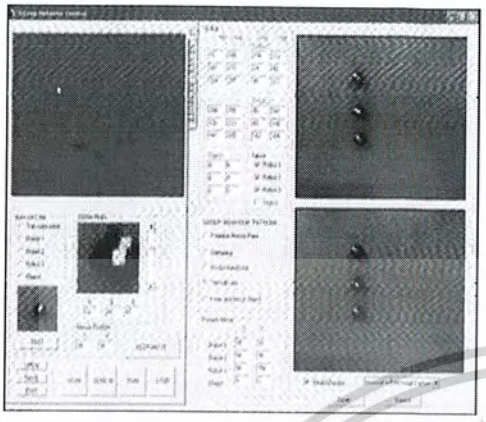


รูปที่ 4.12 ค.

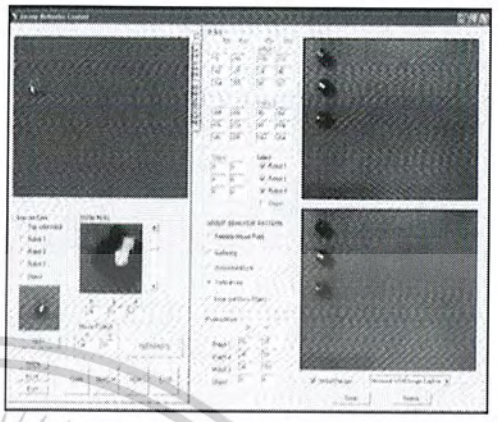


รูปที่ 4.12 ง.

เมื่อหุ่นยนต์จัดเป็นกลุ่มแบบหน้าตอนแล้วทำการคลิกเมาส์ลากให้หุ่นยนต์เคลื่อนที่ไปยังตำแหน่งใหม่ในรูปที่ 4.12 ง. จากนั้นหุ่นยนต์จะเคลื่อนที่แบบเป็นกลุ่มไปยังตำแหน่งที่เมาส์อยู่ดังรูปที่ 4.12 จ. และ รูปที่ 4.12 ฉ.

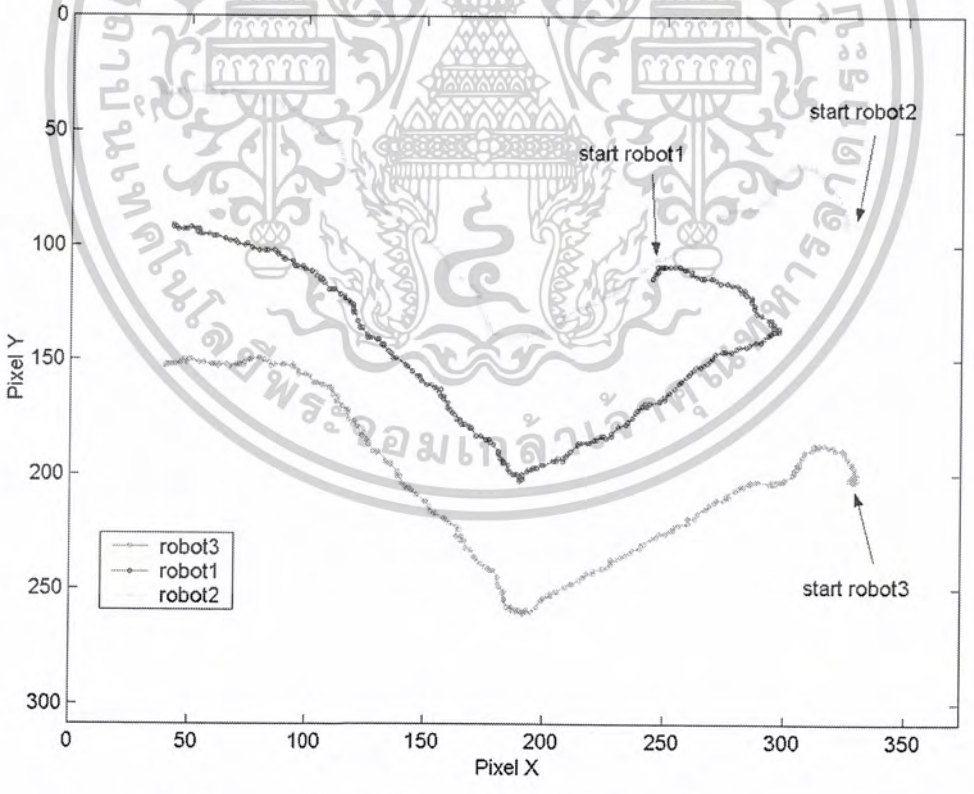


รูปที่ 4.12 จ.



รูปที่ 4.12 ฉ.

ผลจากการทดลองนี้เราสามารถแสดงพิกัดการเคลื่อนที่ของหุ่นยนต์ในรูปแบบของกราฟเทียบกับพิกัดพิกเซลแกน X และ Y ในรูปที่ 4.13 ดังนั้นตำแหน่งที่ได้คือพิกัดการเคลื่อนที่จริงของหุ่นยนต์แต่ละตัว



รูปที่ 4.13 แสดงพิกัดการเคลื่อนที่จริงของหุ่นยนต์แบบแถวตอน

4.6 การทดลองโปรแกรมการควบคุมหุ่นยนต์เคลื่อนย้ายวัตถุ

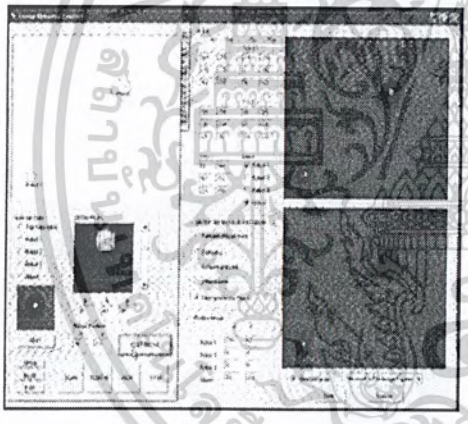
4.6.1 การเคลื่อนย้ายวัตถุมายังจุดเริ่มต้น

การทดลอง

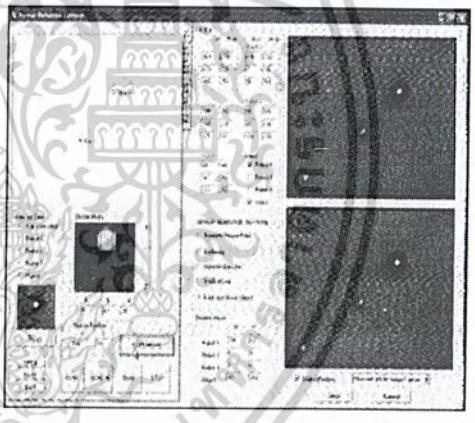
ทำการควบคุมหุ่นยนต์ให้เคลื่อนที่ไปเคลื่อนย้ายวัคถุกับมายังจุดเริ่มต้น โดยการควบคุมแบบอัตโนมัติจากโปรแกรม

ผลการทดลอง

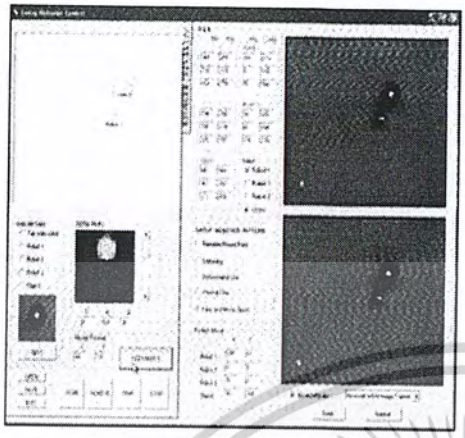
จากการทดลองเมื่อทำการรันโปรแกรมหุ่นยนต์เคลื่อนที่อย่างอัตโนมัติจากตำแหน่งเริ่มต้นดังรูปที่ 4.14 ก. จากนั้นหุ่นยนต์จะทำการเคลื่อนที่ ไปวัตถุดังรูปที่ 4.14 ข. หลังจากนั้นหุ่นยนต์จะทำการหลบหลีกวัตถุดังในรูปที่ 4.14 ค. แล้วอ้อมวัตถุเพื่อหาตำแหน่งที่จะทำการผลักวัตถุกลับมายังที่เดิม ดังรูปที่ 4.14 ง.



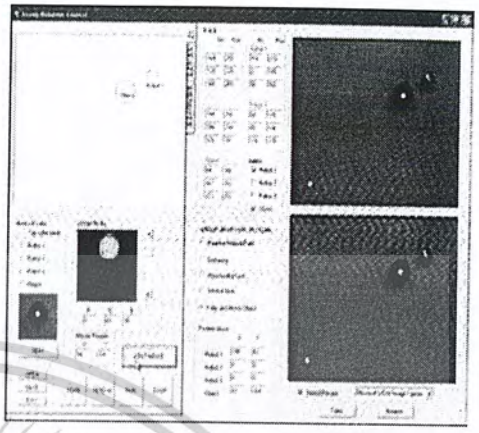
รูปที่ 4.14 ก.



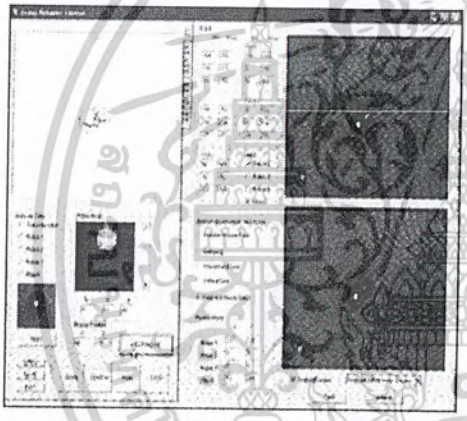
รูปที่ 4.14 ข.



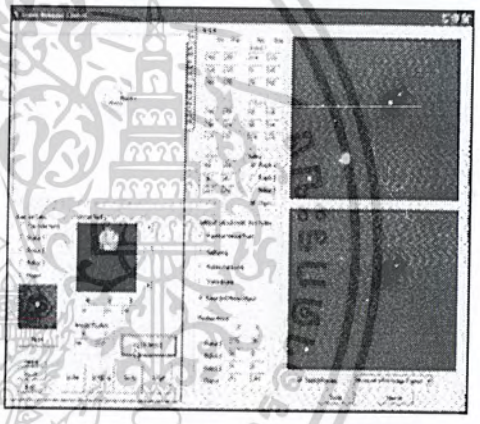
รูปที่ 4.14 ค.



รูปที่ 4.14 ง.

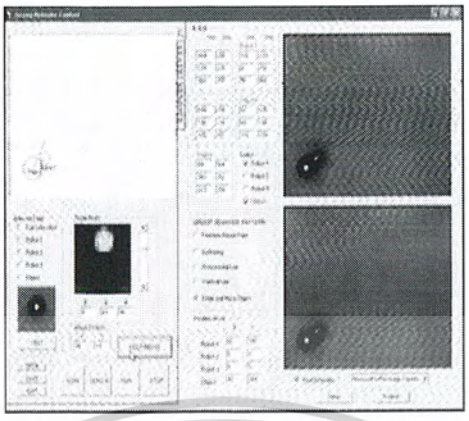


รูปที่ 4.14 จ.



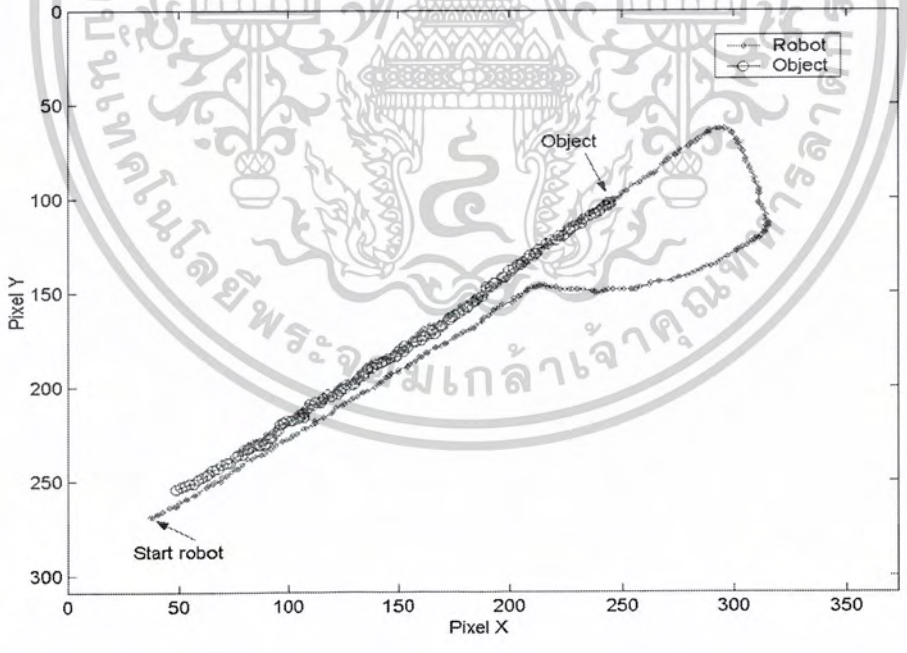
รูปที่ 4.14 ฉ.

เมื่อหุ่นยนต์เคลื่อนที่ผ่านวัตถุได้แล้วจะทำการผลักกล่องกลับมายังจุดเริ่มต้นดังรูปที่ 4.14 จ. จนถึงรูปที่ 4.14 ช.



รูปที่ 4.14 ช.

ผลจากการทดลองนี้เราสามารถแสดงพิกัดการเคลื่อนที่ของหุ่นยนต์และวัตถุในรูปแบบของกราฟเทียบกับพิกัดพิกเซลแกน X และ Y ในรูปที่ 4.15 ดังนั้นตำแหน่งที่ได้คือพิกัดการเคลื่อนที่จริงของหุ่นยนต์และวัตถุ



รูปที่ 4.15 กราฟแสดงการเคลื่อนที่ของหุ่นยนต์และวัตถุในขณะเคลื่อนย้ายวัตถุมายังจุดเริ่มต้น

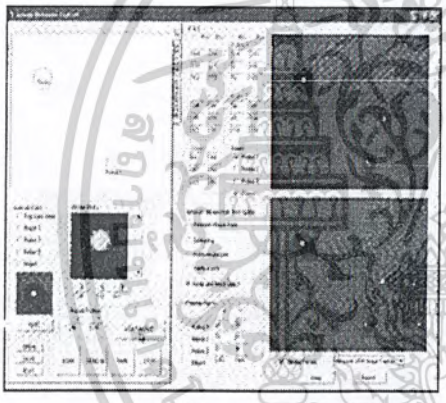
4.6.2 การเคลื่อนย้ายวัตถุจากที่หมายไปยังจุดที่เราต้องการ

การทดลอง

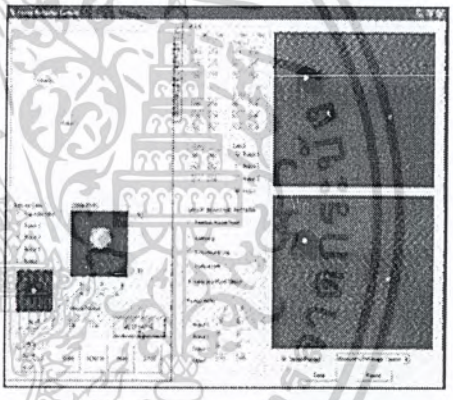
ทำการควบคุมหุ่นยนต์ให้เคลื่อนที่ไปเคลื่อนย้ายวัคถุไปยังตำแหน่งที่เราต้องการโดยการควบคุมแบบอัตโนมัติจากโปรแกรม

ผลการทดลอง

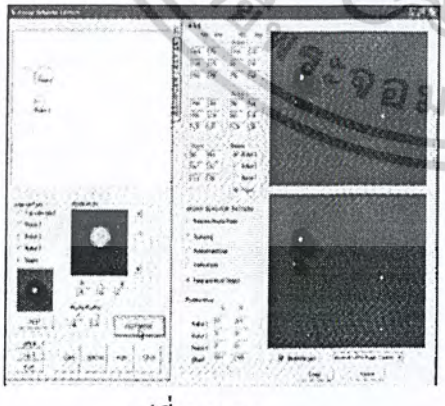
จากการทดลองเมื่อทำการกำหนดตำแหน่งจุดเคลื่อนย้ายวัตถุและทำการรันโปรแกรมหุ่นยนต์เคลื่อนที่อย่างอัตโนมัติจากตำแหน่งเริ่มต้นดังรูปที่ 4.16 ก. จากนั้นหุ่นยนต์จะทำการเคลื่อนที่ ไปวัตถุดังรูปที่ 4.16 ข. หลังจากนั้นหุ่นยนต์จะทำการหลบหลีกวัตถุดังในรูปที่ 4.16 ค. แล้วอ้อมวัตถุเพื่อหาตำแหน่งที่จะทำการผลักวัตถุไปยังที่หมาย ดังรูปที่ 4.14 ง.



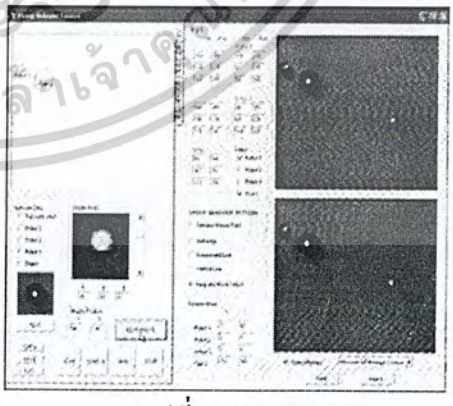
รูปที่ 4.16 ก.



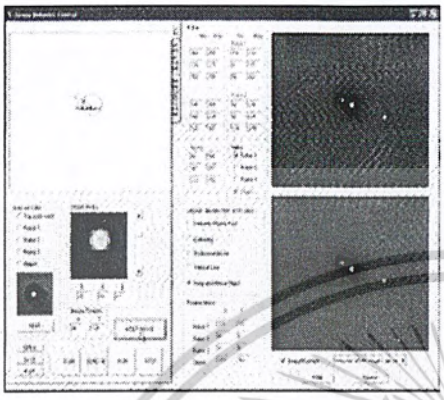
รูปที่ 4.16 ข.



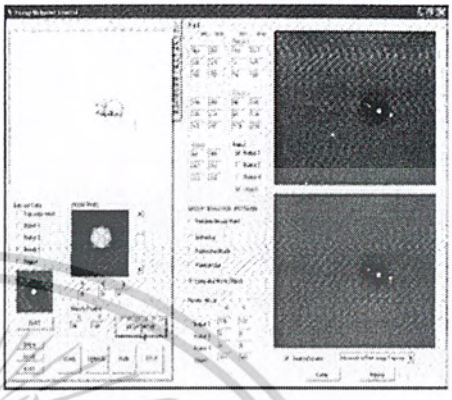
รูปที่ 4.16 ค.



รูปที่ 4.16 ง.



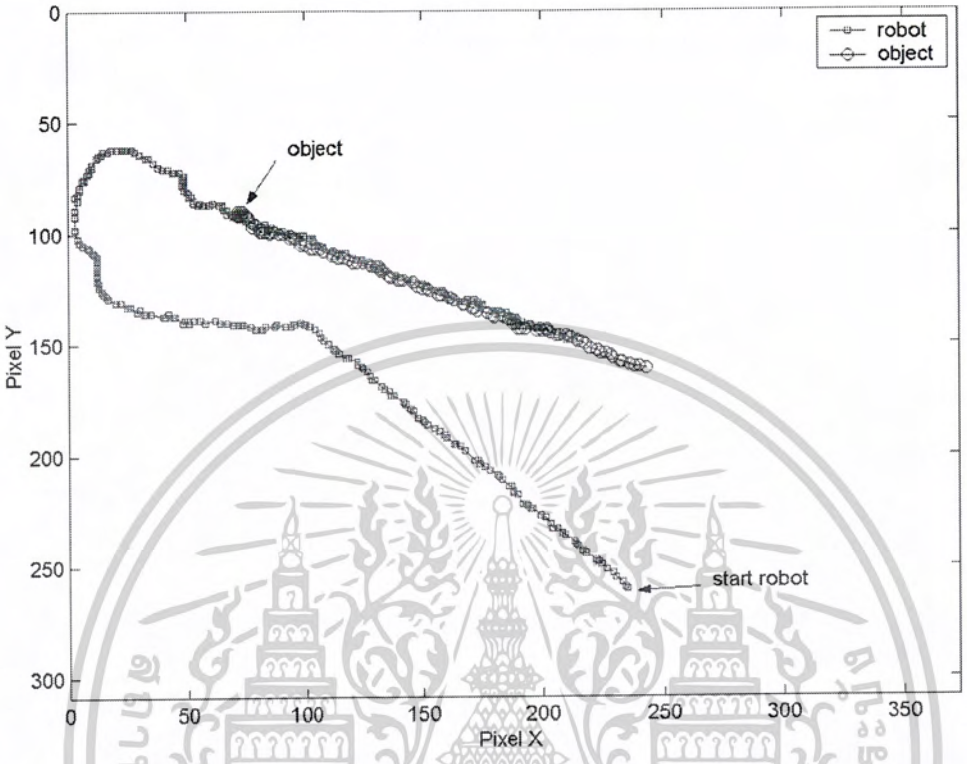
รูปที่ 4.16 จ.



รูปที่ 4.16 ฉ.

เมื่อหุ่นยนต์เคลื่อนที่ผ่านวัตถุได้แล้วจะทำการผลักกล่องกลับมายังจุดเริ่มต้นดังรูปที่ 4.16 จ. จนถึงรูปที่ 4.16 ช.

ผลจากการทดลองนี้เราสามารถแสดงพิกัดการเคลื่อนที่ของหุ่นยนต์และวัตถุในรูปแบบของกราฟเทียบกับพิกัดพิกเซลแกน X และ Y ในรูปที่ 4.17 ดังนั้นตำแหน่งที่ได้คือพิกัดการเคลื่อนที่จริงของหุ่นยนต์และวัตถุ



รูปที่ 4.15 กราฟแสดงการเคลื่อนที่ของหุ่นยนต์ในขณะเคลื่อนย้ายวัตถุไปยังตำแหน่งที่ต้องการ

4.6 สรุปผลจากการทดลอง

จากการทดลองคอมพิวเตอร์และโปรแกรมสามารถรับภาพจากกล้องวีดีโอ เข้ามาประมวลผลภาพและส่งข้อมูลติดต่อกับหุ่นยนต์โดยสามารถแยกรหัสแล้วส่งไปยังหุ่นยนต์ได้อย่างถูกต้อง ในส่วนของการปรับแต่งแม่สี RGB แบบอัตโนมัติ โดยการดึงแม่สีจริงมาอ่านจึงสามารถที่จะเซทสีได้ในขณะแสงที่ส่องลงมามีความเข้มไม่เท่ากันได้แม่นยำและแยกย่านสีที่เราสนใจได้ในระยะเวลาที่รวดเร็วอีกทั้งยังสามารถแก้ย่านสีที่เราสนใจได้ในขณะทำงานโดยไม่ต้องทำหยุดการทำงานของโปรแกรม ส่วนค่าพารามิเตอร์ต่างๆเมื่อทำการเซตค่าเสร็จสามารถที่จะเสพเก็บลงเครื่องได้ทำให้ไม่ต้องทำการเซตค่าใหม่ทุกครั้งเมื่อปิดแล้วเปิดโปรแกรม

ในส่วนของตัวหุ่นยนต์สามารถเคลื่อนที่ไปยังแต่ละตำแหน่งได้อย่างถูกต้อง อยู่ในเกณฑ์ที่น่าพอใจคือมีค่ามุมผิดพลาดที่ 5 องศาและระยะทางที่ 5-10 พิกเซล

บทที่ 5
บทวิจารณ์และสรุป

5.1 สรุปผลการทดลอง

จากทดลองคอมพิวเตอร์สามารถรับภาพเข้ามาประมวลผลในในการทดลองการควบคุมการเคลื่อนที่แบบเป็นกลุ่มสามารถทำงานได้อย่างถูกต้องตามอัลกอลิทึมที่เขียนไว้ ในส่วนการปรับแต่งแม่สี RGB แบบอัตโนมัติ โดยการดึงแม่สีจริงมาอ่านจึงสามารถที่จะเซตได้ในขณะแสงที่ส่องลงมามีความเข้มไม่เท่ากัน ได้และแยกย่านสีที่เราสนใจ ได้ในระยะเวลาที่รวดเร็วอีกทั้งยังสามารถปรับแต่งย่านสีที่เราสนใจได้ในขณะโปรแกรมทำงานโดยไม่ต้องทำการหยุดการทำงานของโปรแกรม ส่วนค่าพารามิเตอร์ต่างๆ เมื่อทำการเซตค่าเสร็จสามารถที่จะเซฟเก็บลงเครื่องได้ ทำให้ไม่ต้องทำการเซตค่าใหม่ทุกครั้ง เมื่อเปิดโปรแกรมใหม่

ส่วนของตัวหุ่นยนต์สามารถเคลื่อนที่ไปยังแต่ละตำแหน่งได้อย่างถูกต้อง อยู่ในเกณฑ์ที่น่าพอใจ คือ มีค่ามุมผิดพลาดที่ 5 องศา และระยะทางที่ 5 – 10 พิกเซล

5.2 วิจารณ์การทดลอง

จากผลการทดลองการับภาพจากกล้องวีดีโอเข้ามาประมวลผลในโปรแกรมและแสดงผลออกทางจอมอนิเตอร์ รวมถึงการประมวลผลภาพได้ผลตามที่ต้องการ ตามอัลกอลิทึมที่เขียนไว้

5.3 ประโยชน์ที่ได้รับจากการทำโครงงานนี้

ในการทำโครงงานต้องอาศัยความรู้หลายๆ ด้าน ทั้งความรู้ด้านการเขียนโปรแกรม และความรู้ด้านการออกแบบและสร้าง ความรู้ในด้านทฤษฎีของระบบควบคุมมาใช้ ความรู้เกี่ยวกับการประมวลผลภาพดิจิตอลอีกทั้งยังทำให้ทักษะในการเขียนโปรแกรมวิชวลเบสิคและความรู้ทางด้านวงจรอิเล็กทรอนิกส์ในการสร้างและออกแบบหุ่นยนต์ที่มีขนาดเล็ก โดยใช้การควบคุมจากคอมพิวเตอร์มอนิเตอร์อีกด้านของการทำงาน นี่คือการแก้ไขปัญหาต่างๆ ที่เกิดขึ้นปัญหาที่เกิดจากปัจจัยภายนอกและปัญหาที่เกิดจากวงจรหรือโปรแกรมและประโยชน์ทางด้านการประสานงานในการทำงานมีการแบ่งงานกันทำ ทำให้มีการฝึกการวางแผนอย่างเป็นระบบและการคิดแก้ปัญหาต่างๆ ที่เกิดขึ้นร่วมกัน

5.4 ปัญหาที่พบในการทำโครงงานนี้

ปัญหาที่จะประสบจากการทำโครงงานนี้ในส่วนแรกคือ ขาดความรู้และประสบการณ์ในการทำงานและปัจจัยที่สำคัญที่สุดและเป็นอุปสรรคในการทำงานมากในช่วงแรกต้องทำการศึกษาเกี่ยวกับการเขียนโปรแกรมทฤษฎีการประมวลผลภาพเกิดมาจากพื้นฐานทางด้านโปรแกรมมีน้อยและไม่เคยศึกษาด้านการประมวลผลภาพมาก่อนในช่วงแรกจึงเสียเวลาในการศึกษาก่อนการทำโครงงานนี้พอสมควร

ส่วนที่สองเป็นในส่วนของการออกแบบตัวหุ่นยนต์ ในส่วนของมอเตอร์มีปัญหาในเรื่องของความเร็วที่มีขนาดไม่เท่ากันทำให้หุ่นยนุต์แต่ละตัวมีความเร็วที่ไม่เท่ากันและในส่วนของตัวส่งคลื่นอินฟราเรดมีอัตราการส่งข้อมูลค่อนข้างต่ำ

5.5 แนวทางในการพัฒนา

แนวทางในการปรับปรุงพัฒนาโครงงานนี้ครรจะออกแบบให้หุนยนต์แต่ละตัวมีหน้าที่ในการทำงานเฉพาะอย่าง เช่นสามารถจับกล่องหรือเจาะสิ่งของได้ เพื่อที่ทำให้หุ่นยนต์สามารถทำงานได้อย่างเป็นทีมได้อย่างมีประสิทธิภาพและควรออกแบบ ให้หุ่นยนต์มีเซนเซอร์ติดตั่งที่ตัวหุ่นเพื่อให้หุ่นยนต์แต่ละตัวสามารถตรวจสอบหุ่นยนต์รอบข้างได้ทำให้หุ่นยนต์สามรถเคลื่อนที่รวมกันเป็นกลุ่มได้โดยไม่ต้องมีอุปกรณ์ภายนอก

ส่วนอัลกอลิทึมที่ได้ออกแบบสามารถนำไปใช้กับหุ่นยนต์ที่มีขนาดเล็กที่สามารถเคลื่อนที่ด้วยความละเอียดสูงหรืออาจใช้กับ งานพวกMachine vision กับการใช้งานในงานอุตสาหกรรม

ภาคผนวก

ภาคผนวก ก

ก โปรแกรมประมวลภาพ

กำหนดตัวแปร

```
Option Explicit
Dim x_click  As Integer
Dim y_click  As Integer
Dim R_min As Long
Dim R_max As Long
Dim G_min As Long
Dim G_max As Long
Dim B_min As Long
Dim B_max As Long
Dim R_min_t As Long
Dim R_max_t As Long
Dim G_min_t As Long
Dim G_max_t As Long
Dim B_min_t As Long
Dim B_max_t As Long
Dim R_min_R1 As Long
Dim R_max_R1 As Long
Dim G_min_R1 As Long
Dim G_max_R1 As Long
Dim B_min_R1 As Long
Dim B_max_R1 As Long
Dim R_min_R2 As Long
Dim R_max_R2 As Long
Dim G_min_R2 As Long
Dim G_max_R2 As Long
Dim B_min_R2 As Long
Dim B_max_R2 As Long
Dim R_min_R3 As Long
Dim R_max_R3 As Long
Dim G_min_R3 As Long
Dim G_max_R3 As Long
Dim B_min_R3 As Long
Dim B_max_R3 As Long
Dim R_min_Ob As Long
Dim R_max_Ob As Long
Dim G_min_Ob As Long
Dim G_max_Ob As Long
Dim B_min_Ob As Long
Dim B_max_Ob As Long
Dim Robot_CR1_i As Long
Dim Robot_CR1_j As Long
Dim count_R1 As Long
Dim Robot_CR2_i As Long
Dim Robot_CR2_j As Long
Dim count_R2 As Long
Dim Robot_CR3_i As Long
Dim Robot_CR3_j As Long
Dim count_R3 As Long
Dim OB_i As Long
Dim OB_j As Long
Dim count_OB As Long
Dim Robot_cenx(1 To 6) As
Long
Dim Robot_ceny(1 To 6) As
Long
Dim Count_cen(1 To 6) As
Long
Dim Cen_obx As Long
Dim Cen_oby As Long
Dim Count_cenob As Long
Dim Robot_Topx(1 To 6) As
Long
Dim Robot_Topy(1 To 6) As
Long
Dim Count_top(1 To 6) As
Long
Dim R_OB As Integer
Dim R_ROBOT As Integer
Dim Angle As Integer
Dim W(1 To 7) As Integer
Dim h(1 To 7) As Integer
Dim rad As Integer
Dim RC_X(1 To 7) As Integer
Dim RC_Y(1 To 7) As Integer
Dim Dx(1 To 6) As Integer
Dim Dy(1 To 6) As Integer
Dim Px(1 To 6) As Integer
Dim Py(1 To 6) As Integer
Dim Tx(1 To 6) As Integer
Dim Ty(1 To 6) As Integer
```

```
Dim W(1 To 7) As Integer
Dim h(1 To 7) As Integer
Dim rad As Integer
Dim RC_X(1 To 7) As Integer
Dim RC_Y(1 To 7) As Integer
Dim Dx(1 To 6) As Integer
Dim Dy(1 To 6) As Integer
Dim Px(1 To 6) As Integer
Dim Py(1 To 6) As Integer
Dim Tx(1 To 6) As Integer
Dim Ty(1 To 6) As Integer
Dim O_R(1 To 6) As Integer
Dim angle_R(1 To 6) As
Integer
Dim t(1 To 4) As Integer
Dim Robot_mx(1 To 6) As
Long
Dim Robot_my(1 To 6) As
Long
Dim Cen_obmx As Long
Dim Cen_obmy As Long
Dim Ro_mx(1 To 6) As Long
Dim Ro_my(1 To 6) As Long
Dim Rob_mx(1 To 3) As Long
Dim Rob_my(1 To 3) As Long
Dim C_obmx As Long
Dim C_obmy As Long
Dim rad_R As Integer
Dim rad_O As Integer
Dim WM(1 To 7) As Integer
Dim hM(1 To 7) As Integer
Dim AngleM As Integer
Dim Pmx(1 To 4) As Integer
Dim Pmy(1 To 4) As Integer
Dim dis_er(1 To 4) As Integer
Dim Dmx(1 To 4) As Double
Dim Dmy(1 To 4) As Double
Dim Tmx(1 To 4) As Integer
Dim Tmy(1 To 4) As Integer
Dim Om_R(1 To 4) As Integer
Dim anglem_R(1 To 4) As
Integer
Dim M_an1 As Integer
Dim M_an2 As Integer
Dim an_sl1 As Integer
Dim an_sl2 As Integer
Dim distance As Integer
Dim distance_ob As Integer
Dim W_R As Integer
Dim an_check As Integer
Dim disx(1 To 10) As Integer
Dim disy(1 To 10) As Integer
Dim angledis(1 To 10) As
Integer
Dim O_dis(1 To 10) As Integer
Dim dis_obs As Double
Dim dis_obx As Double
Dim dis_oby As Double
Dim angle_obs As Double
Dim O_ob As Integer
Dim error_robot As Double
Dim an_rot As Integer
Dim quit As Integer
Const pi As Double = 3.141578
Const dOneDegree As Double =
pi / 180
Dim R_mx As Integer
Dim R_my As Integer
Dim dis_ro As Double
Dim dis_rox As Double
Dim dis_roy As Double
Dim angle_ro As Double
Dim O_ro As Integer
Dim Oer_RO As Integer
Dim dis_rm As Double
Dim dis_rmx As Double
Dim dis_rmy As Double
Dim angle_rm As Double
Dim O_rm As Integer
Dim dis_rp As Double
Dim dis_rpx As Double
Dim dis_rpy As Double
Dim angle_rp As Double
Dim O_rp As Integer
Dim dis_op As Double
Dim dis_opx As Double
Dim dis_opy As Double
Dim angle_op As Double
Dim O_op As Integer
Dim jump1 As Integer
Dim jump2 As Integer
```

ฟังชั่นในการแม่สี RGB

```
Private Function ColorRGB(Color1 As Long, r As Integer, G As Integer, b As Integer)
r = Color1 Mod 256
b = Color1 \ 65536
G = (Color1 \ 256) Mod 256
End Function
```

แสกนแม่สีรอบแรก

```
Private Sub Command1_Click()
DoEvents
Dim i As Integer, j As Integer
Dim Color1 As Long, r As Integer, G As Integer, b As Integer
Robot_CR1_i = 0
Robot_CR1_j = 0
count_R1 = 0
Robot_CR2_i = 0
Robot_CR2_j = 0
count_R2 = 0
Robot_CR3_i = 0
Robot_CR3_j = 0
count_R3 = 0
OB_i = 0
OB_j = 0
count_OB = 0
R_min_R1 = RMI1.Text
R_max_R1 = RMA1.Text
G_min_R1 = GMI1.Text
G_max_R1 = GMA1.Text
B_min_R1 = BMI1.Text
B_max_R1 = BMA1.Text
R_min_R2 = RMI2.Text
R_max_R2 = RMA2.Text
G_min_R2 = GMI2.Text
G_max_R2 = GMA2.Text
B_min_R2 = BMI2.Text
B_max_R2 = BMA2.Text
R_min_R3 = RMI3.Text
R_max_R3 = RMA3.Text
G_min_R3 = GMI3.Text
G_max_R3 = GMA3.Text
B_min_R3 = BMI3.Text
B_max_R3 = BMA3.Text
R_min_Ob = RMIO.Text
R_max_Ob = RMAO.Text
G_min_Ob = GMIO.Text
G_max_Ob = GMAO.Text
B_min_Ob = BMIO.Text
B_max_Ob = BMAO.Text
```

```
BitBlt Picture3.hdc, 0, 0, 373, 309, Picture2.hdc, 0, 0, vbSrcCopy
For i = 0 To 373
For j = 0 To 309
Color1 = GetPixel(Picture3.hdc, i, j)
Call ColorRGB(Color1, r, G, b)
If Check1.Value = 1 And (r + G + b) <= (R_max_R1 + G_max_R1 + B_max_R1) And r >= R_min_R1 And r <=
R_max_R1 And G >= G_min_R1 And G <= G_max_R1 And b >= B_min_R1 And b <= B_max_R1 Then
Color1 = 255
Robot_CR1_i = Robot_CR1_i + i
Robot_CR1_j = Robot_CR1_j + j
count_R1 = count_R1 + 1
ElseIf Check2.Value = 1 And (r + G + b) <= (R_max_R2 + G_max_R2 + B_max_R2) And r >= R_min_R2 And r
<= R_max_R2 And G >= G_min_R2 And G <= G_max_R2 And b >= B_min_R2 And b <= B_max_R2 Then
Color1 = 589823
Robot_CR2_i = Robot_CR2_i + i
Robot_CR2_j = Robot_CR2_j + j
count_R2 = count_R2 + 1
ElseIf Check3.Value = 1 And (r + G + b) <= (R_max_R3 + G_max_R3 + B_max_R3) And r >= R_min_R3 And r
<= R_max_R3 And G >= G_min_R3 And G <= G_max_R3 And b >= B_min_R3 And b <= B_max_R3 Then
Color1 = 16711935
Robot_CR3_i = Robot_CR3_i + i
Robot_CR3_j = Robot_CR3_j + j
count_R3 = count_R3 + 1
ElseIf Check4.Value = 1 And (r + G + b) <= (R_max_Ob + G_max_Ob + B_max_Ob) And r >= R_min_Ob And r
<= R_max_Ob And G >= G_min_Ob And G <= G_max_Ob And b >= B_min_Ob And b <= B_max_Ob Then
Color1 = 16711680
OB_i = OB_i + i
OB_j = OB_j + j
count_OB = count_OB + 1
Else
Color1 = 0
End If
```

```
SetPixel Picture3.hdc, i, j, Color1
DoEvents
Next j
Next i
If Check1.Value = 1 Then
Robot_CR1_i = Robot_CR1_i / count_R1
Robot_CR1_j = Robot_CR1_j / count_R1
Text14.Text = Robot_CR1_i
Text15.Text = Robot_CR1_j
End If
If Check2.Value = 1 Then
Robot_CR2_i = Robot_CR2_i / count_R2
Robot_CR2_j = Robot_CR2_j / count_R2
Text16.Text = Robot_CR2_i
Text17.Text = Robot_CR2_j
End If
If Check3.Value = 1 Then
Robot_CR3_i = Robot_CR3_i / count_R3
Robot_CR3_j = Robot_CR3_j / count_R3
Text18.Text = Robot_CR3_i
Text19.Text = Robot_CR3_j
End If
If Check4.Value = 1 Then
OB_i = OB_i / count_OB
OB_j = OB_j / count_OB
Text20.Text = OB_i
Text21.Text = OB_j
End If
End Sub
```

แสกนจริงพร้อมกับคำนวน

```
Private Sub Command2_Click()
DoEvents
quit = 0
Do
DoEvents
Call scancen
Call cul_angle
Call send_AI
Robot_CR1_i = Robot_cenx(1)
Robot_CR1_j = Robot_ceny(1)
Robot_CR2_i = Robot_cenx(2)
Robot_CR2_j = Robot_ceny(2)
Robot_CR3_i = Robot_cenx(3)
Robot_CR3_j = Robot_ceny(3)
OB_i = Cen_obx
OB_j = Cen_oby
Loop Until quit = 1
End Sub
```

โปรแรมย่อยแสกน

```
Sub scancen()
Dim i As Integer, j As Integer
Dim Color1 As Long, r As Integer, G As Integer, b As Integer
Dim i_1 As Long, i_2 As Long, j_1 As Long, j_2 As Long

Robot_cenx(1) = 0
Robot_ceny(1) = 0
Count_cen(1) = 0
Robot_cenx(2) = 0
Robot_ceny(2) = 0
Count_cen(2) = 0
Robot_cenx(3) = 0
Robot_ceny(3) = 0
```

```
Count_cen(3) = 0
Cen_obx = 0
Cen_oby = 0
Count_cenob = 0
Robot_Topx(1) = 0
Robot_Topy(1) = 0
Count_top(1) = 0
Robot_Topx(2) = 0
Robot_Topy(2) = 0
Count_top(2) = 0
Robot_Topx(3) = 0
'----------Robot 1 ----------
If Check1.Value = 1 Then
Do
DoEvents
BitBlt Picture3.hdc, 0, 0, 373, 309, Picture2.hdc, 0, 0, vbSrcCopy
R_ROBOT = R_ROBOT + 5
i_1 = Robot_CR1_i - R_ROBOT
i_2 = Robot_CR1_i + R_ROBOT
j_1 = Robot_CR1_j - R_ROBOT
j_2 = Robot_CR1_j + R_ROBOT
For i = i_1 To i_2
For j = j_1 To j_2
Color1 = GetPixel(Picture3.hdc, i, j)
Call ColorRGB(Color1, r, G, b)
If (r + G + b) <= (R_max_R1 + G_max_R1 + B_max_R1) And r >= R_min_R1 And r <= R_max_R1 And G >=
G_min_R1 And G <= G_max_R1 And b >= B_min_R1 And b <= B_max_R1 Then
Color1 = 255
Robot_cenx(1) = Robot_cenx(1) + i
Robot_ceny(1) = Robot_ceny(1) + j
Count_cen(1) = Count_cen(1) + 1
Robot_Topy(3) = 0
Count_top(3) = 0
R_min_t = RMIT.Text
R_max_t = RMAT.Text
G_min_t = GMIT.Text
G_max_t = GMAT.Text
B_min_t = BMIT.Text
B_max_t = BMAT.Text
R_OB = Text34.Text
R_ROBOT = Text35.Text
```

```
ElseIf (r + G + b) <= (R_max_t + G_max_t + B_max_t) And r >= R_min_t And r <= R_max_t And G >= G_min_t
And G <= G_max_t And b >= B_min_t And b <= B_max_t Then
Color1 = 16777215
Robot_Topx(1) = Robot_Topx(1) + i
Robot_Topy(1) = Robot_Topy(1) + j
Count_top(1) = Count_top(1) + 1
Else
Color1 = 0
End If
SetPixel Picture3.hdc, i, j, Color1
DoEvents
Next j
Next i
Loop While Count_cen(1) = 0 Or Count_top(1) = 0
End If
R_ROBOT = Text35.Text
'----------------Robot 2 ----------------
If Check2.Value = 1 Then
Do
DoEvents
BitBlt Picture3.hdc, 0, 0, 373, 309, Picture2.hdc, 0, 0, vbSrcCopy
R_ROBOT = R_ROBOT + 5
i_1 = Robot_CR2_i - R_ROBOT
i_2 = Robot_CR2_i + R_ROBOT
j_1 = Robot_CR2_j - R_ROBOT
j_2 = Robot_CR2_j + R_ROBOT
For i = i_1 To i_2
For j = j_1 To j_2
Color1 = GetPixel(Picture3.hdc, i, j)
Call ColorRGB(Color1, r, G, b)
If (r + G + b) <= (R_max_R2 + G_max_R2 + B_max_R2) And r >= R_min_R2 And r <= R_max_R2 And G >=
G_min_R2 And G <= G_max_R2 And b >= B_min_R2 And b <= B_max_R2 Then
```

```
Color1 = 589823
Robot_cenx(2) = Robot_cenx(2) + i
Robot_ceny(2) = Robot_ceny(2) + j
Count_cen(2) = Count_cen(2) + 1
ElseIf (r + G + b) <= (R_max_t + G_max_t + B_max_t) And r >= R_min_t And r <= R_max_t And G >= G_min_t
And G <= G_max_t And b >= B_min_t And b <= B_max_t Then
Color1 = 16777215
Robot_Topx(2) = Robot_Topx(2) + i
Robot_Topy(2) = Robot_Topy(2) + j
Count_top(2) = Count_top(2) + 1
Else
Color1 = 0
End If
SetPixel Picture3.hdc, i, j, Color1
DoEvents
Next j
Next i
Loop While Count_cen(2) = 0 Or Count_top(2) = 0
End If
R_ROBOT = Text35.Text
'----------Robot 3 ----------
If Check3.Value = 1 Then
Do
DoEvents
BitBlt Picture3.hdc, 0, 0, 373, 309, Picture2.hdc, 0, 0, vbSrcCopy
R_ROBOT = R_ROBOT + 5
i_1 = Robot_CR3_i - R_ROBOT
i_2 = Robot_CR3_i + R_ROBOT
j_1 = Robot_CR3_j - R_ROBOT
j_2 = Robot_CR3_j + R_ROBOT
For i = i_1 To i_2
For j = j_1 To j_2
```

```
Color1 = GetPixel(Picture3.hdc, i, j)
Call ColorRGB(Color1, r, G, b)

If (r + G + b) <= (R_max_R3 + G_max_R3 + B_max_R3) And r >= R_min_R3 And r <= R_max_R3 And G >=
G_min_R3 And G <= G_max_R3 And b >= B_min_R3 And b <= B_max_R3 Then
Color1 = 16711935
Robot_cenx(3) = Robot_cenx(3) + i
Robot_ceny(3) = Robot_ceny(3) + j
Count_cen(3) = Count_cen(3) + 1
ElseIf (r + G + b) <= (R_max_t + G_max_t + B_max_t) And r >= R_min_t And r <= R_max_t And G >= G_min_t
And G <= G_max_t And b >= B_min_t And b <= B_max_t Then
Color1 = 16777215
Robot_Topx(3) = Robot_Topx(3) + i
Robot_Topy(3) = Robot_Topy(3) + j
Count_top(3) = Count_top(3) + 1
Else
Color1 = 0
End If
SetPixel Picture3.hdc, i, j, Color1
DoEvents
Next j
Next i
Loop While Count_cen(3) = 0 Or Count_top(3) = 0
End If
'----------object----------
If Check4.Value = 1 Then
Do
DoEvents
BitBlt Picture3.hdc, 0, 0, 373, 309, Picture2.hdc, 0, 0, vbSrcCopy
R_OB = R_OB + 5
i_1 = OB_i - R_OB
i_2 = OB_i + R_OB
```

```
j_1 = OB_j - R_OB
j_2 = OB_j + R_OB
For i = i_1 To i_2
For j = j_1 To j_2
Color1 = GetPixel(Picture3.hdc, i, j)
Call ColorRGB(Color1, r, G, b)
If (r + G + b) <= (R_max_Ob + G_max_Ob + B_max_Ob) And r >= R_min_Ob And r <= R_max_Ob And G >=
G_min_Ob And G <= G_max_Ob And b >= B_min_Ob And b <= B_max_Ob Then
Color1 = 16711680
Cen_obx = Cen_obx + i
Cen_oby = Cen_oby + j
Count_cenob = Count_cenob + 1
Else
Color1 = 0
End If
SetPixel Picture3.hdc, i, j, Color1
DoEvents
Next j
Next i
Loop While Count_cenob = 0
End If
'-------cul position -------
If Check1.Value = 1 Then
Robot_cenx(1) = Robot_cenx(1) / Count_cen(1)
Robot_ceny(1) = Robot_ceny(1) / Count_cen(1)
Text14.Text = Robot_cenx(1)
Text15.Text = Robot_ceny(1)
Robot_Topx(1) = Robot_Topx(1) / Count_top(1)
Robot_Topy(1) = Robot_Topy(1) / Count_top(1)
End If
If Check2.Value = 1 Then
Robot_cenx(2) = Robot_cenx(2) / Count_cen(2)
```

```
Robot_ceny(2) = Robot_ceny(2) / Count_cen(2)
Text16.Text = Robot_cenx(2)
Text17.Text = Robot_ceny(2)
Robot_Topx(2) = Robot_Topx(2) / Count_top(2)
Robot_Topy(2) = Robot_Topy(2) / Count_top(2)
End If
If Check3.Value = 1 Then
Robot_cenx(3) = Robot_cenx(3) / Count_cen(3)
Robot_ceny(3) = Robot_ceny(3) / Count_cen(3)
Text18.Text = Robot_cenx(3)
Text19.Text = Robot_ceny(3)
Robot_Topx(3) = Robot_Topx(3) / Count_top(3)
Robot_Topy(3) = Robot_Topy(3) / Count_top(3)
End If
If Check4.Value = 1 Then
Cen_obx = Cen_obx / Count_cenob
Cen_oby = Cen_oby / Count_cenob
Text20.Text = Cen_obx
Text21.Text = Cen_oby
End If
End Sub
```

โปรแกรมย่อยคำนวนหามุม

```
Sub cul_angle()
DoEvents
Tx(1) = Robot_Topx(1)
Ty(1) = Robot_Topy(1)
Tx(2) = Robot_Topx(2)
Ty(2) = Robot_Topy(2)
Tx(3) = Robot_Topx(3)
Ty(3) = Robot_Topy(3)
If Check1.Value = 1 Then
```

```
Px(1) = Robot_CR1_i
Py(1) = Robot_CR1_j
Dx(1) = Tx(1) - Px(1)
Dy(1) = Ty(1) - Py(1)
If Dx(1) = 0 Then
Dx(1) = 1
End If
angle_R(1) = Int(((Atn(Dy(1) / Dx(1))) * 180) / pi)
If Dx(1) >= 0 And Dy(1) <= 0 Then
O_R(1) = angle_R(1) * (-1)
ElseIf Dx(1) <= 0 And Dy(1) <= 0 Then
O_R(1) = (90 - angle_R(1)) + 90
ElseIf Dx(1) <= 0 And Dy(1) >= 0 Then
O_R(1) = (angle_R(1) * (-1)) + 180
ElseIf Dx(1) >= 0 And Dy(1) >= 0 Then
O_R(1) = 360 - angle_R(1)
End If
Text22.Text = O_R(1)
End If
If Check2.Value = 1 Then
Px(2) = Robot_CR2_i
Py(2) = Robot_CR2_j
Dx(2) = Tx(2) - Px(2)
Dy(2) = Ty(2) - Py(2)
If Dx(2) = 0 Then
Dx(2) = 1
End If
angle_R(2) = Int(((Atn(Dy(2) / Dx(2))) * 180) / pi)
If Dx(2) >= 0 And Dy(2) <= 0 Then
O_R(2) = angle_R(2) * (-1)
ElseIf Dx(2) <= 0 And Dy(2) <= 0 Then
O_R(2) = (90 - angle_R(2)) + 90
```

```
ElseIf Dx(2) <= 0 And Dy(2) >= 0 Then
O_R(2) = (angle_R(2) * (-1)) + 180
ElseIf Dx(2) >= 0 And Dy(2) >= 0 Then
O_R(2) = 360 - angle_R(2)
End If
Text23.Text = O_R(2)
End If
If Check3.Value = 1 Then
Px(3) = Robot_CR3_i
Py(3) = Robot_CR3_j
Dx(3) = Tx(3) - Px(3)
Dy(3) = Ty(3) - Py(3)
If Dx(3) = 0 Then
Dx(3) = 1
End If
angle_R(3) = Int(((Atn(Dy(3) / Dx(3))) * 180) / pi)
If Dx(3) >= 0 And Dy(3) <= 0 Then
O_R(3) = angle_R(3) * (-1)
ElseIf Dx(3) <= 0 And Dy(3) <= 0 Then
O_R(3) = (90 - angle_R(3)) + 90
ElseIf Dx(3) <= 0 And Dy(3) >= 0 Then
O_R(3) = (angle_R(3) * (-1)) + 180
ElseIf Dx(3) >= 0 And Dy(3) >= 0 Then
O_R(3) = 360 - angle_R(3)
End If
Text24.Text = O_R(3)
End If
End Sub
```

โปรแกรม โยนพิกัดให้หน้าต่างควบคุม

```
Sub send_AI()
Picture1.Cls
```

```
DoEvents
rad_R = Text32.Text
rad_O = Text33.Text
RC_X(1) = Robot_cenx(1)
RC_Y(1) = Robot_ceny(1)
RC_X(2) = Robot_cenx(2)
RC_Y(2) = Robot_ceny(2)
RC_X(3) = Robot_cenx(3)
RC_Y(3) = Robot_ceny(3)
If Check1.Value = 1 Then
Angle = O_R(1)
W(1) = Int(Cos(Angle * dOneDegree) * rad_R)
If Sgn(W(1)) = -1 Then
W(1) = W(1) * (-1)
End If
h(1) = Int(Sin(Angle * dOneDegree) * rad_R)
If Sgn(h(1)) = -1 Then
h(1) = h(1) * (-1)
End If
Picture1.ScaleMode = 3
Picture1.Circle (RC_X(1), RC_Y(1)), rad_R, RGB(255, 0, 0)
If Angle >= 0 And Angle <= 90 Then
Picture1.Line (RC_X(1), RC_Y(1))-((RC_X(1) + W(1)), (RC_Y(1) - h(1))), vbWhite
ElseIf Angle >= 91 And Angle <= 180 Then
Picture1.Line (RC_X(1), RC_Y(1))-((RC_X(1) - W(1)), (RC_Y(1) - h(1))), vbWhite
ElseIf Angle >= 181 And Angle <= 270 Then
Picture1.Line (RC_X(1), RC_Y(1))-((RC_X(1) - W(1)), (RC_Y(1) + h(1))), vbWhite
ElseIf Angle >= 271 And Angle <= 360 Then
Picture1.Line (RC_X(1), RC_Y(1))-((RC_X(1) + W(1)), (RC_Y(1) + h(1))), vbWhite
End If
Picture1.CurrentX = RC_X(1) - rad_R
Picture1.CurrentY = RC_Y(1) + rad_R
```

```
Picture1.Print "Robot 1"
End If
If Check2.Value = 1 Then
Angle = O_R(2)
W(2) = Int(Cos(Angle * dOneDegree) * rad_R)
If Sgn(W(2)) = -1 Then
W(2) = W(2) * (-1)
End If
h(2) = Int(Sin(Angle * dOneDegree) * rad_R)
If Sgn(h(2)) = -1 Then
h(2) = h(2) * (-1)
End If
Picture1.ScaleMode = 3
Picture1.Circle (RC_X(2), RC_Y(2)), rad_R, RGB(255, 255, 0)
If Angle >= 0 And Angle <= 90 Then
Picture1.Line (RC_X(2), RC_Y(2))-((RC_X(2) + W(2)), (RC_Y(2) - h(2))), vbWhite
ElseIf Angle >= 91 And Angle <= 180 Then
Picture1.Line (RC_X(2), RC_Y(2))-((RC_X(2) - W(2)), (RC_Y(2) - h(2))), vbWhite
ElseIf Angle >= 181 And Angle <= 270 Then
Picture1.Line (RC_X(2), RC_Y(2))-((RC_X(2) - W(2)), (RC_Y(2) + h(2))), vbWhite
ElseIf Angle >= 271 And Angle <= 360 Then
Picture1.Line (RC_X(2), RC_Y(2))-((RC_X(2) + W(2)), (RC_Y(2) + h(2))), vbWhite
End If
Picture1.CurrentX = RC_X(2) - rad_R
Picture1.CurrentY = RC_Y(2) + rad_R
Picture1.Print "Robot 2"
End If
If Check3.Value = 1 Then
Angle = O_R(3)
W(3) = Int(Cos(Angle * dOneDegree) * rad_R)
If Sgn(W(3)) = -1 Then
W(3) = W(3) * (-1)
```

```
End If
h(3) = Int(Sin(Angle * dOneDegree) * rad_R)
If Sgn(h(3)) = -1 Then
h(3) = h(3) * (-1)
End If
Picture1.ScaleMode = 3
Picture1.Circle (RC_X(3), RC_Y(3)), rad_R, RGB(255, 0, 255)
If Angle >= 0 And Angle <= 90 Then
Picture1.Line (RC_X(3), RC_Y(3))-((RC_X(3) + W(3)), (RC_Y(3) - h(3))), vbWhite
ElseIf Angle >= 91 And Angle <= 180 Then
Picture1.Line (RC_X(3), RC_Y(3))-((RC_X(3) - W(3)), (RC_Y(3) - h(3))), vbWhite
ElseIf Angle >= 181 And Angle <= 270 Then
Picture1.Line (RC_X(3), RC_Y(3))-((RC_X(3) - W(3)), (RC_Y(3) + h(3))), vbWhite
ElseIf Angle >= 271 And Angle <= 360 Then
Picture1.Line (RC_X(3), RC_Y(3))-((RC_X(3) + W(3)), (RC_Y(3) + h(3))), vbWhite
End If
Picture1.CurrentX = RC_X(3) - rad_R
Picture1.CurrentY = RC_Y(3) + rad_R
Picture1.Print "Robot 3"
End If
If Check4.Value = 1 Then
Picture1.ScaleMode = 3
Picture1.Circle (Cen_obx, Cen_oby), rad_O, RGB(0, 0, 255)
Picture1.CurrentX = Cen_obx - rad_R
Picture1.CurrentY = Cen_oby
Picture1.Print "Object"
End If
End Sub
```

โปรแกรมในส่วนของการรันส่วนของ GUI

```
Private Sub Command18_Click()
Do
```

```
DoEvents
quit = 0
'/------------------------ open file to save data position ----------------------/
If Check9.Value = 1 Then
Timer1.Enabled = True    '/ start read data position
End If
'/---------------------------- Jump to select Program --------------------------/
If Option6.Value = True Or Check1.Value = 1 Or Check2.Value = 1 Or Check3.Value = 1 Then
Call program1
End If
If Option7.Value = True And Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
Call program2
End If
If Option8.Value = True And Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
Call program3
End If
If Option9.Value = True And Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
Call program4
End If
'/---------------------------- return  loop until parameter quit =1  ----------------/
Loop Until quit = 1
End Sub

Sub program1()
If Check1.Value = 1 Then
'--------------robot1---------------
Call scancen1
Call cul_angle1
Call send_AI1
Robot_CR1_i = Robot_cenx(1)
Robot_CR1_j = Robot_ceny(1)
Call Error_point1
```

```
Call Control1
End If
If Check2.Value = 1 Then
'-----------------robot2-------------
Call scancen2
Call cul_angle2
Call send_AI2
Robot_CR2_i = Robot_cenx(2)
Robot_CR2_j = Robot_ceny(2)
Call Error_point2
Call Control2
End If
If Check3.Value = 1 Then
'--------------------robot3-----------
Call scancen3
Call cul_angle3
Call send_AI3
Robot_CR3_i = Robot_cenx(3)
Robot_CR3_j = Robot_ceny(3)
Call Error_point3
'Call error_dis3
'Call sum_error3
Call Control3
End If
End Sub

Sub program2()
If Check1.Value = 1 Then
'---------------robot1---------------
Call scancen1
Call cul_angle1
Call send_AI1
```

```
Robot_CR1_i = Robot_cenx(1)
Robot_CR1_j = Robot_ceny(1)
Call Error_point1
Call Control1
End If
If Check2.Value = 1 Then
'-----------------robot2-------------
Call scancen2
Call cul_angle2
Call send_AI2
Robot_CR2_i = Robot_cenx(2)
Robot_CR2_j = Robot_ceny(2)
Call Error_point2
'Call error_dis2
Call Control2
End If
If Check3.Value = 1 Then
'--------------------robot3-----------
Call scancen3
Call cul_angle3
Call send_AI3
Robot_CR3_i = Robot_cenx(3)
Robot_CR3_j = Robot_ceny(3)
Call Error_point3
Call Control3
End If
End Sub
Sub program3()
If Check1.Value = 1 Then
'---------------robot1---------------
Call scancen1
Call cul_angle1
```

```
Call send_AI1
Robot_CR1_i = Robot_cenx(1)
Robot_CR1_j = Robot_ceny(1)
Call Error_point1
Call Control1
End If
If Check2.Value = 1 Then
'----------robot2----------
Call scancen2
Call cul_angle2
Call send_AI2
Robot_CR2_i = Robot_cenx(2)
Robot_CR2_j = Robot_ceny(2)
Call Error_point2
Call Control2
End If
If Check3.Value = 1 Then
'----------robot3----------
Call scancen3
Call cul_angle3
Call send_AI3
Robot_CR3_i = Robot_cenx(3)
Robot_CR3_j = Robot_ceny(3)
Call Error_point3
Call Control3
End If
End Sub

Sub program4()
If Check1.Value = 1 Then
'----------robot1----------
Call scancen1
```

```
Call cul_angle1
Call send_AI1
Robot_CR1_i = Robot_cenx(1)
Robot_CR1_j = Robot_ceny(1)
Call Error_point1
Call Control1
End If
'----------------robot2------------
If Check2.Value = 1 Then
Call scancen2
Call cul_angle2
Call send_AI2
Robot_CR2_i = Robot_cenx(2)
Robot_CR2_j = Robot_ceny(2)
Call Error_point2
Call Control2
End If
'--------------------robot3----------
If Check3.Value = 1 Then
Call scancen3
Call cul_angle3
Call send_AI3
Robot_CR3_i = Robot_cenx(3)
Robot_CR3_j = Robot_ceny(3)
Call Error_point3
Call Control3
End If
End Sub

Sub scancen1()
Dim i As Integer, j As Integer
Dim Color1 As Long, r As Integer, G As Integer, b As Integer
```

```
Dim i_1 As Long, i_2 As Long, j_1 As Long, j_2 As Long
 Robot_cenx(1) = 0
 Robot_ceny(1) = 0
 Count_cen(1) = 0
 Robot_Topx(1) = 0
 Robot_Topy(1) = 0
 Count_top(1) = 0
 R_min_t = RMIT.Text
 R_max_t = RMAT.Text
 G_min_t = GMIT.Text
 G_max_t = GMAT.Text
 B_min_t = BMIT.Text
 B_max_t = BMAT.Text
  R_ROBOT = Text35.Text
'----------------Robot 1 -------------------
If Check1.Value = 1 Then
Do
DoEvents
BitBlt Picture3.hdc, 0, 0, 373, 309, Picture2.hdc, 0, 0, vbSrcCopy
R_ROBOT = R_ROBOT + 5
i_1 = Robot_CR1_i - R_ROBOT
i_2 = Robot_CR1_i + R_ROBOT
j_1 = Robot_CR1_j - R_ROBOT
j_2 = Robot_CR1_j + R_ROBOT
For i = i_1 To i_2
For j = j_1 To j_2
Color1 = GetPixel(Picture3.hdc, i, j)
Call ColorRGB(Color1, r, G, b)
If (r + G + b) <= (R_max_R1 + G_max_R1 + B_max_R1) And r >= R_min_R1 And r <= R_max_R1 And G >=
G_min_R1 And G <= G_max_R1 And b >= B_min_R1 And b <= B_max_R1 Then
Color1 = 255
Robot_cenx(1) = Robot_cenx(1) + i
```

```
Robot_ceny(1) = Robot_ceny(1) + j
Count_cen(1) = Count_cen(1) + 1
ElseIf (r + G + b) <= (R_max_t + G_max_t + B_max_t) And r >= R_min_t And r <= R_max_t And G >= G_min_t
And G <= G_max_t And b >= B_min_t And b <= B_max_t Then
Color1 = 16777215
Robot_Topx(1) = Robot_Topx(1) + i
Robot_Topy(1) = Robot_Topy(1) + j
Count_top(1) = Count_top(1) + 1
Else
Color1 = 0
End If
SetPixel Picture3.hdc, i, j, Color1
DoEvents
Next j
Next i
Loop While Count_cen(1) = 0 Or Count_top(1) = 0
End If
R_ROBOT = Text35.Text
If Check1.Value = 1 Then
Robot_cenx(1) = Robot_cenx(1) / Count_cen(1)
Robot_ceny(1) = Robot_ceny(1) / Count_cen(1)
Text14.Text = Robot_cenx(1)
Text15.Text = Robot_ceny(1)
Robot_Topx(1) = Robot_Topx(1) / Count_top(1)
Robot_Topy(1) = Robot_Topy(1) / Count_top(1)
End If
End Sub

Sub scancen2()
Dim i As Integer, j As Integer
Dim Color1 As Long, r As Integer, G As Integer, b As Integer
Dim i_1 As Long, i_2 As Long, j_1 As Long, j_2 As Long
```

```
Robot_cenx(2) = 0
Robot_ceny(2) = 0
Count_cen(2) = 0
Robot_Topx(2) = 0
Robot_Topy(2) = 0
Count_top(2) = 0
R_min_t = RMIT.Text
R_max_t = RMAT.Text
G_min_t = GMIT.Text
G_max_t = GMAT.Text
B_min_t = BMIT.Text
B_max_t = BMAT.Text
R_ROBOT = Text35.Text
'----------Robot 2 ----------
If Check2.Value = 1 Then
Do
DoEvents
BitBlt Picture3.hdc, 0, 0, 373, 309, Picture2.hdc, 0, 0, vbSrcCopy
R_ROBOT = R_ROBOT + 5
i_1 = Robot_CR2_i - R_ROBOT
i_2 = Robot_CR2_i + R_ROBOT
j_1 = Robot_CR2_j - R_ROBOT
j_2 = Robot_CR2_j + R_ROBOT
For i = i_1 To i_2
For j = j_1 To j_2
Color1 = GetPixel(Picture3.hdc, i, j)
Call ColorRGB(Color1, r, G, b)
If (r + G + b) <= (R_max_R2 + G_max_R2 + B_max_R2) And r >= R_min_R2 And r <= R_max_R2 And G >=
G_min_R2 And G <= G_max_R2 And b >= B_min_R2 And b <= B_max_R2 Then
Color1 = 589823
Robot_cenx(2) = Robot_cenx(2) + i
Robot_ceny(2) = Robot_ceny(2) + j
Count_cen(2) = Count_cen(2) + 1
ElseIf (r + G + b) <= (R_max_t + G_max_t + B_max_t) And r >= R_min_t And r <= R_max_t And G >= G_min_t
And G <= G_max_t And b >= B_min_t And b <= B_max_t Then
Color1 = 16777215
Robot_Topx(2) = Robot_Topx(2) + i
Robot_Topy(2) = Robot_Topy(2) + j
```

```
Count_top(2) = Count_top(2) + 1
Else
Color1 = 0
End If
SetPixel Picture3.hdc, i, j, Color1
DoEvents
Next j
Next i
Loop While Count_cen(2) = 0 Or Count_top(2) = 0
End If
R_ROBOT = Text35.Text
'-------cul position -------
If Check2.Value = 1 Then
Robot_cenx(2) = Robot_cenx(2) / Count_cen(2)
Robot_ceny(2) = Robot_ceny(2) / Count_cen(2)
Text16.Text = Robot_cenx(2)
Text17.Text = Robot_ceny(2)
Robot_Topx(2) = Robot_Topx(2) / Count_top(2)
Robot_Topy(2) = Robot_Topy(2) / Count_top(2)
End If
End Sub

Sub cul_angle1()
DoEvents
Tx(1) = Robot_Topx(1)
Ty(1) = Robot_Topy(1)
If Check1.Value = 1 Then
Px(1) = Robot_CR1_i
Py(1) = Robot_CR1_j
Dx(1) = Tx(1) - Px(1)
Dy(1) = Ty(1) - Py(1)
If Dx(1) = 0 Then
```

```
Dx(1) = 1
End If
angle_R(1) = Int(((Atn(Dy(1) / Dx(1))) * 180) / pi)
If Dx(1) >= 0 And Dy(1) <= 0 Then
O_R(1) = angle_R(1) * (-1)
ElseIf Dx(1) <= 0 And Dy(1) <= 0 Then
O_R(1) = (90 - angle_R(1)) + 90
ElseIf Dx(1) <= 0 And Dy(1) >= 0 Then
O_R(1) = (angle_R(1) * (-1)) + 180
ElseIf Dx(1) >= 0 And Dy(1) >= 0 Then
O_R(1) = 360 - angle_R(1)
End If
Text22.Text = O_R(1)
End If
End Sub

Sub cul_angle2()
DoEvents
Tx(2) = Robot_Topx(2)
Ty(2) = Robot_Topy(2)
If Check2.Value = 1 Then
Px(2) = Robot_CR2_i
Py(2) = Robot_CR2_j
Dx(2) = Tx(2) - Px(2)
Dy(2) = Ty(2) - Py(2)
If Dx(2) = 0 Then
Dx(2) = 1
End If
angle_R(2) = Int(((Atn(Dy(2) / Dx(2))) * 180) / pi)
If Dx(2) >= 0 And Dy(2) <= 0 Then
O_R(2) = angle_R(2) * (-1)
ElseIf Dx(2) <= 0 And Dy(2) <= 0 Then
```

```
O_R(2) = (90 - angle_R(2)) + 90
ElseIf Dx(2) <= 0 And Dy(2) >= 0 Then
O_R(2) = (angle_R(2) * (-1)) + 180
ElseIf Dx(2) >= 0 And Dy(2) >= 0 Then
O_R(2) = 360 - angle_R(2)
End If
Text23.Text = O_R(2)
End If
End Sub

Sub cul_angle2()
DoEvents
Tx(2) = Robot_Topx(2)
Ty(2) = Robot_Topy(2)
If Check2.Value = 1 Then
Px(2) = Robot_CR2_i
Py(2) = Robot_CR2_j
Dx(2) = Tx(2) - Px(2)
Dy(2) = Ty(2) - Py(2)
If Dx(2) = 0 Then
Dx(2) = 1
End If
angle_R(2) = Int(((Atn(Dy(2) / Dx(2))) * 180) / pi)
If Dx(2) >= 0 And Dy(2) <= 0 Then
O_R(2) = angle_R(2) * (-1)
ElseIf Dx(2) <= 0 And Dy(2) <= 0 Then
O_R(2) = (90 - angle_R(2)) + 90
ElseIf Dx(2) <= 0 And Dy(2) >= 0 Then
O_R(2) = (angle_R(2) * (-1)) + 180
ElseIf Dx(2) >= 0 And Dy(2) >= 0 Then
O_R(2) = 360 - angle_R(2)
End If
```

```
Text23.Text = O_R(2)
End If
End Sub

Sub send_AI1()
'----------clear display  AI ----------'
Picture1.Cls
'----------------------------------------
DoEvents
rad_R = Text32.Text
rad_O = Text33.Text
RC_X(1) = Robot_cenx(1)
RC_Y(1) = Robot_ceny(1)
If Check1.Value = 1 Then
Angle = O_R(1)
W(1) = Int(Cos(Angle * dOneDegree) * rad_R)
If Sgn(W(1)) = -1 Then
W(1) = W(1) * (-1)
End If
h(1) = Int(Sin(Angle * dOneDegree) * rad_R)
If Sgn(h(1)) = -1 Then
h(1) = h(1) * (-1)
End If
Picture1.ScaleMode = 3
Picture1.Circle (RC_X(1), RC_Y(1)), rad_R, RGB(255, 0, 0)
If Angle >= 0 And Angle <= 90 Then
Picture1.Line (RC_X(1), RC_Y(1))-((RC_X(1) + W(1)), (RC_Y(1) - h(1))), RGB(0, 0, 0)
ElseIf Angle >= 91 And Angle <= 180 Then
Picture1.Line (RC_X(1), RC_Y(1))-((RC_X(1) - W(1)), (RC_Y(1) - h(1))), RGB(0, 0, 0)
ElseIf Angle >= 181 And Angle <= 270 Then
Picture1.Line (RC_X(1), RC_Y(1))-((RC_X(1) - W(1)), (RC_Y(1) + h(1))), RGB(0, 0, 0)
ElseIf Angle >= 271 And Angle <= 360 Then
```

```
Picture1.Line (RC_X(1), RC_Y(1))-((RC_X(1) + W(1)), (RC_Y(1) + h(1))), RGB(0, 0, 0)
End If
Picture1.CurrentX = RC_X(1) - rad_R
Picture1.CurrentY = RC_Y(1) + rad_R
Picture1.Print "Rc bot 1"
End If
End Sub

Sub send_AI2()
'----------clear display  AI ----------'
Picture1.Cls
'----------------------------------------
DoEvents
rad_R = Text32.Text
rad_O = Text33.Text
RC_X(2) = Robot_cenx(2)
RC_Y(2) = Robot_ceny(2)
If Check2.Value = 1 Then
Angle = O_R(2)
W(2) = Int(Cos(Angle * dOneDegree) * rad_R)
If Sgn(W(2)) = -1 Then
W(2) = W(2) * (-1)
End If
h(2) = Int(Sin(Angle * dOneDegree) * rad_R)
If Sgn(h(2)) = -1 Then
h(2) = h(2) * (-1)
End If
Picture1.ScaleMode = 3
Picture1.Circle (RC_X(2), RC_Y(2)), rad_R, RGB(255, 255, 0)
If Angle >= 0 And Angle <= 90 Then
Picture1.Line (RC_X(2), RC_Y(2))-((RC_X(2) + W(2)), (RC_Y(2) - h(2))), vbWhite
ElseIf Angle >= 91 And Angle <= 180 Then
```

```
Picture1.Line (RC_X(2), RC_Y(2))-((RC_X(2) - W(2)), (RC_Y(2) - h(2))), vbWhite
ElseIf Angle >= 181 And Angle <= 270 Then
Picture1.Line (RC_X(2), RC_Y(2))-((RC_X(2) - W(2)), (RC_Y(2) + h(2))), vbWhite
ElseIf Angle >= 271 And Angle <= 360 Then
Picture1.Line (RC_X(2), RC_Y(2))-((RC_X(2) + W(2)), (RC_Y(2) + h(2))), vbWhite
End If
Picture1.CurrentX = RC_X(2) - rad_R
Picture1.CurrentY = RC_Y(2) + rad_R
Picture1.Print "Robot 2"
End If
End Sub

Sub send_AI3()
'----------clear display  AI ----------'
Picture1.Cls
'---------------------------------------
DoEvents
rad_R = Text32.Text
rad_O = Text33.Text
RC_X(3) = Robot_cenx(3)
RC_Y(3) = Robot_ceny(3)
If Check3.Value = 1 Then
Angle = O_R(3)
W(3) = Int(Cos(Angle * dOneDegree) * rad_R)
If Sgn(W(3)) = -1 Then
W(3) = W(3) * (-1)
End If
h(3) = Int(Sin(Angle * dOneDegree) * rad_R)
If Sgn(h(3)) = -1 Then
h(3) = h(3) * (-1)
End If
Picture1.ScaleMode = 3
```

```
Picture1.Circle (RC_X(3), RC_Y(3)), rad_R, RGB(255, 0, 255)
If Angle >= 0 And Angle <= 90 Then
Picture1.Line (RC_X(3), RC_Y(3))-((RC_X(3) + W(3)), (RC_Y(3) - h(3))), vbWhite
ElseIf Angle >= 91 And Angle <= 180 Then
Picture1.Line (RC_X(3), RC_Y(3))-((RC_X(3) - W(3)), (RC_Y(3) - h(3))), vbWhite
ElseIf Angle >= 181 And Angle <= 270 Then
Picture1.Line (RC_X(3), RC_Y(3))-((RC_X(3) - W(3)), (RC_Y(3) + h(3))), vbWhite
ElseIf Angle >= 271 And Angle <= 360 Then
Picture1.Line (RC_X(3), RC_Y(3))-((RC_X(3) + W(3)), (RC_Y(3) + h(3))), vbWhite
End If
Picture1.CurrentX = RC_X(3) - rad_R
Picture1.CurrentY = RC_Y(3) + rad_R
Picture1.Print "Robot 3"
End If
End Sub

Sub send_AI4()
'-----------clear display  AI ------------'
Picture1.Cls
'----------------------------------------
DoEvents
rad_O = Text33.Text
If Check4.Value = 1 Then
Picture1.ScaleMode = 3
Picture1.Circle (Cen_obx, Cen_oby), rad_O, RGB(0, 0, 255)
Picture1.CurrentX = Cen_obx - rad_R
Picture1.CurrentY = Cen_oby
Picture1.Print "Object"
End If
End Sub

Sub Error_point1()
```

```
Pmx(1) = Text6.Text
Pmy(1) = Text7.Text
Tmx(1) = Text14.Text
Tmy(1) = Text15.Text
If Check1.Value = 1 Then
Dmx(1) = Pmx(1) - Tmx(1)
Dmy(1) = Pmy(1) - Tmy(1)
If Dmx(1) = 0 Then
Dmx(1) = 1
End If
anglem_R(1) = Int(((Atn(Dmy(1) / Dmx(1))) * 180) / pi)
If Dmx(1) >= 0 And Dmy(1) <= 0 Then
Om_R(1) = anglem_R(1) * (-1)
ElseIf Dmx(1) <= 0 And Dmy(1) <= 0 Then
Om_R(1) = (90 - anglem_R(1)) + 90
ElseIf Dmx(1) <= 0 And Dmy(1) >= 0 Then
Om_R(1) = (anglem_R(1) * (-1)) + 180
ElseIf Dmx(1) >= 0 And Dmy(1) >= 0 Then
Om_R(1) = 360 - anglem_R(1)
End If
Text25.Text = Om_R(1)
dis_er(1) = Int(Sqr((Dmx(1) * Dmx(1)) + (Dmy(1) * Dmy(1))))
Text28.Text = dis_er(1)
End If
End Sub

Sub Error_point2()
Pmx(2) = Text8.Text
Pmy(2) = Text9.Text
Tmx(2) = Text16.Text
Tmy(2) = Text17.Text
```

```
If Check2.Value = 1 Then
Dmx(2) = Pmx(2) - Tmx(2)
Dmy(2) = Pmy(2) - Tmy(2)
If Dmx(2) = 0 Then
Dmx(2) = 1
End If
anglem_R(2) = Int(((Atn(Dmy(2) / Dmx(2))) * 180) / pi)
If Dmx(2) >= 0 And Dmy(2) <= 0 Then
Om_R(2) = anglem_R(2) * (-1)
ElseIf Dmx(2) <= 0 And Dmy(2) <= 0 Then
Om_R(2) = (90 - anglem_R(2)) + 90
ElseIf Dmx(2) <= 0 And Dmy(2) >= 0 Then
Om_R(2) = (anglem_R(2) * (-1)) + 180
ElseIf Dmx(2) >= 0 And Dmy(2) >= 0 Then
Om_R(2) = 360 - anglem_R(2)
End If
Text26.Text = Om_R(2)
dis_er(2) = Int(Sqr((Dmx(2) * Dmx(2)) + (Dmy(2) * Dmy(2))))
Text29.Text = dis_er(2)
End If
End Sub

Sub Error_point3()
Pmx(3) = Text10.Text
Pmy(3) = Text11.Text
Tmx(3) = Text18.Text
Tmy(3) = Text19.Text
If Check3.Value = 1 Then
Dmx(3) = Pmx(3) - Tmx(3)
Dmy(3) = Pmy(3) - Tmy(3)
```

```
If Dmx(3) = 0 Then
Dmx(3) = 1
End If
anglem_R(3) = Int(((Atn(Dmy(3) / Dmx(3))) * 180) / pi)
If Dmx(3) >= 0 And Dmy(3) <= 0 Then
Om_R(3) = anglem_R(3) * (-1)
ElseIf Dmx(3) <= 0 And Dmy(3) <= 0 Then
Om_R(3) = (90 - anglem_R(3)) + 90
ElseIf Dmx(3) <= 0 And Dmy(3) >= 0 Then
Om_R(3) = (anglem_R(3) * (-1)) + 180
ElseIf Dmx(3) >= 0 And Dmy(3) >= 0 Then
Om_R(3) = 360 - anglem_R(3)
End If
Text27.Text = Om_R(3)
dis_er(3) = Int(Sqr((Dmx(3) * Dmx(3)) + (Dmy(3) * Dmy(3))))
Text30.Text = dis_er(3)
End If
End Sub

Sub Control1()
M_an1 = Text37.Text
M_an2 = Text38.Text
an_sl1 = Text41.Text
an_sl2 = Text42.Text
distance = Text39.Text
W_R = Text43.Text
If Check1.Value = 1 Then
If dis_er(1) < distance Then
Out &H378, &H9F
ElseIf Om_R(1) > O_R(1) Then
If (Om_R(1) - O_R(1)) < M_an1 Then
'//driver_speed GO
```

```
Out &H378, &H9E
ElseIf (Om_R(1) - O_R(1)) < an_sl1 Then
'//driver_slow left
Out &H378, &H9B
ElseIf (Om_R(1) - O_R(1)) <= 180 Then
'//driver_speed left
Out &H378, &H8F
ElseIf (Om_R(1) - O_R(1)) > M_an2 Then
'//driver_GO
Out &H378, &H9E
ElseIf (Om_R(1) - O_R(1)) > an_sl2 Then
'//driver_slow Right
Out &H378, &H97
ElseIf (Om_R(1) - O_R(1)) > 180 Then
'//driver_speed Right
Out &H378, &H93
End If
ElseIf Om_R(1) < O_R(1) Then
If (Abs(Om_R(1) - O_R(1))) < M_an1 Then
'//driver_ GO
Out &H378, &H9E
ElseIf (Abs(Om_R(1) - O_R(1))) < an_sl1 Then
'//driver_slow Right
Out &H378, &H97
ElseIf (Abs(Om_R(1) - O_R(1))) <= 180 Then
'//driver_speed Right
Out &H378, &H93
ElseIf (Abs(Om_R(1) - O_R(1))) > M_an2 Then
'//driver_GO
Out &H378, &H9E
ElseIf (Abs(Om_R(1) - O_R(1))) > an_sl2 Then
'//driver_slow left
```

```
Out &H378, &H9B
ElseIf (Abs(Om_R(1) - O_R(1))) > 180 Then
'//driver_speed left
Out &H378, &H8F
End If
End If
End If
End Sub

Sub Control2()
M_an1 = Text37.Text
M_an2 = Text38.Text
an_sl1 = Text41.Text
an_sl2 = Text42.Text
distance = Text39.Text
W_R = Text43.Text
error_robot = ((2 * W_R) + (2 * rad_R))
If Check2.Value = 1 Then
If dis_er(2) < distance Then
Out &H378, &HBF
'///////////////// set  angle robot /////////////////
' If Option6.Value = True Or Option7.Value = True Or Option9.Value = True Then
' Call angle_rr2
' End If
'////////////////////////////////////////////////////
ElseIf Om_R(2) > O_R(2) Then
If (Om_R(2) - O_R(2)) < M_an1 Then
'//driver_speed GO
Out &H378, &HBE
ElseIf (Om_R(2) - O_R(2)) < an_sl1 Then
'//driver_slow left
Out &H378, &HBB
```

```
ElseIf (Om_R(2) - O_R(2)) <= 180 Then
'//driver_speed left
Out &H378, &HAF
ElseIf (Om_R(2) - O_R(2)) > M_an2 Then
'//driver_GO
Out &H378, &HBE
ElseIf (Om_R(2) - O_R(2)) > an_sl2 Then
'//driver_slow Right
Out &H378, &HB7
ElseIf (Om_R(2) - O_R(2)) > 180 Then
'//driver_speed Right
Out &H378, &HB3
End If
ElseIf Om_R(2) < O_R(2) Then
If (Abs(Om_R(2) - O_R(2))) < M_an1 Then
'//driver_ GO
Out &H378, &HBE
ElseIf (Abs(Om_R(2) - O_R(2))) < an_sl1 Then
'//driver_slow Right
Out &H378, &HB7
ElseIf (Abs(Om_R(2) - O_R(2))) <= 180 Then
'//driver_speed Right
Out &H378, &HB3
ElseIf (Abs(Om_R(2) - O_R(2))) > M_an2 Then
'//driver_GO
Out &H378, &HBE
ElseIf (Abs(Om_R(2) - O_R(2))) > an_sl2 Then
'//driver_slow left
Out &H378, &HBB
ElseIf (Abs(Om_R(2) - O_R(2))) > 180 Then
'//driver_speed left
Out &H378, &HAF
```

```
End If
End If
End If
End Sub

Sub Control3()
M_an1 = Text37.Text
M_an2 = Text38.Text
an_sl1 = Text41.Text
an_sl2 = Text42.Text
distance = Text39.Text
W_R = Text43.Text
error_robot = ((2 * W_R) + (2 * rad_R))
If Check3.Value = 1 Then
If dis_er(3) < distance Then
Out &H378, &HDF
"////////////////// set  angle robot //////////////////
'If Option6.Value = True Or Option7.Value = True Or Option9.Value = True Then
'Call angle_rr3
'End If
'////////////////////////////////////////////////
ElseIf Om_R(3) > O_R(3) Then
If (Om_R(3) - O_R(3)) < M_an1 Then
'//driver_speed GO
Out &H378, &HDE
ElseIf (Om_R(3) - O_R(3)) < an_sl1 Then
'//driver_slow left
Out &H378, &HDB
ElseIf (Om_R(3) - O_R(3)) <= 180 Then
'//driver_speed left
Out &H378, &HCF
ElseIf (Om_R(3) - O_R(3)) > M_an2 Then
```

```
'//driver_GO
Out &H378, &HDE
ElseIf (Om_R(3) - O_R(3)) > an_sl2 Then
'//driver_slow Right
Out &H378, &HD7
ElseIf (Om_R(3) - O_R(3)) > 180 Then
'//driver_speed Right
Out &H378, &HD3
End If
ElseIf Om_R(3) < O_R(3) Then
If (Abs(Om_R(3) - O_R(3))) < M_an1 Then
'//driver_ GO
Out &H378, &HDE
ElseIf (Abs(Om_R(3) - O_R(3))) < an_sl1 Then
'//driver_slow Right
Out &H378, &HD7
ElseIf (Abs(Om_R(3) - O_R(3))) <= 180 Then
'//driver_speed Right
Out &H378, &HD3
ElseIf (Abs(Om_R(3) - O_R(3))) > M_an2 Then
'//driver_GO
Out &H378, &HDE
ElseIf (Abs(Om_R(3) - O_R(3))) > an_sl2 Then
'//driver_slow left
Out &H378, &HDB
ElseIf (Abs(Om_R(3) - O_R(3))) > 180 Then
'//driver_speed left
Out &H378, &HCF
End If
End If
End If
End Sub
```

```
Private Sub Command5_Click()
quit = 1
jump1 = 1
jump2 = 1
If Check9.Value = 1 Then
Timer1.Enabled = False
If Check1.Value = 1 Then
Close #1
Close #2
End If
If Check2.Value = 1 Then
Close #3
Close #4
End If
If Check3.Value = 1 Then
Close #5
Close #6
End If
If Check4.Value = 1 Then
Close #7
Close #8
End If
End If
Out &H378, &H9F
Call delay
Out &H378, &HBF
Call delay
Out &H378, &HDF
Call delay
End Sub
```

```
Private Sub Command14_Click()
'/----------------------  open file to save data position ----------------------/
If Check9.Value = 1 Then
Timer1.Enabled = True    '/ start read data position
End If
If Option10.Value = True And Check1.Value = 1 Then
'-----------------------------------
jump1 = 0
Do
DoEvents
Call program5
'-----------check loop ---------
If dis_rm < distance Then
jump1 = 1
End If
If quit = 1 Then
jump1 = 1
End If
'----------end check ------------
Loop Until jump1 = 1
Call move_ob
'-----------------------------------
End If
End Sub

Sub program5()
'---------------robot1---------------
Call scancen1
Call cul_angle1
Call send_AI1
```

```
Robot_CR1_i = Robot_cenx(1)
Robot_CR1_j = Robot_ceny(1)
'------------------object---------------
Call scancen4
Call cul_ob
OB_i = Cen_obx
OB_j = Cen_oby
'----------------control----------------
Call sum_ob
Call Control11
'----------------------'-
End Sub

Sub scancen4()
Dim i As Integer, j As Integer
Dim Color1 As Long, r As Integer, G As Integer, b As Integer
Dim i_1 As Long, i_2 As Long, j_1 As Long, j_2 As Long
Cen_obx = 0
Cen_oby = 0
Count_cenob = 0
R_min_t = RMIT.Text
R_max_t = RMAT.Text
G_min_t = GMIT.Text
G_max_t = GMAT.Text
B_min_t = BMIT.Text
B_max_t = BMAT.Text
R_OB = Text34.Text
'----------------object---------------------
If Check4.Value = 1 Then
Do
DoEvents
BitBlt Picture3.hdc, 0, 0, 373, 309, Picture2.hdc, 0, 0, vbSrcCopy
```

```
R_OB = R_OB + 5
i_1 = OB_i - R_OB
i_2 = OB_i + R_OB
j_1 = OB_j - R_OB
j_2 = OB_j + R_OB
For i = i_1 To i_2
For j = j_1 To j_2
Color1 = GetPixel(Picture3.hdc, i, j)
Call ColorRGB(Color1, r, G, b)
If (r + G + b) <= (R_max_Ob + G_max_Ob + B_max_Ob) And r >= R_min_Ob And r <= R_max_Ob And G >=
G_min_Ob And G <= G_max_Ob And b >= B_min_Ob And b <= B_max_Ob Then
Color1 = 16711680
Cen_obx = Cen_obx + i
Cen_oby = Cen_oby + j
Count_cenob = Count_cenob + 1
Else
Color1 = 0
End If
SetPixel Picture3.hdc, i, j, Color1
DoEvents
Next j
Next i
Loop While Count_cenob = 0
End If
'-------cul position -------
If Check4.Value = 1 Then
Cen_obx = Cen_obx / Count_cenob
Cen_oby = Cen_oby / Count_cenob
Text20.Text = Cen_obx
Text21.Text = Cen_oby
Picture1.Circle (Cen_obx, Cen_oby), rad_O, RGB(0, 0, 255)
Picture1.CurrentX = Cen_obx - rad_R
```

```
Picture1.CurrentY = Cen_oby
Picture1.Print "Object"
End If
End Sub

Sub move_ob()
jump2 = 0
'---------------------------------------
Text14.Text = Robot_CR1_i
Text15.Text = Robot_CR1_j
Text20.Text = OB_i
Text21.Text = OB_j
'---------------------------------------
Do
DoEvents
Call scancen1
Call cul_angle1
Call send_AI1
Robot_CR1_i = Robot_cenx(1)
Robot_CR1_j = Robot_ceny(1)
Call scancen4
Call cul_ob
OB_i = Cen_obx
OB_j = Cen_oby
Call Control13
Call error_control
Text31.Text = dis_obs
'---------- check  loop----------
If dis_obs < distance Then
Out &H378, &H9F
jump2 = 1
quit = 1
```

```
End If
'-----------------------------------
Loop Until jump2 = 1
'------------ return position-----------------
Text14.Text = Robot_CR1_i
Text15.Text = Robot_CR1_j
Text20.Text = OB_i
Text21.Text = OB_j
Robot_CR1_i = Robot_cenx(1)
Robot_CR1_j = Robot_ceny(1)
OB_i = Cen_obx
OB_j = Cen_oby
'---------- ----------------------
Timer1.Enabled = False
If Check1.Value = 1 Then
Close #1
Close #2
End If
If Check2.Value = 1 Then
Close #3
Close #4
End If
If Check3.Value = 1 Then
Close #5
Close #6
End If
If Check4.Value = 1 Then
Close #7
Close #8
End If
'-----------------------------------------
Out &H378, &H9F
```

```
End Sub

Sub cul_ob()
Cen_obx = Text20.Text
Cen_oby = Text21.Text
C_obmx = Text12.Text
C_obmy = Text13.Text
dis_obx = (C_obmx - Cen_obx)
dis_oby = (C_obmy - Cen_oby)
If dis_obx = 0 Then
dis_obx = 1
End If
angle_obs = Int(((Atn(dis_oby / dis_obx)) * 180) / pi)
If dis_obx >= 0 And dis_oby <= 0 Then
O_ob = angle_obs * (-1)
ElseIf dis_obx <= 0 And dis_oby <= 0 Then
O_ob = (90 - angle_obs) + 90
ElseIf dis_obx <= 0 And dis_oby >= 0 Then
O_ob = (angle_obs * (-1)) + 180
ElseIf dis_obx >= 0 And dis_oby >= 0 Then
O_ob = 360 - angle_obs
End If
Text79.Text = O_ob
dis_obs = Int(Sqr((dis_obx * dis_obx) + (dis_oby * dis_oby)))
Text31.Text = dis_obs
OB_i = Cen_obx
OB_j = Cen_oby
If O_ob >= 0 And O_ob <= 180 Then
Angle = O_ob + 180
ElseIf O_ob > 180 And O_ob <= 360 Then
Angle = O_ob - 180
End If
```

```
W(4) = Int(Cos(Angle * dOneDegree) * (rad_O * 3))
If Sgn(W(4)) = -1 Then
W(4) = W(4) * (-1)
End If
h(4) = Int(Sin(Angle * dOneDegree) * (rad_O * 3))
If Sgn(h(4)) = -1 Then
h(4) = h(4) * (-1)
End If
Picture1.ScaleMode = 3
If Angle >= 0 And Angle <= 90 Then
Picture1.Line (Cen_obx, Cen_oby)-((Cen_obx + W(4)), (Cen_oby - h(4))), RGB(255, 0, 0)
R_mx = (Cen_obx + W(4))
R_my = (Cen_oby - h(4))
ElseIf Angle >= 91 And Angle <= 180 Then
Picture1.Line (Cen_obx, Cen_oby)-((Cen_obx - W(4)), (Cen_oby - h(4))), RGB(255, 0, 0)
R_mx = (Cen_obx - W(4))
R_my = (Cen_oby - h(4))
ElseIf Angle >= 181 And Angle <= 270 Then
Picture1.Line (Cen_obx, Cen_oby)-((Cen_obx - W(4)), (Cen_oby + h(4))), RGB(255, 0, 0)
R_mx = (Cen_obx - W(4))
R_my = (Cen_oby + h(4))
ElseIf Angle >= 271 And Angle <= 360 Then
Picture1.Line (Cen_obx, Cen_oby)-((Cen_obx + W(4)), (Cen_oby + h(4))), RGB(255, 0, 0)
R_mx = (Cen_obx + W(4))
R_my = (Cen_oby + h(4))
End If
Text6.Text = R_mx
Text7.Text = R_my
'--------------------------------------------------------'
dis_rox = (Cen_obx - Robot_cenx(1))
dis_roy = (Cen_oby - Robot_ceny(1))
If dis_rox = 0 Then
```

```
dis_rox = 1
End If
angle_ro = Int(((Atn(dis_roy / dis_rox)) * 180) / pi)
If dis_rox >= 0 And dis_roy <= 0 Then
O_ro = angle_ro * (-1)
ElseIf dis_rox <= 0 And dis_roy <= 0 Then
O_ro = (90 - angle_ro) + 90
ElseIf dis_rox <= 0 And dis_roy >= 0 Then
O_ro = (angle_ro * (-1)) + 180
ElseIf dis_rox >= 0 And dis_roy >= 0 Then
O_ro = 360 - angle_ro
End If
Text60.Text = O_ro
'-------------------------------------------------'
dis_ro = Int(Sqr((dis_rox * dis_rox) + (dis_roy * dis_roy)))
Text62.Text = dis_ro
'-------------------------------------------------'
OB_i = Cen_obx
OB_j = Cen_oby
'-------------------------------------------------'
dis_rmx = (R_mx - Robot_cenx(1))
dis_rmy = (R_my - Robot_ceny(1))
If dis_rmx = 0 Then
dis_rmx = 1
End If
angle_rm = Int(((Atn(dis_rmy / dis_rmx)) * 180) / pi)
If dis_rmx >= 0 And dis_rmy <= 0 Then
O_rm = angle_rm * (-1)
ElseIf dis_rmx <= 0 And dis_rmy <= 0 Then
O_rm = (90 - angle_rm) + 90
ElseIf dis_rmx <= 0 And dis_rmy >= 0 Then
O_rm = (angle_rm * (-1)) + 180
```

```
ElseIf dis_rmx >= 0 And dis_rmy >= 0 Then
O_rm = 360 - angle_rm
End If
Text63.Text = O_rm
dis_rm = Int(Sqr((dis_rmx * dis_rmx) + (dis_rmy * dis_rmy)))
Text61.Text = dis_rm
dis_rpx = (C_obmx - Robot_cenx(1))
dis_rpy = (C_obmy - Robot_ceny(1))
If dis_rpx = 0 Then
dis_rpx = 1
End If
angle_rp = Int(((Atn(dis_rpy / dis_rpx)) * 180) / pi)
If dis_rpx >= 0 And dis_rpy <= 0 Then
O_rp = angle_rp * (-1)
ElseIf dis_rpx <= 0 And dis_rpy <= 0 Then
O_rp = (90 - angle_rp) + 90
ElseIf dis_rpx <= 0 And dis_rpy >= 0 Then
O_rp = (angle_rp * (-1)) + 180
ElseIf dis_rpx >= 0 And dis_rpy >= 0 Then
O_rp = 360 - angle_rp
End If
Text72.Text = O_rp
dis_rp = Int(Sqr((dis_rpx * dis_rpx) + (dis_rpy * dis_rpy)))
Text65.Text = dis_rp
Oer_RO = O_ro - O_rm
End Sub

Sub Control13()
M_an1 = Text37.Text
M_an2 = Text38.Text
an_sl1 = Text41.Text
an_sl2 = Text42.Text
```

```
distance = Text39.Text
W_R = Text43.Text
If Check1.Value = 1 Then
If dis_obs < distance Then
Out &H378, &H9F
ElseIf O_rp > O_R(1) Then
If (O_rp - O_R(1)) < M_an1 Then
'//driver_speed GO
Out &H378, &H9E
ElseIf (O_rp - O_R(1)) < an_sl1 Then
'//driver_slow left
Out &H378, &H9B
ElseIf (O_rp - O_R(1)) <= 180 Then
'//driver_speed left
Out &H378, &H8F
ElseIf (O_rp - O_R(1)) > M_an2 Then
'//driver_GO
Out &H378, &H9E
ElseIf (O_rp - O_R(1)) > an_sl2 Then
'//driver_slow Right
Out &H378, &H97
ElseIf (O_rp - O_R(1)) > 180 Then
'//driver_speed Right
Out &H378, &H93
End If
ElseIf O_rp < O_R(1) Then
If (Abs(O_rp - O_R(1))) < M_an1 Then
'//driver_ GO
Out &H378, &H9E
ElseIf (Abs(O_rp - O_R(1))) < an_sl1 Then
'//driver_slow Right
Out &H378, &H97
```

```
ElseIf (Abs(O_rp - O_R(1))) <= 180 Then
'//driver_speed Right
Out &H378, &H93
ElseIf (Abs(O_rp - O_R(1))) > M_an2 Then
'//driver_GO
Out &H378, &H9E
ElseIf (Abs(O_rp - O_R(1))) > an_sl2 Then
'//driver_slow left
Out &H378, &H9B
ElseIf (Abs(O_rp - O_R(1))) > 180 Then
'//driver_speed left
Out &H378, &H8F
End If
End If
End If
End Sub

Sub error_control()
M_an1 = Text37.Text
M_an2 = Text38.Text
an_sl1 = Text41.Text
an_sl2 = Text42.Text
distance = Text39.Text
W_R = Text43.Text
If Check1.Value = 1 Then
If O_ro > O_rp Then
If (O_ro - O_rp) < M_an1 Then
'//end check
ElseIf (O_ro - O_rp) < an_sl1 Then
Call program5
ElseIf (O_ro - O_rp) <= 180 Then
Call program5
```

```
ElseIf (O_ro - O_rp) > M_an2 Then
'//end check
ElseIf (O_ro - O_rp) > an_sl2 Then
Call program5
ElseIf (O_ro - O_rp) > 180 Then
Call program5
End If
ElseIf O_ro < O_rp Then
If (Abs(O_ro - O_rp)) < M_an1 Then
'//end check
ElseIf (Abs(O_ro - O_rp)) < an_sl1 Then
Call program5
ElseIf (Abs(O_ro) - O_rp) <= 180 Then
Call program5
ElseIf (Abs(O_ro - O_rp)) > M_an2 Then
'//end check
ElseIf (Abs(O_ro - O_rp)) > an_sl2 Then
Call program5
ElseIf (Abs(O_ro - O_rp)) > 180 Then
Call program5
End If
End If
End If
End Sub
```

หน้าต่างปรับจูนแม่สี

```
Private Sub Picture3_Click()
DoEvents
x_click = Text1.Text
y_click = Text2.Text
```

```
StretchBlt Picture5.hdc, 0, 0, Picture5.ScaleWidth, Picture5.ScaleHeight, Picture2.hdc, (x_click -
((Picture5.ScaleWidth) / 2)), (y_click - ((Picture5.ScaleHeight) / 2)), Picture5.ScaleWidth, Picture5.ScaleHeight,
vbSrcCopy
StretchBlt Picture4.hdc, 0, 0, Picture4.ScaleWidth, Picture4.ScaleHeight, Picture2.hdc, (Text1.Text -
(VScroll1.Value / 2) - 2), (Text2.Text - (VScroll1.Value / 2) + 2), VScroll1.Value, VScroll1.Value, vbSrcCopy
End Sub

Private Sub Picture4_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
Dim p_rgb_x As Long, p_rgb_y As Long
Dim Color1 As Long, r As Integer, G As Integer, b As Integer
Dim i As Integer, j As Integer
Text1.Text = X
Text2.Text = Y
p_rgb_x = Text1.Text
p_rgb_y = Text2.Text
Color1 = GetPixel(Picture4.hdc, p_rgb_x, p_rgb_y)
Call ColorRGB(Color1, r, G, b)
Text3.Text = r
Text4.Text = G
Text5.Text = b
If Button = 1 Then
'////// Condition ///////
If Option1.Value = True Then
If Text3.Text < R_min_t Then
R_min_t = Text3.Text
RMIT.Text = R_min_t
ElseIf Text3.Text > R_max_t Then
R_max_t = Text3.Text
RMAT.Text = R_max_t
ElseIf Text4.Text < G_min_t Then
G_min_t = Text4.Text
GMIT.Text = G_min_t
```

```
ElseIf Text4.Text > G_max_t Then
G_max_t = Text4.Text
GMAT.Text = G_max_t
ElseIf Text5.Text < B_min_t Then
B_min_t = Text5.Text
BMIT.Text = B_min_t
ElseIf Text5.Text > B_max_t Then
B_max_t = Text5.Text
BMAT.Text = B_max_t
End If
ElseIf Option2.Value = True Then
If Text3.Text < R_min_R1 Then
R_min_R1 = Text3.Text
RMI1.Text = R_min_R1
ElseIf Text3.Text > R_max_R1 Then
R_max_R1 = Text3.Text
RMA1.Text = R_max_R1
ElseIf Text4.Text < G_min_R1 Then
G_min_R1 = Text4.Text
GMI1.Text = G_min_R1
ElseIf Text4.Text > G_max_R1 Then
G_max_R1 = Text4.Text
GMA1.Text = G_max_R1
ElseIf Text5.Text < B_min_R1 Then
B_min_R1 = Text5.Text
BMI1.Text = B_min_R1
ElseIf Text5.Text > B_max_R1 Then
B_max_R1 = Text5.Text
BMA1.Text = B_max_R1
End If
ElseIf Option3.Value = True Then
If Text3.Text < R_min_R2 Then
```

```
R_min_R2 = Text3.Text
RMI2.Text = R_min_R2
ElseIf Text3.Text > R_max_R2 Then
R_max_R2 = Text3.Text
RMA2.Text = R_max_R2
ElseIf Text4.Text < G_min_R2 Then
G_min_R2 = Text4.Text
GMI2.Text = G_min_R2
ElseIf Text4.Text > G_max_R2 Then
G_max_R2 = Text4.Text
GMA2.Text = G_max_R2
ElseIf Text5.Text < B_min_R2 Then
B_min_R2 = Text5.Text
BMI2.Text = B_min_R2
ElseIf Text5.Text > B_max_R2 Then
B_max_R2 = Text5.Text
BMA2.Text = B_max_R2
End If
ElseIf Option4.Value = True Then
If Text3.Text < R_min_R3 Then
R_min_R3 = Text3.Text
RMI3.Text = R_min_R3
ElseIf Text3.Text > R_max_R3 Then
R_max_R3 = Text3.Text
RMA3.Text = R_max_R3
ElseIf Text4.Text < G_min_R3 Then
G_min_R3 = Text4.Text
GMI3.Text = G_min_R3
ElseIf Text4.Text > G_max_R3 Then
G_max_R3 = Text4.Text
GMA3.Text = G_max_R3
ElseIf Text5.Text < B_min_R3 Then
```

```
B_min_R3 = Text5.Text
BMI3.Text = B_min_R3
ElseIf Text5.Text > B_max_R3 Then
B_max_R3 = Text5.Text
BMA3.Text = B_max_R3
End If
ElseIf Option8.Value = True Then
If Text3.Text < R_min_Ob Then
R_min_Ob = Text3.Text
RMIO.Text = R_min_Ob
ElseIf Text3.Text > R_max_Ob Then
R_max_Ob = Text3.Text
RMAO.Text = R_max_Ob
ElseIf Text4.Text < G_min_Ob Then
G_min_Ob = Text4.Text
GMIO.Text = G_min_Ob
ElseIf Text4.Text > G_max_Ob Then
G_max_Ob = Text4.Text
GMAO.Text = G_max_Ob
ElseIf Text5.Text < B_min_Ob Then
B_min_Ob = Text5.Text
BMIO.Text = B_min_Ob
ElseIf Text5.Text > B_max_Ob Then
B_max_Ob = Text5.Text
BMAO.Text = B_max_Ob
End If
End If
End If
End Sub

Private Sub Picture4_DblClick()
If Option1.Value = True Then
```

```
R_min_t = Text3.Text
RMIT.Text = R_min_t
R_max_t = Text3.Text
RMAT.Text = R_max_t
G_min_t = Text4.Text
GMIT.Text = G_min_t
G_max_t = Text4.Text
GMAT.Text = G_max_t
B_min_t = Text5.Text
BMIT.Text = B_min_t
B_max_t = Text5.Text
BMAT.Text = B_max_t
ElseIf Option2.Value = True Then
R_min_R1 = Text3.Text
RMI1.Text = R_min_R1
R_max_R1 = Text3.Text
RMA1.Text = R_max_R1
G_min_R1 = Text4.Text
GMI1.Text = G_min_R1
G_max_R1 = Text4.Text
GMA1.Text = G_max_R1
B_min_R1 = Text5.Text
BMI1.Text = B_min_R1
B_max_R1 = Text5.Text
BMA1.Text = B_max_R1
ElseIf Option3.Value = True Then
R_min_R2 = Text3.Text
RMI2.Text = R_min_R2
R_max_R2 = Text3.Text
RMA2.Text = R_max_R2
G_min_R2 = Text4.Text
GMI2.Text = G_min_R2
```

```
G_max_R2 = Text4.Text
GMA2.Text = G_max_R2
B_min_R2 = Text5.Text
BMI2.Text = B_min_R2
B_max_R2 = Text5.Text
BMA2.Text = B_max_R2
ElseIf Option4.Value = True Then
R_min_R3 = Text3.Text
RMI3.Text = R_min_R3
R_max_R3 = Text3.Text
RMA3.Text = R_max_R3
G_min_R3 = Text4.Text
GMI3.Text = G_min_R3
G_max_R3 = Text4.Text
GMA3.Text = G_max_R3
B_min_R3 = Text5.Text
BMI3.Text = B_min_R3
B_max_R3 = Text5.Text
BMA3.Text = B_max_R3
ElseIf Option8.Value = True Then
R_min_Ob = Text3.Text
RMIO.Text = R_min_Ob
R_max_Ob = Text3.Text
RMAO.Text = R_max_Ob
G_min_Ob = Text4.Text
GMIO.Text = G_min_Ob
G_max_Ob = Text4.Text
GMAO.Text = G_max_Ob
B_min_Ob = Text5.Text
BMIO.Text = B_min_Ob
B_max_Ob = Text5.Text
BMAO.Text = B_max_Ob
```

```
End If
End Sub

Private Sub Command6_Click()
DoEvents
Dim i As Integer, j As Integer
Dim Color1 As Long, r As Integer, G As Integer, b As Integer
If Option1.Value = True Then
R_min = RMIT.Text
R_max = RMAT.Text
G_min = GMIT.Text
G_max = GMAT.Text
B_min = BMIT.Text
B_max = BMAT.Text
ElseIf Option2.Value = True Then
R_min = RMI1.Text
R_max = RMA1.Text
G_min = GMI1.Text
G_max = GMA1.Text
B_min = BMI1.Text
B_max = BMA1.Text
ElseIf Option3.Value = True Then
R_min = RMI2.Text
R_max = RMA2.Text
G_min = GMI2.Text
G_max = GMA2.Text
B_min = BMI2.Text
B_max = BMA2.Text
ElseIf Option4.Value = True Then
R_min = RMI3.Text
R_max = RMA3.Text
G_min = GMI3.Text
```

```
G_max = GMA3.Text
B_min = BMI3.Text
B_max = BMA3.Text

ElseIf Option8.Value = True Then
R_min = RMIO.Text
R_max = RMAO.Text
G_min = GMIO.Text
G_max = GMAO.Text
B_min = BMIO.Text
B_max = BMAO.Text
End If
StretchBlt Picture5.hdc, 0, 0, 85, 85, Picture2.hdc, (x_click - (85 / 2)), (y_click - (85 / 2)), 85, 85, vbSrcCopy
For i = 0 To 85
For j = 0 To 85
Color1 = GetPixel(Picture5.hdc, i, j)
Call ColorRGB(Color1, r, G, b)
If (r + G + b) <= (R_max + G_max + B_max) And r >= R_min And r <= R_max And G >= G_min And G <=
G_max And b >= B_min And b <= B_max Then
If Option1.Value = True Then Color1 = 16777215
If Option2.Value = True Then Color1 = 255
If Option3.Value = True Then Color1 = 589823
If Option4.Value = True Then Color1 = 16711935
If Option8.Value = True Then Color1 = 16711680
Else
Color1 = 0
End If
SetPixel Picture5.hdc, i, j, Color1
DoEvents
Next j
Next i
End Sub
```

โปรแกรมหน้าต่างควบคุม

```
Private Sub Picture1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
DoEvents
'----------Freedom Mouse Point----------
If Option5.Value = True Then
If Check1.Value = 1 Then
If X <= (Ro_mx(1) + rad_R) And X >= (Ro_mx(1) - rad_R) And Y <= (Ro_my(1) + rad_R) And Y >= (Ro_my(1) -
rad_R) Then
Picture1.Circle (X, Y), rad_R, RGB(255, 0, 0)
Robot_mx(1) = X
Robot_my(1) = Y
Text6.Text = Robot_mx(1)
Text7.Text = Robot_my(1)
Picture1.Line (Ro_mx(1), Ro_my(1))-(X, Y), RGB(255, 0, 0)
Picture1.CurrentX = X
Picture1.CurrentY = Y + rad_R
Picture1.Print "Move Robot 1"
t(1) = 1
End If
End If
If Check2.Value = 1 Then
If X <= (Ro_mx(2) + rad_R) And X >= (Ro_mx(2) - rad_R) And Y <= (Ro_my(2) + rad_R) And Y >= (Ro_my(2) -
rad_R) Then
Picture1.Circle (X, Y), rad_R, RGB(255, 255, 0)
Robot_mx(2) = X
Robot_my(2) = Y
Text8.Text = Robot_mx(2)
Text9.Text = Robot_my(2)
Picture1.Line (Ro_mx(2), Ro_my(2))-(X, Y), RGB(255, 255, 0)
Picture1.CurrentX = X
Picture1.CurrentY = Y + rad_R
Picture1.Print "Move Robot 2"
```

```
t(2) = 1
End If
End If
If Check3.Value = 1 Then
If X <= (Ro_mx(3) + rad_R) And X >= (Ro_mx(3) - rad_R) And Y <= (Ro_my(3) + rad_R) And Y >= (Ro_my(3) -
rad_R) Then
Picture1.Circle (X, Y), rad_R, RGB(255, 0, 255)
Robot_mx(3) = X
Robot_my(3) = Y
Text10.Text = Robot_mx(3)
Text11.Text = Robot_my(3)
Picture1.Line (Ro_mx(3), Ro_my(3))-(X, Y), RGB(255, 0, 255)
Picture1.CurrentX = X
Picture1.CurrentY = Y + rad_R
Picture1.Print "Move Robot 3"
t(3) = 1
End If
End If
If t(1) = 1 Then
Picture1.Circle (X, Y), rad_R, RGB(255, 0, 0)
Robot_mx(1) = X
Robot_my(1) = Y
Text6.Text = Robot_mx(1)
Text7.Text = Robot_my(1)
Picture1.Line (Ro_mx(1), Ro_my(1))-(X, Y), RGB(255, 0, 0)
Picture1.CurrentX = X
Picture1.CurrentY = Y + rad_R
Picture1.Print "Move Robot 1"
End If
If t(2) = 1 Then
Picture1.Circle (X, Y), rad_R, RGB(255, 255, 0)
Robot_mx(2) = X
```

```
Robot_my(2) = Y
Text8.Text = Robot_mx(2)
Text9.Text = Robot_my(2)

Picture1.Line (Ro_mx(2), Ro_my(2))-(X, Y), RGB(255, 255, 0)
Picture1.CurrentX = X
Picture1.CurrentY = Y + rad_R
Picture1.Print "Move Robot 2"
End If
If t(3) = 1 Then
Picture1.Circle (X, Y), rad_R, RGB(255, 0, 255)
Robot_mx(3) = X
Robot_my(3) = Y
Text10.Text = Robot_mx(3)
Text11.Text = Robot_my(3)
Picture1.Line (Ro_mx(3), Ro_my(3))-(X, Y), RGB(255, 0, 255)
Picture1.CurrentX = X
Picture1.CurrentY = Y + rad_R
Picture1.Print "Move Robot 3"
End If
End If
'------------------Gethering------------------
If Option6.Value = True And Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
If Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
Robot_mx(1) = X
Robot_my(1) = Y
Text6.Text = Robot_mx(1)
Text7.Text = Robot_my(1)
Picture1.Circle (X, Y), rad_R, RGB(255, 0, 0)
Robot_mx(2) = X - 50
Robot_my(2) = Y + 60
Text8.Text = Robot_mx(2)
```

```
Text9.Text = Robot_my(2)
Picture1.Circle (Robot_mx(2), Robot_my(2)), rad_R, RGB(255, 255, 0)
Robot_mx(3) = X + 50
Robot_my(3) = Y + 60
Text10.Text = Robot_mx(3)
Text11.Text = Robot_my(3)
Picture1.Circle (Robot_mx(3), Robot_my(3)), rad_R, RGB(255, 0, 255)
Picture1.Line (Ro_mx(1), Ro_my(1))-(X, Y), RGB(255, 0, 0)
Picture1.Line (Ro_mx(2), Ro_my(2))-(Robot_mx(2), Robot_my(2)), RGB(255, 255, 0)
Picture1.Line (Ro_mx(3), Ro_my(3))-(Robot_mx(3), Robot_my(3)), RGB(255, 0, 255)
Picture1.CurrentX = Robot_mx(2) + (rad_R / 2)
Picture1.CurrentY = Robot_my(3) + rad_R
Picture1.Print "Gethering"
End If
End If
'---------- Horizonnatal Line ----------
If Option7.Value = True And Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
If Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
Robot_mx(1) = X
Robot_my(1) = Y
Text6.Text = Robot_mx(1)
Text7.Text = Robot_my(1)
Picture1.Circle (X, Y), rad_R, RGB(255, 0, 0)
Robot_mx(2) = X - 60
Robot_my(2) = Y
Text8.Text = Robot_mx(2)
Text9.Text = Robot_my(2)
Picture1.Circle (Robot_mx(2), Robot_my(2)), rad_R, RGB(255, 255, 0)
Robot_mx(3) = X + 60
Robot_my(3) = Y
Text10.Text = Robot_mx(3)
Text11.Text = Robot_my(3)
```

```
Picture1.Circle (Robot_mx(3), Robot_my(3)), rad_R, RGB(255, 0, 255)
Picture1.Line (Ro_mx(1), Ro_my(1))-(X, Y), RGB(255, 0, 0)
Picture1.Line (Ro_mx(2), Ro_my(2))-(Robot_mx(2), Robot_my(2)), RGB(255, 255, 0)
Picture1.Line (Ro_mx(3), Ro_my(3))-(Robot_mx(3), Robot_my(3)), RGB(255, 0, 255)
Pic ure1.CurrentX = Robot_mx(2) + (rad_R / 2)
Picture1.CurrentY = Y + rad_R
Picture1.Print "Horizonnatal Line"
End If
End If
'------------------ Vertical Line -------------
If Option9.Value = True And Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
If Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
Robot_mx(1) = X
Robot_my(1) = Y - 60
Text6.Text = Robot_mx(1)
Text7.Text = Robot_my(1)
Picture1.Circle (Robot_mx(1), Robot_my(1)), rad_R, RGB(255, 0, 0)
Robot_mx(2) = X
Robot_my(2) = Y
Text8.Text = Robot_mx(2)
Text9.Text = Robot_my(2)
Picture1.Circle (X, Y), rad_R, RGB(255, 255, 0)
Robot_mx(3) = X
Robot_my(3) = Y + 60
Text10.Text = Robot_mx(3)
Text11.Text = Robot_my(3)
Picture1.Circle (Robot_mx(3), Robot_my(3)), rad_R, RGB(255, 0, 255)
Picture1.Line (Ro_mx(1), Ro_my(1))-(Robot_mx(1), Robot_my(1)), RGB(255, 0, 0)
Picture1.Line (Ro_mx(2), Ro_my(2))-(X, Y), RGB(255, 255, 0)
Picture1.Line (Ro_mx(3), Ro_my(3))-(Robot_mx(3), Robot_my(3)), RGB(255, 0, 255)
Picture1.CurrentX = Robot_mx(2) + (rad_R / 2)
Picture1.CurrentY = Y + rad_R
```

```
Picture1.Print "Vertical Line"
End If
End If
'-----------------Keep and Move Object---------
If Option10.Value = True And Check1.Value = 1 Then
C_obmx = X
C_obmy = Y
Text12.Text = C_obmx
Text13.Text = C_obmy
Picture1.Line (X, Y)-(X, Y + 10), RGB(0, 0, 255)
Picture1.Line (X, Y)-(X, Y - 10), RGB(0, 0, 255)
Picture1.Line (X, Y)-(X + 10, Y), RGB(0, 0, 255)
Picture1.Line (X, Y)-(X - 10, Y), RGB(0, 0, 255)
Picture1.CurrentX = X - 5
Picture1.CurrentY = Y + 5
Picture1.Print "Move"
End If
End Sub

Private Sub Picture1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
DoEvents
quit = 0
Text1.Text = X
Text2.Text = Y
Ro_mx(1) = Text14.Text
Ro_my(1) = Text15.Text
Ro_mx(2) = Text16.Text
Ro_my(2) = Text17.Text
Ro_mx(3) = Text18.Text
Ro_my(3) = Text19.Text
C_obmx = Text20.Text
C_obmy = Text21.Text
```

```
If Button = 1 Then
'-------------Freedom Mouse Point-------------
If Option5.Value = True Then
If Check1.Value = 1 Then
If X <= (Ro_mx(1) + rad_R) And X >= (Ro_mx(1) - rad_R) And Y <= (Ro_my(1) + rad_R) And Y >= (Ro_my(1) -
rad_R) Then
Picture1.Circle (X, Y), rad_R, RGB(255, 0, 0)
Robot_mx(1) = X
Robot_my(1) = Y
Text6.Text = Robot_mx(1)
Text7.Text = Robot_my(1)
Picture1.Line (Ro_mx(1), Ro_my(1))-(X, Y), RGB(255, 0, 0)
Picture1.CurrentX = X
Picture1.CurrentY = Y + rad_R
Picture1.Print "Move Robot 1"
t(1) = 1
End If
End If
If Check2.Value = 1 Then
If X <= (Ro_mx(2) + rad_R) And X >= (Ro_mx(2) - rad_R) And Y <= (Ro_my(2) + rad_R) And Y >= (Ro_my(2) -
rad_R) Then
Picture1.Circle (X, Y), rad_R, RGB(255, 255, 0)
Robot_mx(2) = X
Robot_my(2) = Y
Text8.Text = Robot_mx(2)
Text9.Text = Robot_my(2)
Picture1.Line (Ro_mx(2), Ro_my(2))-(X, Y), RGB(255, 255, 0)
Picture1.CurrentX = X
Picture1.CurrentY = Y + rad_R
Picture1.Print "Move Robot 2"
t(2) = 1
End If
```

```
End If
If Check3.Value = 1 Then
If X <= (Ro_mx(3) + rad_R) And X >= (Ro_mx(3) - rad_R) And Y <= (Ro_my(3) + rad_R) And Y >= (Ro_my(3) -
rad_R) Then
Picture1.Circle (X, Y), rad_R, RGB(255, 0, 255)
Robot_mx(3) = X
Robot_my(3) = Y
Text10.Text = Robot_mx(3)
Text11.Text = Robot_my(3)
Picture1.Line (Ro_mx(3), Ro_my(3))-(X, Y), RGB(255, 0, 255)
Picture1.CurrentX = X
Picture1.CurrentY = Y + rad_R
Picture1.Print "Move Robot 3"
t(3) = 1
End If
End If
If t(1) = 1 Then
Picture1.Circle (X, Y), rad_R, RGB(255, 0, 0)
Robot_mx(1) = X
Robot_my(1) = Y
Text6.Text = Robot_mx(1)
Text7.Text = Robot_my(1)
Picture1.Line (Ro_mx(1), Ro_my(1))-(X, Y), RGB(255, 0, 0)
Picture1.CurrentX = X
Picture1.CurrentY = Y + rad_R
Picture1.Print "Move Robot 1"
End If
If t(2) = 1 Then
Picture1.Circle (X, Y), rad_R, RGB(255, 255, 0)
Robot_mx(2) = X
Robot_my(2) = Y
Text8.Text = Robot_mx(2)
```

```
Text9.Text = Robot_my(2)
Picture1.Line (Ro_mx(2), Ro_my(2))-(X, Y), RGB(255, 255, 0)
Picture1.CurrentX = X
Picture1.CurrentY = Y + rad_R
Picture1.Print "Move Robot 2"
End If
If t(3) = 1 Then
Picture1.Circle (X, Y), rad_R, RGB(255, 0, 255)
Robot_mx(3) = X
Robot_my(3) = Y
Text10.Text = Robot_mx(3)
Text11.Text = Robot_my(3)
Picture1.Line (Ro_mx(3), Ro_my(3))-(X, Y), RGB(255, 0, 255)
Picture1.CurrentX = X
Picture1.CurrentY = Y + rad_R
Picture1.Print "Move Robot 3"
End If
End If
'----------Gethering----------
If Option6.Value = True And Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
If Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
Robot_mx(1) = X
Robot_my(1) = Y
Text6.Text = Robot_mx(1)
Text7.Text = Robot_my(1)
Picture1.Circle (X, Y), rad_R, RGB(255, 0, 0)
Robot_mx(2) = X - 50
Robot_my(2) = Y + 60
Text8.Text = Robot_mx(2)
Text9.Text = Robot_my(2)
Picture1.Circle (Robot_mx(2), Robot_my(2)), rad_R, RGB(255, 255, 0)
Robot_mx(3) = X + 50
```

```
Robot_my(3) = Y + 60
Text10.Text = Robot_mx(3)
Text11.Text = Robot_my(3)
Picture1.Circle (Robot_mx(3), Robot_my(3)), rad_R, RGB(255, 0, 255)
Picture1.Line (Ro_mx(1), Ro_my(1))-(X, Y), RGB(255, 0, 0)
Picture1.Line (Ro_mx(2), Ro_my(2))-(Robot_mx(2), Robot_my(2)), RGB(255, 255, 0)
Picture1.Line (Ro_mx(3), Ro_my(3))-(Robot_mx(3), Robot_my(3)), RGB(255, 0, 255)
Picture1.CurrentX = Robot_mx(2) + (rad_R / 2)
Picture1.CurrentY = Robot_my(3) + rad_R
Picture1.Print "Gethering"
End If
End If
'---------- Horizonnatal Line ----------
If Option7.Value = True And Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
If Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
Robot_mx(1) = X
Robot_my(1) = Y
Text6.Text = Robot_mx(1)
Text7.Text = Robot_my(1)
Picture1.Circle (X, Y), rad_R, RGB(255, 0, 0)
Robot_mx(2) = X - 60
Robot_my(2) = Y
Text8.Text = Robot_mx(2)
Text9.Text = Robot_my(2)
Picture1.Circle (Robot_mx(2), Robot_my(2)), rad_R, RGB(255, 255, 0)
Robot_mx(3) = X + 60
Robot_my(3) = Y
Text10.Text = Robot_mx(3)
Text11.Text = Robot_my(3)
Picture1.Circle (Robot_mx(3), Robot_my(3)), rad_R, RGB(255, 0, 255)
Picture1.Line (Ro_mx(1), Ro_my(1))-(X, Y), RGB(255, 0, 0)
Picture1.Line (Ro_mx(2), Ro_my(2))-(Robot_mx(2), Robot_my(2)), RGB(255, 255, 0)
```

```
Picture1.Line (Ro_mx(3), Ro_my(3))-(Robot_mx(3), Robot_my(3)), RGB(255, 0, 255)
Picture1.CurrentX = Robot_mx(2) + (rad_R / 2)
Picture1.CurrentY = Y + rad_R
Picture1.Print "Horizonnatal Line"
End If
End If
'------------------ Vertical Line -------------
If Option9.Value = True And Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
If Check1.Value = 1 And Check2.Value = 1 And Check3.Value = 1 Then
Robot_mx(1) = X
Robot_my(1) = Y - 60
Text6.Text = Robot_mx(1)
Text7.Text = Robot_my(1)
Picture1.Circle (Robot_mx(1), Robot_my(1)), rad_R, RGB(255, 0, 0)
Robot_mx(2) = X
Robot_my(2) = Y
Text8.Text = Robot_mx(2)
Text9.Text = Robot_my(2)
Picture1.Circle (X, Y), rad_R, RGB(255, 255, 0)
Robot_mx(3) = X
Robot_my(3) = Y + 60
Text10.Text = Robot_mx(3)
Text11.Text = Robot_my(3)
Picture1.Circle (Robot_mx(3), Robot_my(3)), rad_R, RGB(255, 0, 255)
Picture1.Line (Ro_mx(1), Ro_my(1))-(Robot_mx(1), Robot_my(1)), RGB(255, 0, 0)
Picture1.Line (Ro_mx(2), Ro_my(2))-(X, Y), RGB(255, 255, 0)
Picture1.Line (Ro_mx(3), Ro_my(3))-(Robot_mx(3), Robot_my(3)), RGB(255, 0, 255)
Picture1.CurrentX = Robot_mx(2) + (rad_R / 2)
Picture1.CurrentY = Y + rad_R
Picture1.Print "Vertical Line"
End If
End If
```

```
'-----------------Keep and Move Object---------
If Option10.Value = True And Check1.Value = 1 And Check4.Value = 1 Then
Picture1.Line (X, Y)-(X, Y + 10), RGB(0, 0, 255)
Picture1.Line (X, Y)-(X, Y - 10), RGB(0, 0, 255)
Picture1.Line (X, Y)-(X + 10, Y), RGB(0, 0, 255)
Picture1.Line (X, Y)-(X - 10, Y), RGB(0, 0, 255)
Picture1.CurrentX = X - 5
Picture1.CurrentY = Y + 5
Picture1.Print "Move"
End If
End If
End Sub

Private Sub Picture1_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
DoEvents
t(1) = 0
t(2) = 0
t(3) = 0
t(4) = 0
End Sub
```

โปรแรมขณะเริ่มต้น

```
Private Sub Form_Load()
Timer1.Enabled = False
DoEvents
Check1.Value = 1
Check2.Value = 1
Check3.Value = 1
Check9.Value = 1
C_obmx = 0
C_obmy = 0
rad_R = Text32.Text
```

```
rad_O = Text33.Text
R_OB = Text34.Text
R_ROBOT = Text35.Text
HScroll1.Value = Text36.Text
quit = 0
jump1 = 0
jump2 = 0
ezVidCap1.Preview = True
ezVidCap1.AutoSize = False
ezVidCap1.FrameEventEnabled = False
ezVidCap1.Width = 373
ezVidCap1.Height = 309
Dim i As Long
Call EnableButtons 'check device caps and enable appropriate btns
Me.Show 'show form
Me.Refresh
If 0 < ezVidCap1.NumCapDevs Then
For i = 0 To ezVidCap1.NumCapDevs - 1
cbDriver.AddItem (ezVidCap1.GetDriverName(i))
Next
cbDriver.ListIndex = ezVidCap1.DriverIndex
Else
cbDriver.AddItem ("<none>")
cbDriver.ListIndex = 0
MsgBox "No Video Capture Device!", vbInformation, App.Title
End If
Out &H378, &H9F
Call delay
Out &H378, &HBF
Call delay
Out &H378, &HDF
Call delay
```

```
End Sub
```

โปรแกรมในโมดูลต่างๆที่ใช้ในการดึงภาพ ควบคุม และเก็บข้อมูล

- **ฟังชั่น API ที่เลือกใช้งาน**

```
Public Declare Function BitBlt Lib "gdi32" (ByVal hDestDC As Long, ByVal X As Long, ByVal Y As Long,
ByVal nWidth As Long, ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal xSrc As Long, ByVal ySrc As
Long, ByVal dwRop As Long) As Long
Public Declare Function StretchBlt Lib "gdi32" (ByVal hdc As Long, ByVal X As Long, ByVal Y As Long, ByVal
nWidth As Long, ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal xSrc As Long, ByVal ySrc As Long,
ByVal nSrcWidth As Long, ByVal nSrcHeight As Long, ByVal dwRop As Long) As Long
Public Declare Function SetPixel Lib "gdi32" (ByVal hdc As Long, ByVal X As Long, ByVal Y As Long, ByVal
crColor As Long) As Long
Public Declare Function GetPixel Lib "gdi32" (ByVal hdc As Long, ByVal X As Long, ByVal Y As Long) As Long
```

- **ฟังชั่น ที่ใช้ติดต่อกับพอรต**

```
#If Win32 Then
Public Declare Sub Out Lib "io.dll" Alias "PortOut" (ByVal Port As Integer, ByVal Data As Byte)
Public Declare Function Inp Lib "io.dll" Alias "PortIn" (ByVal Port As Integer) As Byte
#Else
Declare Function Inp Lib "InpOut.DLL" (ByVal Port As Integer) As Byte
Declare Sub Out Lib "InpOut.DLL" (ByVal Port As Integer, ByVal Value As Byte)
#End If
```

ข โปรแกรมของหุ่นยนต์และตัวส่งอินฟราเรด

ตัวหุ่นยนต์

```
Include "MODEDEFS.bas"
TRISB = %00000000
TRISA 0.2 = 1
```

```
TRISA 0# = 0
TRISA 0.1 = 0
OO var byte
portb =%00000000
LOOP:SERIN PORTA.2,T2400,OO
PORTA.1=PORTB.7
PORTA.0=PORTB.6
IF OO=%10011110 THEN GO
IF OO=%10011101 THEN BACK
IF OO=%10011011 THEN RIGHT
IF OO=%10010111 THEN LEFT
IF OO=%10001111 THEN RR
IF OO=%10010011 THEN RL
IF OO=%10011111 THEN ST
GOTO LOOP
GO:PORTB =%01100110
pause 50
GOTO LOOP
BACK:PORTB =%10011001
pause 50
GOTO LOOP
LEFT:PORTB =%00000110
pause 50
GOTO LOOP
RIGHT:PORTB =%01100000
pause 50
GOTO LOOP
RL:PORTB =%10010110
pause 50
GOTO LOOP
RR:PORTB =%01101001
pause 50
```

```
GOTO LOOP
ST:PORTB =%00000000
pause 50
GOTO LOOP
End
```

ภาคส่ง

```
Include "modedefs.bas"
TRISB = %11111111
II VAR BYTE
Loop: II = PORTB
serout PORTA.0,N2400,[II]
pause 10
GOTO LOOP
End
```

หนังสืออ้างอิง

1.พุฒิพงษ์ นาคะปัท , "การเขียนเกมส์บนวินโดวส์ด้วย Visual Basic" , บริษัท ซีเอ็ดยูเคชั่น จำกัด(มหาชน), 352 หน้า, 2543

2. สังวาล บกสุวรรณ , สุเมธ จันทรท่าจีน," ระบบติดตามวัตถุ", ปริญญานิพนธ์ปีการศึกษา 2545, คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

3. อรรถพล บุญยะโภคา , ชัยวัฒน์ ลิ้มวิจิตรวิไล , "เรียนรู้และปฏิบัติการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอกผ่านพอร์ตขนาน", บริษัท อินโนเวตีฟ เอ็กเพอริเมนต์ จำกัด, พิมพ์ครั้งที่ 2

4.ฉัตรทวุฒิ พืชผล,พิชิตสันติกุลานนท์ , "คู่มือการเรียน Visual Basic 6", โปรวิชั่น, 2544 , พิมพ์ครั้งที่ 5

กิตติกรรมประกาศ

โครงงานการควบคุมหุ่นยนต์หลายตัวนี้ ผ่านสำเร็จไปได้ด้วยนี้ ต้องขอขอบคุณอาจารย์ภาควิชาวิศวกรรมระบบควบคุมที่สอนและให้ความรู้ อาจารย์ที่ปรึกษาที่ช่วยให้แนวคิดกับวิธีแก้ปัญหาต่าง ๆ รวมถึงเพื่อนๆที่คอยช่วยเหลือและเป็นกำลังใจ โครงงานนี้จะสำเร็จไปไม่ได้ จึงขอขอบพระคุณบิดามารดา ผู้ที่ให้กำเนิด ขอบคุณอาจารย์ดอนที่ช่วยเหลือในเรื่องอุปกรณ์งบประมาณและคำที่ปรึกษา อาจารย์ถาวรที่ให้คำปรึกษาในเรื่องโปรแกรม และขอคุณเพื่อนเบิ้ม พร และ น้องก้อย ในการช่วยเหลือยามคับขัน จากการทำงานโครงงานครั้งนี้จะนำความรู้และประสบการณ์ที่ได้ไปประยุกต์ใช้กับการทำงานอื่นๆอีกต่อไป