รถเข็นคนพิการที่ควบคุมด้วยการกรอกตา
EYE TRACKING FOR WHEELCHAIR CONTROL

อนิวัฒน์ จูห้อง
Aniwat Juhong

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมชีวการแพทย์
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ.2558

รถเข็นคนพิการที่ควบคุมด้วยการกรอกตา

EYE TRACKING FOR WHEELCHAIR CONTROL

โดย

นาย อนิวัฒน์ จูห้อง

อาจารย์ที่ปรึกษา

รศ.ดร.ชูชาติ ปิณฑวิรุจน์

ปริญญานิพนธ์เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตบัณฑิต

สาขาวิชาวิศวกรรมชีวการแพทย์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ.2558

ปริญญานิพนธ์ ปีการศึกษา 2558

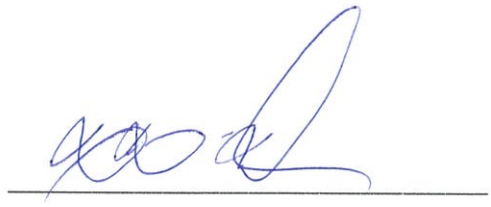
สาขาวิชา วิศวกรรมชีวการแพทย์

คณะ วิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง รถเข็นคนพิการที่ควบคุมด้วยการกรอกตา
(Eye tracking for wheelchair control)

ผู้จัดทำ นายอนิวัฒน์ จูห้อง รหัสนักศึกษา 55011400

รายงานนี้ผ่านการตรวจสอบโดยอาจารย์ที่ปรึกษาแล้ว



(รศ.ดร.ชูชาติ ปิณฑวิรุจน์)
อาจารย์ที่ปรึกษา

หัวข้อโครงงาน	รถเข็นคนพิการที่ควบคุมด้วยการกรอกตา
นักศึกษา	นายอนิวัฒน์ จูห้อง
รหัสนักศึกษา	55011400
ปริญญา	วิศวกรรมศาสตร์บัณฑิต
สาขาวิชา	วิศวกรรมชีวการแพทย์
ปีการศึกษา	2558
อาจารย์ที่ปรึกษาปริญญานิพนธ์	รศ.ดร. ชูชาติ ปิณฑวิรุจน์

บทคัดย่อ

โปรเจคนี้พัฒนาขึ้นเพื่อช่วยเหลือผู้พิการในการควบคุม wheelchair โดยใช้การเคลื่อนไหวของตา ซึ่งจะหลักการนี้จะช่วยผู้พิการที่ไม่สามารถจะขยับได้ตั้งแต่ส่วนคอเป็นต้นมา ซึ่งหลักการนั้นจะเกี่ยวข้อง 3 stages คือ image detection , image processing และ การควบคุม wheelchair การเคลื่อนไหวของตานั้นจะDetect โดยใช้ Webcam แล้วส่งภาพไปประมวณผลที Raspberry Pi เพื่อหาตำแหน่งของลูกตาใช้โปรแกรม C++ หลังจากที่ประมวลผลหาตำแหน่งของตาเสร็จแล้วก็จะส่งข้อไปมูลตำแหน่งของตาไปที่ Arduino เพื่อไปควบคุม wheelchair

Project Title	Eye-tracking for wheelchair control
Student	Mr. Aniwat Juhong
Student ID	55011400
Degree	Bachelor of Engineering
Program	Biomedical Engineering
Year	2014
Thesis Advisor	Assoc.Prof.Dr. Chuchart Pintavrooj

ABSSTRACT

This Project developed for disable people to control wheelchair base on eye movement. This concept can be used for people with loco-motor disabilities. The propose system involves three stages: image detection, image processing and motor control of wheelchair. The eye movement is detected by webcam, the image of the eye will be sent to Raspberry Pi (small computer) where the image will be processed using C++ software. The corresponding output signal send to Arduino after that Arduino use this signal to control wheelchair joystick.

กิติกรรมประกาศ

วิทยานิพนธ์เล่มนี้สำเร็จลุล่วงได้ด้วยความกรุณาจาก รศ.ดร. ชูชาติ ปิณฑวิรุจน์ ซึ่งเป็นอาจารย์ที่ปรึกษาให้ความรู้ คำแนะนำ อีกทั้งความช่วยเหลือ ในการแก้ไขปัญญาต่างๆที่เกิดขึ้น อีกทั้งยังแบ่งปันประสบการณ์ต่างๆมากมายให้แก่ข้าเจ้าตลอดเวลาการศึกษา

ขอขอบพระคุณ สถาบันเทคโนโยลีพระจอมเกล้าเจ้าคุณทหารลาดกระบังที่สนับสนุนการทำวิยานิพนธ์ฉบับนี้ รวมทั้งเอื้อเฟื้อนสานที่ ห้องแลป B-204 ในการทำโครงงานตลอดที่ข้าพเจ้าเป็นศึกษาที่สถาบันแห่งนี้

ขอขอบคุณเพื่อนๆ พี่ๆ น้องๆ ทุกคนที่คอยให้กำลังใจและความช่วยเหลือในการทำวิทยานิพนธ์เล่มนี้ตลอดจนคำปรึกษาที่ขำให้ข้าพเจ้ามีแนวทางที่จะทำวิทยานิพนธ์เล่มนี้ให้เสร็จสมบูรณ์

สุดท้ายนี้ขอขอบพระคุณพระคุณ บิดา มารดา และสมาชิกในครอบครัว ทุกคน ที่เป็นกำลังใจข้าพเจ้าตลอดจนประสบความสำเร็จ

อนิวัฒน์ จูห้อง

สารบัญ

สารบัญ(ต่อ)

สารบัญตาราง

สารบัญรูป

สารบัญรูป(ต่อ)

สารบัญรูป(ต่อ)

บทที่1

บทนำ

1.1 ความเป็นมาและความสำคัญของโครงงาน

ในปัจจุบันมีผู้พิการที่ไม่สามารถจะเคลื่อนไหว ตั้งแต่ส่วนคอลงมา ทำให้ต้องใช้ชีวิตอยากอย่างลำบาก ไม่สามารถที่จะเดินทางไปไหนมาเดินด้วยตนเองได้ ทางเราจะได้มีแนวความคิดเกี่ยวกับการใช้ ตา ในการควบคุม วีลแชร์ ซึ่งสามารถใช้งานได้ง่ายมีความเสถียรและที่สำคัญราคาไม่แพง ซึ่งระบบจะมี 3 ขั้นตอน คือ การตรวจจับภาพ, การประมวลผลภาพ และ การควบคุมจอยสติ๊กของวีลแชร์ ซึ่งการเคลื่อนไหวของตาจะถูกถ่ายโดย webcam ซึ่งถูกติดไว้กับแว่น แล้วจะส่งภาพไปประมวลผลที่ ราสเบอรี่พาย(Raspberry Pi) โดยใช้ภาษา C++ในการประมวลผล หลังจากนั้นก็จะส่งคำสั่งไปควบคุม เซอร์โว(Servo) ที่ ควบคุม จอยสติ๊ก

1.2 วัตถุประสงค์ของการศึกษา

1.2.1 ศึกษาการ การประมวลผลภาพOpencv c++ ในDetect ลูกตา

12.2 ศึกษาส่วนของ Serial port communication ระหว่าง Arduino และ ราสเบอรี่พาย (Raspberry Pi)

12.2 ศึกษาการทำงานของ กลไกลควบคุมการทำงานของ wheelchair จอยสติ๊ก

1.3 วิธีการวิจัย

1.3.1 ศึกษาข้อ การประมวลผลภาพขั้นต้น (Pre-image processing) ในการปรับปรุงภาพ สามารถ Threshold ได้ง่าย

1.3.2.ศึกษาข้อ การ Detect Pupil ของลูกตาที่เป็นทรงกลมโดยใช้ Hough transform

1.3.3 เขียน Algorithm ในการคำนวณ หา ROI และ ระบุพิกัดตำแหน่งของ Pupil ใช้สำหรับการควบคุม Motor

1.4 ขอบเขตการวิจัย

ใช้กล้อง Detect การเคลื่อนไหวของ Pupil แล้วนำการเคลื่อนไหวนี้ไป ควบคุม Wheelchair โดยมีการCalibrate คำนวณ ROI ทุกครั้งก่อนใช้งาน

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1.5.1 ได้รับความรู้การทำงานเกี่ยวกับ การประมวลผลภาพเบื้องต้น

1.5.2 ได้รับความรู้เกี่ยวกับการใช้งาน ราสเบอรี่พาย(Raspberry Pi) & Arduino

15.3 ได้รับความรู้เกี่ยวกับการออกแบบ Wheelchair

15.4 สามารถนำงานนี้ไปใช้จริงกับผู้พิการได้จริง

บทที่ 2

ทฤษฎีและความรู้พื้นฐานที่เกี่ยวข้อง

2.1 งานวิจัยที่เกี่ยวข้อง

2.1.1 Detection of Eyes by Circular Hough Transform and Histogram of Gradient Yasutaka Ito, Wataru Ohyama, Tetsushi Wakabayashi, Fumitaka Kimura Graduate School of Engineering, Mie University

2.1.2 Feasibility of Hough Transform based Iris Localisation for real time application Klaus D. Toennies, Frank Behrens, Melanie Aurnhammer Dept. of Computer Science, Otto-von-Guericke Universität Magdeburg, Germany

2.2 ความรู้พื้นฐาน

2.2.1 การประมวลผลสัญญาณภาพดิจิตอล

การประมวณผลสัญญาณภาพดิจิตอล หรือ Digital การประมวลผลภาพเป็นการนำภาพเข้าสู่การแปลงข้อมูลภาพให้อยู่ในรูปแบบข้อมูลดิติตอลที่สามารถนำข้อมูลนี้ผ่านกระบวนการต่างๆเพื่อให้ได้ผลลัพธ์แบบใหม่ที่บ่งบอกถึงลักษณะและคุณสมบัติของภาพ เช่นนำภาพสี RGB แปลงเป็นภาพเฉดขาวดำ (Gray level) ,การปรับเท่าฮีสโตแกรม(Histogram Equalization),การตรวจวัตถุทรงกลมโดยใช้ function Hough transform เป็นต้น

2.2.2 ทฤษฎีสีและเงา

สี คือ ลักษณะความเข้มของแสงที่ปรากฏแก่สายตาให้เห็นเป็นสี โดยผ่านกระบวนการรับรู้ด้วยตา โดยที่ตาได้ผ่านกระบวนการวิเคราะห์ข้อมูลพลังงานแสงมาแล้ว ผ่านประสาทสัมผัสการมองเห็น ผ่านศูนย์สับเปลี่ยนในสมองไปสู่ศูนย์การเห็นภาพ การสร้างภาพหรือการมองเห็นก็คือการที่ข้อมูลได้ผ่านการวิเคราะห์แยกแยะให้เรารับรู้ถึงสรรพสิ่งรอบตัว โดยการมองเห็นภาพจากคอมพิวเตอร์นั้นเป็นการประกอบขึ้นขอ

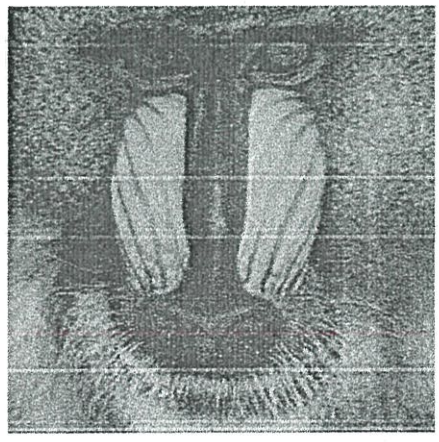
จุดภาพ (pixel) ที่มีค่าสีอยู่ในจุดภาพนั้น จำนวนสีสูงสุดที่เป็นไปได้ของแต่ละจุดภาพขึ้นอยู่กับจำนวนบิตที่ใช้ เมื่อมีการกำหนดให้ขนาดของบิตต่อจุดมากขึ้นจะทำให้จำนวนของสีมากขึ้นด้วย

1 บิต	=	2^1	=	2	สี
2 บิต	=	2^2	=	4	สี
4 บิต	=	2^4	=	16	สี
8 บิต	=	2^8	=	256	สี
16บิต	=	2^{16}	=	65536	สี

2.2.3 มาตรฐานของสี

มาตรฐานของสีที่ใช้อยู่ในปัจจุบันมีอยู่หลายระบบด้วยกัน ทั้งนี้จะขึ้นอยู่กับการนำไปใช้ แต่โดยทั่วไปแล้วทุกมาตรฐานจะมีแนวคิดเดียวกันคือ การแทนจุดสีด้วยจุดที่อยู่ภายในพิกัด 3 มิติ โดยจะมีแกนอ้างอิงสำหรับจุดีสีนั้นในระนาบซึ่งแต่ละแกนจะมีความเป็นอิสระต่อกัน

2.2.3.1 ระบบสีแบบ Gray Scale ภาพโทนสีขาวดำหรืออาพที่มีการเปลี่ยนแปลงตามความเข้มของแสง (Intensity Image) ค่าในแต่ละพิกเซลคือค่าความเข้มของแสง ณ แต่ละตำแหน่งของพิกเซลซึ่งอยู่ในรูปแบบของระดับค่าขาวดำ(Gray Scale หรือ Gray Level) ค่าที่เป็นไปได้ของระดับขาวดำ จะขึ้นอยู่กับจำนวนบิตที่ใช้ ตัวอย่างเช่น 8 Bit จะมีระดับของโทนสีขาวดำทั้งหมด 256 ระดับ

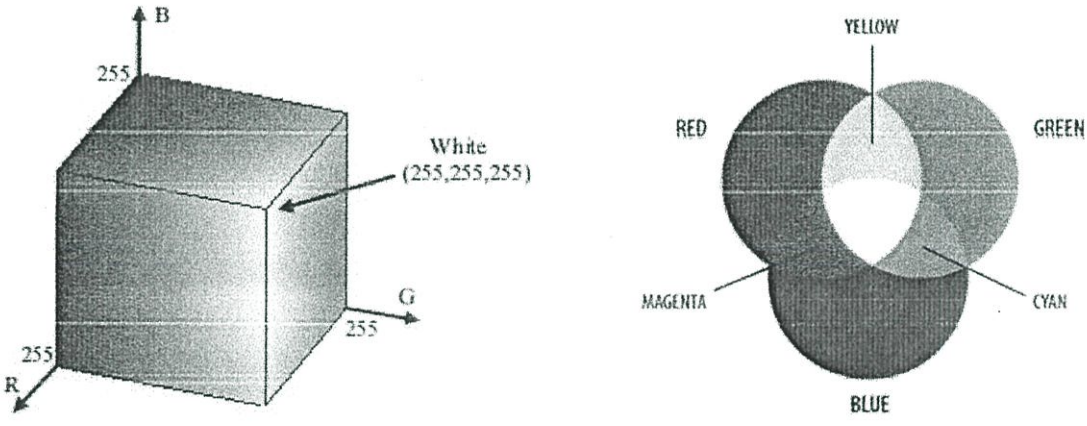


รูปที่ 2.1 ภาพโทนสีขาวดำ

2.2.3.2 ระบบสำแบบ RGB เป็นระบบสีที่เกิดจากการรวมกันของแสงสีแดง เขียวและน้ำเงิน รวมกันเป็นแสงสีขาวโดยมีการรวมกันแบบบวก ใช้ในการแสดงผลทางหน้าจอคอมพิวเตอร์ และหน้าจอโทรทัศน์

ในระบบพิกัด Color Space ดัง Figure 2.2 (a) โดยแต่ละสีจะมีค่าตั้งแต่ศูนย์จนถึงหนึ่ง โดยที่ศูนย์หมายถึง สีนั้นมีความเข้มมากจึงดูมือและหนึ่งหมายถึงสีนั้นมีความเข้มน้อยจึงดูสว่างจะด้าพการผสมสีทางแสงหรือการบวกแม่สี (Additive Primary Color) เข้าด้วยยกันดัง Figure 2.2 (b)

โดยทั่วไปจำนวนบิตข้อมูลที่ใช้ในการแทนความเข้มของแม่สีแต่ละสีมี 256 ระดับ (0-255) จำนวน 8 Bit รวมแม่สีทั้งสามแล้วใช้จำนวน 8x3 เท่ากับ 24 Bit ซึ่งสามารถใช้สร้างสีได้ถึง 256x256x256 =16,777,216 สี



(a) (b)

รูปที่ 2.2 (a) แสดงModel ในระบบพิกัด Color Space

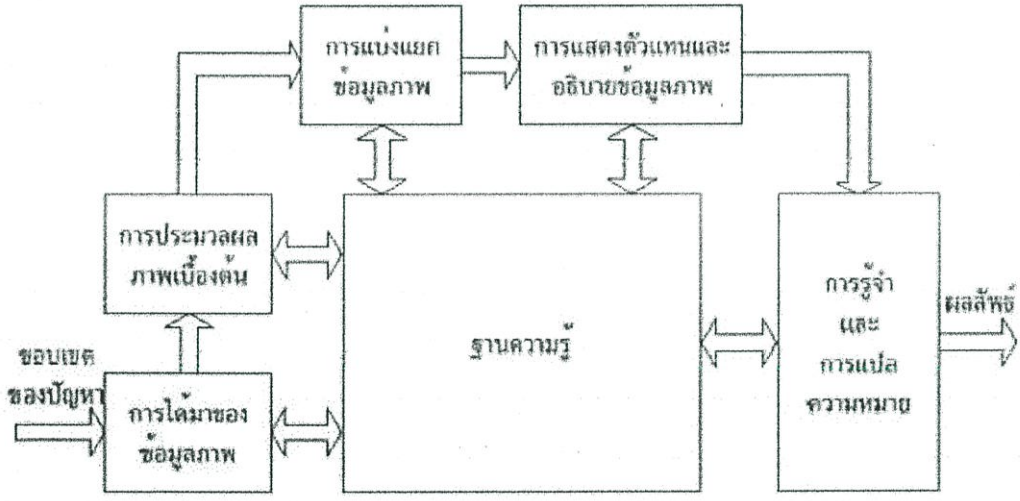
รูปที่ 2.2 (b) แสดงการผสมสีทางแสง (Additive Primary Color)

2.2.4 กระบวนการทางด้านการประมวลภาพด้วยคอมพิวเตอร์

การประมวลผลภาพด้วยคอมพิวเตอร์หรือที่นิยมเรียกกันว่า การประมวลผลภาพดิจิตอล (Digital Image Processing) เป็นกระบวนการที่มีเทคนิควิธีในการประมวณผลข้อมูลตัวเลขของภาพที่มีหลากหลายวิธี ซึ่งสามารถเลือกไปประยุกต์ใช้งานให้เหมาะสมกับข้อมูลภาพที่นำเข้ามาประมวลผล โดยปกติแล้วข้อมูลภาพจะมีลักษณะเด่นทางด้านรูปร่าง พื้นผิว สีสันและโครงสร้างต่างๆที่แตกต่างกันไปขึ้นอยู่วัตถุและสภาพแวดล้อมโดยรอบของวัตถุ

2.2.4.1 ขั้นตอนพื้นฐานของการประมวณผลภาพดิตจิตอล

การประมวลผลภาพดิจิตอลสามารถทำงานในรูปแบบของฮาร์ดแวร์หรือซอฟต์แวร์ได้หลักทฤษฎีการประมวลผลภาพ มีการทำงานตามขั้นตอนอย่างเหมาะสมและเป็นไปตามเทคนิคทางด้านการประมวลผลภาพ ดัง figure 2.3



รูปที่ 2.3 ขั้นตอนพื้นฐานของการประมวณผลภาพดิจิตอล

2.2.4.1.1 การได้มาของข้อมูลภาพ (Image Acquisition)

เป็นการนำข้อมูลภาพเข้าสู่คอมพิวเตอร์ โดยอาศัยตัวรับรู้สัญญาณภาพและสามารถแปลงให้เป็นสัญญาณระบบดิจิตอลด้วยตัวรับรู้ เช่น กล้องถ่ายภาพดิจิตอล กล้องวีดีทัศน์ กล้องเว็บแคม เครื่องสแกน หรืออุปกรณ์รับสัญญาภาพอื่นๆ ที่เหมาะสมกับระบบงานแต่ละระบบ อย่างไรก็ตามรูปแบบของข้อมูลจะถูกจัดเก็บให้อยู่ในลักษณะของภาพ 2 มิติ ที่มีความส่างของแสงหรือความคมชัดแตกต่างกันของแตจ่ละจุดภาพในตำแหน่งต่างๆ

2.2.4.1.2 การประมวลผลภาพเบื้องต้น (Image Preprocessing)

เป็นเทคนิควิธีของการปรับปรุงคุณภาพของข้อมูลภาพดิจิตอลที่ได้จากขั้นตอนการนำเข้าภาพ เพื่อให้ข้อมูลมีความถูกต้องสมบูรณ์ตามความเป็นจริงก่อนนำไปประมวลผล โดยปกติแล้วการปรับปรุงคุณภาพของข้อมูลภาพดิจิตอลมีหลายหลายเทคนิค เช่น การปรับความคมชัด การปรับความว่าง การกำจัดสัญญาณรบกวน การหมุนและการกรองช่วงความถี่ของภาพเป็นต้น

2.2.4.1.3 การแบ่งแยกข้อมูลภาพ (Image Segmentation)

เป็นวิธีการแบ่งแยกข้อมูลภาพออกเป็นส่วนๆ เพื่อทำให้ได้ข้อมูลที่ต้องการออกจากพื้นหลัง โดยทั่วไปผลลัพธ์ของการแย่งแยกข้อมูลภาพจะได้เป็นข้อมูลดิบของจุดภาพที่ประกอบด้วยขอบภาพของแต่ละบริเวณหรือ จุดภาพภายในบริเวณนั้น ในแต่ละกรณีจะต้องทำการแปลงข้อมูลให้มีรูปแบบที่เหมาะสมสำหรับการประมวณผลที่บังคับไว้ จะทำให้การตัดสินใจข้อมูลมีการแสดงตัวแทนของขอบภาพหรือบริเวณนั้นๆ อย่างสมบูรณ์

2.2.4.1.4 การแสดงตัวแทนและอธิบายข้อมูล (Representation and Description)

สำหรับการแสดงภาพหลังจากการแบ่งแยกข้อมูลภาพแล้ว เพื่อให้เห็นถึงลักษณะเด่นและ อธิบายข้อมูลภาพของบริเวณต่างๆของภาพนำเข้า การเลือกตัวแทนสำหรับแสดงข้อมูลเป็นส่วนเดียวของการแก้ปัญหาสำหรับการแปลงมูลดิบเป็นรูปแบบที่เหมาะสมสำหรับการประมวลผลของคอมพิวเตอร์ต่อไป วิธีการ

ที่จะอธิบายลักษณะเด่นของข้อมูลที่สนใจถือเป็นสิ่งสำคัญซึ่งเรียกว่าการเลือกลักษณะเด่น (Feature Extraction) ผลลัพธ์ที่ได้จากการแยกลักษณะเด่นหรือความแตกต่างของข้อมูลที่สนใจออกจากข้อมูลอื่นๆ ก็คือกลุ่มของวัตถุ (Class of Object) ที่ต้องการนั้นเอง

2.2.4.1.5 การรู้จำและการแปลความหมาย (Recognition and Interpretation)

ขั้นตอนสุดท้าของการประมวลผลภาพดิจิตอลหลังจากขั้นตอนการแสดงตัวแทนและอธิบายข้อมูลก็คือการรู้จำภาพ (Image Recognition) ซึ่งเป็นแขนงหนึ่งของการรู้จำแบบรูป (Pattern Recognition) โดยการรู้จำภาพจะต้องรู้จำแบบรูปของแต่ละภาพเป้าหมายเพื่อการให้คำตอบว่าแบบรูปของภาพนำเข้ามีความคล้ายกับแบบรู)ของภาพอ้างอิงภาพใดมากที่สุด และการแปลความหมายนำไปสู่การกำหนดความหมายของชุดข้อมูลรู้จำวัตถุ การได้มาของแบบรูปอ้างอิงนั้นสามารถทำได้หลายวิธี เช่น รูปแบบอ้างอิงอาจอยู่ในรูปแบบจำลองทางคณิตศาสตร์ ซึ่งจะต้องมีวิธีเฉพาะในการเปรียบเทียบ การสร้างแบบจำลองทางคณิตศาสตอร์สามารถทำได้จากขั้นตอนการฝึกฝน (Training Phrase) ซึ่งโดยทั่วไปแล้วจะต้องมีตัวอย่างภาพที่มีลักษณะเดียวกันหลายๆ ภาพ จากนั้นจะทำการคำนวณหาค่าลักษณะเด่นของแต่ละภาพซึ่งผลลัพธ์ที่ได้ก็คือ แบบรูปของภาพเหล่านั้นนั่นเองแบบจำลองของภาพในแต่ละกลุ่มสามารถคำนวณได้จากค่าสถิติต่างๆ ของแบบรูปของภาพในกลุ่มเดียวกัน บางครั้งอาจจะอยู่ในรูปของฐานความรู้ (Knowledge Base) จำนวนมากจนมีกลุ่มข้อมูลที่เก็บไว้เป็นฐานความรู้ในรูปแบบของฐานข้อมูลความรู้ (Knowledge Database)

2.2.4.1.6 ฐานความรู้(Knowledge base) ระบบฐานความรู้ (Knowledge Base)

การสร้างระบบคอมพิวเตอร์ที่สมารถรับรู้และมีความเชี่ยวชาญในด้านใดด้านหนึ่ง โดยนำความรู้จากบุคคลผู้เชี่ยวชาญในสาขาความรู้นั้นมาสร้างเป็นระบบฐานความรู้ (Knowledge Bas) ขึ้นนอกจากนี้ยังมีโปรแกรมที่ควบคุมการค้นหาความรู้ที่ต้องการและโปรแกรมที่ตรวจสอบกฎเกณฑ์และทฤษฎีเพื่อให้ได้คำตอบของปัญหา เพื่อให้คอมพิวเตอร์สามารถแก้ปัญหาตามกฎเกณฑ์และเงื่อนไขที่ได้รับได้

2.2.5 การปรับเท่าฮีสโตแกรม(Histogram Equalization)

การปรับเท่าฮีสโตแกรมเป็นวิธีการปรับคอนทราสท์โดยใช้ฮีสโตแกรมของภาพทำได้โดยการกระจายค่าความเข้มที่เกิดขึ้นในภาพ ซึ่งได้จากการคำนวณหาค่าความเข้มของตุดนั้นๆใหม่ทำให้ฮีสโตแกรมของภาพผลลัพธ์มีการกระจายอย่างสม่ำเสมอ จะใช้สมการในการคำนวณความเข้มของจุดดังนี้

$$p(x_i) = \frac{n_i}{n}, i \in 0, \ldots, L-1 \quad (1)$$

โดย $p(x_i)$ ความน่าจะเป็นที่เกิดขึ้นของพิกเซล(Pixel) ในระดับความเข้ม i

n_i จำนวนจุดที่เกิดขึ้นของภาพที่มีความเข้ม i

n จำนวนพิกเซล (Pixel) ทั้งหมดในภาพ

L จำนวนความเข้มทั้งหมดในภาพ

$$c(i) = \sum_{j=0}^{i} p(x_j) \quad (2)$$

โดย $c(i)$ ฟังก์ชันการกระจายสะสม (Cumulative distribution function)

2.2.6 การหารัศมีวงกลมโดยทฤษฎีฮัฟทรานสฟอร์ม (Hough Transform)

การทำ Hough Transform สำหรับวงกลมจะเป็นการคำนวณในระบบ 3 มิติ (3 Dimensional Space) คือ (a,b,r) โดยที่ (a,b) คือตำแหน่งจุดศูนย์กลางของวงกลมและสมการวงกลมเป็นดังสมการ

$$(x-a)^2 + (y-b)^2 = r^2 \quad (3)$$

โดยที่

$$x = a + r\cos\theta \quad (4)$$

$$y = b + r\sin\theta$$

ซึ่งวิธีนี้จะเป็นการพยายามสร้างวงกลมที่เหมาะสมที่สุดกับจุดที่เป็นขอบของชิ้นงาน โดยอาศัยอาร์เรย์สะสม (Accumulator Array) M[a,b] ซึ่งจะทำการเลือกค่าที่มากที่สุดในอาร์เรย์สะสม M [a,b] และกรณีที่ทราบค่ารัศมี r วิธีการทำฮัฟทรานฟอร์ม อธิบายได้ดังนี้

1.เริ่มต้นด้วยการกำหนดปริมาณพารามิเตอร์ สำหรับ a และ b

2.กำหนดค่าเริ่มต้นให้กับอาร์เรย์สะสม M [a,b] จากนั้นทำการคำนวณหาค่าขนาด G [x,y]และทิศทางแกรเดียนต์ Q[x,y]

3.แล้วทำการการคำนวณค่า a,b วาดเป็นวงกลมที่ $0<\theta<360$ ซึ่ง a,b สามารถหาได้ดังนี้

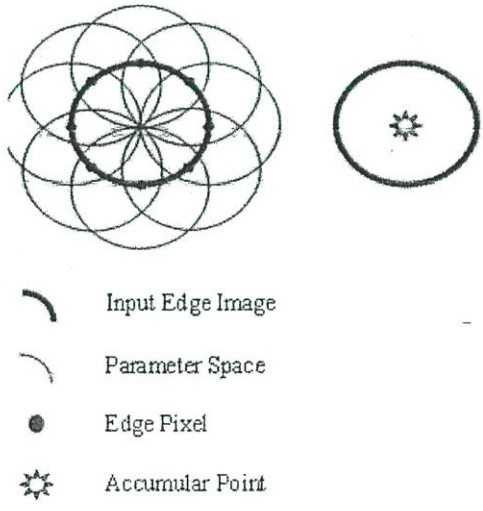
$$a = x - r\cos\theta \tag{5}$$

$$b = y - r\sin\theta$$

โดยที่ x , y คือจุดของขอบภาพ

4. จากนั้นวาดวงกลมในพื้นที่สะสม (Accumulator Space) ซึ่งจะกระทำทุกจุดของขอบภาพ ดังตัวอย่าง figure() จากนั้นเพิ่มค่าที่ละหนึ่งในอาร์เรย์สะสม ณ ตำแหน่งที่วงกลมวาดผ่าน M[a,b]

5.เมื่อกระทำทุกจุดของขอบภาพแล้วทำการเลือกค่าที่มากที่สุดในอาร์เรย์สะสม M [a,b] ซึ่ง a,b ที่ได้คือจุดศูนย์กลางของวงกลมในภาพที่รัศมีเท่ากับ r



รูปที่ 2.4 การ Detect วงกลมโดยใช้ Hough Transform

2.2.7 Morphological Image Processing

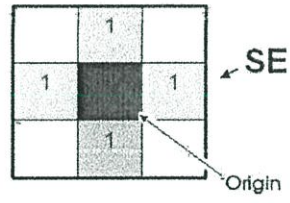
Mathematical morphology เป็นเครื่องมือที่ใช้งานด้าน Digital การประมวลผลภาพสำหรับตัดต่อหรือแต่งเติมส่วนขอบของภาพ โครงสร้างของภาพ โดยใช้ทฤษฎีของเซ็ต ซึ่งเซ็ตใน Morphology จะแทนรูปร่างหรือรูปทรงของวัตถุในภาพ เช่นกลุ่มของสีดำทั้งหมดในภาพไบนารีสำหรับการทำ Morphological สามารถใช้ในการกำจัดสัญญาณรบกวน ขยายพื้นที่ของวัตถุ และกำจัดส่วนเกินของวัตถุได้

2.2.7.1 Dilation and Erosion

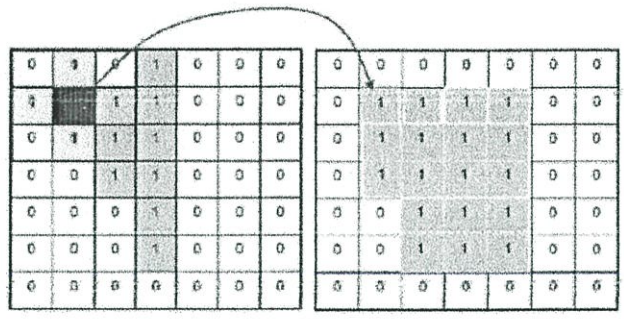
Dilation คือ การขยายพิกเซลของภาพ โดยการสแกนค่าของ SE (Structuring Element) ดังรูปที่ 14 บนแต่ละค่าของพิกเซลภาพ โดยทำการสแกนจากตำแหน่งบนซ้ายไปยังตำแหน่งล่างขวา ซึ่งจะเปลี่ยนค่าของพิกเซลที่มีค่าเป็น 0 ให้มีค่าเป็น 1 เมื่อค่าของพิกเซลใด ๆพิกเซลหนึ่งบน SE มีค่าตรงกับค่าของพิกเซลภาพ และจะมีค่าคงเดิม เมื่อทุกค่าของ SE มีค่าตรงกับทุกค่าของพิกเซลภาพ แสดงดังรูปที่ 2.5 โดยมีสมการดังนี้

$$A \oplus B = \bigcup_{x \in B} (A_x) \qquad (6)$$

เรียก B ว่าเป็น Structuring element in dilation ความหมายคือทุก ๆ พิกเซล ทำการเคลื่อนย้ายไปยัง A โดยการยูเนียนไปตามพิกัดของ x

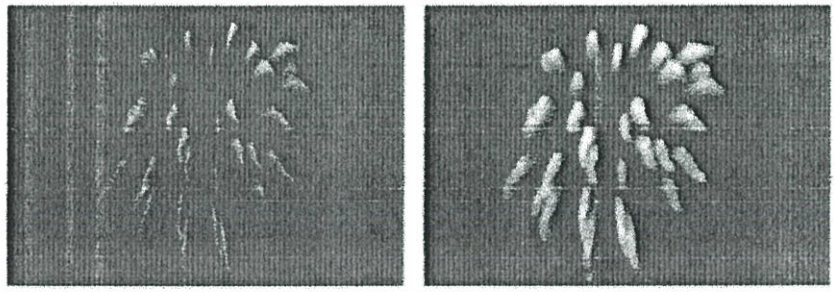


รูปที่ 2.5 ค่าของ SE (Structuring Element)



รูปที่ 2.6 การทำงานของ Dilation

จากรูปที่ 2.6 เมื่อค่าของพิกเซลใน SE ตรงกับค่าของพิกเซลใด ๆ พิกเซลหนึ่งของภาพพิกเซลที่ตำแหน่ง Origin จะเปลี่ยนเป็น 1 ผลลัพธ์ของ Dilation แสดงดังรูปที่ 2.7 (b)



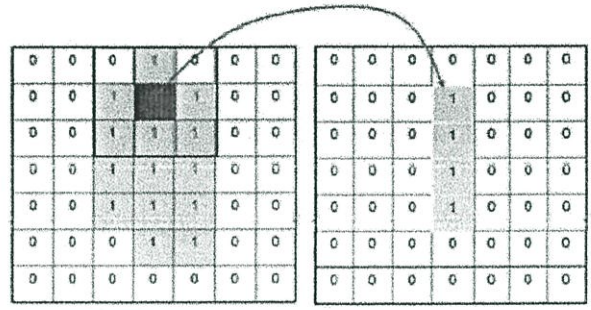
(a) (b)

รูปที่ 2.7 ภาพการทำ Dilation (a) ภาพต้นฉบับ (b) ผลลัพธ์จากการทำ Dilation

Erosion เป็นวิธีการที่ตรงข้ามกับ Dilation คือจะลดขนาดของพิกเซล โดยการสแกนค่าของ SE บนแต่ละค่าของพิกเซลภาพ โดยทำการสแกนจากตำแหน่งบนซ้ายไปยังตำแหน่งล่างขวา ซึ่งจะเปลี่ยนค่าของพิกเซลที่มีค่าเป็น 1ให้มีค่าเป็น 0 เมื่อพิกเซลใดพิกเซลหนึ่งบน SE มีค่าตรงกับค่าของพิกเซลภาพ และจะมีค่าคงเดิม เมื่อทุกพิกเซลของ SE มีค่าตรงกับค่าของพิกเซลภาพแสดงดังรูปที่ 2.8โดยมีสมการดังนี้

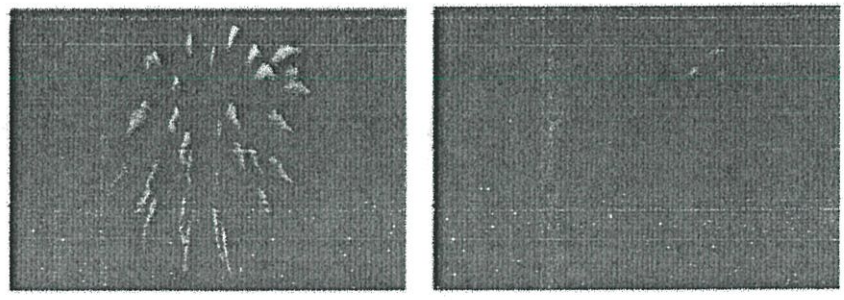
$$A\Theta B = \{w : B_w \subset A\} \quad (7)$$

เรียก B ว่าเป็น Structuring element in Erosion ความหมายคือ *Bw* เป็นสับเซตของ A โดยที่ค่าของ B จะต้องประกอบด้วยทุก ๆ พิกเซลของ w มีพิกัดเป็น (x, y) ซึ่งค่า *Bw* จะต้องอยู่ใน A



รูปที่ 2.8 การทำงานของ Erosion

จากรูปที่ 2.8 เมื่อค่าของพิกเซลใน SE ทุก ๆ พิกเซลมีค่าตรงกับค่าของพิกเซลในภาพทุกตำแหน่ง พิกเซลที่ตำแหน่ง Origin จะมีค่าคงเดิม และจะมีค่าเป็น 0 เมื่อค่าของ SE ตรงกับค่าของพิกเซลใดพิกเซลหนึ่งของภาพ ผลลัพธ์ของ Erosion แสดงดังรูปที่ 18 (b)



(a) (b)

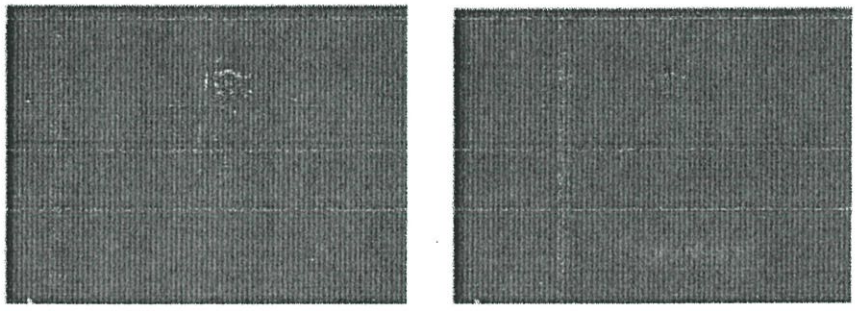
Figure 2.9 ภาพการทำ Erosion (a) ภาพต้นฉบับ (b) ผลลัพธ์จากการทำ Erosion

2.2.7.2 Opening and Closing

Opening ใช้เพื่อกำจัดรายละเอียดขนาดเล็กของภาพ และการทำ Opening จะทำให้พิกเซลของภาพจะถูกเปิดกว้างมากขึ้นดังรูปที่ 19 และวิธีการของ Opening คือการทำ Erosion ก่อน จากนั้นจึงทำ Dilation ดังสมการ

$$A \circ B = (A \oplus B) \ominus B \qquad (8)$$

เรียก B ว่าเป็น Structuring element



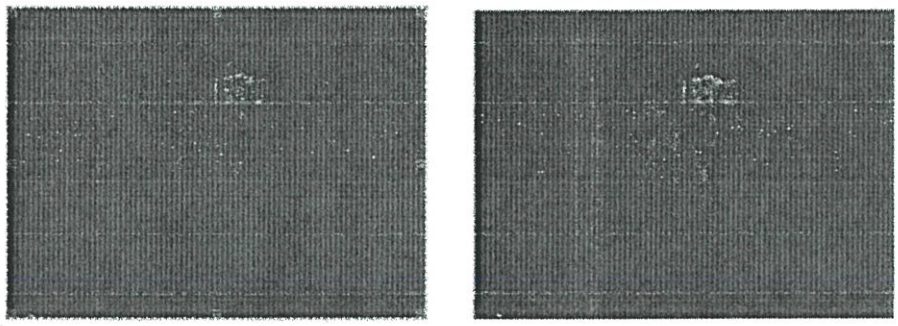
(a) (b)

รูปที่ 2.10 ภาพการทำ Opening (a) ภาพต้นฉบับ (b) ผลลัพธ์จากการทำ Opening

Closing ทำในวิธีตรงข้ามกับ Opening จะเป็นการทำให้ภาพมีการเชื่อมต่อกันมากขึ้นและการทำ Closing จะทำให้พิกเซลของภาพจะถูกปิดเชื่อมต่อกันมากขึ้นดังรูปที่ 20 วิธีการทำ Closing คือการทำ Dilation ก่อน จากนั้นจึงทำ Erosion ดังสมการ

$$A \bullet B = (A \oplus B) \ominus B \qquad (9)$$

เรียก B ว่าเป็น Structuring element



(a) (b)

รูปที่ 2.11 ภาพการทำ closing (a) ภาพต้นฉบับ (b) ผลลัพธ์จากการทำ closing

2.2.8 มอเตอร์กระแสตรง

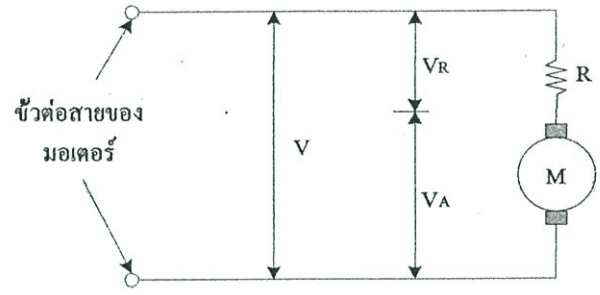
2.2.8.1 หลักการทำงานของมอเตอร์กระแสตรง

เมื่อมีการผ่านกระแสไฟฟ้าเข้าไปยังขดลวดในสนามแม่เหล็กจะทำให้เกิดแรงแม่เหล็กซึ่งมีสัดส่วนของแรงขึ้นกับกระแสแรงของสนามแม่เหล็ก โดยแรงจะเกิดขึ้นเป็นมุมฉากกับกระแสและสนามแม่เหล็ก ขณะที่ทิศทางของแรงกลับตรงกันข้ามกัน ถ้าหากกระแสของสนามแม่เหล็กไหลย้อนกลับจะทำให้เกิดการเปลี่ยนแปลงของกระแส และ สนามแม่เหล็กเป็นผลทำให้ทิศทางของแรงเปลี่ยนไป ด้วยคุณสมบัตินี้ทำให้มอเตอร์กระแสตรงกลับทิศทางการหมุนได้

สนามแม่เหล็กของมอเตอร์ส่วนหนึ่งเกิดขึ้นจากแม่เหล็กถาวรซึ่งจะถูกยึดติดกับแผ่นเหล็ก หรือเหล็กกล้า โดยปกติส่วนนี้จะเป็นส่วนที่ยึดอยู่กับที่ และ ขดลวดเหนี่ยวนำจะพันอยู่กับส่วนที่เป็นแกนหมุนของมอเตอร์

2.2.8.2 คุณสมบัติของมอเตอร์กระแสตรง

ในการอธิบายคุณสมบัติของมอเตอร์กระแสตรงให้ละเอียดนั้นต้องพิจารณาแรงดันที่ป้อนและความต้านทานของโรเตอร์ด้วย วงจรภายในของมอเตอร์เขียนได้ดังรูปที่ 3



รูปที่ 2.12 วงจรภายในของมอเตอร์กระแสตรง

โดยสมมติให้ทุ่นโรเตอร์ไม่มีความต้านทานอยู่เลย อนุกรมกับความต้านทานซึ่งในที่นี้ก็คือความต้านทานของขดลวดนั่นเอง แรงดันที่ขั้วต่อสายของมอเตอร์ก็คือผลบวกระหว่างแรงดันที่ทุ่นโรเตอร์ (V_A) และแรงดันตกคร่อมความต้านทานขดลวด (V_R)

แรงดัน V_A ถูกเรียกว่า แรงเคลื่อนเหนี่ยวนำป้อนกลับ (BACK EMF) ซึ่งเกิดขึ้นในโรเตอร์ขณะที่หมุน แรงดันที่เกิดขึ้นนี้เป็นไปตากกฎของการเหนี่ยวนำแม่เหล็กไฟฟ้าจากการเคลื่อนที่ของตัวนำในสนามแม่เหล็ก สัมพันธ์กับแรงเคลื่อนเหนี่ยวนำแม่เหล็ก และ ความเร็วในการเคลื่อนที่ของตัวนำ แรงดันที่เกิดขึ้นจะมีขั้วตรงกันข้ามกับแรงดันที่ป้อนให้กับมอเตอร์ และ แปรผันตรงกับความเร็วในการหมุน ผลบวกของแรงดันที่ทุ่นโรเตอร์ (V_A) และแรงดันตกคร่อมขดลวด (V_R) ต้องเท่ากับแรงดันที่ป้อนให้กับมอเตอร์ (V)

$$V = V_A + V_R \qquad (V)$$

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

$$I \equiv V_R / R \qquad (A)$$

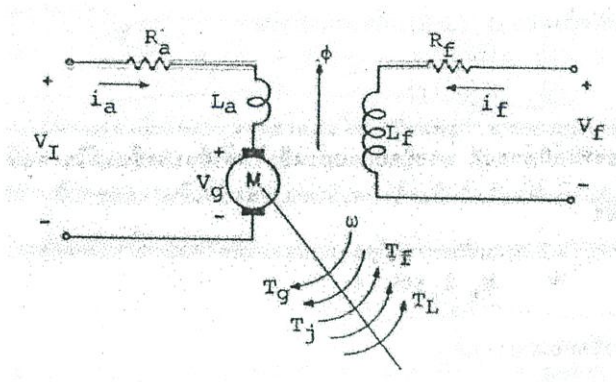
เมื่อมอเตอร์เริ่มหมุนจะมีความเร็ว และ V_A เพิ่มขึ้นเป็นเส้นตรงตามความเร็ว V_R ซึ่งมีค่าเท่ากับความแตกต่างระหว่าง V_A และ V จะเริ่มลดลงกระแส I ก็จะเริ่มลดลงเช่นกันขณะที่มอเตอร์ยังมีความเร่งอยู่ ความเร็วจะเพิ่มขึ้น แรงบิดจะลดลงจนกว่าจะถึงจุดซึ่งแรงบิดจองมอเตอร์รับภาระโหลดได้สมดุลพอดี ขณะที่มอเตอร์ไม่มีโหลด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

2.2.8.3 โมเดลคณิตศาสตร์ของดีซีมอเตอร์

ดีซีมอเตอร์ที่ใช้ร่วมกับดีซีแอมปลิไฟล์ทั้งในระบบการบังคับตำแหน่งและการบังคับความเร็วมักจะได้รับการประยุกต์ใช้เป็นส่วนประกอบสร้างกำลังงานในระบบการนำร่องและระบบบังคับต่างๆ และเนื่องจากวิทยาการเกี่ยวกับสารแม่เหล็กและการขยายด้วยโซลิคสเตททำให้ดีซีมอเตอร์แบบแม่เหล็กถาวรได้รับความนิยมใช้เป็นส่วนประกอบการขับเคลื่อนในระบบการบังคับแบบปิดลูฟต่างๆ มากขึ้น การออกแบบและการชดเชยระบบดังกล่าวได้อย่างเหมาะสมจะต้องใช้โมเดลทางคณิตศาสตร์ของส่วนประกอบทั้งหมดในระบบ

รูปที่ 2.13 แสดงอินพุตและเอาต์พุตของโมเดลทางคณิตศาสตร์ของมอเตอร์

โมเดลอิเล็คโทรแมคคานิคอล ส่วนสำคัญของดีซีมอเตอร์แบบฟิลด์แยกกระตุ้นมีโมเดลดังแสดงในรูป 2.14



รูปที่ 2.14 แสดงโมเดลของดีซีมอเตอร์แบบฟิลด์แยกกระตุ้น

R_a: ความต้านทานของอาร์เมเจอร์

L_a: อินดัคแต้นซ์ของอาร์เมเจอร์

V_g: โวลท์เต็จกำเนิดในอาร์เมเจอร์(โวลท์เต็จย้อนกลับ)

R_f: ความต้านทานของฟิลด์

L_f: อินดัคแต้นซ์ของฟิลด์

ϕ: ช่องว่างอากาศของเส้นแรงสนามแม่เหล็ก

ω: ความเร็วของเพลาอาร์เมเจอร์

T_g: แรงบิดที่พัฒนาขึ้นในมอเตอร์

T_f: แรงบิดเสียดทานของมอเตอร์

T_J: แรงเฉื่อยของมอเตอร์

T_L: แรงบิดโหลดบนเพลาของมอเตอร์

ขั้นแรกเราจะหาสมการพื้นฐานโมเดลของดีซีมอเตอร์ได้จากลูฟของอาร์เมเจอร์

$$V_f(t) = R_a i_a(t) + L_a \frac{di_a(t)}{dt} + V_g(t) \qquad \text{=====} (1)$$

เทอมโวลท์เต็จ $V_g(t)$ ในสมการ (1) คือโวลท์เต็จย้อนกลับของมอเตอร์ซึ่งเกิดขึ้นเมื่อเส้นลวดตัวนำของอาร์มาเจอร์หมุนตัดเส้นแรงแม่เหล็กซึ่งเกิดขึ้นในกระแสของฟิลด์ (i_f) ตามกฎของฟาราเดย์ลูฟของเส้นลวดตัวนำหมุนในฟิลด์แม่เหล็กคงที่จะมีการเหนี่ยวนำโวลท์เต็จขึ้นในขดลวดนั้น

$$V_f(t) = \frac{d\lambda(t)}{dt} \qquad \text{-----} (2)$$

เมื่อ $\lambda(t)$ คือเส้นแรงแม่เหล็กที่ลิงเคจ(linkages) ไปยังขดลวดและ t คือเวลาในการหมุนของค็อมนิวเทเตอร์ของมอเตอร์ การควบคุมวงจรของแต่ละส่วนของตัวนำในโรเตอร์จะเกิดโวลท์เต็จขึ้นในส่วนของตัวนำนั้นตามสมการ (2) เมื่อ $\frac{d\lambda(t)}{dt}$ จะเป็นสัดส่วนต่อเส้นแรงแม่เหล็กในช่องว่างอากาศและความเร็วเชิงมุม $\omega(t)$ เราจะได้ว่า

$$V_g(t) = K\phi(t)\omega(t) \qquad \text{-----} (3)$$

สมมติให้กระแสของฟิลด์มีค่าคงที่และไม่คิดถึงส่วนการเปลี่ยนแปลงในเส้นแรงฟิลด์เนื่องจากอาร์เมเจอร์รีแอคชั่นเส้นแรงฟิลด์ก็จะมีค่าคงที่ดังนั้นสมการ (3) ก็จะเป็น

$$V_g(t) = K_e\omega(t) \qquad \text{----} (4)$$

เมื่อเราสมมติให้เส้นแรงของฟิลมีค่าคงที่ แรงบิดของแม่เหล็กไฟฟ้าซึ่งเกิดขึ้นแก่โรเตอร์ของมอเตอร์จะเป็นสัดส่วนกับกระแสของอาร์มาเจอร์

$$T_g(t) = K_t\, i_a(t) \qquad \text{----} (5)$$

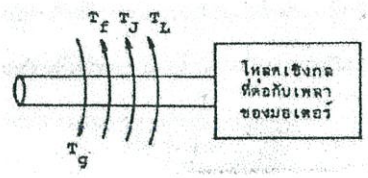
เมื่อ K_t คือ ค่าคงที่ของแรงบิดของมอเตอร์

กำลังงานเชิงกลที่เกิดขึ้นในโรเตอร์คือผลคูณของแรงบิดที่เกิดขึ้นและความเร็วเชิงมุม

$$P_g(t) = T_g(t)\,\omega(t) \qquad \text{----} (6)$$

กำลังงานเชิงกลที่เกิดขึ้นในโรเตอร์ทั้งหมดนี้จะจ่ายไปยังโหลดที่ต่ออยู่กับเพลาของมอเตอร์แต่กำลังงานนี้บางส่วนจะสูญเสียไปในมอเตอร์ การสูญเสียจากแรงเสียดทาน หมายถึงความหน่วงเนื่องจากลมที่มีต่อโร

เตอร์ แรงเสียดทานตัวรองรับโรเตอร์ กระแสที่ไหลวนในเหล็กของโรเตอร์และฮีสเทรีซิส โดยแรงบิดต่างๆแสดงดังนี้



รูปที่ 2.15 แสดงถึงแรงบิดต่างๆที่เกิดขึ้นต่อโหลดของมอเตอร์

$T_g(t)$: แรงบิดของมอเตอร์

$T_f(t)$: แรงบิดที่ต้องชนะการสูญเสียเนื่องจากการเสียดทาน

$T_J(t)$: แรงบิดเพื่อใช้เพิ่มอัตราเร่งแก่ความเฉื่อยของโหลด

$T_L(t)$: แรงบิดโหลด

ในช่วงเวลาใดๆก็ตาม แรงบิดของมอเตอร์จะต้องเท่ากับและมีทิศทางตรงข้ามกับผลรวมของแรงบิด $T_f(t)$ $T_J(t)$ และ $T_L(t)$ ดังนั้น

$$T_g(t) = T_f(t) + T_L(t) + J\frac{d\omega(t)}{dt} \qquad \text{----- (7)}$$

เมื่อ J คือผลรวมของโมเม้นท์แรงเฉื่อยของโรเตอร์และโหลดที่ต่ออยู่ที่เพาของมอเตอร์

ผลรวมของแรงบิดเสียดทานที่ประกอบกันขึ้นที่เพลาของมอเตอร์ซึ่งเป็นลิเนียร์ฟังก์ชันกับความเร็วเชิงมุมของโรเตอร์เรียกว่า ส่วนประกอบของวิสคอสฟริคชั่นและมักจะอยู่ในเทอมที่แยกออกจากฟริคชั่นอื่นๆ ซึ่งแสดงได้ด้วยสมการต่อไปนี้

$$T_g(t) = T_f(t) + T_L(t) + J\frac{d\omega(t)}{dt} + B\omega(t) \qquad \text{----- (8)}$$

เมื่อ B คือสัมประสิทธิ์ของวิสคอสฟริคชั่นของมอเตอร์และโหลดที่ต่ออยู่กับเพลาของมอเตอร์ $T_f(t)$ คือผลรวมของฟริคชั่นของโหลดและมอเตอร์ทั้งหมด มีแรงต้านของลมและการสูญเสียกำลังในเหล็กของเพลามอเตอร์ยกเว้นวิสคอสฟริคชั่น

สมการ (1) (4) (5) และ (8) เป็นชุดสมการพื้นฐานของดีซีมอเตอร์โมเดลและสมการเหล่านี้เราสามารถจะหาทรานสเฟอร์ฟังก์ชั่นของดีซีมอเตอร์ได้ โดยใส่ลาปลาสทรานสฟอร์มทั้งสองข้างของชุดสมการพื้นฐานได้เป็น

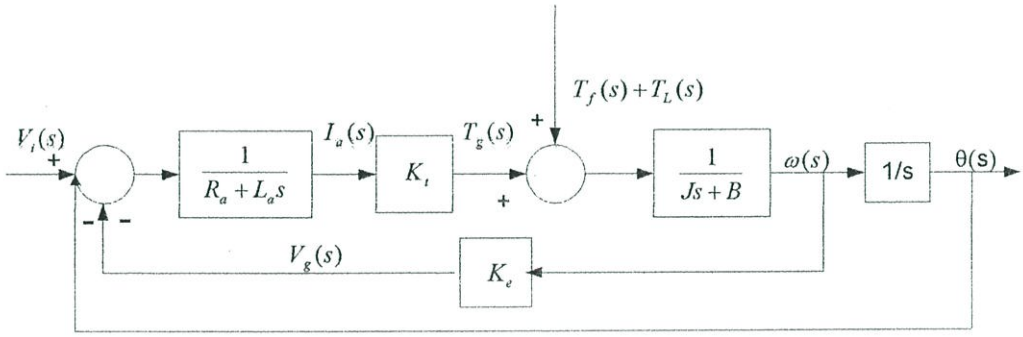
$$V_i(s) - V_g(s) = (R_a + sL_a)\, I_a(s) \qquad \text{----- (9)}$$

$$V_g(s) = K_e \omega(s) \qquad \text{----- (10)}$$

$$T_g(s) = K_t\, I_a(s) \qquad \text{----- (11)}$$

$$T_g(s) - T_f(s) - T_L(s) = (B + sJ)\, \omega(s) \qquad \text{----- (12)}$$

สามารถเขียนเป็นบล็อกไดอะแกรมที่แสดงสมการพื้นฐานเหล่านี้ได้ดังนี้



รูปที่ 2.16 แสดงบล็อกไดอะแกรมของดีซีมอเตอร์โมเดล

ข้อสังเกต

สมมติว่าโวลท์เต็จที่ป้อนให้กับวงจรอาร์มาเจอร์ของมอเตอร์มีค่าคงที่ดังนั้นมอเตอร์จะหมุนด้วยความเร็วคงที่คือทำงานอยู่ที่สภาวะสงบนิ่งด้วยโหลดที่คงที่ กำลังงานเชิงกลที่เกิดขึ้นโดยโรเตอร์จะหาได้จากสมการ (6) จะได้

$$P_g = T_g \omega = K_t\, I_a\, \omega \qquad \text{---- (13)}$$

เมื่อทุกเทอมในสมการสุดท้ายมีค่าคงที่เนื่องจากมอเตอร์ทำงานอยู่ที่สภาวะสงบนิ่งกำลังไฟฟ้าที่ถูกดูดกลืนโดยอาร์เมเจอร์ต้องเท่ากับ

$$P = V_s I_a = K_e \omega I_a \qquad \text{---- (14)}$$

ดังนั้นเราจะได้ว่ากำลังงานเชิงกลที่เกิดขึ้นต้องเท่ากับกำลังงานไฟฟ้าที่ถูกดูดกลืนในโรเตอร์คือสรุปได้ว่า $K_e = K_t$

ทรานสเฟอร์ฟังก์ชั่นของดีซีมอเตอร์

บล็อกไดอะแกรมของรูปที่ 5 แสดงถึงระบบที่มีสองอินพุท และมีเอาต์พุทเป็นความเร็วเชิงมุม $\omega(s)$ และการเคลื่อนที่แบบเชิงมุม □(s) จากรูปที่ 5 ความเร็วเอาท์พุทของระบบเขียนได้เป็น

$$\omega(s) = G_1(s)V_i(s) + G_2(s)[T_f(s) + T_L(s)] \qquad \text{---- (15)}$$

เมื่อ

$$G_1(s) = \left.\frac{\omega(s)}{V_i(s)}\right|_{T_f(s)+T_L(s)=0} \qquad \text{---- (16)}$$

$$G_2(s) = \left.\frac{\omega(s)}{T_f(s)+T_L(s)}\right|_{V_i(s)=0} \qquad \text{---- (17)}$$

$G_1(s)$ คือทรานสเฟอร์ฟังก์ชั่นระหว่างโวลท์เต็จและความเร็ว

$$G_1(s) = \frac{\omega(s)}{V_i(s)} = \frac{K_t}{(L_a s + R_a)(Js + B) + K_t K_e}$$

$$= \frac{K_m}{\alpha s^2 + \beta s + 1} \qquad \text{----- (18)}$$

เมื่อ

$$K_m = \frac{K_t}{R_a B + K_t K_e}$$

$$\alpha = \frac{L_a J}{R_a B + K_t K_e}$$

$$\beta = \frac{R_a J + L_a B}{R_a B + K_t K_e}$$

สมการ (18) เป็นโวลท์เต็จทรานสเฟอร์ฟังก์ชั่นของดีซีมอเตอร์ในเมื่อสมมติว่า $\bar{T}_f$ และ$\bar{T}_L$มีค่าเป็นศูนย์ สมการ (18) สามารถเขียนใหม่ได้เป็น

$$G_1(s) = \frac{K_t}{R_a B(1+\tau_e s)(1+\tau_m s) + K_t K_e}$$

เมื่อ $\tau_e = \frac{L_a}{R_a}$ = ไทม์คอนสแต้นท์ทางไฟฟ้า

$\tau_m = \frac{J}{B}$ = ไทม์คอนสแต้นท์ทางเชิงกล

ถ้าอินดัคแต้นท์ของอาร์มาเจอร์มีค่าน้อย ไทม์คอนสแต้นท์ทางไฟฟ้าสามารถตัดทิ้งได้และได้สมการเป็น

$$G_V(s) = \frac{\omega(s)}{V_i(s)} = \frac{K_t}{R_a(Js+B) + K_t K_e}$$

$$= \frac{K_m}{\tau s + 1} \qquad \text{----- (19)}$$

เมื่อ $\tau = \frac{R_a J}{R_a B + K_t K_e}$

ในสมการ (19) ค่าคงที่ K_m อาจเรียกได้ว่าเป็นค่าคงที่ของมอเตอร์

ทรานสเฟอร์ฟังก์ชั่นแรงบิดโหลด $G_2(s)$ หาได้เป็น

$$G_2(s) = \frac{\omega(s)}{T_f(s) + T_L(s)} = \frac{\frac{1}{Js+B}}{1 + \frac{K_t K_e}{(Js+B)(L_a s + R_a)}}$$

$$= \frac{-\frac{R_a}{K_t}K_m\left[\frac{L_a}{R_a}s+1\right]}{\alpha s^2+\beta s+1} \qquad \text{---- (20)}$$

ซึ่งถ้าอินดัคแต้นท์ของอาร์มาเจอร์ไม่นำมาคิด จะทำให้ได้สมการ ดังนี้

$$G_L(s)=\frac{\omega(s)}{T_f(s)+T_L(s)}=\frac{-\frac{R_a}{K_t}K_m}{\tau s+1}$$

ซึ่งจากสมการที่ (15) เมื่อให้ค่าของ ว่า T_f และT_Lมีค่าเป็นศูนย์จะทำให้ค่าทรานสเฟอร์ฟังก์ชั่นมีค่าดังนี้

$$\boxed{\omega(s)=G_v(s)V_i(s)=\frac{K_m}{\tau s+1}V_i(s)}$$

และจากรูปที่ 5 ตำแหน่งเอาท์พุทของระบบเป็นดังนี้

โมเดลคณิตศาสตร์ในการควบคุมตำแหน่งของมอเตอร์ จาก transfer function ในหัวข้อที่ 3.3 โมเดลคณิตศาสตร์ของดีซีมอเตอร์ ซึ่งจะได้ model ของการควบคุมความเร็วของมอเตอร์ในรูปของสมการอันดับหนึ่งเป็นดังนี้ คือ

$$\boxed{\omega(s)=G_v(s)V_i(s)=\frac{K_m}{\tau s+1}V_i(s)}$$

จากสมการข้างต้นดังกล่าวนั้นจะเห็นถึงความสัมพันธ์ระหว่างความเร็วเชิงมุม (output) และ ค่าแรงดันที่ป้อน (input)

และในการควบคุมตำแหน่งจะมีการผ่านตัว Integrator (1/s) ทำให้ได้เอาท์พุตคือ $\Theta(s)$ ซึ่งจากรูปที่ 5. และ transfer function ที่ได้จากหัวข้อที่ 3.2 ข้างต้นนั้นเมื่อทำการผ่าน Integrator (1/s) เข้าไปจะทำให้ได้ค่าของ output เป็นมุมในการเคลื่อนที่ของมอเตอร์ ซึ่งเขียนในรูปของสมการได้ดังนี้

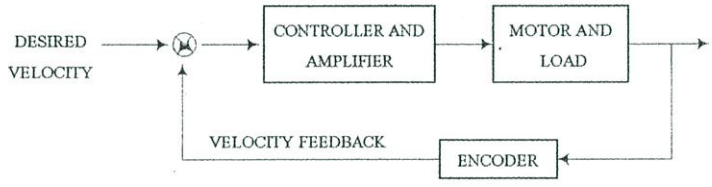
$$\theta(s) = \frac{\omega(s)}{s} = \left[\frac{1}{s}\right]\left[\frac{K_m V_i(s)}{\varpi + 1}\right]$$

ส่วนในการหาโมเดลมอเตอร์ที่ใช้ในการทดลองนั้น เรากำหนดให้โมเดลของมอเตอร์เป็นโมเดลมอเตอร์อันดับหนึ่ง (First Order)

หลักการควบคุมความเร็ว และ ตำแหน่ง

รูปที่ 6 คือ บล็อกไดอะแกรมของระบบควบคุมความเร็วในระบบกลไกแบบเซอร์โวนี้มอเตอร์ถูกเรียกว่า มอเตอร์แบบเซอร์โว (เซอร์โว(Servo) motor) เครื่องวัดรอบในลูปแบบป้อนกลับจะวัดความเร็วของมอเตอร์แบบเซอร์โว และ ส่งป้อนกลับมาในรูปของสัญญาณไฟฟ้า (แรงดัน หรือ กรพะแส) ซึ่ง แปลตามความเร็วเพลาของมอเตอร์ ในที่นี้ลูปป้อนกลับจะทำให้ความเร็วเอาท์พุทของมอเตอร์มีค่าคงที่มากขึ้น

ระบบแบบ closed loop ถูกใช้รวมอยู่ในระบบเครื่องมือ เครื่องจักร หรือ สว่านกำลังเพื่อชดเชยความแตกต่างของโหลดหรือวัสดุที่ถูกตัดหรือถูกเจาะ วัสดุเนื้อแข็งจะหน่วงความเร็วของสว่าน หรือ เครื่องมือที่ใช้ในการตัด และ สว่านจะมีความเร็วเพิ่มขึ้นในวัสดุเนื้ออ่อน



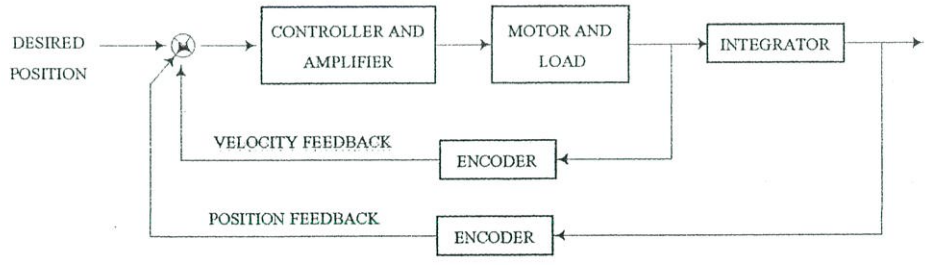
รูปที่ 2.17 ระบบควบคุมความเร็ว ที่ประกอบด้วยลูปการควบคุมป้อนกลับเพียงลูปเดียวเหมาะสำหรับอุปกรณ์สว่านไฟฟ้าเคลื่อนที่

ในลูปป้อนกลับความเร็วของมอเตอร์ ที่ประกอบด้วยเครื่องวัดความเร็ว ความเร็วของเครื่องมือจะยังคงมีค่าคงที่ เนื่องจากเมื่อเครื่องมือที่ใช้ในการตัดมีความเร็วลดลง สัญญาณป้อนกลับจะควบคุมมอเตอร์ให้เพิ่มความเร็วขึ้น ในขณะเดียวกัน เมื่อเครื่องมือตัดชิ้นงานที่เป็นวัสดุอ่อน ลูปป้อนกลับจะป้องกันไม่ให้มอเตอร์เร่งความเร็วเกินขนาด

อย่างไรก็ตาม ถ้าการประยุกต์ใช้งานมีความต้องการ วงจรควบคุมเพิ่มเติมสามารถถูกเพิ่มขึ้นมาเพื่อทำให้มอเตอร์มีความเร็วค่อย ๆ เพิ่มขึ้น และ ค่อย ๆ ลดลงจนกระทั่งหยุด กราฟของความเร็วเขียนได้เป็นรูปของสามเหลี่ยมที่ประกอบด้วยส่วนที่ลาดขึ้น และ ส่วนที่ลาดลง หรือ อาจเป็นรูปแทรปซอยด์ (Trapezoid) ซึ่งแบ่งเป็น 3 ส่วนคือ ส่วนเพิ่มความเร็วขึ้น (Ramp up) ส่วนความเร็วคงที่ในช่วงเวลาหนึ่ง และ ส่วนความเร็วลดลง

รูปที่ 2.18 คือ บล็อกไดอะแกรมของระบบควบคุมตำแหน่งนอกจากเครื่องวัดความเร็วในลูปป้อนกลับความเร็วแล้ว ระบบกลไกแบบเซอร์โวนี้จะมีลูปป้อนกลับการกำหนดตำแหน่ง

ตัวเซนเซอร์ในการตรวจจับตำแหน่ง และ ความเร็ว จะรู้ว่าเมื่อใดที่เพลาของมอเตอร์เซอร์โวอยู่ในตำแหน่งมุมที่ต้องการโดยการนับสัญญาณพัลส์ และ เปรียบเทียบพัลส์กับสัญญาณอินพุท ก่อนที่จะหยุดเพลาเมื่อนับได้เท่ากัน ซึ่งก็คือ บล็อกที่ชื่อตัวอินทิเกรท (Integrator) ในรูปที่ 5 โดยทั่วไป จะเป็นวงจรอิเลคทรอนิคส์ควบคุมตำแหน่งของเพลา ตัวเซนเซอร์ความเร็วในลูปป้อนกลับของระบบควบคุมตำแหน่ง ช่วยทำให้ระบบเกิดเสถียรภาพขึ้น



รูปที่ 2.18 ระบบควบคุมตำแหน่ง ประกอบด้วยลูปการควบคุมตำแหน่งป้อนกลับ และ ลูปควบคุมความเร็วป้อนกลับ

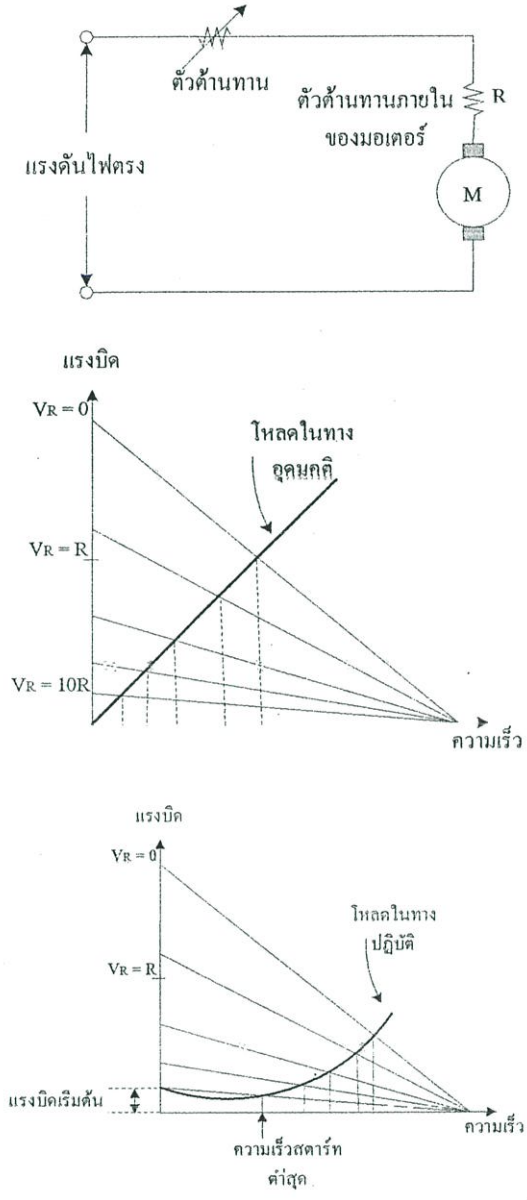
ในระบบควบคุมกำลังบิด (Torque control system) กำลังบิดของมอเตอร์เซอร์โวจะถูกรักษาให้มีค่าคงที่ เนื่องจากแรงบิดของมอเตอร์แปรตามกระแสของมอเตอร์ ดังนั้น กระแสที่ป้อนให้กับมอเตอร์ เพื่อรักษาค่าแรงบิดให้คงที่เอาไว้ วิธีนี้สามารถทำได้ด้วยวงจรที่ทำการเปรียบเทียบกระแสเอาท์พุทของมอเตอร์กับกระแสอินพุทของมอเตอร์ และ ขยายผลต่างเพื่อใช้เป็นวงจรควบคุมแรงบิดป้อนกลับ (Torque control feedback circuit)

ระบบควบคุมการเคลื่อนที่แบบเพิ่ม (Incremental motion control system) ทำหน้าที่สับเปลี่ยนโหมดควบคุมจากโหมดหนึ่งไปเป็นอีกโหมดหนึ่ง เพื่อให้เกิดสมรรถนะการใช้งานที่ต้องการ ตัวอย่างเช่น การควบคุมความเร็วและตำแหน่ง ทำหน้าที่ควบคุมให้ได้ความเร็วที่ต้องการ แต่ก็สามารถปรับเปลี่ยนเป็นการควบคุมตำแหน่ง เพื่อหยุดเพลาให้ได้ถูกต้องแม่นยำขึ้น

2.2.8.4 การควบคุมความเร็วของมอเตอร์ขั้นพื้นฐาน

1. การควบคุมด้วยตัวต้านทานที่ปรับค่าได้

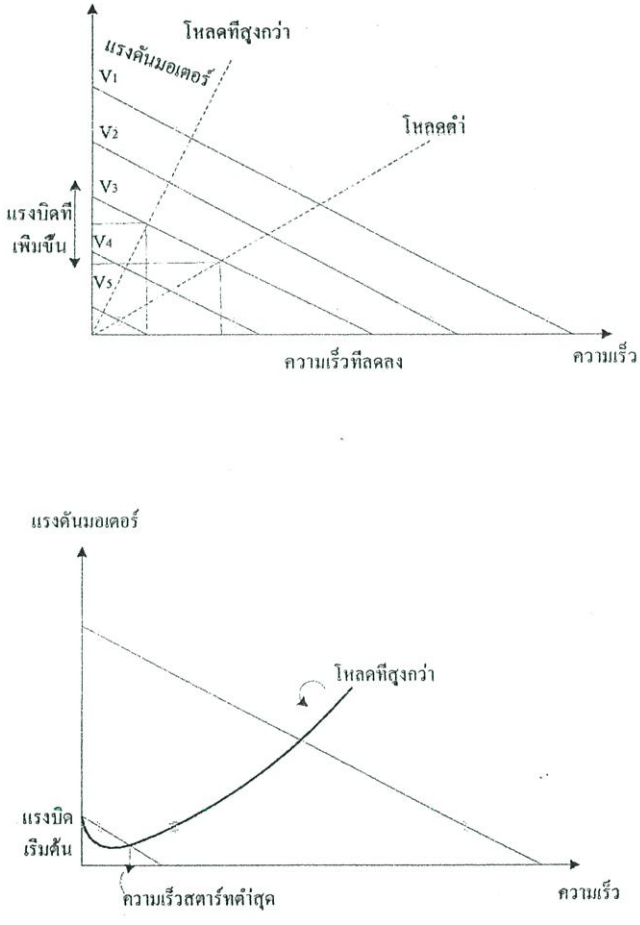
เป็นรูปแบบพื้นฐานที่สุดของการควบคุมมอเตอร์คือ ใช้ตัวต้านทานปรับค่าได้อนุกรมกับมอเตอร์ โดยตัวต้านทานที่ปรับค่าได้จะเป็นตัวกำหนดความเร็วในการหมุนของมอเตอร์ การบังคับแบบนี้ไม่มีประสิทธิภาพเพราะกำลังไฟสูญเสียไปในตัวความต้านทาน มักนิยมใช้กับมอเตอร์ตัวเล็กๆ การบังคับแบบนี้ให้คุณสมบัติการสตาร์ทดี (ให้แรงบิดสูงที่ความเร็วต่ำ) แต่จะให้ความเร็วสูงมากเมื่อมอเตอร์อยู่ในภาวะที่มีโหลดน้อยๆ ดังนั้นการบังคับแบบนี้มีประโยชน์เฉพาะภาวะที่แรงต้านคงที่ เช่น การบังคับความเร็วของเครื่องจักรเย็บผ้า เป็นต้น



รูปที่ 2.19 วงจรควบคุมความเร็วของมอเตอร์กระแสตรงแบบใช้ตัวต้านทานอนุกรมและกราฟแสดงคุณสมบัติ

2. การควบคุมด้วยวิธีเปลี่ยนค่าแรงดัน

วิธีการนี้ดีกว่าวิธีการแรกแต่จะซับซ้อนกว่าต้องใช้อุปกรณ์อิเล็กทรอนิคส์ที่ทีอัตราขยายกำลังสูง และ มอเตอร์จะถูกป้อนด้วยแรงดันที่เปลี่ยนแปลงค่าได้ จากแหล่งจ่ายที่มีอิมพีแดนซ์ต่ำ ข้อดีของการควบคุมวิธีนี้คือ ถ้าความเร็วลดลงจากผลของแรงบิด แรงดันที่ป้อนให้กับมอเตอร์จะเพิ่มขึ้นเพื่อรักษาระดับความเร็ว ส่วนข้อเสียจากการควบคุมวิธีนี้คือ เมื่อมอเตอร์มีความเร็วต่ำแรงดันที่ป้อนให้กับมอเตอร์จะมีค่าต่ำเช่นกัน



รูปที่ 2.20 การควบคุมความเร็วโดยเปลี่ยนค่าแรงดัน

3. การควบคุมด้วยตัวต้านทานที่ปรับค่าได้

การควบคุมแบบนี้สามารถขับดีซีมอเตอร์ได้ความเร็ว 10 : 1 และให้การเร็คกูเลทที่ดีกว่ากระแสถูกปล่อยให้ฟิลด์คงที่ ผลของคุณสมบัติ ความเร็วและแรงบิดได้รับการปรับปรุงดีขึ้นกว่าการบังคับด้วยความต้านทานที่ปรับค่าได้ และให้การเร็คกูเลทความเร็วคงที่ได้ดีขึ้นตลอดช่วงความเร็วที่กว้างกว่า

4. การควบคุมแบบ PWM (Pulse Width Modulation)

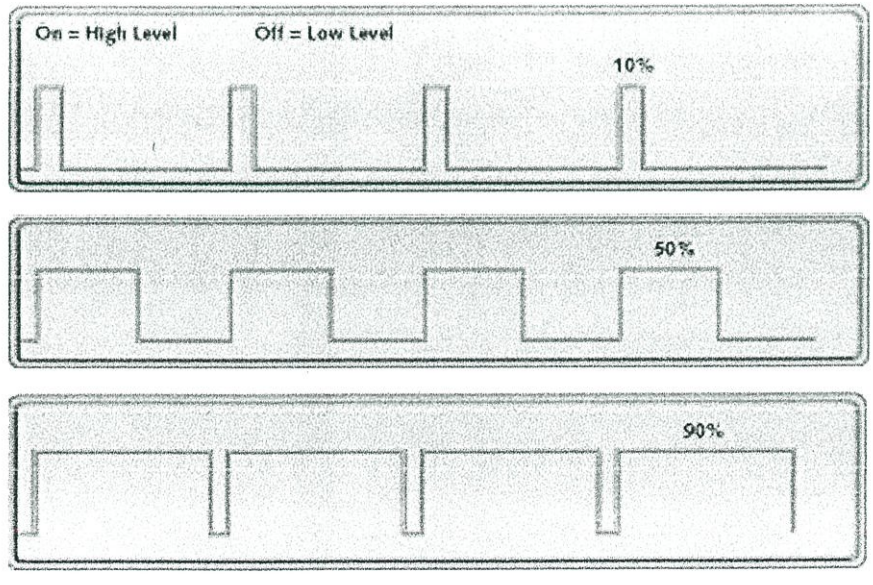
Pulse width modulation (PWM) คือ เทคนิคสำหรับควบคุมวงจรทางด้านฮาร์ดแวร์โดยใช้สัญญาณเอาท์พุตแบบดิจิตอลของไมโครโปรเซสเซอร์ควบคุม

การทำงานของสัญญาณ PWM

รูปที่ 2.21 แสดงสัญญาณ PWM ที่แตกต่างกัน 3 สัญญาณ

- โดย 10a แสดงสัญญาณ PWM ที่ 10% duty cycle คือ สัญญาณในการออนจะเป็น 10% ของคาบสัญญาณ และ จะออฟเป็น 90% ของคาบสัญญาณ
- โดย 10b แสดงสัญญาณ PWM ที่ 50% duty cycle คือ สัญญาณในการออนจะเป็น 10% ของคาบสัญญาณ และ จะออฟเป็น 50% ของคาบสัญญาณ
- โดย 10c แสดงสัญญาณ PWM ที่ 90% duty cycle คือ สัญญาณในการออนจะเป็น 10% ของคาบสัญญาณ และ จะออฟเป็น 10% ของคาบสัญญาณ

เช่น ถ้า Power Supply มี 9V และ duty cycle เป็น 10% จะได้เอาท์พุต 0.9V



รูปที่ 2.21 แสดงสัญญาณ PWM ซึ่งแสดงค่า duty cycles ที่ต่างๆกัน

ทำไมถึงใช้ PWM ในการควบคุมความเร็วมอเตอร์

มีหลายเหตุผลว่าทำไม PWM ถึงถูกเลือกใช้ในการควบคุมความเร็วของมอเตอร์ เช่น :

- PWM ง่ายในการอินเตอร์เฟสกับไมโครคอนโทรลเลอร์ และ ใช้เพียงแค่เอาท์พุตสัญญาณเดียวในการควบคุมความเร็ว
- PWM มีประสิทธิภาพ คือ Power Supply จะจ่ายกำลังได้เต็มที่ทั้ง ON และ OFF(FULL ON and FULL OFF)
- PWM ทำให้ได้ค่า ทอร์ค และ ความเร็วสูงสุดของมอเตอร์ เป็นเพราะ Power Supply จะจ่ายกำลังได้เต็มที่ทั้ง ON และ OFF(FULL ON and FULL OFF)

ซึ่งในโครงงานนี้ที่ใช้ทรานซิสเตอร์เป็นตัวตัดต่อวงจร เราสามารถจะควบคุมจังหวะในการจ่ายกระแสได้ การ on และ off ในสัดส่วนต่างๆกัน ด้วยความถี่ที่เหมาะสมก็จะทำให้ motor หมุนที่ความเร็วต่างๆ กันตามความต้องการได้ ถ้าความถี่ต่ำไป motor ก็จะหมุนแบบกระตุกๆ ไม่เรียบ และ อาจจะได้ยินเสียง จากการสั่นของ ขดลวดทองแดง ถ้าความถี่สูงกว่า 20 kHz หูเราก็จะไม่ได้ยินเสียงขดลวดสั่นอีกต่อไป และ ที่ความถี่สูงขึ้นไป มากๆ ก็จะเกิดการสูญเสีย พลังงาน ในวงจรมากเกินความจำเป็น

2.2.9 ราสเบอรี่พาย(Raspberry Pi)

2.2.9.1 การกำเนิด ราสเบอรี่พาย(Raspberry Pi)

Raspberry ถูกวางตลาดครั้งแรกในเดือนกุมภาพันธ์ ค.ศ.2012 โดยกำเนิดจากนักออกแบบคอมพิวเตอร์ชาวสหราชอาณาจักร Eben Upton โดยมีแนวความคิดที่จะออกแบบและสร้าง Raspberry คือในขณะการทำงานของเขากับห้องปฏิบัติการคอมพิวเตอร์ของ Cambridge University เกิดความไม่พอใจในการทำงานเพราะนักศึกษามีความสามารถในการเขียนโปรแกรมลดลงเมื่อเทียบกับนักศึกษาที่ผ่านมา อีกทั้งผู้คนทั่วไปก็คิดว่าคอมพิวเตอร์ก็คือ เครื่องมือสืบค้นข้อมูลทางอินเทอร์เน็ต พิมพ์เอกสาร หรือ คำนวณตาราง Excel เท่านั้น ไม่ได้รู้ว่าคอมพิวเตอร์สามารถทำงานได้อีกหลากหลาย

เมื่อ Upton คิดอย่างนั้น จึงตั้งใจว่าจะต้องสร้างคอมพิวเตอร์ขนาดเล็ก และราคาถูกสุด (ในตอนแรกได้ตั้งใจให้อยู่ที่ 25$) และยังต้องทำงานได้หลายอย่างแบบที่คอมพิวเตอร์สมัยนี้ทำได้อีก เขาคาดหวังว่าจะทำได้คนกลับมาเข้าใจการทำงานของคอมพิวเตอร์มากขึ้น สามารถเข้าถึงได้มากขึ้นเพราะมีราคาที่ถูกลง และฝึกเขียนโปรแกรมคอมพิวเตอร์กันมากขึ้น เขาได้คิดและลงมือทำอยู่ 6 ปี ราสเบอรี่พาย(Raspberry Pi) Model A จึงได้เปิดตัวขึ้นและได้ให้เปิดจองกันในเดือนกุมภาพันธ์ ค.ศ.2012 และแล้วภายใน 3 เดือนก็มีคนสั่งจองกันราว

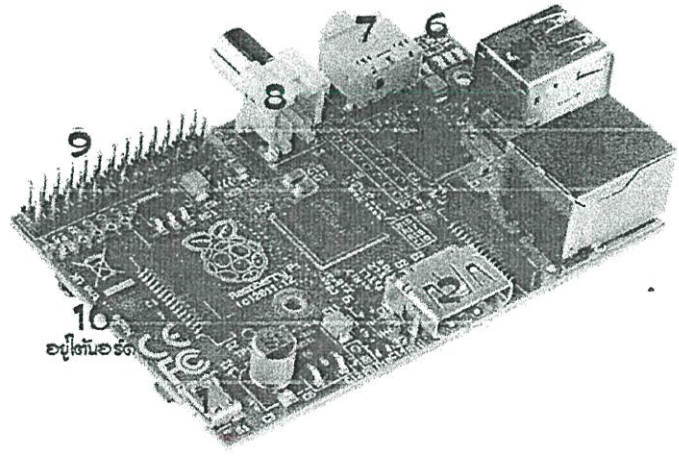
5 แสนเครื่อง และต้องรอนานกันเป็นเดือนกว่าจะได้ของมา และเมื่อผ่านไปเพียงปีเดียวก็สามารถขายได้เกิน 1 ล้านเครื่องทั่วโลก และในเดือนตุลาคม ค.ศ.2012 ได้ปล่อย ราสเบอรี่พาย(Raspberry Pi) Model B ออกมาขาย โดยปรับปรุง RAM เป็น 512 MB และเพิ่มพอร์ต LAN เข้าไปด้วย และยังมีการพัฒนาเพิ่มขึ้นอีก

เนื่องจากต้องการทำให้ราคาถูก นอกจาก Hardware ที่ใช้อย่างคุ้มค่าแล้ว Software ก็จำเป็นต้องใช้แบบที่เป็น Open source นั้นก็คือ Linux ดังนั้นระบบปฏิบัติการต่างๆ ของ ราสเบอรี่พาย(Raspberry Pi) จึงมีพื้นฐานมาจาก Linux แล้วพัฒนาต่อให้มี Graphic User interface ที่คล้ายกับ Windows

ความแตกต่างของ ราสเบอรี่พาย(Raspberry Pi) เมื่อเทียบกับ Arduino นั้น ถ้าเปรียบเทียบกันในเรื่องของการใช้งานแล้ว ราสเบอรี่พาย(Raspberry Pi) จะคล้ายกับเครื่องคอมพิวเตอร์ที่มี Windows หรือเครื่อง Mac นั้นคือสามารถเปิดไปยัง website ต่างๆ สามารถลงโปรแกรมที่เป็น Open Office เครื่องคิดเลข ดูหนัง ฟังเพลง ฝึกเขียนโปรแกรม และอื่นๆอีกเยอะ จึงสามาถกล่าวได้ว่า ราสเบอรี่พาย(Raspberry Pi) สามาถทำงานได้เช่นเดียวกับ Windows เลยทีเดียว ซึ่งจะต่างจาก Arduino ที่เป็นเพียงแค่ Microcontroller เท่านั้น ไม่มี Operating system เป็นของตัวเอง เขียนโปรแกรมแล้วสั่งให้ทำงานเป็นเรื่องๆไปเท่านั้น

ถ้าต้องการให้ ราสเบอรี่พาย(Raspberry Pi) มาทำงานแบบ Arduino ก็สามารถทำได้ โดยเราสามารถเขียนโปรแกรมเพื่อควบคุมพอร์ด GPIO โดยใช้ภาษา Python หรือ ภาษา C ก็ได้ และยังมีความเร็วสูงกว่าอีกด้วย งานที่ ราสเบอรี่พาย(Raspberry Pi) จะถูกเลือกใช้เป็นส่วนมากนั้น จะที่เกี่ยวข้องกับกล้องถ่ายภาพนิ่ง วีดีโอ ประมวลผลภาพ เพราะงานในด้านนี้ Arduino แบบทั่วไปจะประมวลผลภาพได้ไม่ทัน

2.2.9.2 จุดเชื่อมต่ออุปกรณ์และความสามารถของแต่ละตำแหน่งบนบอร์ด ราสเบอรี่พาย(Raspberry Pi)



รูปที่ 2.22 ส่วนประกอบของ ราสเบอรี่พาย(Raspberry Pi)

หมายเลข 1 คือ ช่องเสียบไฟหรือช่องเสียบ adapter เวลาต้องการจะจ่ายไฟให้กับบอร์ ราสเบอรี่พาย(Raspberry Pi) เราจำเป็นต้องเสียบไฟจากแหล่งจ่ายเข้าตรงนี้และจากแหล่งจ่ายต้องมีแรงดันไฟ ไม่เกิน 3.3 v. ดังนั้นจำเป็นต้องหา adapter ที่มีความเหมาะสมต่อการใช้งานด้วย

หมายเลข 2 คือช่องเสียบสาย HDMI (High Definition Multimedia Interface) ซึ่งมีไว้เพื่อเสียบสาย HDMI จากตัวบอร์ด ราสเบอรี่พาย(Raspberry Pi) เพื่อเข้าสู่จอแสดงผลที่รองรับการเชื่อมต่อภาพด้วย HDMI เช่น จอคอมพิวเตอร์รุ่นใหม่ หรือ Smart TV ในปัจจุบันก็รองรับการเชื่อมต่อเช่นกัน ซึ่งสามารถหาอุปกรณ์เหล่านี้และสาย HDMI ได้ง่ายตามร้านขายอุปกรณ์อิเลคทรอนิกส์ทั่วไป

หมายเลข 3 คือ CSI Connector ไว้สำหรับต่อกับโมดูลกล้องที่ออกแบบมาเพื่อบอร์ด ราสเบอรี่พาย (Raspberry Pi) โดยเฉพาะ โดยลักษณะการต่อจะเป็นสายแพรที่มากับตัวกล้องต่อเข้าไปกับพอร์ทนี้และสามารถตั้งค่าการใช้งานกล้องได้อย่างสมบูรณ์แบบกว่าการนำกล้องแบบอื่นมาต่อเพราะกล้องที่นำมาต่อกับพอร์ตนี้ถูกออกแบบมาเพื่อ ราสเบอรี่พาย(Raspberry Pi) โดยเฉพาะนั่นเอง

หมายเลข 4 คือ Port Ethernet มีไว้สำหรับเชื่อมต่อบอร์ด ราสเบอรี่พาย(Raspberry Pi) เข้ากับระบบเครือข่าย คอมพิวเตอร์โดยผ่านสายแลน(หัว RJ 45) โดยการเชื่อมต่อนี้จะนำบอร์ดเข้าสู่ระบบเครือข่ายท้องถิ่น (LAN) และสามารถออก Internet ได้ด้วยทำให้เราสามารถอัพเดทซอฟท์แวร์ได้อิสระตามต้องการมากขึ้น และซึ่งเป็นปัจจัยหลักในการติดตั้งซอฟท์แวร์ต่างๆอีกด้วยเพราะเกือบจะ 100 % จะติดตั้งซอฟแวร์ผ่านระบบ Internet (ยกเว้น OS) เพิ่มเติมในส่วนของระบบเครือข่าย บอร์ด ราสเบอรี่พาย(Raspberry Pi) สามารถเชื่อมต่อกับระบบเครือข่ายไร้สายหรือ wireless ได้โดยการเพิ่ม wifi dongle เสียบเข้าทาง พอร์ต USB ก็สามารถทำงานได้เหมือนกันกับการเสียบสายและทุกประการแต่มีข้อเสียคือความเร็วต่ำกว่าการเสียบสายอยู่พอสมควร

หมายเลข 5 คือพอร์ตที่จำเป็นที่มีในทุกๆอุปกรณ์ในปัจจุบันนั่นก็คือพอร์ต USB นั่นเอง เช่นเดียวกันกับพอร์ต USB ของ ราสเบอรี่พาย(Raspberry Pi) นั้นก็สามารถเชื่อมต่ออุปกรณ์ต่างๆ ที่เป็น USB ได้หลากหลายมาก เช่น กล้องเว็บแคม, เม้าส์, คีย์บอร์ด,หรือแม้แต่ wifi dongle ก็สามารถเชื่อมต่อและใช้งานได้เช่นกัน และยังมีอุปกรณ์อีกมากมายที่สามารถใช้งานเข้ากันได้

หมายเลข 6 คือสถานะไฟ LED เพื่อแสดงว่าสถานะของบอร์ด เช่น แสดงว่าเราเสียบไฟเข้าหรือไม่, สาย Lan เชื่อมต่อหรือไม่ และมีไฟกระพริบที่บ่งบอกว่าบอร์ดมีการส่งข้อมูลกันกับเครือข่ายนั่นเอง

หมายเลข 7 คือ ช่องเสียบแจ็คเสียงเอ้าพุท หรือ ก็คือช่องเสียบลำโพงนั่นเองแน่นอนว่า ราสเบอรี่พาย(Raspberry Pi) สามารถมีเสียงและเปิดเพลงหรือ media ที่มีเสียงได้ในซอฟแวร์ของบอร์ด ราสเบอรี่พาย (Raspberry Pi) นั่นเอง

หมายเลข 8 คือ ช่องต่อสัญญาณภาพเป็นชนิด แจ็ค RCA แบบธรรมดาสามารถต่อเข้ากับ ทีวีทั่วไปได้ โดยประโยชน์ของมันคือเวลาเราเริ่มต้นใช้งานสามารถต่อสัญญาณภาพออกจากตรงนี้

หมายเลข 9 คือ GPIO ส่วนที่เห็นเป็นเสาแหลมๆ สามารถต่อสายเพิ่มเข้าไปได้ เพราะ GPIO (General Purpose Input Output) ซึ่งคือพอร์ตเชื่อมต่อที่เราสามารถกำหนดให้มันเป็นอินพุต หรือ เอาท์พุต ได้โดยการเขียนโปรแกรมขึ้นเองหรือใช้ software ที่มีมาให้ดาวน์โหลดใช้ เช่น wiring pi เป็นต้น ทั้งนี้ GPIO สามารถทำงานได้หลายอย่างสามารถที่จะเชื่อมต่อเซ็นเซอร์ต่างๆ ที่เราต้องการแล้วป้อนค่าให้กับโปรแกรมตามที่เราต้องการได้เลยขึ้นอยู่กับการเขียนโปรแกรม เช่น สามารถใช้ GPIO แล้วเขียนโปแกรมให้ส่งค่า logic 1 ออกไปนั่นหมายถึงว่าให้ ขาที่ถูกกำหนดมีแรงดันไฟส่งออกไป ให้กับ Relay เพื่อทำการเปิดไฟห้องให้สว่างได้ ทั้งนี้สามารถที่จะเขียน UI ของมันออกมาเป็นหน้าเว็บเพจเพื่อการใช้งานที่สะดวกได้อีก

หมายเลข 10 (อยู่ด้านล่างบอร์ด) คือ ช่องเสียบ SD Card ซึ่งเป็นหน่วยความจำที่เก็บระบบปฏิบัติการ และ ซอฟท์แวร์ทั้งหมดที่เราติดตั้งลงไปไว้

หมายเลข 11 คือ DSI Display ทำหน้าที่ไว้ต่อกับจอชนิดที่เป็นสายแพร

2.2.9.3 ข้อดีและข้อเสียของ ราสเบอรี่พาย(Raspberry Pi)

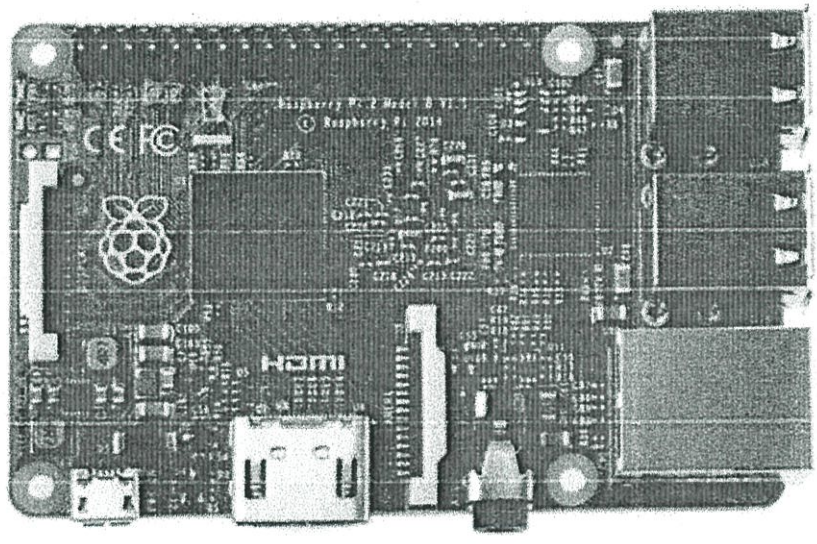
ข้อดีคือ

- มีความเล็กกระทัดรัด เคลื่อนย้ายสะดวก
- ประหยัดพลังงาน
- ใช้ซอฟท์แวร์ที่เป็น open source ทั้งหมด
- เป็นคอมพิวเตอร์ขนาดจิ๋วที่เหมาะกับการเริ่มต้นเขียนโปรแกรม
- สามารถทำเป็น Server เพื่อรองรับการร้องขอจากเครื่อง Client ได้
- มี GPIO สามารถสามารถเชื่อมต่อกัน เซ็นเซอร์ต่างๆหรือบอร์คอนโทรลอื่นๆ ได้อย่างอิสระ
- มีทั้ง หน้าจอ GUI และ Command Line (Terminal)
- สามารถ SSH (Secure Shell) เข้าไปสั่งงานได้โดยใช้ โปรแกรม Putty เป็นต้น
- ราคาประหยัดและสามารถหาอุปกรณ์เสริมอื่นๆได้ง่าย

ข้อเสีย

- มีความเร็วค่อนข้างต่ำ (แต่เมื่อใช้งานในโหมด command line (Terminal) จะเร็ว)
- รากฐานเป็น Linux OS ซึ่งอาจจะยากแก่ผู้ที่ไม่คุ้นเคย
- การ config บางอย่างจำเป็นต้องใช้คำสั่ง command line (ซึ่งจะขึ้นอยู่กับความถนัดแต่ละคน)

2.2.9.4 ราสเบอรี่พาย(Raspberry Pi) Model B+



รูปที่ 2.23 ราสเบอรี่พาย(Raspberry Pi) 2 Model B+

ในงานวิจัยนี้ได้ใช้ ราสเบอรี่พาย(Raspberry Pi) Model B+ ในการประมวลผลข้อมูล เชื่อมต่อกับกล้องเว็บแคมทั้ง 4 ตัว รวมไปถึง SD card, หน้าจอ, เมาส์และคีย์บอร์ดที่จำเป็นต้องใช้ โดยมีคุณสมบัติหลักๆ ดังนี้

- แหล่งจ่ายไฟ Dual step-down (buck) ที่ 3.3V and 1.8V
- 5V supply has polarity protection, 2A fuse and hot-swap protection (สามารถเสียบและถอด USB โดยไม่ต้อง reset บอร์ด)
- USB ports มีจำนวน 4 ช่อง

- 40 GPIO pins. 26 pins แรกจะเหมือนกับรุ่นเดิม, 9 GPIO pins เพิ่มเติม และ 2 EEPROM Plate identification pins

- Composite (NTSC/PAL) video now integrated into 4-pole 3.5mm 'headphone' jack

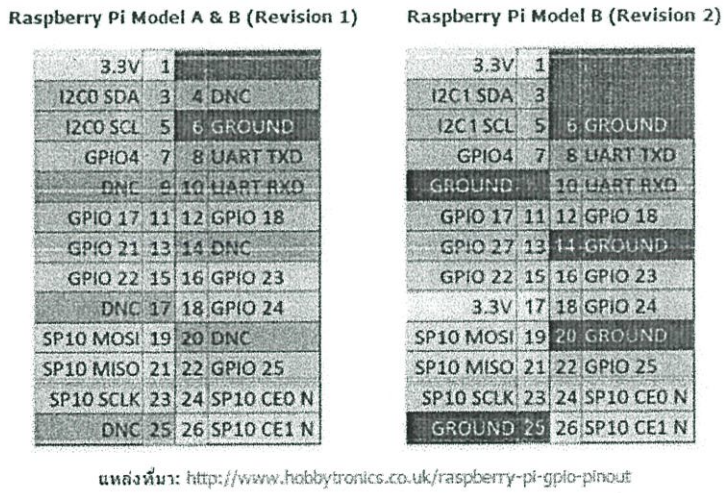
- MicroSD card socket

- ขนาดของ PCB คือ 56mm x 85mm x 1.4mm.

- ความสูงเมื่อรวมช่อง USB คือ 17mm.

- น้ำหนัก 42 กรัม

1.พอร์ต GPIO ซึ่งในโมเดล A และ B (Revision 1) ทุก Pin จะเหมือนกัน แต่โมเดล B (Revision 2) จะแตกต่างกัน รายละเอียดดังรูป



Raspberry Pi Model A & B (Revision 1)

3.3V	1	2	
I2C0 SDA	3	4	DNC
I2C0 SCL	5	6	GROUND
GPIO4	7	8	UART TXD
DNC	9	10	UART RXD
GPIO 17	11	12	GPIO 18
GPIO 21	13	14	DNC
GPIO 22	15	16	GPIO 23
DNC	17	18	GPIO 24
SP10 MOSI	19	20	DNC
SP10 MISO	21	22	GPIO 25
SP10 SCLK	23	24	SP10 CE0 N
DNC	25	26	SP10 CE1 N

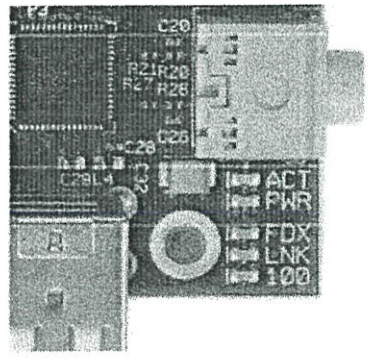
Raspberry Pi Model B (Revision 2)

3.3V	1	2	
I2C1 SDA	3		
I2C1 SCL	5	6	GROUND
GPIO4	7	8	UART TXD
GROUND		10	UART RXD
GPIO 17	11	12	GPIO 18
GPIO 27	13	14	GROUND
GPIO 22	15	16	GPIO 23
3.3V	17	18	GPIO 24
SP10 MOSI	19	20	GROUND
SP10 MISO	21	22	GPIO 25
SP10 SCLK	23	24	SP10 CE0 N
GROUND	25	26	SP10 CE1 N

แหล่งที่มา: http://www.hobbytronics.co.uk/raspberry-pi-gpio-pinout

รูปที่ 2.24 port GPIO ของ ราสเบอรี่พาย(Raspberry Pi)

-LED แสดงสถานะของบอร์ด อยู่ภายในบริเวณกรอบสีแดง ดังภาพ



รูปที่ 2.25 LED แสดงสถานะ ของ ราสเบอรี่พาย(Raspberry Pi)

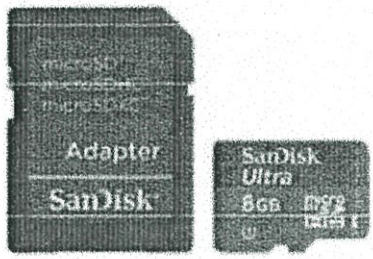
- ACT คือ ไฟสถานะ SD Card Access (สีเขียว)
- PWR คือ ไฟสถานะ 3.3V Power (สีแดง)
- FDX คือ ไฟสถานะ Full Duplex LAN Model B (สีเขียว)
- LNK คือ ไฟสถานะ Link/Activity LAN Model B (สีเขียว)
- 100 คือ ไฟสถานะ 10/100Mbps LAN Model B (สีเหลือง)

2.2.9.5 ติดตั้งระบบปฏิบัติการ

เริ่มต้นการติดตั้งระบบปฏิบัติการ

ก่อนเริ่มต้นการใช้งานบอร์ด ราสเบอรี่พาย(Raspberry Pi) จำเป็นที่จะต้องติดตั้งระบบปฏิบัติการให้กับบอร์ดก่อนเนื่องจากบอร์ดไม่มีหน่วยความจำแบบแฟลชเมมโมรี่มาบนบอร์ดด้วย ดังนั้น จำเป็นที่จะต้องเตรียมอุปกรณ์ต่างๆ ให้พร้อมเพื่อให้สามารถใช้งานบอร์ดได้ ซึ่งมีรายละเอียดอุปกรณ์ดังนี้

1. บอร์ด ราสเบอรี่พาย(Raspberry Pi)
2. SD Card สำหรับติดตั้งระบบปฏิบัติการ Linux ต้องมีความจุมากกว่า 2GB ขึ้นไป แต่แนะนำให้ใช้ ขนาด 4GB หรือมากกว่า สำหรับคู่มือฉบับนี้จะใช้ขนาด 8GB ควรเลือกใช้การ์ดที่มีความเร็วสูงอย่าง Class 10 เพื่อประสิทธิภาพการทำงานของระบบโดยรวม



รูปที่ 2.26 SD card

3. เมาส์และคีย์บอร์ดแบบ USB

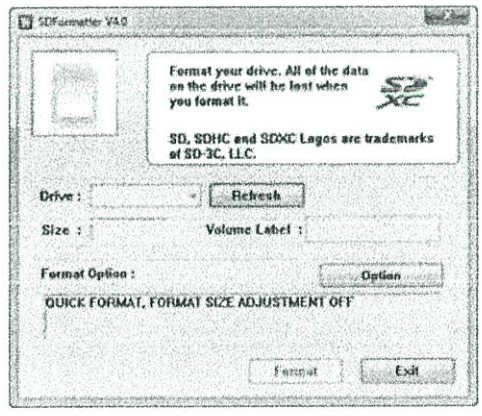
4. สาย Micro USB เพื่อจ่ายไฟเลี้ยงวงจร สามารถเลือกใช้แหล่งจ่ายไฟจากพอร์ต USB ของเครื่องคอมพิวเตอร์ได้

5. สาย HDMI เพื่อเชื่อมต่อกับจอแสดงผล หากเลือกใช้จอ Monitor ที่ไม่มีพอร์ต HDMI รองรับต้องใช้ตัวแปลง HDMI to VGA ด้วย หรือเชื่อมต่อสายวีดีโอ RCA ก็ได้เช่นเดียวกัน (เลือกอย่างใดอย่างหนึ่ง)

เตรียม Software สำหรับติดตั้งระบบปฏิบัติการ Linux ลงบนบอร์ด ราสเบอรี่พาย(Raspberry Pi)

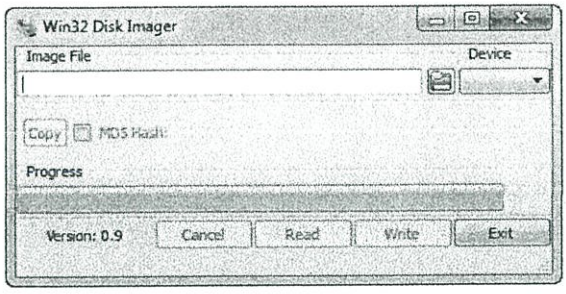
คู่มือฉบับนี้จะจัดเตรียมซอฟต์แวร์ที่รองรับระบบปฏิบัติการ Windows 7 เป็นหลัก และต้องติดตั้งลงบนเครื่องคอมพิวเตอร์ดังนี้

1. โปรแกรม SD Formatter 4.0 ใช้สำหรับ Format Disk สามารถดาวน์โหลดได้จากลิงค์



รูปที่ 2.27 โปรแกรม SD Formatter Version 4.0

2.โปรแกรม Win32 Disk Imager ใช้สำหรับเขียนไฟล์ระบบปฏิบัติการที่เป็นไฟล์ Image (*.img) ลงบน SD Card สามารถดาวน์โหลดได้จากลิงค์



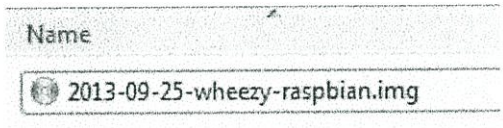
รูปที่ 2.28 โปรแกรม Win32 Disk Imager

3.ไฟล์ระบบปฏิบัติการ **คู่มือนี้ติดตั้งระบบปฏิบัติการ Raspbian** เป็นระบบปฏิบัติการ Debian Wheezy ที่ถูกปรับแต่งให้ใช้สำหรับบอร์ด ราสเบอรี่พาย(Raspberry Pi) โดยเฉพาะ เป็น Linux ที่ให้ใช้งานได้ฟรี สามารถดาวน์โหลดได้จากลิงค์

ขั้นตอนการติดตั้งระบบปฏิบัติการ Raspbian ให้กับบอร์ด ราสเบอรี่พาย(Raspberry Pi)

1. หากมีข้อมูลอยู่ใน SD Card ให้ทำการ Format ด้วยโปรแกรม SD Formatter 4.0 หรือโปรแกรมอื่นๆ ก็ได้ ถ้าหาก Format แล้วให้ข้ามขั้นตอนนี้ได้เลย

2. เมื่อดาวน์โหลดไฟล์ระบบปฏิบัติการ Raspbian มาแล้วจะได้เป็นไฟล์ Zip ให้แตกไฟล์จะได้เป็นไฟล์ Image (*img) มาแสดงดังรูป 2.29



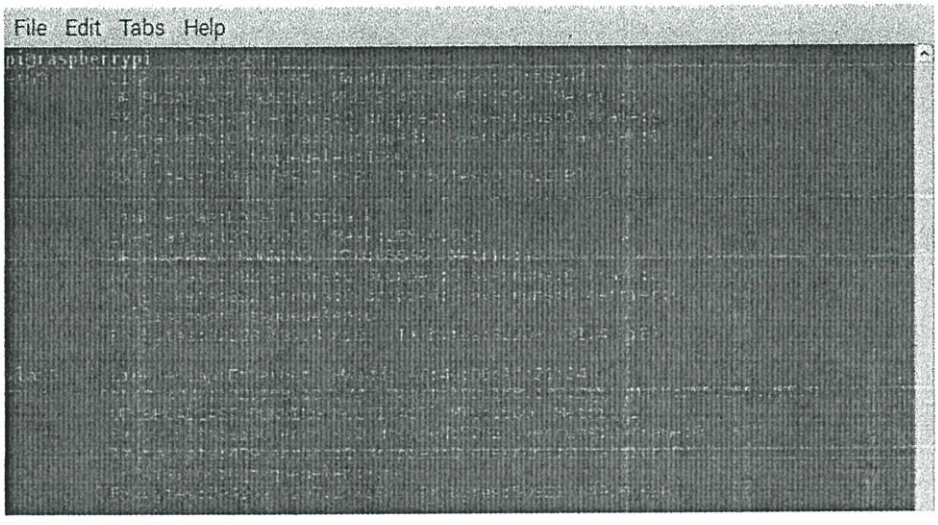
รูปที่ 2.29 การแตกไฟล์ Zip ของไฟล์ระบบปฏิบัติการ Raspbian

2.9.6 IP Address Lan บน ราสเบอรี่พาย(Raspberry Pi)

IP Address จำเป็นในการเชื่อมต่อกับคอมพิวเตอร์เพื่อการถ่ายโอนข้อมูลจากคอมพิวเตอร์มายัง ราสเบอรี่พาย(Raspberry Pi) หรือ จาก ราสเบอรี่พาย(Raspberry Pi) ไปยังคอมพิวเตอร์ โดยการกำหนดค่าเกี่ยวกับ network นั้นใช้

ifconfig

โดยจะแสดงข้อมูลเกี่ยวกับการเชื่อมต่อเครือข่ายในขณะนั้น ซึ่งแสดงดังรูปที่ 2.17



รูปที่ 2.30 การแสดงผลลักษณะการเชื่อมต่อเครือข่าย

โดยที่ eth0, lo, wlan0 เป็นชื่อของลักษณะการเชื่อมต่อเครือข่ายที่ใช้งานภายในระบบ

- eth0 มักจะแสดงถึงการเชื่อมต่อ NIC กับเครือข่ายผ่านสายเคเบิล ซึ่งรวมถึงสาย Lan ด้วย

- lo เป็นการเชื่อมต่อพิเศษที่ระบบสามารถสื่อสารกับตัวเองได้

- wlan0 เป็นชื่อการเชื่อมต่อเครือข่ายไร้สายอันแรกในระบบ

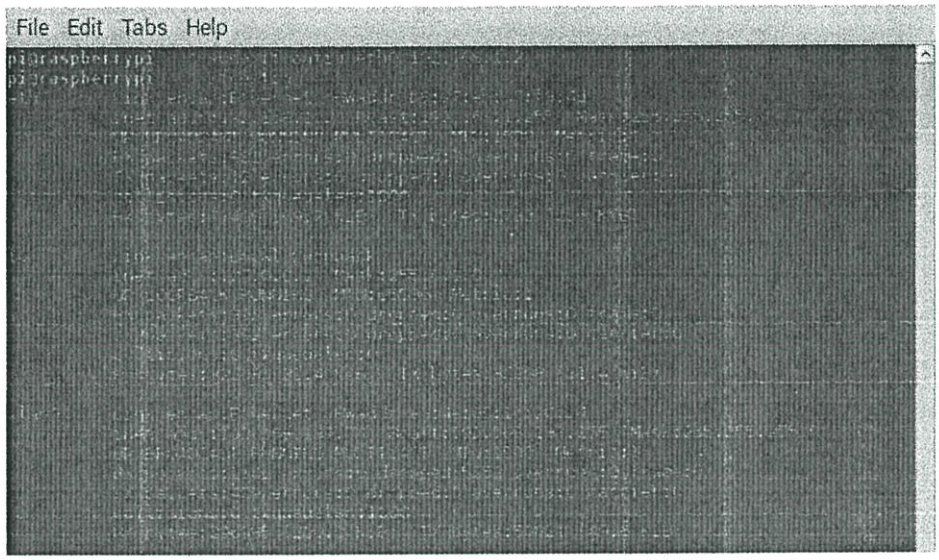
ในการตั้งค่า IP Address ของ ราสเบอรี่พาย(Raspberry Pi) เพื่อการเชื่อมต่อแลกเปลี่ยนข้อมูลกับคอมพิวเตอร์ผ่านสาย Lan นั้นจึงต้องตั้งค่าในส่วนของ eth0 โดยพิมพ์คำสั่งบน command line ดังนี้

sudo ifconfig eth0 192.168.1.2

เมื่อพิมพ์คำสั่งนี้ลงไปใน command line แล้ว ค่า IP Address จะถูกเปลี่ยนไป ซึ่งแสดงอยู่ในรูปที่ 2.18 ในผลลัพธ์บรรทัดที่ 2 ของ eth0 จะพบดังนี้

inet addr:192.168.1.2

แสดงให้เห็นว่า ค่า IP จะถูกตั้งค่าเป็น 192.168.1.2 แล้วเพื่อสามารถเชื่อมต่อกับคอมพิวเตอร์ผ่านสาย Lan ได้ สร้างความสะดวกในการสื่อสารกันระหว่าง ราสเบอรี่พาย(Raspberry Pi) และ คอมพิวเตอร์



รูปที่ 2.31 แสดงการตั้งค่า IP Address บน Raspberry

2.2.10 การเขียนโปรแกรมคอมพิวเตอร์

2.10.1 Computer Programing

การเขียนโปรแกรมคอมพิวเตอร์ (Computer programming) หรือเรียกให้สั้นลงว่า การเขียนโปรแกรม (Programming) หรือ การเขียนโค้ด (Coding) เป็นขั้นตอนการเขียน ทดสอบ และดูแลซอร์สโค้ดของโปรแกรมคอมพิวเตอร์ ซึ่งซอร์สโค้ดนั้นจะเขียนด้วยภาษาโปรแกรม ขั้นตอนการเขียนโปรแกรมต้องการความรู้ในหลายด้านด้วยกัน เกี่ยวกับโปรแกรมที่ต้องการจะเขียน และขั้นตอนวิธีที่จะใช้ ซึ่งในวิศวกรรมซอฟต์แวร์นั้น การเขียนโปรแกรมถือเป็นเพียงขั้นหนึ่งในวงจรชีวิตของการพัฒนาซอฟต์แวร์

การเขียนโปรแกรมจะได้มาซึ่งซอร์สโค้ดของโปรแกรมนั้นๆ โดยปกติแล้วจะอยู่ในรูปแบบของ ข้อความธรรมดา ซึ่งไม่สามารถนำไปใช้งานได้ จะต้องผ่านการคอมไพล์ตัวซอร์สโค้ดนั้นให้เป็นภาษาเครื่อง (Machine Language) เสียก่อนจึงจะได้เป็นโปรแกรมที่พร้อมใช้งาน

ภาษาโปรแกรมแต่ละภาษาจะมีลักษณะหรือรูปแบบการเขียนที่แตกต่างกัน การเลือกภาษาโปรแกรมหรือภาษาคอมพิวเตอร์เพื่อนำมาเขียนโปรแกรมนั้นขึ้นอยู่กับปัจจัยหลาย ๆ อย่าง เช่น นโยบายของบริษัท, ความเหมาะสมของโปรแกรมกับลักษณะงานที่จะถูกนำไปใช้, การเข้ากันได้กับโปรแกรมอื่น ๆ, หรืออาจเป็นความถนัดของแต่ละคน ภาษาโปรแกรมที่มีแนวโน้มในการนำมาเขียนมักเป็นภาษาที่มีคนที่สามารถเขียนได้ทันที หรือหากมีความจำเป็นที่จะต้องเลือกใช้ภาษาอื่น เช่นต้องการเน้นประสิทธิภาพในการทำงานของโปรแกรม ก็อาจจำเป็นต้องหานักเขียนโปรแกรมขึ้นมาจำนวนหนึ่งซึ่งมีความรู้ความเข้าใจในภาษาโปรแกรมที่ต้องการ และต้องมีคอมไพเลอร์ที่รองรับภาษาเหล่านั้นด้วย

2.10.2 ภาษาที่ใช้เขียนโปรแกรม

2.10.2.1 ภาษาเครื่อง (Machine Languages)

ภาษาเครื่อง คือ กลุ่มของคำสั่งเครื่องที่กระทำการโดยตรงโดยหน่วยประมวลผลกลาง (CPU) ของคอมพิวเตอร์ คำสั่งเครื่องแต่ละคำสั่งจะปฏิบัติงานเฉพาะกิจงานเดียวเท่านั้น เช่น การบรรจุ, การกระโดด หรือการดำเนินการผ่านหน่วยคำนวณและตรรกะ (ALU) บนหน่วยของข้อมูลในหน่วยความจำหรือเรจิสเตอร์ ทุกๆ โปรแกรมที่กระทำการโดยหน่วยประมวลผลกลางสร้างขึ้นจากอนุกรมของคำสั่งเครื่องเช่นว่านั้น

รหัสเครื่องเชิงตัวเลข (ซึ่งไม่ใช่รหัสแอสเซมบลี) อาจพิจารณาได้ว่าเป็นตัวแทนระดับต่ำสุดของโปรแกรมคอมพิวเตอร์ที่ได้คอมไพล์และ/หรือเขียนด้วยภาษาแอสเซมบลี หรือเป็นภาษาโปรแกรมแบบดั้งเดิม

และขึ้นอยู่กับฮาร์ดแวร์ ถึงแม้ว่าเราจะสามารถเขียนโปรแกรมด้วยรหัสเครื่องเชิงตัวเลขโดยตรงก็ได้ แต่การจัดการบิตต่าง ๆ เป็นเอกเทศ และการคำนวณตำแหน่งที่อยู่กับค่าคงตัวเชิงตัวเลขด้วยมือ จะทำให้น่าเบื่อหน่ายและมีแนวโน้มที่จะเกิดความผิดพลาด ดังนั้นการเขียนรหัสเครื่องจึงไม่ค่อยกระทำกันในทุกวันนี้ เว้นแต่ในสถานการณ์ที่ต้องการทำให้เหมาะสมอย่างที่สุดหรือแก้จุดบกพร่อง

ปัจจุบันนี้โปรแกรมเกือบทั้งหมดในทางปฏิบัติเขียนขึ้นด้วยภาษาแอสเซมบลีหรือภาษาระดับสูงกว่าแล้วแปลเป็นรหัสเครื่องที่กระทำการได้โดยคอมไพเลอร์และ/หรือแอสเซมเบลอร์ กับลิงเกอร์ อย่างไรก็ดีโปรแกรมที่เขียนด้วยภาษาที่แปลด้วยอินเทอร์พรีเตอร์จะไม่ถูกแปลเป็นรหัสเครื่อง ถึงแม้ว่าอินเทอร์พรีเตอร์ (ตัวกระทำการ หรือ ตัวประมวลผล) โดยทั่วไปประกอบขึ้นจากรหัสเครื่องที่กระทำการได้โดยตรง

หน่วยประมวลผลหรือตระกูลของหน่วยประมวลผลทุก ๆ ชิ้นมีชุดของคำสั่งเครื่องที่เป็นรหัสของมันเอง คำสั่งเครื่องคือแบบรูปต่าง ๆ ของบิต ซึ่งการออกแบบเชิงกายภาพสอดคล้องกับคำสั่งงานที่แตกต่างกันของเครื่อง ดังนั้นชุดของคำสั่งเครื่องจึงใช้ได้กับประเภทของหน่วยประมวลผลที่ใช้สถาปัตยกรรมเดียวกัน การออกแบบหน่วยประมวลผลรุ่นหลังหรือรุ่นต่อยอดก็มักจะรวมคำสั่งทั้งหมดของรุ่นก่อนหน้าไว้ และอาจเพิ่มเติมคำสั่งใหม่เข้าไปอีกด้วย หน่วยประมวลผลรุ่นหลังอาจยกเลิกหรือเปลี่ยนแปลงรหัสของคำสั่งเครื่องในบางครั้ง (สาเหตุทั่วไปก็คือมันจำเป็นสำหรับจุดประสงค์ใหม่) ส่งผลกระทบต่อความเข้ากันได้ของรหัสในบางขอบข่าย แม้กระทั่งหน่วยประมวลผลที่เข้ากันได้เกือบสมบูรณ์ก็อาจแสดงพฤติกรรมต่างไปจากเดิมเล็กน้อยสำหรับบางคำสั่ง แต่ปัญหานี้พบได้น้อยมาก ระบบต่าง ๆ ก็อาจแตกต่างกันในรายละเอียดอื่น เช่น การจัดการหน่วยความจำ ระบบปฏิบัติการ หรืออุปกรณ์รอบข้าง เนื่องจากตามปกติแล้วโปรแกรมจะยึดถือปัจจัยดังกล่าว ระบบที่แตกต่างกันก็จะไม่ทำงานด้วยรหัสเครื่องที่เหมือนกัน ถึงแม้ว่าใช้หน่วยประมวลผลชนิดเดียวกันก็ตาม

รหัสของชุดของคำสั่งเครื่องอาจมีความยาวเท่ากันหมดทุกคำสั่งหรือมีความยาวแปรผันก็ได้ วิธีการจัดการแบบรูปของรหัสเครื่องขึ้นอยู่กับสถาปัตยกรรมนั้น ๆ เป็นอย่างยิ่ง และมักจะขึ้นอยู่กับชนิดของคำสั่ง คำสั่งเครื่องส่วนมากมีฟิลด์ออปโคด (opcode) หนึ่งฟิลด์หรือมากกว่าซึ่งใช้ระบุชนิดของคำสั่งพื้นฐาน (เช่นเลขคณิต ตรรกศาสตร์ การกระโดด ฯลฯ) และการดำเนินการแท้จริง (เช่นการบวก การเปรียบเทียบ) และมีฟิลด์อื่น ๆ ที่อาจใช้สำหรับระบุชนิดของตัวถูกดำเนินการ (operand) ภาวะการกำหนดตำแหน่งที่อยู่ (addressing mode) ออฟเซตของตำแหน่งที่อยู่หรือดัชนี หรือค่าแท้จริงโดยตัวมันเอง (ตัวถูกดำเนินการที่เป็นค่าคงตัวที่บรรจุอยู่ในคำสั่งเครื่องเช่นนั้นเรียกว่า ค่าใช้ทันที, immediate)

ไม่ใช่ว่าทุกเครื่องหรือทุกคำสั่งจะมีตัวถูกดำเนินการชัดแจ้ง เครื่องที่ใช้ตัวสะสม (accumulator machine) มีตัวถูกดำเนินการข้างซ้ายแบบผสม และคืนค่าผลลัพธ์ในตัวสะสม (accumulator) ปริยาย สำหรับคำสั่งเลขคณิตส่วนใหญ่ สถาปัตยกรรมอื่น (เช่น 8086 และตระกูล x86) มีคำสั่งธรรมดาในรุ่นของตัวสะสมอยู่ด้วย ซึ่งคำสั่งที่ยาวกว่าจะทำเรจิสเตอร์อันหนึ่งในเรจิสเตอร์ทั่วไปเป็นตัวสะสม เครื่องที่ใช้กองซ้อน (stack machine) มีตัวถูกดำเนินการส่วนใหญ่หรือทั้งหมดอยู่บนกองซ้อนปริยาย คำสั่งเครื่องที่มีจุดประสงค์พิเศษก็มักจะขาดตัวถูกดำเนินการชัดแจ้ง (ตัวอย่างเช่น ซีพียูไอดีในสถาปัตยกรรม x86 เขียนค่าต่าง ๆ ลงในเรจิสเตอร์ปลายทางเป็นปริยาย 4 เรจิสเตอร์) ความแตกต่างระหว่างตัวถูกดำเนินการชัดแจ้งกับปริยายเช่นนี้สำคัญต่อโปรแกรมสร้างรหัสเครื่อง โดยเฉพาะอย่างยิ่งในเรื่องการจัดสรรเรจิสเตอร์และส่วนติดตามพิสัยแบบสด โปรแกรมทำรหัสให้เหมาะที่สุด (code optimizer) ที่ดีสามารถติดตามตัวถูกดำเนินการทั้งชัดแจ้งและปริยาย ซึ่งอาจช่วยให้เกิดการแพร่กระจายค่าคงตัว (constant propagation) ได้บ่อยยิ่งขึ้น การพับทบค่าคงตัว (constant folding) ของเรจิสเตอร์ (เรจิสเตอร์ที่กำหนดให้เป็นผลลัพธ์จากนิพจน์ของค่าคงตัวจะถูกแทนที่ด้วยค่าคงตัวนั้น) และการปรับปรุงรหัสอื่น ๆ ให้ดีขึ้น

2.10.2.2 ภาษาแอสเซมบลี (Assembly)

ภาษาแอสเซมบลี (อังกฤษ: Assembly Language) หมายถึง ภาษาที่ใช้ในการเขียนโปรแกรมภาษาหนึ่งซึ่งจะทำงานโดยขึ้นกับรุ่นของไมโครโพรเซสเซอร์ หรือ "หน่วยประมวลผล" (CPU) ของเครื่องคอมพิวเตอร์

การใช้ภาษาแอสเซมบลีจำเป็นต้องผ่านการแปลภาษาด้วยคอมไพเลอร์เฉพาะเรียกว่า แอสเซมเบลอร์ (assembler) ให้อยู่ในรูปของรหัสคำสั่งก่อน (เช่น .OBJ) โดยปกติ ภาษานี้ค่อนข้างมีความยุ่งยากในการใช้งาน และการเขียนโปรแกรมเป็นจำนวนบรรทัดมากมากกว่า เมื่อเปรียบเทียบกับการใช้ภาษาระดับสูง เช่น ภาษา C หรือภาษา BASIC แต่จะทำให้ได้ผลลัพธ์การทำงานของโปรแกรมเร็วกว่า และขนาดของตัวโปรแกรมมีขนาดเนื้อที่น้อยกว่าโปรแกรมที่สร้างจากภาษาอื่นมาก จึงนิยมใช้ภาษานี้เมื่อต้องการประหยัดเวลาทำงานของเครื่องคอมพิวเตอร์ และเพิ่มประสิทธิภาพของโปรแกรม

เนื่องจากตัวคำสั่งภายในภาษาอ้างอิงเฉพาะกับรุ่นของหน่วยประมวลผล ดังนั้นถ้ามีการเปลี่ยนแปลงไปใช้กับหน่วยประมวลผลอื่นหรือระบบอื่น (เช่น หน่วยประมวลผล x86 ไม่เหมือนกับ z80) จะต้องมีการปรับแก้ตัวคำสั่งภายในซึ่งบางครั้งอาจไม่สามารถปรับปรุงแก้ไขได้อย่างสมบูรณ์

2.10.2.3 ภาษาระดับสูง (High-level Languages)

ภาษาโปรแกรมระดับสูง หมายถึงภาษาโปรแกรมที่มีภาวะนามธรรมอย่างสูงจากรายละเอียดการทำงานของคอมพิวเตอร์ หากเปรียบเทียบกับภาษาโปรแกรมระดับต่ำแล้ว ภาษาโปรแกรมระดับสูงอาจมี

องค์ประกอบเป็นภาษาธรรมชาติ ใช้งานง่ายกว่า ทำให้กระบวนการพัฒนาโปรแกรมตามข้อกำหนดเรียบง่ายกว่าและสามารถทำความเข้าใจได้ดีกว่า ระดับของภาวะนามธรรมที่ภาษาโปรแกรมจัดเตรียมไว้ให้ เป็นตัวกำหนดว่าภาษานั้นมี "ระดับสูง" มากน้อยแค่ไหน

ประเภทของภาษาโปรแกรมระดับสูงที่ใช้นั้น มีดังนี้

- ภาษาซี (C)

ภาษาซี (C) เป็นภาษาโปรแกรมสำหรับวัตถุประสงค์ทั่วไป เริ่มพัฒนาขึ้นช่วง พ.ศ. 2512-2516 (ค.ศ. 1969-1973) โดยเดนนิส ริชชี่ (Denis Retchie) ที่เอทีแอนด์ทีเบลล์แล็บส์ (AT&T Bell Labs) ภาษาซีเป็นภาษาที่มีความยืดหยุ่นในการเขียนโปรแกรมและมีเครื่องมืออำนวยความสะดวกสำหรับการเขียนโปรแกรมเชิงโครงสร้างและอนุญาตให้มีขอบข่ายตัวแปร (scope) และการเรียกซ้ำ (recursion) ในขณะที่ระบบชนิดตัวแปรอพลวัตก็ช่วยป้องกันการดำเนินการที่ไม่ตั้งใจหลายอย่าง เหมือนกับภาษาโปรแกรมเชิงคำสั่งส่วนใหญ่ในแบบแผนของภาษาอัลกอล การออกแบบของภาษาซีมีคอนสตรักต์ (construct) ที่โยงกับชุดคำสั่งเครื่องทั่วไปได้อย่างพอเพียง จึงทำให้ยังมีการใช้ในโปรแกรมประยุกต์ซึ่งแต่ก่อนลงรหัสเป็นภาษาแอสเซมบลี คือซอฟต์แวร์ระบบอันโดดเด่นอย่างระบบปฏิบัติการคอมพิวเตอร์ ยูนิกซ์

ภาษาซีเป็นภาษาโปรแกรมหนึ่งที่ใช้กันอย่างแพร่หลายมากที่สุดตลอดกาล และตัวแปลโปรแกรมของภาษาซีมีให้ใช้งานได้สำหรับสถาปัตยกรรมคอมพิวเตอร์และระบบปฏิบัติการต่าง ๆ เป็นส่วนมาก

ภาษาหลายภาษาในยุคหลังได้หยิบยืมภาษาซีไปใช้ทั้งทางตรงและทางอ้อม ตัวอย่างเช่น ภาษาดี ภาษาโก ภาษารัสต์ ภาษาจาวา จาวาสคริปต์ภาษาลิมโบ ภาษาแอลพีซี ภาษาซีชาร์ป ภาษาอ็อบเจกทีฟ-ซี ภาษาเพิร์ล ภาษาพีเอชพี ภาษาไพทอน ภาษาเวอริล็อก (ภาษาพรรณนาฮาร์ดแวร์) และซีเชลล์ของยูนิกซ์ ภาษาเหล่านี้ได้ดึงโครงสร้างการควบคุมและคุณลักษณะพื้นฐานอื่น ๆ มาจากภาษาซี ส่วนใหญ่มีวากยสัมพันธ์คล้ายคลึงกับภาษาซีเป็นอย่างมากโดยรวม (ยกเว้นภาษาไพทอนที่ต่างออกไปอย่างสิ้นเชิง) และตั้งใจที่จะผสานนิพจน์และข้อความสั่งที่จำแนกได้ของวากยสัมพันธ์ของภาษาซี ด้วยระบบชนิดตัวแปร ตัวแบบข้อมูล และอรรถศาสตร์ที่อาจแตกต่างกันโดยมูลฐาน ภาษาซีพลัสพลัสและภาษาอ็อบเจกทีฟ-ซีเดิมเกิดขึ้นในฐานะตัวแปลโปรแกรมที่สร้างรหัสภาษาซี ปัจจุบันภาษาซีพลัสพลัสแทบจะเป็นเซตใหญ่ของภาษาซี ในขณะที่ภาษาอ็อบเจกทีฟ-ซีก็เป็นเซตใหญ่อันเคร่งครัดของภาษาซี

ก่อนที่จะมีมาตรฐานภาษาซีอย่างเป็นทางการ ผู้ใช้และผู้พัฒนาต่างก็เชื่อถือในข้อกำหนดอย่างไม่เป็นทางการในหนังสือที่เขียนโดยเดนนิส ริตชี และไบรอัน เคอร์นิกัน (Brian Kernighan) ภาษาซีรุ่นนั้นจึงเรียกกันโดยทั่วไปว่า ภาษาเคแอนด์อาร์ซี (K&R C) ต่อมา พ.ศ. 2532 สถาบันมาตรฐานแห่งชาติของสหรัฐอเมริกา (ANSI) ได้ตีพิมพ์มาตรฐานสำหรับภาษาซีขึ้นมา เรียกกันว่า ภาษาแอนซีซี (ANSI C) หรือ ภาษาซี89 (C89) ในปีถัดมา องค์การระหว่างประเทศว่าด้วยการมาตรฐาน (ISO) ได้อนุมัติให้ข้อกำหนดเดียวกันนี้เป็นมาตรฐานสากล เรียกกันว่า ภาษาซี90 (C90) ในเวลาต่อมาอีก องค์การฯ ก็ได้เผยแพร่ส่วนขยาย

มาตรฐานเพื่อรองรับสากลวิวัตน์ (internationalization) เมื่อ พ.ศ. 2538 และมาตรฐานที่ตรวจชำระใหม่เมื่อ พ.ศ. 2542 เรียกกันว่า ภาษาซี99 (C99) มาตรฐานรุ่นปัจจุบันก็ได้รับอนุมัติเมื่อเดือนธันวาคม พ.ศ. 2554 เรียกกันว่า ภาษาซี11 (C11)

ภาษาซีเป็นภาษาที่ใช้ในการมีปฏิสัมพันธ์เช่น เชิงคำสั่ง (หรือเชิงกระบวนงาน) ถูกออกแบบขึ้นเพื่อใช้แปลด้วยตัวแปลโปรแกรมแบบการเชื่อมโยงที่ตรงไปตรงมา สามารถเข้าถึงหน่วยความจำในระดับล่าง เพื่อสร้างภาษาที่จับคู่อย่างมีประสิทธิภาพกับชุดคำสั่งเครื่อง และแทบไม่ต้องการสนับสนุนใด ๆ ขณะทำงาน ภาษาซีจึงเป็นประโยชน์สำหรับหลายโปรแกรมที่ก่อนหน้านี้เคยเขียนในภาษาแอสเซมบลีมาก่อน

หากคำนึงถึงความสามารถในระดับล่าง ภาษานี้ถูกออกแบบขึ้นเพื่อส่งเสริมการเขียนโปรแกรมที่ขึ้นอยู่กับเครื่องใดเครื่องหนึ่ง (machine-independent) โปรแกรมภาษาซีที่เขียนขึ้นตามมาตรฐานและเคลื่อนย้ายได้ สามารถแปลได้บนแพลตฟอร์มคอมพิวเตอร์และระบบปฏิบัติการต่าง ๆ อย่างกว้างขวาง โดยแก้ไขรหัสต้นฉบับเพียงเล็กน้อยหรือไม่ต้องแก้ไขเลย

ภาษาซียังมีลักษณะเฉพาะต่อไปนี้เพิ่มเติม

- ตัวแปรอาจถูกซ่อนในบล็อกซ้อนใน
- ชนิดตัวแปรไม่เคร่งครัด เช่นข้อมูลตัวอักษรสามารถใช้เป็นจำนวนเต็ม
- เข้าถึงหน่วยความจำคอมพิวเตอร์ในระดับต่ำโดยแปลงที่อยู่ในเครื่องด้วยชนิดตัวแปรตัวชี้ (pointer)
- ฟังก์ชันและตัวชี้ข้อมูลรองรับการทำงานในภาวะหลายรูปแบบ (polymorphism)
- การกำหนดดัชนีแถวลำดับสามารถทำได้ด้วยวิธีรอง คือนิยามในพจน์ของเลขคณิตของตัวชี้
- ตัวประมวลผลก่อนสำหรับการนิยามแมโคร การรวมไฟล์รหัสต้นฉบับ และการแปลโปรแกรมแบบมีเงื่อนไข
- ความสามารถที่ซับซ้อนเช่น ไอ/โอ การจัดการสายอักขระ และฟังก์ชันทางคณิตศาสตร์ รวมอยู่ในไลบรารี่
- คำหลักที่สงวนไว้มีจำนวนค่อนข้างน้อย
- ตัวดำเนินการแบบประสมจำนวนมาก อาทิ `+=`, `-=`, `*=`, `++` ฯลฯ

โครงสร้างการเขียน คล้ายภาษาบีมากกว่าภาษาอัลกอล ตัวอย่างเช่น

- ใช้วงเล็บปีกกา `{ ... }` แทนที่จะเป็น `begin ... end` ในภาษาอัลกอล 60 หรือวงเล็บโค้ง `( ... )` ในภาษาอัลกอล 68
- เท่ากับ `=` ใช้สำหรับกำหนดค่า เหมือนภาษาฟอร์แทรน แทนที่จะเป็น `:=` ในภาษาอัลกอล

- เท่ากับสองตัว == ใช้สำหรับเปรียบเทียบความเท่ากัน แทนที่จะเป็น .EQ. ในภาษาฟอร์แทรนหรือ = ในภาษาเบสิกและภาษาอัลกอล
- ตรรกะ "และ" กับ "หรือ" แทนด้วย && กับ || ตามลำดับ แทนที่จะเป็นตัวดำเนินการ ∧ กับ ∨ ในภาษาอัลกอล แต่ตัวดำเนินการดังกล่าวจะไม่ประเมินค่าตัวถูกดำเนินการทางขวา ถ้าหากผลลัพธ์จากทางซ้ายสามารถพิจารณาได้แล้ว เหตุการณ์เช่นนี้เรียกว่าการประเมินค่าแบบลัดวงจร (short-circuit evaluation) และตัวดำเนินการดังกล่าวก็มีความหมายต่างจากตัวดำเนินการระดับบิต & กับ |

- ภาษาซีพลัสพลัส (C++)

ภาษาซีพลัสพลัส (อังกฤษ: C++) เป็นภาษาโปรแกรมคอมพิวเตอร์อเนกประสงค์ มีโครงสร้างภาษาที่มีการจัดชนิดข้อมูลแบบสแตติก (statically typed) และสนับสนุนรูปแบบการเขียนโปรแกรมที่หลากหลาย (multi-paradigm language) ได้แก่ การโปรแกรมเชิงกระบวนคำสั่ง, การนิยามข้อมูล, การโปรแกรมเชิงวัตถุ, และการโปรแกรมแบบเจเนริก (generic programming)

เบียเนอ สเดราสดร็อบ (Bjarne Stroustrup) จากเบลล์แล็บส์ (Bell Labs) เป็นผู้พัฒนาภาษาซีพลัสพลัส (เดิมใช้ชื่อ "C with classes") ในปี ค.ศ. 1983 เพื่อพัฒนาภาษาซีดั้งเดิม สิ่งที่พัฒนาขึ้นเพิ่มเติมนั้นเริ่มจากการเพิ่มเติมการสร้างคลาสจากนั้นก็เพิ่มคุณสมบัติต่างๆ ตามมา ได้แก่ เวอร์ชวลฟังก์ชัน การโอเวอร์โหลดโอเปอเรเตอร์ การสืบทอดหลายสาย เทมเพลต และการจัดการเอกเซพชันมาตรฐานของภาษาซีพลัสพลัสได้รับการรับรองในปี ค.ศ. 1998 เป็นมาตรฐาน ISO/IEC 14882:1998 เวอร์ชันล่าสุดคือเวอร์ชันในปี ค.ศ. 2003 ซึ่งเป็นมาตรฐาน ISO/IEC 14882:2003 ในปัจจุบันมาตรฐานของภาษาในเวอร์ชันใหม่ (รู้จักกันในชื่อ C++0x) กำลังอยู่ในขั้นพัฒนา รูปแบบของการออกแบบมีดังนี้

- ภาษาซีพลัสพลัสได้ถูกออกแบบมาเพื่อเป็นภาษาสำหรับการเขียนโปรแกรมทั่วไป สามารถรองรับการเขียนโปรแกรมในระดับภาษาเครื่องได้ เช่นเดียวกับภาษาซี
- ในทางทฤษฎี ภาษาซีพลัสพลัสควรจะมีความเร็วเทียบเท่าภาษาซี แต่ในการเขียนโปรแกรมจริงนั้น ภาษาซีพลัสพลัสเป็นภาษาที่มีการเปิดกว้างให้โปรแกรมเมอร์เลือกรูปแบบการเขียนโปรแกรม ซึ่งทำให้มีแนวโน้มที่โปรแกรมเมอร์อาจใช้รูปแบบที่ไม่เหมาะสม ทำให้โปรแกรมที่เขียนมีประสิทธิภาพต่ำกว่าที่ควรจะเป็น และภาษาซีพลัสพลัสนั้นเป็นภาษาที่มีความซับซ้อนมากกว่าภาษาซี จึงทำให้มีโอกาสเกิดบั๊กขณะคอมไพล์มากกว่า
- ภาษาซีพลัสพลัสได้รับการออกแบบเพื่อเข้ากันได้กับภาษาซีในเกือบทุกกรณี
- มาตรฐานของภาษาซีพลัสพลัส ถูกออกแบบมาเพื่อไม่ให้มีการเจาะจงแพลตฟอร์มคอมพิวเตอร์
- ภาษาซีพลัสพลัสถูกออกแบบมาให้รองรับการเขียนโปรแกรมที่หลากหลาย (multi-paradigm)

- ภาษาซีชาร์ป (C#)

ภาษาซีชาร์ป (C# Programming Language) เป็นภาษาโปรแกรมแบบหลายโมเดล ที่ใช้ระบบชนิดข้อมูลแบบรัดกุม (strong typing) และสนับสนุนการเขียนโปรแกรมเชิงคำสั่ง การเขียนโปรแกรมเชิงประกาศ การเขียนโปรแกรมเชิงฟังก์ชัน การเขียนโปรแกรมเชิงกระบวนการ การเขียนโปรแกรมเชิงวัตถุ (แบบคลาส) และการเขียนโปรแกรมเชิงส่วนประกอบ พัฒนาเริ่มแรกโดยบริษัทไมโครซอฟท์เพื่อทำงานบนดอตเน็ตเฟรมเวิร์ก โดยมีแอนเดอร์ เฮลส์เบิร์ก (Anders Hejlsberg) เป็นหัวหน้าโครงการ และมีรากฐานมาจากภาษาซีพลัสพลัสและภาษาอื่นๆ (โดยเฉพาะภาษาเดลไฟและจาวา) โดยมีจุดมุ่งหมายให้เป็นภาษาสมัยใหม่ที่ไม่ซับซ้อน ใช้งานได้ทั่วไป (general-purpose) และเป็นเชิงวัตถุเป็นหลัก

ปัจจุบันภาษาซีชาร์ปมีการรับรองให้เป็นมาตรฐานโดยเอ็กมาอินเตอร์เนชันแนล (Ecma International) และองค์การระหว่างประเทศว่าด้วยการมาตรฐาน (ISO) และมีรุ่นล่าสุดคือ C# 5.0 ที่ออกมาเมื่อวันที่ 15 สิงหาคม พ.ศ. 2555

- ภาษาโคบอล (COBOL)

ภาษาโคบอล (COBOL programming language) เป็นภาษาโปรแกรมระดับสูงภาษาหนึ่งที่อยู่มาอย่างยาวนาน COBOL ย่อมาจาก Common Business Oriented Language เป็นภาษาที่นิยมนำไปใช้ทางธุรกิจ ถูกพัฒนาขึ้นเมื่อ ค.ศ. 1959 โดยนักคอมพิวเตอร์กลุ่มหนึ่งที่เรียกตัวเองว่า Conference on Data Systems Languages (CODASYL) และตั้งแต่ปี ค.ศ. 1959 ภาษาโคบอลมีการแก้ไขและปรับปรุงอยู่ตลอด ดังนั้น เพื่อขจัดปัญหาความแตกต่างของตัวภาษาโคบอลในแต่ละเวอร์ชัน สถาบันมาตรฐานแห่งชาติอเมริกัน (ANSI) จึงได้พัฒนามาตรฐานกลางขึ้นมาในปี ค.ศ. 1968 เป็นที่รู้จักกันในนามของ ANS COBOL ต่อมาเมื่อ ปี ค.ศ. 1974 ทาง ANSI ได้นำเสนอ ANS COBOL รุ่นใหม่ที่มีคุณสมบัติที่ดีกว่ารุ่น 1968 และในปี ค.ศ. 1985 ANSI ก็นำเสนออีกรุ่นหนึ่งที่มีคุณสมบัติมากกว่ารุ่นปี 1974

รูปแบบภาษาโคบอลแบ่งออกเป็น 4 ดิวิชั่น คือ

1. Identification division การกำหนดชื่อโปรแกรมและชื่อผู้เขียน
2. Environment division การอธิบายเกี่ยวกับคอมพิวเตอร์
3. Data division การอธิบายเกี่ยวกับการประมวลผลข้อมูล
4. Procedure division การอธิบายเกี่ยวกับขั้นตอนการประมวลผล

- ภาษาปาสกาล (Pascal)

ภาษาปาสกาล เป็นภาษาโปรแกรมที่ใช้กันอย่างกว้างขวาง โดยเฉพาะในวงการศึกษา คิดค้นขึ้นโดย นิเคลาส์ แวร์ท (Niklaus Wirth) นักวิทยาการคอมพิวเตอร์ชาวสวิตเซอร์แลนด์ ในปี ค.ศ. 1970 เพื่อช่วยในการเรียนการสอนการเขียนโปรแกรมโครงสร้าง (structured programming) ภาษาปาสกาลนั้นพัฒนาขึ้นมาจาก ภาษาอัลกอล (Algol), และชื่อปาสกาลนั้น ตั้งเพื่อเป็นเกียรติแก่ แบลส ปาสกาล (Blaise Pascal) นอกเหนือจากภาษาปาสกาลแล้ว แวร์ทได้พัฒนา ภาษาโมดูลาทู (Modula-2) และ โอบีรอน (Oberon) ซึ่งมีโครงสร้างคล้ายกับภาษาปาสกาล แต่สามารถรองรับการเขียนโปรแกรมเชิงวัตถุ (object-oriented programming)

โปรแกรมภาษาปาสกาลทุกอัน จะเริ่มต้นด้วยคีย์เวิร์ด Program และส่วนของโค้ดจะอยู่ระหว่างคีย์เวิร์ด Begin และ End ภาษาปาสกาลนั้นไม่สนใจความแตกต่างระหว่างตัวพิมพ์ใหญ่และตัวพิมพ์เล็ก ("end" มีผลเท่ากับ "End"). เซมิโคลอน (;) ใช้เพื่อแบ่งคำสั่ง และ มหัพภาค(.) ใช้เมื่อจบโปรแกรม (หรือยูนิต)

ภาษาปาสกาลเป็นภาษาที่มีโครงสร้างที่ตายตัว เช่นการประกาศตัวแปร จะอยู่ระหว่าง Program กับ Begin โดยไม่สามารถไปประกาศที่อื่นได้เหมือนกับภาษา VB,C หรือภาษาอื่น ๆ ทำให้ผู้เรียนได้ทราบถึงขั้นตอนการเขียนโปรแกรมที่ถูกต้อง เพื่อง่ายต่อการตรวจสอบในภายหลัง

- ภาษาเบสิก (BASIC)

ภาษาเบสิก (BASIC programming language) เป็นภาษาโปรแกรมที่ออกแบบมาให้ใช้งานได้ง่าย และยังได้รับความนิยมมาจนถึงทุกวันนี้ เบสิกออกแบบมาให้ใช้กับคอมพิวเตอร์ตามบ้าน

ชื่อภาษาเบสิก หรือ BASIC ย่อมาจาก Beginner's All-purpose Symbolic Instruction Code ต้องเขียนด้วยตัวพิมพ์ใหญ่เสมอ

บริษัทไมโครซอฟท์ได้นำภาษาเบสิกมาปรับปรุงให้ทันสมัย และพัฒนาเครื่องมือพัฒนาโปรแกรม Visual Basic ทำให้เบสิกได้รับความนิยมในการพัฒนาโปรแกรมยุคใหม่ รุ่นล่าสุดของวิชวลเบสิกเรียกว่า VB.NET

- ภาษาฟอร์แทรน (FORTRAN)

ภาษาฟอร์แทรน (Fortran programming language หรือ FORTRAN) เป็นภาษาที่เก่าแก่ที่สุดของวงการคอมพิวเตอร์ ถูกพัฒนาขึ้นในยุคคริสต์ทศวรรษ 1950 นิยมนำไปใช้ในการคำนวณทางคณิตศาสตร์และวิทยาศาสตร์ จนถึงปัจจุบันนี้ ภาษาฟอร์แทรนก็ยังถูกใช้ในทางวิทยาศาสตร์อยู่

- ภาษาจาวา (Java)

ภาษาจาวา (Java programming language) เป็นภาษาโปรแกรมเชิงวัตถุ (อังกฤษ: Object Oriented Programming) พัฒนาโดย เจมส์ กอสลิง และวิศวกรคนอื่นๆ ที่ ซัน ไมโครซิสเต็มส์ ภาษาจาวาถูกพัฒนาขึ้นในปี พ.ศ. 2534 (ค.ศ. 1991) โดยเป็นส่วนหนึ่งของ โครงการกรีน (the Green Project) และสำเร็จออกสู่สาธารณะในปี พ.ศ. 2538 (ค.ศ. 1995) ซึ่งภาษานี้มีจุดประสงค์เพื่อใช้แทนภาษาซีพลัสพลัส (C++) โดยรูปแบบที่เพิ่มเติมขึ้นคล้ายกับภาษาอ็อบเจกต์ทีฟซี (Objective-C) แต่เดิมภาษานี้เรียกว่า ภาษาโอ๊ก (Oak) ซึ่งตั้งชื่อตามต้นโอ๊กใกล้ที่ทำงานของ เจมส์ กอสลิง แต่ว่ามีปัญหาทางลิขสิทธิ์ จึงเปลี่ยนไปใช้ชื่อ "จาวา" ซึ่งเป็นชื่อกาแฟแทน

และแม้ว่าจะมีชื่อคล้ายกัน แต่ภาษาจาวาไม่มีความเกี่ยวข้องใด ๆ กับภาษาจาวาสคริปต์ (JavaScript) ปัจจุบันมาตรฐานของภาษาจาวาดูแลโดย Java Community Process ซึ่งเป็นกระบวนการอย่างเป็นทางการที่อนุญาตให้ผู้ที่สนใจเข้าร่วมกำหนดความสามารถในจาวาแพลตฟอร์มได้

- ภาษาจาวาสคริปต์ (JavaScript)

จาวาสคริปต์ (JavaScript) เป็นภาษาสคริปต์ ทีมีลักษณะการเขียนแบบโปรโตไทพ (Prototyped-based Programming) ส่วนมากใช้ในหน้าเว็บเพื่อประมวลผลข้อมูลที่ฝั่งของผู้ใช้งาน แต่ก็ยังมีใช้เพื่อเพิ่มเติมความสามารถในการเขียนสคริปต์โดยฝังอยู่ในโปรแกรมอื่นๆ

ซัน ไมโครซิสเต็มส์เป็นเจ้าของเครื่องหมายการค้า "JavaScript" โดยมันถูกนำไปใช้ภายใต้สัญญาอนุญาตเพื่อการพัฒนาเทคโนโลยีโดย เน็ตสเคป และมูลนิธิมอซิลลา

- ภาษาเพิร์ล (Perl)

ภาษาเพิร์ล (อังกฤษ: Perl) (ย่อมาจาก Practical Extraction and Report Language) เป็นภาษาโปรแกรมแบบไดนามิก พัฒนาโดยนายแลร์รี วอลล์ (Larry Wall) ในปี ค.ศ. 1987 เพื่อใช้งานกับระบบปฏิบัติการยูนิกซ์ ภาษาเพิร์ลนั้นถูกออกแบบมาให้ใช้งานได้ง่าย โครงสร้างของภาษาจึงไม่ซับซ้อน มีลักษณะคล้ายกับภาษาซี นอกจากนี้เพิร์ลยังได้แนวคิดบางอย่างมาจากเชลล์สคริปต์, ภาษา AWK, sed และ Lisp ปัจจุบันเวอร์ชันล่าสุดคือ 5.18.0

- ภาษาพีเอชพี (PHP)

พีเอชพี (PHP) คือ ภาษาคอมพิวเตอร์ในลักษณะเซิร์ฟเวอร์-ไซด์ สคริปต์ โดยลิขสิทธิ์อยู่ในลักษณะโอเพนซอร์ส ภาษาพีเอชพีใช้สำหรับจัดทำเว็บไซต์ และแสดงผลออกมาในรูปแบบ HTML โดยมีรากฐานโครงสร้างคำสั่งมาจากภาษา ภาษาซี ภาษาจาวา และ ภาษาเพิร์ล ซึ่ง ภาษาพีเอชพี นั้นง่ายต่อการเรียนรู้ ซึ่งเป้าหมายหลักของภาษานี้ คือให้นักพัฒนาเว็บไซต์สามารถเขียน เว็บเพจ ที่มีการตอบโต้ได้อย่างรวดเร็ว พีเอชพีรุ่นล่าสุดคือ PHP 5.4.0 ส่วนรุ่นพัฒนาคือ PHP 6.0.0-dev

- ภาษาไพทอน (Python)

ภาษาไพทอน (Python programming language) เป็นภาษาโปรแกรมระดับสูง เพื่อใช้งานทั่วไป แบบอินเทอร์พรีเตอร์ ที่สร้างโดยกีโด ฟาน รอสซัม (Guido van Rossum) ในพ.ศ. 2533 ปัจจุบันดูแลโดย มูลนิธิซอฟต์แวร์ไพทอน จุดเด่นของภาษาไพทอน คือ

ความเป็นภาษาสคริปต์

เนื่องจากไพทอนเป็นภาษาสคริปต์ ทำให้ใช้เวลาในการเขียนและคอมไพล์ไม่มาก ทำให้เหมาะกับงานด้านการดูแลระบบ (System administration) เป็นอย่างยิ่ง

ไวยากรณ์ที่อ่านง่าย

ไวยากรณ์ของไพทอนได้กำจัดการใช้สัญลักษณ์ที่ใช้ในการแบ่งบล็อกของโปรแกรม และใช้การย่อหน้าแทน ทำให้สามารถอ่านโปรแกรมที่เขียนได้ง่าย นอกจากนั้นยังมีการสนับสนุนการเขียน docstring ซึ่งเป็นข้อความสั้นๆ ที่ใช้อธิบายการทำงานของฟังก์ชัน, คลาส, และโมดูลอีกด้วย

ความเป็นภาษา Glue

ไพทอนเป็นภาษา Glue (Glue Language) ได้อย่างดีเนื่องจากสามารถเรียกใช้ภาษาโปรแกรมอื่นๆ ได้หลายภาษา ทำให้เหมาะที่จะใช้เขียนเพื่อประสานงานโปรแกรมที่เขียนในภาษาต่างกันได้

- ภาษาโปรล็อก (Prolog)

ภาษาโปรล็อก (อังกฤษ: Prolog) เป็นภาษาสำหรับการเขียนโปรแกรมเชิงตรรกะ ได้ชื่อมาจาก *PROgrammation en LOGique* (logic programming) สร้างขึ้นโดย Alain Colmerauer ราว ค.ศ. 1972 ภาษาโปรล็อกเกิดจากความพยายามที่จะสร้างภาษาที่อาศัยวิธีการทางตรรกศาสตร์แทนที่จะกำหนดคำสั่งอย่างละเอียดให้กับคอมพิวเตอร์

ภาษาโปรล็อกถูกนำไปใช้ในโปรแกรมสำหรับปัญญาประดิษฐ์ และภาษาศาสตร์เชิงคำนวณ โดยเฉพาะการประมวลผลภาษาธรรมชาติ ไวยากรณ์และความหมายของภาษานั้นเรียบง่ายและชัดเจน งานวิจัยจำนวนมากที่ทำให้เกิดการพัฒนาภาษาโปรล็อกในปัจจุบันนั้น เป็นผลมาจากโครงการระบบคอมพิวเตอร์ยุคที่ห้า (fifth generation computer systems project - FGCS) ซึ่งเลือกรูปแบบหนึ่งของภาษาโปรล็อกเป็นภาษาแก่น (Kernel Language) ของระบบปฏิบัติการ

ภาษาโปรล็อกมีพื้นฐานมาจากแคลคูลัสภาคแสดง (predicate calculus) หรือเรียกเต็ม ๆ ว่า แคลคูลัสภาคแสดงอันดับที่หนึ่ง (first-order predicate calculus) โดยจำกัดให้ใช้เฉพาะอนุประโยคของฮอร์น (Horn clause) การดำเนินการของโปรแกรมโปรล็อก ก็คือการประยุกต์วิธีพิสูจน์ทฤษฎีบทโดยใช้รีโซลูชันอันดับหนึ่ง (first-order resolution) แนวคิดพื้นฐานที่เกี่ยวข้องได้แก่ การทำให้เท่ากัน(unification), การเรียกซ้ำจากส่วนท้าย (tail recursion), การย้อนรอย (backtracking)

- ภาษาอ็อบเจกทีฟ-ซี (Objective-C)

ภาษาอ็อบเจกทีฟ-ซี (อังกฤษ: Objective-C หรือ ObjC) เป็นภาษาโปรแกรมเชิงวัตถุและมีสมบัติการสะท้อน โดยแรกเริ่ม ภาษาอ็อบเจกทีฟ-ซี พัฒนาขึ้นจากภาษาซีโดยยังคงคุณลักษณะของภาษาซีไว้ครบทุกประการเพียงแต่เพิ่มระบบส่งข้อความ (messaging) แบบเดียวกับภาษาสมอลล์ทอล์กเข้าไปเท่านั้น (Objective-C runtime) ปัจจุบันภาษาอ็อบเจกทีฟ-ซีมีคุณสมบัติอื่นๆเพิ่มเติมจากการพัฒนาภาษาอ็อบเจกทีฟ-ซี 2.0 โดยบริษัทแอปเปิล

ปัจจุบันภาษาอ็อบเจกทีฟ-ซีถูกใช้มากใน Cocoa (API) ใน Mac OS X, GNUstep (API) และ Cocotron (API) เป็นต้น ซึ่งระบบเหล่านี้ได้รับการพัฒนาขึ้นโดยมีพื้นฐานจากมาตรฐาน OpenStep (API) ใน Nextstep (Operating system) โดยมีภาษาอ็อบเจกทีฟ-ซีเป็นภาษาหลัก ปัจจุบัน Mac OS Xใช้ Cocoa เป็นเฟรมเวิร์กสำหรับสร้างโปรแกรมประยุกต์ โดย ไลบรารีและ/หรือ API เหล่านี้เป็นเพียงส่วนเพิ่มขยาย (Software extension) เท่านั้น โปรแกรมที่ใช้ภาษาอ็อบเจกทีฟ-ซีทั่วไปที่ไม่ได้ใช้ส่วนเพิ่มขยายเหล่านี้ก็ยังสามารถคอมไพล์ได้ เช่นอาจใช้แต่ gcc ซึ่งรองรับภาษาอ็อบเจกทีฟ-ซี

- ภาษารูบี้ (Ruby)

ภาษารูบี (Ruby) เป็นภาษาโปรแกรมเชิงวัตถุ ที่ได้รับอิทธิพลของโครงสร้างภาษามาจาก ภาษาเพิร์ลกับภาษาเอดา มีความสามารถในเชิงวัตถุแบบเดียวกับภาษาสมอลทอล์ค และมีความสามารถหลายอย่างจากภาษาไพทอน, ภาษาลิสป์, ภาษา Dylan และภาษา CLU ตัวแปลภาษารูบีตัวหลักเป็นซอฟต์แวร์เสรี และเป็นตัวแปลแบบอินเตอร์พรีเตอร์

2.10.2 Open Source Computer Vision (OpenCV)

OpenCV เป็น library ที่มีใบอนุญาต BSD และเป็น open-source เป็น library ที่มีอัลกอริทึมอยู่มากมาย OpenCV มีโครงสร้างแบบแยกส่วน ซึ่งหมายความว่า แพคเกจจะรวม library ต่างๆที่ใช้ร่วมกันไว้โดย modules มีดังต่อไปนี้

- core

Module สำหรับการอธิบายถึงโครงสร้างของข้อมูลขั้นพื้นฐาน รวมถึง array หลายมิติ และฟังก์ชั่นพื้นฐานต่างๆที่ใช้ใน modules อื่นๆ

- improc

Module สำหรับการประมวลผลภาพ รวมถึงการกรองภาพ (filtering) ทั้งแบบ linear และ non-linear การเปลี่ยนแปลงทางเรขาคณิตของภาพ (การปรับขนาด, warping, remapping) การทำ histograms และอื่นๆ

- video

Module เพื่อการวิเคราะห์วิดีโอ รวมถึงการประมาณการเคลื่อนที่ และขั้นตอนการตรวจจับวัตถุ

- calib3d

Algorithms ทางเรขาคณิตแบบหลายมุมมอง, การสอบเทียบกล้องเพื่อการสร้าง 3D

- features2d

การตรวจจับลักษณะเด่นของวัตถุ

- objdetect

การตรวจจับวัตถุและกรณีของกลุ่มที่กำหนดไว้ล่วงหน้า (เช่น ใบหน้า, ดวงตา, รถ หรืออื่นๆ)

- highgui

เพื่อให้ง่ายต่อการจับภาพจากวิดีโอ การแปลงสัญญาณวิดีโอและรูปภาพ

- gpu

Algorithms ของ GPU-accelerated จาก modules อื่น

- อื่นๆ

2.11 การเขียนโปรแกรมด้วยภาษา c++

2.11.1 โครงสร้างของภาษา c++

โปรแกรมแรกสำหรับผู้เริ่มต้นในการเขียนโปรแกรมในทุกๆภาษาคือโปรแกรมที่เรียกว่า "Hello World" ซึ่งเป็นโปรแกรมที่จะแสดงผลคำว่า "Hello World" ออกทางจอคอมพิวเตอร์

```
//my first world program
#include <iostream>

int main(){
  std::cout << "Hello World!";
}
```

โดยผลลัพธ์ของโปรแกรมจะแสดงดังนี้

```
Hello World!
```

ในส่วนแรกที่นั้นคือซอสโค้ดของโปรแกรม และส่วนที่สองคือผลลัพธ์ของโปรแกรม เมื่อรันโปรแกรมจะเห็นคำว่า "Hello world" ถูกแสดงผลออกทางหน้าจอคอมพิวเตอร์ โดยสามารถทดสอบกับคำหรือประโยคอื่นๆ เพื่อดูผลลัพธ์

comment

comment เป็นส่วนของซอสโค้ดที่ไม่มีผลกับโปรแกรม มันทำให้โปรแกรมเมอร์สามารถอธิบายโปรแกรมของพวกเขาและเพื่อให้ตรวจสอบได้ง่ายในภายหลัง ในภาษา C++ มีสองทางที่จะสามารถ coment ได้ คือ

```
// line comment
/* block comment */
```

comment ประเภทแรกคือการ comment แบบบรรทัด มันถูกใช้สำหรับเพื่อ comment ในหนึ่งบรรทัดและจะไม่สนใจโค้ดของโปรแกรมหลังจากเครื่องหมาย // แบบที่สองคือ ละเว้นทุกอย่างเริ่มจาก /* และสิ้นสุดที่ */ ซึ่งมันมักจะใช้กับการคอมเม้นต์ในหลายบรรทัด อย่างไรก็ตามมันยังสามารถที่จะใช้ในการคอมเม้นต์หนึ่งบรรทัดได้

Using namespace std

ภาษา C++ มีไลบรารี่มาตรฐานที่เรียกว่า namespace (เนมสเปลซ) จากตัวอย่างของโค้ดโปรแกรมก่อนหน้าที่เราได้ใช้มันไปแล้ว

```
using namespace std;
```

เพื่อใช้ฟังก์ชันในไลบรารี่ของ namespace std โดยไม่ต้องมี std:prefix เราจำเป็นต้องใช้คำสั่ง using namespace std อย่างเช่น ฟังก์ชัน cout เราจะสามารถเรียกใช้ฟังก์ชันนี้ได้ในทันที

```
#include <iostream>

int main ()
{
  std::cout << "Hello C++";
  std::cout << "My name is Mateo";
}
```

```
#include <iostream>

using namespace std;
int main ()
{
  cout << "Hello C++";
  cout << "My name is Mateo";
}
```

โปรแกรมสองโปรแกรมด้านบนให้ผลลัพธ์ที่เหมือนกัน โปรแกรมแรกเราไม่ได้ประกาศโดยการใช้คำสั่ง namespace std และเราจำเป็นต้องเข้าถึงฟังก์ชัน cout โดยการใช้ std:cout ในการที่จะอนุญาติให้สามารถเข้าถึงทุกอย่างใน namespace ได้โดยไม่ต้องใช้ std:prefix เราจำเป็นต้องใช้คำสั่ง using namespace std เหมือนในโปรแกรมที่สอง

2.11.2 ตัวแปรและประเภทของข้อมูล

ตัวแปรนั้นเป็นสิ่งที่สำคัญสำหรับในการเขียนโปรแกรมทุกภาษา มันถูกใช้เพื่อเก็บข้อมูลในหน่วยความจำและช่วยให้เราสามารถจัดการกับข้อมูลได้อย่างง่ายดาย

ยกตัวอย่างเช่น ถ้าต้องการที่จะจำบางอย่าง จะต้องเขียนมันลงไปบนสมุดบันทึก เพราะจะทำให้ไม่ลืมเมื่อบันทึกลงไป สามารถนำมาใช้เมื่อไหร่ก็ตามที่ต้องการ ดังนั้นในการเขียนโปรแกรมคอมพิวเตอร์ ตัวแปรถูกนำมาใช้ในแนวคิดเดียวกัน

```
int a = 2;
int b = 2;

int sum = a + b;
```

ในตัวอย่าง เราเห็นค่า 2 ในตัวแปร a และ b เพื่อหาผลรวมของตัวแปร a และตัวแปร b เราจำเป็นต้องจดจำ 2 ไว้ใสตัวแปร a ก่อนจนกว่าเราจะมีค่าของ b และหลังจากนั้นเราได้ทำการรวมค่าของตัวแปรทั้งสองไว้ในอีกตัวแปรคือ sum จะเห็นคำว่า int ก่อนชื่อของตัวแปร ซึ่งเรียกว่าประเภทของตัวแปร หรือการประกาศประเภทของตัวแปร

ประเภทข้อมูล

ในภาษา C++ มีตัวแปรหลายประเภทที่ช่วยให้เราสามารถจัดการกับข้อมูลประเภทต่างๆ เช่น boolean, number, character, string และ object เป็นต้น

ในภาษา C++ จะมีกลุ่มของประเภทตัวแปรอยู่ทั้งหมด 4 กลุ่ม

- Character types: นี่เป็นประเภทของข้อมูลที่ใช้เก็บตัวอักษรหนึ่งตัว เช่น 'a', 'b' หรือ '8'.
- Numerical integer types: ประเภทของข้อมูลชนิดนี้ใช้สำหรับเก็บค่าของจำนวนธรรมชาติ เช่น 1 หรือ 1000 ซึ่งจะแบ่งย่อยไปตามขนาดที่ใช้เก็บ โดยปกตินั้นจะเป็นแบบ int และ long.
- Floating-point types: ประเภทของข้อมูลชนิดนี้ใช้เพื่อเก็บจำนวนจริง เช่น 1.8 100.05 หรือ -5.5
- Boolean type: ประเภทของข้อมูลชนิดนี้สามารถเก็บค่าได้เพียงสองค่าคือ true และ false.

ตารางข้างล่างนี้แสดงประเภทของตัวแปรทั้งหมดในภาษา C++

ตารางที่ 2.1 ประเภทของตัวแปรในภาษา c++

ชนิดของตัวแปร	ขนาด (bits)	ขอบเขต	ข้อมูลที่เก็บ
char	8	-128 ถึง 127	ข้อมูลชนิดอักขระ ใช้เนื้อที่ 1 byte
unsigned char	8	0 ถึง 255	ข้อมูลชนิดอักขระ ไม่คิดเครื่องหมาย
int	16	-32,768 ถึง 32,767	ข้อมูลชนิดจำนวนเต็ม ใช้เนื้อที่ 2 byte
unsigned int	16	0 ถึง 65,535	ข้อมูลชนิดจำนวนเต็ม ไม่คิดเครื่องหมาย
short	8	-128 ถึง 127	ข้อมูลชนิดจำนวนเต็มแบบสั้น ใช้เนื้อที่ 1 byte

unsigned short	8	0 ถึง 255	ข้อมูลชนิดจำนวนเต็มแบบสั้น ไม่คิดเครื่องหมาย
long	32	-2,147,483,648 ถึง 2,147,483,649	ข้อมูลชนิดจำนวนเต็มแบบยาว ใช้เนื้อที่ 4 byte
unsigned long	32	0 ถึง 4,294,967,296	ข้อมูลชนิดจำนวนเต็มแบบยาว ไม่คิดเครื่องหมาย
float	32	3.4*10e(-38) ถึง 3.4*10e(38)	ข้อมูลชนิดเลขทศนิยม ใช้เนื้อที่ 4 byte
double	64	3.4*10e(-308) ถึง 3.4*10e(308)	ข้อมูลชนิดเลขทศนิยม ใช้เนื้อที่ 8 byte
long double	128	3.4*10e(-4032) ถึง 1.1*10e(4032)	ข้อมูลชนิดเลขทศนิยม ใช้เนื้อที่ 16 byte

การประกาศตัวแปร

ภาษา C++ เป็นภาษาที่เข้มงวดในการเขียน ตัวแปรจำเป็นต้องประกาศก่อนที่จะมีการใช้งาน นั่นหมายความว่าคอมไพเลอร์จะมีการจองหน่วยความจำที่เพียงพอสำหรับตัวแปร รูปแบบในการประกาศตัวแปรในภาษา C++ นั้นตรงไปตรงมา ตัวอย่างสำหรับการประกาศตัวแปรในภาษา C++

```
int n;
float money;
bool t;
```

ในตัวอย่างเราได้ประกาศ 3 ตัวแปร ตัวแปรแรกเราประกาศตัวแปรที่มีชนิดข้อมูลเป็น int และมีชื่อของตัวแปรว่า n ตัวแปรที่สองประกาศตัวแปรที่มีชนิดข้อมูลเป็น float และมีชื่อของตัวแปรว่า money ตัวแปรที่สามนั้นเป็นประเภท boolean ที่มีชื่อว่า t เมื่อตัวแปรถูกสร้าง มันจะสามารถถูกใช้ได้ในขอบเขตของโปรแกรม

การกำหนดค่าเริ่มต้นให้กับตัวแปร

หลังจากที่เราได้ประกาศตัวแปรแล้ว เราสามารถกำหนดค่าให้มันในตอนเริ่มต้นหรือในตอนที่โปรแกรมรันได้ทันที

String เป็นตัวอักษรที่เรียงต่อกันที่ถูกแสดงในรูปแบบอาเรย์ของตัวอักษร แต่ string ในภาษา C++ เป็นคลาสที่มากับ C++ ไลบรารี่ ในการที่จะใช้ string เราจำเป็นต้องนำเข้าไลบรารี่ของ string โดยใช้คำสั่ง #include <string> ข้อมูลประเภทสตริงนั้นจะเก็บข้อมูลในรูปแบบของคำหรือประโยค ที่เป็นตัวอย่างเพื่อใช้ string ในภาษา C++

```
// string example
#include <iostream>
#include <string>

using namespace std;

int main(){
  string name = "Mateo";
  cout << "My name is " << name;
}
```

2.11.3 ค่าคงที่

ค่าคงที่ (constant) เป็นตัวแปรประเภทหนึ่งที่ไม่สามารถเปลี่ยนแปลงค่าได้ในขณะที่โปรแกรมทำงาน นี่หมายความว่าเราจะต้องกำหนดค่าให้ตัวแปรในเวลาที่คอมไพเลอร์ทำงานหรือในตอนแรกที่เราสร้างตัวแปรแบบค่าคงที่ขึ้นมา ค่าคงที่ที่เราใช้กันบ่อยๆ นั้นเรียกว่า literal ซึ่ง literal สามารถแบ่งแยกได้เป็น integer, floating-point, characters, strings, Boolean, pointers และที่ผู้ใช้สร้างขึ้นเอง

Typed constant

เราสามารถประกาศค่าคงที่โดยการตั้งชื่อและกำหนดค่าให้กับมันในตอนที่เราประกาศตัวค่าคงที่เสมอ หลังจากนั้นเราสามารถเรียกใช้ตัวแปรค่าคงที่โดยใช้ชื่อของมันได้ในโปรแกรม ชนิดของตัวแปรประเภทค่าคงที่นั้นเหมือนกับตัวแปรปกติ ซึ่งจะมี integer, floating-point, characters, strings, Boolean, pointers ตัวอย่างเช่น

```
const int length = 100;
const double pi = 3.1415926;
const char n = 'a';
```

Preprocessor definitions

อีกทางหนึ่งในการประกาศค่าคงที่คือการใช้ processor definitions โดยมีรูปแบบดังนี้

```
#define identifier replacement
```

ซึ่งคำสั่งนี้จะถูกประมวลผลโดย preprocessor และเกิดขึ้นในตอนที่โปรแกรมคอมไพล์ และมันไม่จำเป็นต้องจบด้วยเครื่องหมายเซมิโคลอน นี่เป็นตัวอย่างสำหรับการใช้วิธีนี้

```
#define PI 3.14159
#define NAME 'Mateo'
```

ตอนนี้ เราสามารถใช้ค่าคงที่ที่เราเพิ่งได้สร้าง โดยการใช้ชื่อของมัน

```
#include <iostream>
using namespace std;

#define PI 3.14159
#define NAME "Mateo"

int main(){
    cout << "Pi is " << PI << endl;
```

```
    cout << "Name is " << NAME ;
}
```

2.11.4 ตัวดำเนินการ

ตัวดำเนินการถูกใช้เพื่อดำเนินการกับตัวแปรและค่าคงที่ ในภาษา C++ มีตัวดำเนินการประเภทต่างๆ นี่เป็นตัวดำเนินการที่สำคัญที่จะต้องรู้จัก

Assignment operator

ตัวดำเนินการกำหนดค่านั้นแสดงโดยใช้เครื่องหมาย = มันถูกใช้เพื่อกำหนดค่าข้อมูลให้กับตัวแปร

```
x = 10;
y = x;
```

ในตัวอย่าง เรามีตัวแปรสองตัวคือ *x* และ *y* เราได้กำหนดค่า 10 ให้กับตัวแปร x และเรากำหนดค่าของ x ให้กับตัวแปร y ดังนั้นทั้งตัวแปร x และ y จะมีค่าเป็น 10

Arithmetic operators (+, -, *, /, %)

ตัวดำเนินการคณิตศาตร์ ถูกใช้เพื่อดำเนินการเกี่ยวกับทางคณิตระหว่างตัวแปรและค่าคงที่ แสดงตามตารางที่ 2.3

ตารางที่ 2.2 ตัวดำเนินการทางคณิตศาสตร์ในภาษา C++

Symbol	Name	Example
+	Addition	c = a + b
-	Subtraction	c = a - b
*	Multiplication	c = a * b
/	Division	c = a / b
%	Modulo	c = a % b

ตัวอย่างเกี่ยวกับการใช้ตัวดำเนินการทางคณิตศาสตร์

```
#include <iostream>

using namespace std;

int main () {
   int a = 3;
   int b = 2;
   cout << "a + b = " << a + b << endl;
   cout << "a - b = " << a - b << endl;
   cout << "a * b = " << a * b << endl;
   cout << "a / b = " << a / b << endl;
   cout << "a % b = " << a % b << endl;
}
```

เมื่อโปรแกรมทำงาน จะให้ผลลัพธ์ดังต่อไปนี้

```
a + b = 5
a - b = 1
a * b = 6
a / b = 1
a % b = 1
```

Compound assignment (+=, -=, *=, /=, %=, >>=, <<=, &=, ^=, |=)

Compound assignment operators ถูกใช้เพื่อแก้ไขค่าปัจจุบันของตัวแปรโดยการใช้ตัวดำเนินการทางคณิตศาตร์และตัวดำเนินการกำหนดค่าในเวลาเดียวกัน ซึ่งแสดงดังตารางรา 2.4

ตารางที่ 2.3 ตารางของตัวดำเนินการกำหนดเพิ่มค่า

Operator	Example	Equivalent to
+=	a += 2;	a = a + 2
-=	a -= 2;	a = a - 2
*=	a *= 2;	a = a * 2
/=	a /= 2;	a= a / 2
%=	a %= 2;	a = a % 2
>>=	a >>= 2;	a = a >> 2
<<=	a <<= 2	a = a << 2
&=	a & = 2;	a = a & 2
^=	a ^= 2;	a= a ^ 2

|=	a |= 2;	a = a | 2

Increment และ decrement (++, --)

ตัวดำเนินการ Increment และ decrement operators ถูกใช้เพื่อเพิ่มหรือลดค่าของตัวแปร ตัวดำเนินการดำเนินการกับตัวแปรโดยการเพิ่ม ++ หรือ -- หลังจากตัวแปร ตัวอย่างเช่น

```
#include <iostream>

using namespace std;

int main () {
   int x = 1;
   int y = 5;
   x++;
   ++x;
   y--;
   cout << " x = " << x;
   cout << " y = " << y;
}
```

โปรแกรมจะมีผลลัพธ์ตามนี้

```
x = 3
y = 4
```

ตัวดำเนินการนี้เป็นรูปย่อของ x = x + 1 เหมือนที่คุณได้เห็นในโปรแกรม อย่างไรก็ตามมันมีข้อแตกต่างระหว่าง prefix และ postfix increments โดย x++ จะต้องเสร็จสิ้นคำสั่งปัจจุบันก่อนที่จะเพิ่มค่า 1 ไปยังตัวแปร *x* แต่ ++x จะเพิ่มค่า 1 ไปยังตัวแปร *x* ก่อนที่คำสั่งจะทำงาน

Relational และ comparison operators (==, !=, >, <, >=, <=)

ตัวดำเนินการเปรียบเทียบถูกใช้ในการดำเนินการเปรียบเทียบระหว่างตัวแปร ผลลัพธ์ของตัวดำเนินการเหล่านี้จะมีแค่ true และ false อีกนัยหนึ่ง มันเป็นการเขียนโปรแกรมแบบมีเงื่อนไข ที่ใช้ในคำสั่งที่มีการเปรียบเทียบ เช่น if, for, while, do while เป็นต้น

ตารางที่ 2.4 ตัวดำเนินการเปรียบเทียบ

Operater	Example	Result
==	a == b	true if *a* equal to *b*, otherwise false
!=	a != b	true if *a* not equal to *b*, otherwise false
<	a < b	true if *a* less than *b*, otherwise false
>	a > b	true if *a* greater than *b*, otherwise false
<=	a <= b	true if *a* less than or equal to *b*, otherwise false
>=	a >= b	true if *a* greater than or equal to *b*, otherwise false

ตัวอย่างเช่น

```
if (n == 5) {
   cout << "n equal to 5";
}
```

จะเรียกว่า คำสั่งเปรียบเทียบเงื่อนไข ในตัวอย่าง เราใช้ตัวดำเนินการ == เพื่อตรวจสอบว่าตัวแปร n เท่ากับ 5 หรือไม่ ถ้ามันเท่ากับ 5 เราจะแสดงผล n equal to 5 ออกทางหน้าจอ

Logical operators (!, &&, ||)

ตัวดำเนินการตรรกะ ใช้เพื่อประเมิน expression ของตัวแปรซึ่ง expression สามารถประกอบไปด้วยตัวแปรเดียวหรือมากกว่า ตามตารางที่ 2.6

ตารางที่ 2.5 ตารางของตัวดำเนินการตรรกะ

Symbol	Name	Example
!	Not	! (n==1)
&&	And	a == 1 && b == 1
||	Or	a == 1 || b == 1

ดำเนินการตรรกะมักจะใช้เพื่อเปรียบเทียบและสร้างเงื่อนไข ผลลัพธ์สุดท้ายนั้นจะเป็น true หรือ false เท่านั้น ตัวอย่างเช่น

```
bool isRain = true;
if (!isRain) {
    cout << "I will go to beach" ;
}
```

เรามีตัวแปร *isRain* ซึ่งเป็นตัวแปรที่ใช้บ่งบอกว่าฝนตกหรือไม่ ในคำสั่ง if (!isRain) มีความหมายคือ "if it not rain" เราจะแสดงผลข้อความ "I will go to beach" ออกทางจอภาพ

```
char c = 'a';
if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u') {
    cout << "c is a vowel" ;
}
```

ในตัวอย่าง เรามีตัวแปร *c* ซึ่งสามารถเก็บค่าของตัวอักษรภาษาอังกฤษทุกตัวได้ และเราใช้ "or operator" เพื่อเช้คว่าถ้า *c* เป็นสระใดๆ เราจะแสดงผลออกทางจอภาพว่า "c is vowel" อย่างไรก็ตามคุณสามารถลองใช้ "and operator" ได้ด้วยตัวเอง

Conditional ternary operator (?)

Conditional ternary operator ถูกใช้เพื่อประเมิน expression โดยการใช้ตัวดำเนินการที่คุณได้เรียนไปก่อนหน้านี้แล้ว มันจะให้ผลลัพธ์เป็น true ถ้าผลลัพธ์ของ expression เป็นจริงและจะใช้ค่า false ถ้าไม่ โดยมีรูปแบบการใช้งานดังนี้

condition ? value1: value2

Conditional ternary operator นั้นง่ายต่อการใช้ มันเป็นรูปแบบย่อของการใช้คำสั่ง if เช่น

```
if (n%2 == 0) {
    cout << "n is even number.";
} else {
    cout << "n is odd number.";
}

// now using conditional ternary operator
cout << "n is " << (n % 2 == 0 ? "even" : "odd") << " number.";
```

Bitwise operators (&, |, ^, ~, <<, >>)

Bitwise operators จะดำเนินการกับในรูปแบบของหนึ่งบิตหรือมากกว่า หรือในตัวเลขฐานสอง

ตารางที่ 2.6 ตารางของตัวดำเนินการ Bitwise operators

Symbol	Description
&	Bitwise AND
|	Bitwise inclusive OR
^	Bitwise exclusive OR
~	bit inversion
<<	Shift bits left
>>	Shift bits right

Explicit type casting operator

ในภาษา C++ นั้นจะมีข้อมูลประเภทต่างๆ ข้อมูลบางประภทสามารถแปลงไปอีกประเภทได้โดยการใช้ explicit type casting ข้อมูลบางประเภทจำเป็นต้องใช้ฟังก์ชันจากไลบรารี่ของภาษา C++ หรือสามารถสร้างฟังก์ชันของคุณเองเพื่อแปลงข้อมูลเหล่านั้น ในตัวอย่างนี้เราจะใช้การแปลงข้อมูลแบบ explicit type casting เพื่อแปลงค่าข้อมูลพื้นฐานเช่น integer และ float

```
float b = 10.25;
int a = (int) b;
```

ในตัวอย่างนี้ เราได้แปลง 10.25 ไปเป็น integer ดังนั้นตัวแปร *a* จะมีค่าเป็น 10

2.11.5 Input และ Output

สิ่งที่สำคัญที่สุดในการเขียนโปรแกรมคือ Input และ output ของโปรแกรม เพราะว่าทุกๆ โปรแกรมจำเป็นต้องติดต่อกับผู้ใช้ ในภาษา C++ มันใช้ *stream* เพื่อทำการกับ input และ output ของโปรแกรม

Steam นั้นเป็นไลบรารี่มาตรฐานของภาษา C++ เราใช้ *cin*stream สำหรับการ input และ *cout* stream สำหรับ เพื่อใช้โดยไม่ต้องมี prefix std เราจำเป็นต้องใช้คำสั่งนี้ในโปรแกรมของเรา

```
using namespace std;
```

โดยทั่วไป การแสดงผลมักจะถูกแสดงผลออกทางหน้าจอของคอมพิวเตอร์ cout เป็น stream object ที่เราสามารถแสดงผลตัวอักษรใดๆ ออกทางจอภาพได้

```
#include <iostream>

using namespace std;

int main () {
   int myNumber = 5;
   cout << "This is my sentence." << endl;
   cout << 1234 << endl;
   cout << myNumber << endl;
   return 0;
}
```

จากตัวอย่าง เราได้แสดงตัวอักษรออกทางจอภาพในบรรทัดแรก ต่อมาเป็นตัวเลข และสุดท้ายเราได้แสดงค่าของตัวแปร *myNumber* ดังนั้น เราสามารถใช้ cout เพื่อแสดงผลตัวอักษร ตัวแปรหรือสิ่งต่างๆ ออกทางจอภาพได้ และข้างล่างนี้เป็นผลลัพธ์ของโปรแกรม

```
This is my sentence.
1234
5
```

ในโปรแกรม ได้เห็นโค้ดบางส่วนมาก่อนหน้าแล้ว เช่น endl ไลบรารี่มาตาฐานซึ่งมันหมายถึงการขึ้นบรรทัดใหม่ ซึ่งเหมือนกันกับการใช้ \n.

Standard input

แค่การแสดงผลนั้นยังไม่เพียงพอ ในภาษา C++ ยังมี cin stream object ซึ่งให้เราสามารถที่จะรับค่าจากคีบอร์ดได้ เรามักจะใช้ cin กับตัวแปร ดูตัวอย่างการใช้เช่น

```
#include <iostream>
#include <string>

using namespace std;

int main () {
   int age;
   string name;
   cout << "What is your name: ";
   cin << name;    // get name from keyboard
   cout << "Enter you age: ";
   cin << age;    // get age from keyboard
   cout << "Hi " << name ;
   cout << ", your age is " << age ;
}
```

หลังจากโปรแกรมรัน input อันแรกคือ cin << name มันจะรอรับค่าจากคีบอร์ดจนกว่าเราจะกดปุ่ม enter และนี่หมายถึงการรับค่าเสร็จสิ้น และข้อมูลจะถูกบันทึกลงในตัวแปนname สำหรับ input อันที่สองโปรแกรมที่จะทำแบบเดียวกัน แต่มันจะบันทึกข้อมูลลองในตัวแปร age แทน โดยผลลัพธ์ของโปรแกรมเป็นดังนี้

```
What is your name: Marcus
Enter you age: 18
Hi Marcus, your age is 18.
```

คุณยังสามารถกดปุ่ม space ได้สำหรับการรับค่าถัดไป ใน cin steam object การรับค่าจะจบเมื่อมันพบกับ enter key (new line) หรือ space key อย่างไรก็ตาม cin ยังให้เราสามารถรับค่า space key ได้โดยการใช้ฟังก์ชัน getline ซึ่งมากับไลบรารี่ของ string ตัวอย่างเช่น

```
string text;
getline(cin, text);
```

ฟังก์ชันนี้ทำให้เราสามารถที่จะทำการรับค่าของ space key ได้ เช่น ประโยค หรือคำหลายคำที่คั่นด้วยช่องว่าง

2.11.6 คำสั่งควบคุม

คำสั่ง If, if else, และ switch ถูกใช้เพื่อควบคุมโปรแกรมโดยมีเงื่อนไขเป็น expression คำสั่งวนซ้ำ เช่น for, while, และ do-while ถูกใช้เพื่อทำซ้ำส่วนของโค้ดตามเงื่อนไขของมัน

คำสัง If

คำสั่ง if ถูกใช้เพื่อควบคุมโปรแกรมกับเงื่อนไขที่กำหนด โค้ดในบล็อคของ คำสั่ง If จะทำงานถ้าเงื่อนไขตรงหรือเป็นจริง

```
int n = 10;
if (n == 10) {
    cout << "n is 10";
}
```

ในตัวอย่างนี้ ได้ใช้ if เพื่อตรวจสอบว่าตัวแปร n เท่ากับ 10 หรือไม่ ถ้ามันตรงกับเงื่อนไข โปรแกรมจะทำงานในบล็อคของ If คือ cout << "n is 10".

คำสั่ง If else

คำสั่ง If else นั้นคล้ายกับคำสั่ง if คำสั่ง Else เพื่อทำเงื่อนไขที่นอกเหลือจากเงื่อนไขอื่นทั้งหมด และมันจะต้องเริ่มต้นด้วยคำสั่ง if เสมอ

```
int m = -1;
if (n < 0) {
    cout << "Negative number.";
}
else if (n > 0) {
    cout << "Positive number.";
}
else {
   cout << "Zero number";
}
```

ในตัวอย่าง คำสั่ง If สามารถมีเงื่อไนได้หลายอันดดยการใช้ if else () เงื่อนไขสุดท้ายคือ else ซึ่งมันจะทำงานเมื่อไม่ตรงกับเงื่อนไขใดๆก่อนหน้า จากโค้ดด้านบน เรามีตัวแปร *m* ซึ่งมีข้อมูลเป็นแบบ integer โปรแกรมของเรานั้นจะตรวจสอบว่า *m* เป็นจำนวนเต็มบวก เต็มลบ หรือศูนย์ ยิ่งไปกว่านั้น expression สามารถมีได้หลายเงื่อนไข โดยการใช้ตัวดำเนินการตรรกะ

```
int a = 12;
int b = 5;
if (a > 10 && b % 2 == 0) {
    cout << "a is greater than 10 and b is even number.";
}
 else {
```

```
    cout << "Other condition";
}
```

ในตัวอย่างนั้นใช้หลายเงื่อนไขและผลลัพธ์ของโปรแกรมจะเป็น "Other condition" เพราะว่า *a* นั้นมากกว่า 10 แต่ *b* นั้นไม่เป็นจำนวนคู่ คุณสามารถเพิ่มเงื่อนไขได้อีกตามที่คุณต้องการ

คำสั่ง Switch case

คำสั่ง switch-case นั้นคล้ายกับ คำสั่ง If-else เป้าหมายของมันเพื่อตรวจสอบกับค่าคงที่ นี่เป็นตัวอย่างการใช้คำสั่ง switch

```
switch (n) {
   case 1:
      cout << "n is 1";
      break;
   case 2:
      cout << "n is 2";
      break;
  default:
      cout << "Unknown n";
}
```

Switch expression สามารถมีได้แค่หนึ่งค่าเพื่อประเมิน คำสั่ง case เป็นคำสั่งเงื่อนไขเพื่อเปรียบเทียบค่า ในตัวอย่างด้านบน case 1: จะทำงานเมื่อ n มีค่าเท่ากับ 1 หลังจากคำสั่งด้านล่างเราต้องใส่คำสั่ง break เพื่อหยุดสำหรับแต่ละ case คำสั่ง default นั้นเป็นทางเลือกเมื่อโปรแกรมไม่ตรงกับเงื่อนไขใดๆ ก่อนหน้า สามารถถูกเขียนโดยการใช้คำสั่ง if-else ได้ดังด้านล่างนี้

```
#include <iostream>
using namespace std;
```

```
int main ()
{
    if (n == 1) {
        cout << "n is 1";
    }
    else if (n == 2) {
        cout << "n is 2";
    }
    else {
        cout << "Unknown n";
    }
    return 0;
}
```

คำสั่ง while loop

ลูปที่ง่ายและพื้นฐานที่สุดในภาษา C++ นั้นคือ while loop ซึ่งมีรูปแบบการใช้งานคือ

```
while (expression) {
    statements
}
```

คำสั่ง while-loop ใช้เพื่อทำสั่งโค้ดของโปรแกรมในขณะที่ expression เป็นจริง true และมันจะสิ้นสุดการทำงานเมื่อ expression ไม่เท็จและออกจาก while-loop และทำคำสั่งอื่นต่อไป

```
#include <iostream>
using namespace std;

int main ()
{
    int n = 1;
    while (n <= 10) {
```

```
        cout << n << ",";
        n++;
    }
    cout << " end loop";
    return 0;
}
```

ในตัวอย่าง โปรแกรมจะนับจาก 1 ถึง 10 เราได้ประกาศตัวแปร n และกำหนดค่าให้เป็น 1 ก่อนที่มันจะเข้าไปในทำงานใน while-loop while-loop จะทำการตรวจสอบ expression และเข้าสู่ถ้าเงื่อนไขยังคงเป็นจริง; และแสดงค่า n ออกทางจอภาพและเพิ่มค่า n ขึ้น 1 จนกว่า nจะเพิ่มไปถึง 10 ซึ่งจะทำให้ expression และโปรแกรมจะออกจาก loop และทำสั่งอื่นต่อไป และผลลัพธ์เมื่อรันโปรแกรมจะเป็นดังนี้

```
1,2,3,4,5,6,7,8,9,10, end loop
```

คำสั่ง do-while loop

ลูปที่คล้ายกับ while-loop คือ do-while ลูป มันมีรูปแบบดังนี้

```
do {
    statements
} while (condition);
```

มันทำงานเหมือน while loop ยกเว้นในการเปรียบเทียบเงื่อนไขจะทำตอนท้ายหลังจากสิ้นสุดคำสั่งในลูป นั่นหมายความว่า do-while loop จะต้องทำงานอย่างน้อยหนึ่งรอบแน่นอน มันมักจะใช้กับโปรแกรมที่จำเป็นต้องรับค่าจากผู้ใช้ก่อนที่จะทำอย่างอื่นต่อไป ดังตัวอย่าง

```
#include <iostream>
using namespace std;

int main ()
{
    char ch;
```

```
    do {
        cout << "Enter 'n' to exit loop: ";
        cin >> ch;
    } while (ch != 'n');
    cout << "Exit from the loop.";
    return 0;
}
```

โปรแกรมข้างบนต้องการรับค่าจากผู้ใช้ expression ของมันต้องการตัวอักษร 'n' character เพื่อออกจากลูป นี่คือผลลัพธ์ของโปรแกรมเมื่อได้ทดสอบ คุณสามารถลองดูได้เช่นกัน

```
Enter 'n' to exit loop: a
Enter 'n' to exit loop: b
Enter 'n' to exit loop: c
Enter 'n' to exit loop: n
Exit from the loop.
```

คำสั่ง for loop

for loop เป็นลูปที่มีการวนรอบเป็นจำนวนที่แน่นอน รูปแบบของมันคือ

```
for (initialize; condition; increase) {
    statements
}
```

for loop เป็นลูปที่สามารถวนรอบตามตัวเลขที่กำหนดได้ มันทำงานเหมือน while-loop มันจะวนซ้ำจนกว่า expressionจะเป็นเท็จ นอกจากนั้น เรายังสามารถประกาศตัวแปรเริ่มต้น สร้าง expression เพิ่มและลดค่าก่อนที่ลูปจะเริ่ม ตัวอย่างการนับตัวเลขโดยการใช้ for loop เช่น

```
#include <iostream>
using namespace std;
```

```
int main ()
{
    for (int n = 1; n <= 10; n++) {
        cout << n << ",";
    }
    cout << " end loop";
    return 0;
}
```

และนี่เป็นผลลัพธ์เมื่อเรารันโปรแกรม ซึ่งมันเป็นโปรแกรมเดียวกันกับตัวอย่างของ while-loop ก่อนหน้า

```
1,2,3,4,5,6,7,8,9,10, end loop
```

คำสั่ง break

คำสั่ง break เพื่อจบลูปในทันที และมันไม่สนใจว่า expression จะเป็นจริงหรือไม่

```
#include <iostream>
using namespace std;

int main ()
{
    for (int n = 1; n <= 10; n++) {
        if(n == 5) break;
        cout << n << ",";
    }
    cout << " end loop";
    return 0;
}
```

จากตัวอย่างข้างบน โปรแกรมจะออกจากลูปเมื่อ n มีค่าเท่ากับ 5 คำสั่ง break สามารถใช้กับลูป เช่น for, while, do-while, switch และอื่นๆ และผลลัพธ์การทำงานจะเป็นดังนี้

```
1,2,3,4, end loop
```

คำสั่ง continue

ไม่เหมือนคำสั่ง break คำสั่ง continue ถูกใช้เพื่อข้ามการทำงานในรอบปัจจุบัน ซึ่งจะไม่ทำคำสั่งหลังจากมันและไปเริ่มรอบถัดไป

```
#include <iostream>
using namespace std;

int main ()
{
   for (int n = 1; n <= 10; n++) {
      if(n % 2 == 0) continue;
      cout << n << ",";
   }
   cout << " end loop";
   return 0;
}
```

โปรแกรมจะข้ามลูปถ้า n เป็นตัวเลขคู่ ซึ่งผลลัพธ์ได้ดังนี้

```
1, 3, 5, 7, 9, end loop
```

บทที่ 3

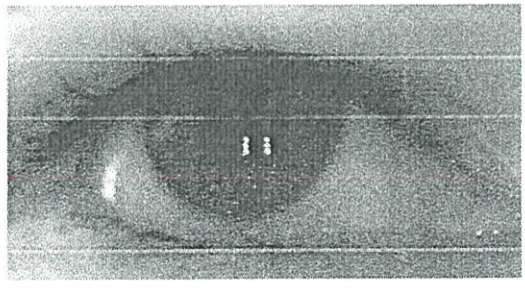
ระเบียบวิธีวิจัยขั้นตอนการดำเนินงานแผนการดำเนินงาน

3.1 การทำงานของระบบ

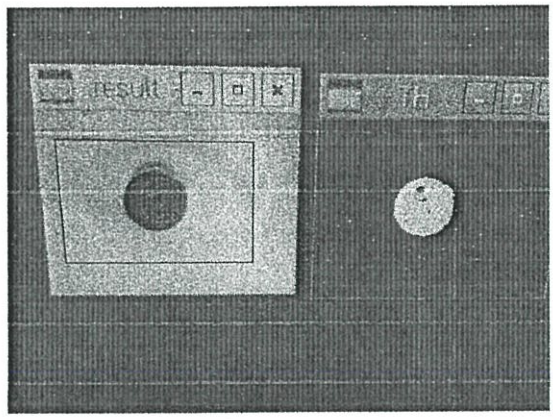
ในบทนี้กล่าวถึงขั้นตอนในการศึกษาวิจัย เพื่อออกแบบ Algorithm สำหรับการใช้Eye-tracking ควบคุม wheelchair โดยใช้ใช้การประมวลผลภาพ ซึ่งภาพรวมของ Algorithm สามารถแบ่งออกเป็น 3 ขั้นตอนหลักๆ ดังภาพ 3.1 กระบวนการเริ่มต้นสำหรับAlhorithm คือ การปรับปรุงภาพบืองต้น (Preprocessing Process) ได้แค่การเปลี่ยนภาพ RGB ให้เป็น Gray scale แล้วทำการ Thresholding การกำจัดสัญญาณรบกวนภาพ ส่วนถัดมาคือ การDetect วงกลมโดยใช้ hough transform และในส่วนสุดท้ายจะเป็นการส่งตำแหน่งของตาที่Detect ได้ไปควบคุม Wheelchair จอยสติ๊ก

รูปที่ 3.1 แสดงระบบการทำงานของงานวิจัยนี้

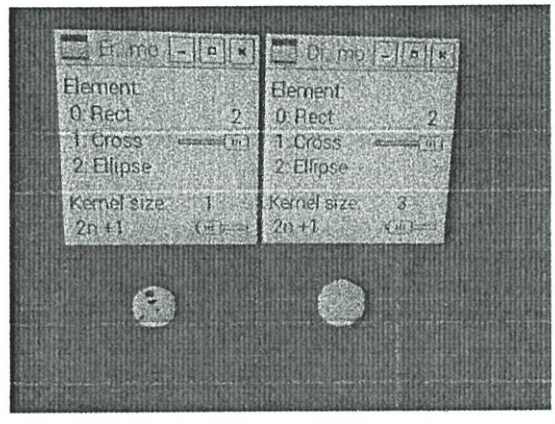
กล้องที่ใช้ในการถ่ายภาพตานั้นจะใช้ Infrared camera เนื่องจากว่า สภาวะแสงจากสิ่งแวดล้อมจะไม่มีผลต่อระบบ จากนั้นนำภาพที่จากกล้องอินฟาเรด เข้าไปประมวลผล การประมวลผลภาพบน ราสเบอรี่พาย (Raspberry Pi)



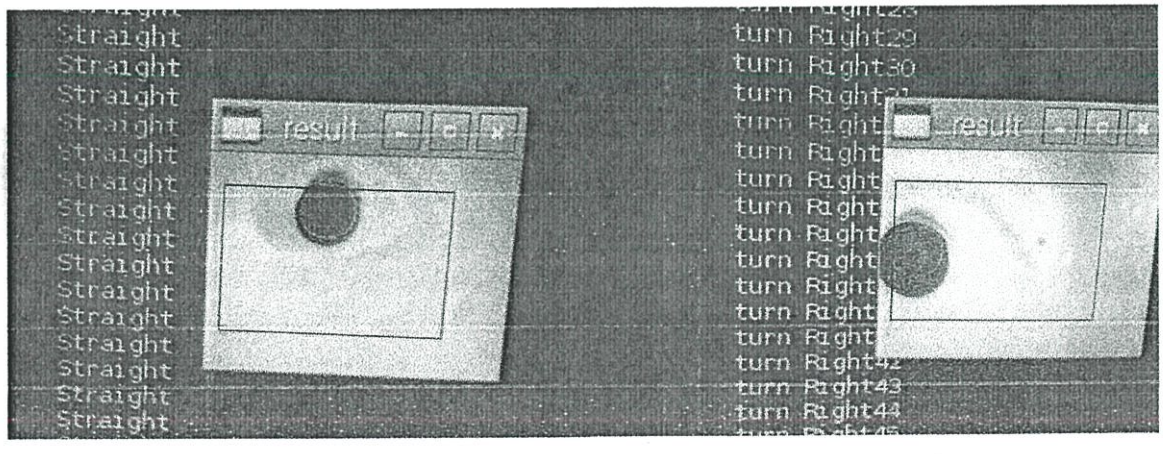
รูปที่ 3.2 ภาพตาที่ได้จากกล้อง Infrared



รูปที่ 3.3 ภาพที่ผ่านกระบวนการPre-processing และทำการ Threshold

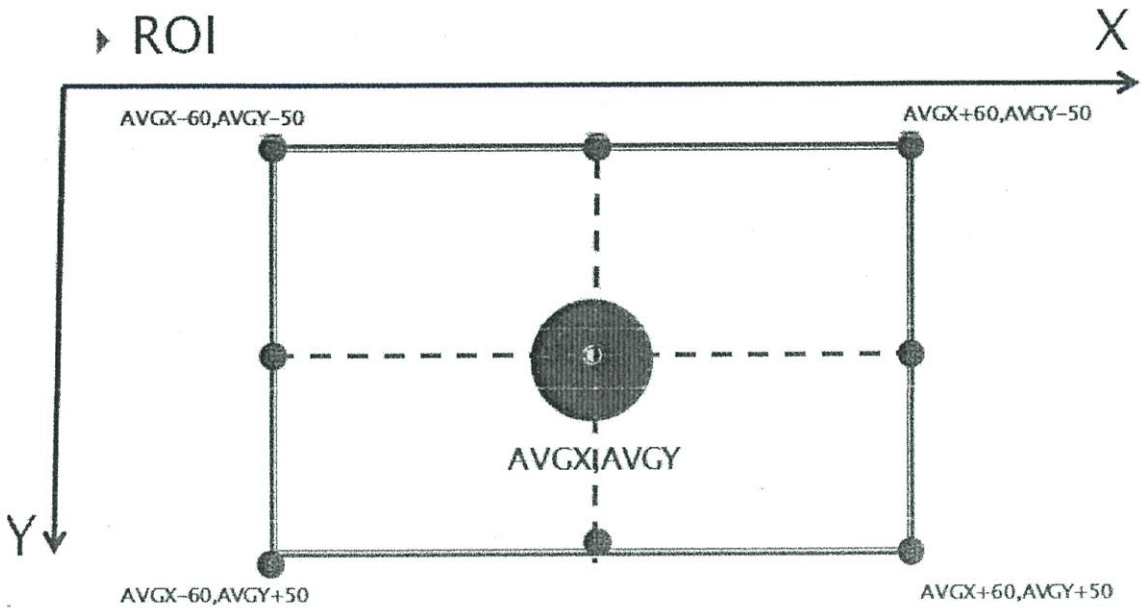


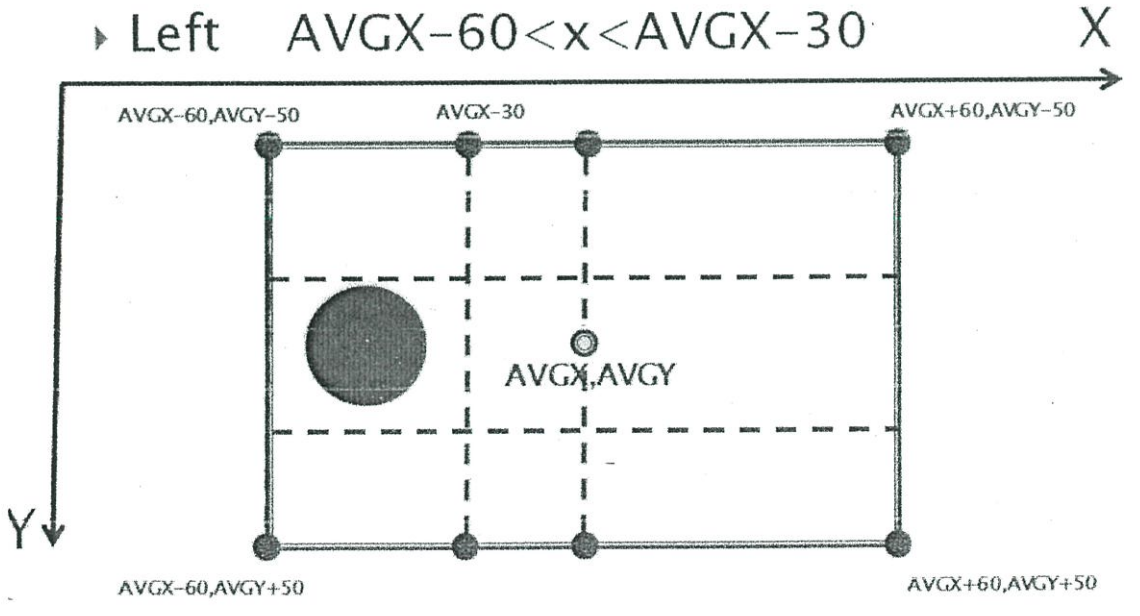
รูปที่ 3.4 ภาพที่ผ่านกระบวนการMorphology



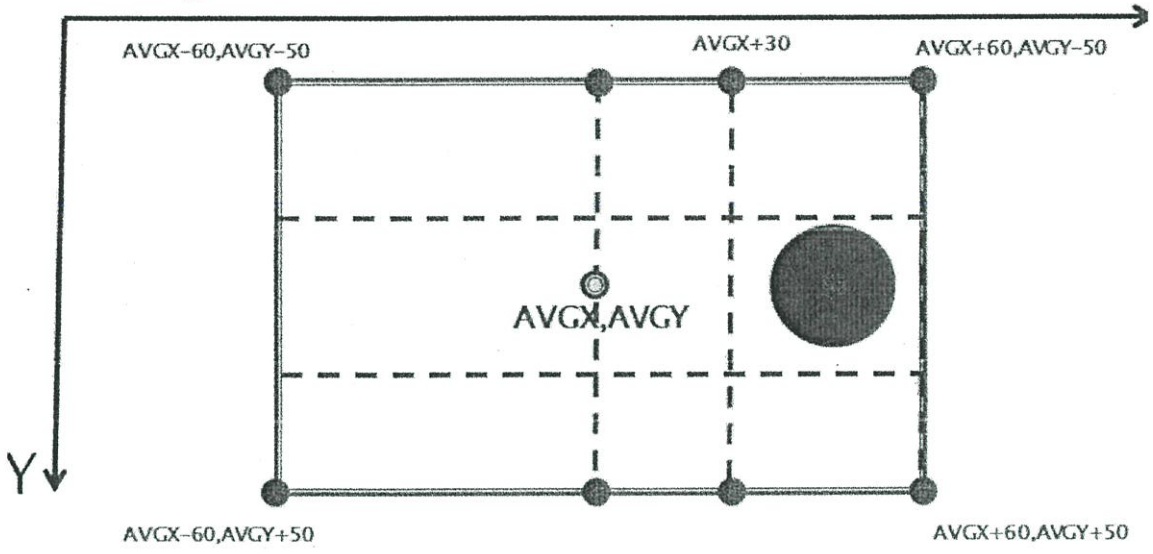
รูปที่ 3.5 ภาพแสดงการใช้ตำแหน่งพิกัดของตาเพื่อควบคุมมอเตอร์

3.2 ตำแหน่งของตาสำหรับการควบคุมจอยสติ๊ก

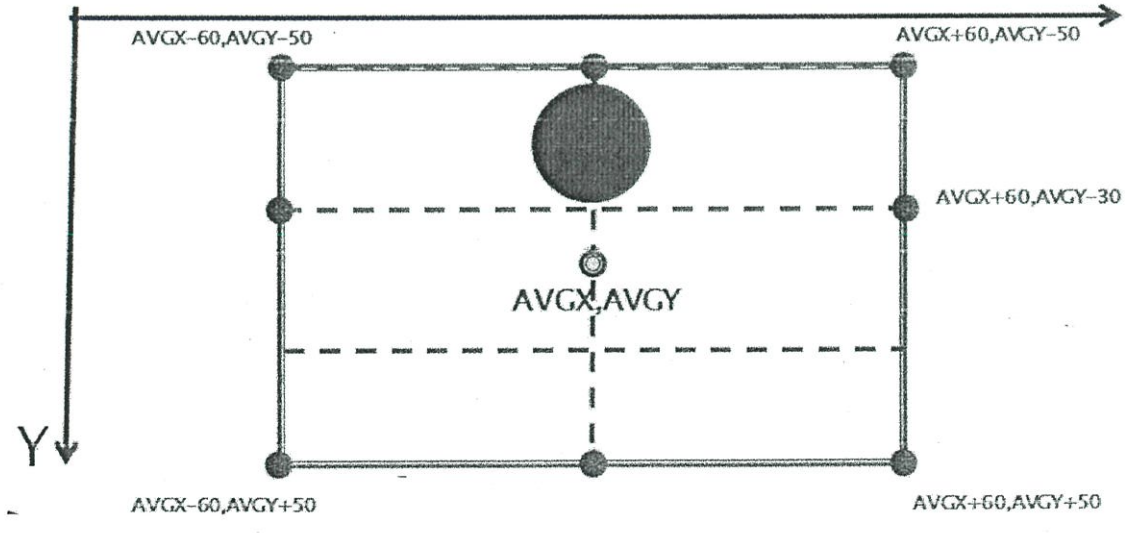


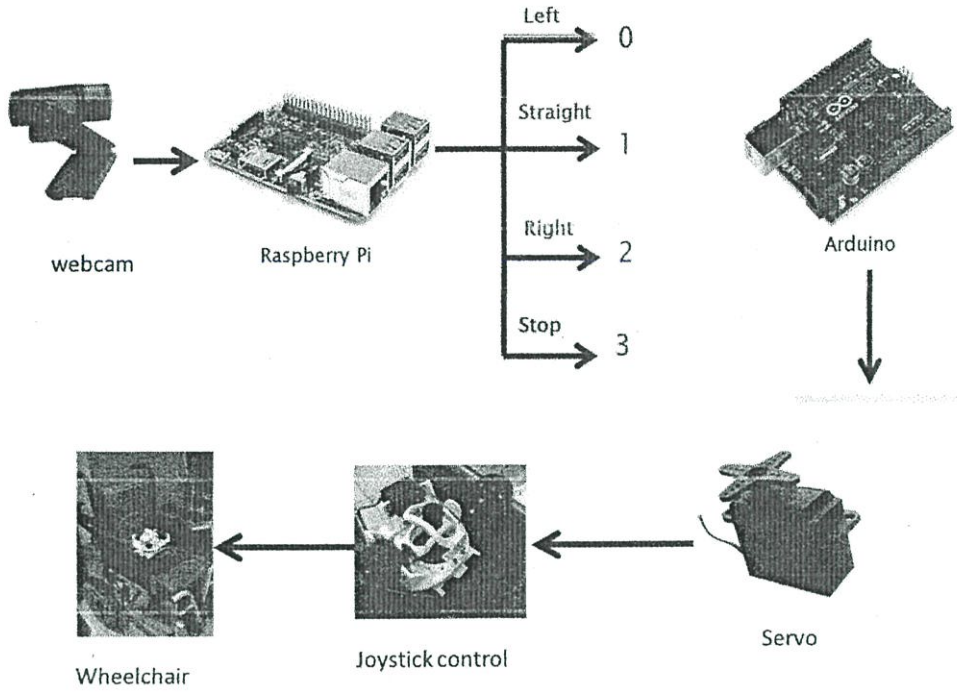


- Right AVGX+30<x<AVGX+60 X



- Straight AVGY-30<Y<AVGY-50 X





รูปที่ 3.6 แผนภาพการทำงานโดยรวมทั้งหมด

3.3 เครื่องมือที่ใช้ในการทดลอง

3.2.1 ขาตั้งกล้อง

3.2.1 วัตถุทรงกลมสีดำขนาดใกล้เคียงกับ Pupil

3.2.3 ราสเบอรี่พาย(Raspberry Pi) สำหรับการแสดงผล, การประมวลผล , การวิเคราะห์

3.2.4 บอร์ด Arduino พร้อม เซอร์โว(Servo) สำหรับควบคุมwheelchair จอยสติ๊ก

3.2.5 Wheelchair ไฟฟ้า

3.4 ขั้นตอนการดำเนินงาน

3.3.1 การทดลองเก็บข้อมูล ตรวจสอบการทำงานของระบบที่สร้างขึ้น โดยDetect วัตถุทรงกลม ทดสอบความถูกต้องในการระบุพิกัดออกมา

3.3.2 การทดสอบการสื่อสาระว่าง ราสเบอรี่พาย(Raspberry Pi) และ Arduino ผ่าน Serial Port

3.3.3 การทดสอบโดยการ Detect ตาจริง

3.5 แผนการดำเนินงาน

ตารางที่ 3.1 ตารางแผนการดำเนินงาน

การดำเนินงาน	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.
1. การเลือกหัวข้อศึกษา และค้นคว้างานวิจัยที่เกี่ยวข้อง	/	/							
2. การออกแบบด้าน Software		/	/						
3. การออกแบบด้าน Hardware		/	/	/					
4. การพัฒนาระบบทั้งด้าน Hardware และ Software			/	/	/	/			
5. การทดลองเก็บข้อมูล						/	/		
6. การจัดเก็บผลการทดลองจากคนไข้จริง							/	/	
7. การวิเคราะห์ผลและการแก้ไข								/	
8. การจัดทำเล่มวิทยานิพนธ์									/

บทที่ 4

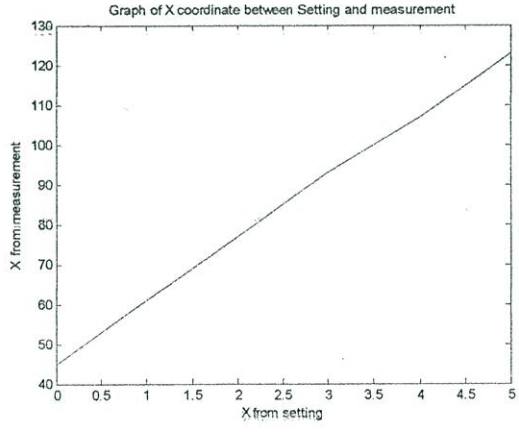
การทดลองและผลการทดลอง

ในบทนี้จะอธิบายถึงขั้นตอนการทดลอง เพื่อนำไปสู่การแสดงผลการทดลองของอัลกอลิทึมสำหรับการดี Detect ตำแหน่งของตาโดยใช้ Hough transform แล้วนำ Position ไปควบคุม wheelchair สำหรับการทดลองนั้นจะใช้โปรแกรมคอมพิวเตอร์ plot graph วงกลมโดยให้รัศมีคงที่ จากนั้นเปลี่ยนจุดศูนย์กลางของวงกลมโดยนำกล้องถ่ายภาพของวงกลมที่เปลี่ยนไปแล้วนำไปประมวลผลในโปรแกรม Eye-tracking อ่านค่าของจุดศูนย์กลางทีเปลี่ยนไปว่าเป็น linear system หรือไม่

การทดลอง 4.1 จุดศูนย์กลางของวงกลมเปลี่ยนไปตามแนวแกน x

ตารางที่ 4.1 แสดงตำแหน่งที่เปลี่ยนของวงกลมตามแนวแกน x (y=0 or x-axis)

	cos(t)+x1,sin(t)+y1 (Setting)	x2,y2 (Measurement)
A	0,0	45,97
B	1,0	61,97
C	2,0	77,99
D	3,0	93,93
E	4,0	107,99
F	5,0	123,99

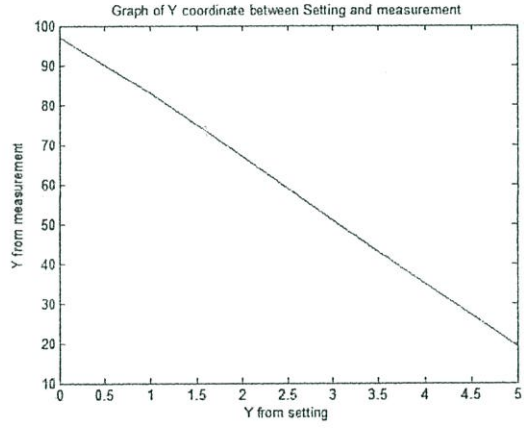


รูปที่ 4.1 กราฟของเปลี่ยนตำแหน่งตามแนวแกน x

การทดลองที่ 4.2 จุดศูนย์กลางของวงกลมเปลี่ยนไปตามแนวแกน y

ตารางที่ 4.2 แสดงตำแหน่งที่เปลี่ยนของวงกลมตามแนวแกน y (x=0 or y-axis)

	cos(t)+x1,sin(t)+y1 (Setting)	x2,y2 (Measurement)
A	0,0	45,97
B	0,1	45,83
C	0,2	43,67
D	0,3	43,51
E	0,4	43,35
F	0,5	43,19

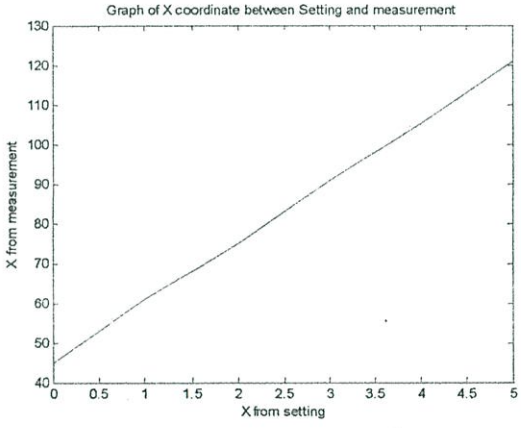


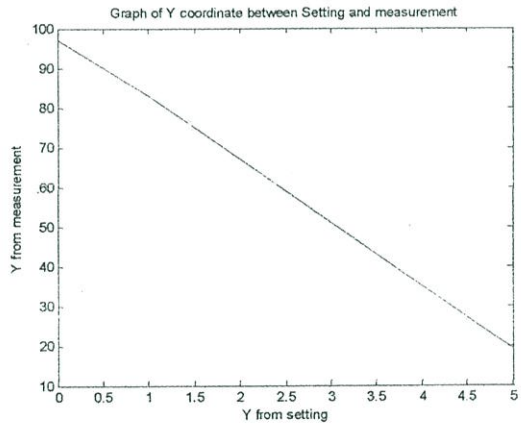
รูปที่ 4.2 กราฟของเปลี่ยนตำแหน่งตามแนวแกน y

การทดลองที่ 4.3 จุดศูนย์กลางของวงกลมเปลี่ยนไปตามแนวเส้นตรง y=x

ตารางที่ 4.3 แสดงตำแหน่งที่เปลี่ยนของวงกลมตามเส้นตรง y=x

	cos(t)+x1,sin(t)+y1 (Setting)	x2,y2 (Measurement)
A	0,0	45,97
B	1,1	61,83
C	2,2	75,67
D	3,3	91,51
E	4,4	105,35
F	5,5	121,19

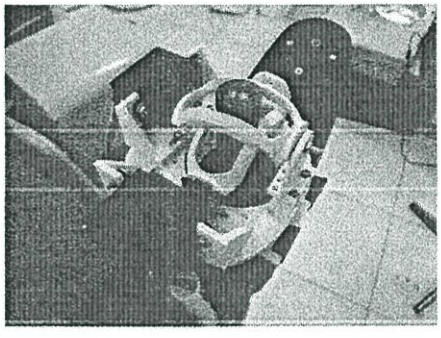




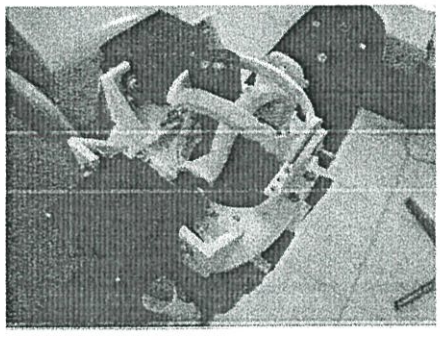
รูปที่ 4.3 กราฟของเปลี่ยนตำแหน่งตามแนวเส้นตรง y=x

การทดลองที่ 4.4 การทดสอบการใช้งานกับฮาร์ดแวร์

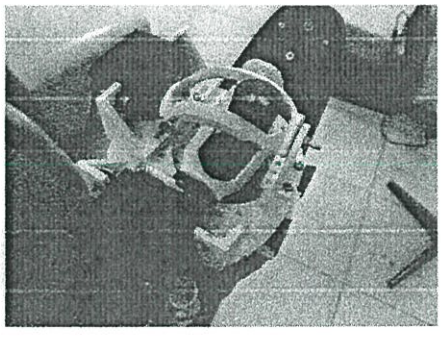
ผลการทดลองการใช้งานกับ วีลแชร์ พบว่าจอยสติ๊กสามารถตอบสนองต่อโปรแกรมได้ดีสามารถควบคุมวีลแชร์ได้อย่างมีประสิทธิภาพ ในตำแหน่งที่ตาอยู่ตรงกลางภาพ หรือมองตรง เซอร์โว ทั้งสองตัวจะไม่มีการเปลี่ยนแปลงทำให้ วีลแชร์หยุดอยู่กับที่ ดังรูปที่ 4.4 แต่เมื่อกรอกตาไปทางขวาจะทำให้ เซอร์โวตัวล่างขยับจอยสติ๊กไปทางขวา ดังรูปที่ 4.5 เมื่อกรอกตาไปทางซ้ายจะทำให้เซอร์โวตัวล่างขยับไปจอยสติ๊กไปทางซ้าย ดังรูปที่4.6 และเมื่อกรอกตาขึ้นด้านบนวีลแชร์ตัวบนจะขยับจอยสติ๊กไปข้างหน้า ดังรูปที่4.7



รูปที่ 4.4 ตำแหน่งจอยสติ๊กตรงกลาง



รูปที่ 4.5 จอยสติ๊กขยับไปด้านซ้าย



รูปที่ 4.6 จอยสติ๊กขยับไปด้านขวา

รูปที่ 4.7 จอยสติ๊กขยับไปด้านหน้า

บทที่ 5

สรุปผลการวิจัยและข้อเสนอแนะ

จากผลการทดลองที่ 4.1 4.2 และ4.3 จะพบว่าการเปลี่ยนไปของ Position ของวงกลมนั้นมีความเป็น Linearและมีความแน่ยำสูง ซึ่งจากการทดลองใช้กับระบบ Hardware พบว่า เซอร์โว(Servo) สามารถ Response ต่อการเปลี่ยนตำแหน่งได้อย่างดี ทำให้ควบคุม เซอร์โว(Servo) ควบคุม จอยสติ๊ก ของ wheelchair ได้แต่เนื่องจากว่าแว่นที่ติดกล้องนั้นยังมีนำหนักมากอยู่ทำให้สวมใส่ไม่สะบาย ดังนั้นจึงจำเป็นต้องพัฒนาปรับปรุงใช้กล้องที่มีขนาดเล็กลง

เอกสารอ้างอิง

[1] รศ.ดร.ชูชาติ ปิณฑวิรุน์. Digital Signal Processing with C++ สถานบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

[2] Michael J. Spivey, Daniel C. Richardson. Eye tracking : Characteristics and Methods and Research Areas and Applications Department of Psychology, Cornell University

[3]Klaus D. Toennies, Fracnk Behrens, Melanie Aurnhammer. Feasibility of Hough-Transform-based Iris Localisation for Real Time-Applictaion Otto-von-Guericke Universität Magdeburg, Germany