การควบคุมตำแหน่งและแนวการวางตัวของแขนหุ่นยนต์ 2 แกน บนพาหนะขณะเคลื่อนที่

นางสาวผกาสินี สิงห์เจริญกิจ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมเมคคาทรอนิกส์
มหาวิทยาลัยเทคโนโลยีสุรนารี
ปีการศึกษา 2563

POSITION AND ORIENTATION CONTROL OF ROBOT ARMS 2 AXIS ON MOVING VEHICLE

Phakasinee Singcharoenkit

A Thesis Submitted in Partial Fulfillment of the Requirement for the

Degree of Master of Engineering in Mechatronics Engineering

Suranaree University of Technology

Academic Year 2020

การควบคุมตำแหน่งและแนวการวางตัวของแขนหุ่นยนต์ 2 แกน บนพาหนะขณะเคลื่อนที่

มหาวิทยาลัยเทคโนโลยีสุรนารี อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

คณะกรรมการสอบวิทยานิพนธ์

(ผศ. ดร.โศรฎา แข็งการ)
ประธานกรรมการ

(รศ. ร.อ. ดร.กนต์ธร ชำนิประศาสน์)
กรรมการ (อาจารย์ที่ปรึกษาวิทยานิพนธ์)

(อ. ดร.พิจิตรา เอื้องไพโรจน์)
กรรมการ

(ดร.สุพัฒน์ กลิ่นเขียว)
กรรมการ

(รศ. ดร.ฉัตรชัย โชติษฐยางกูร)
รักษาการแทนรองอธิการบดีฝ่ายวิชาการ
และประกันคุณภาพ

(รศ. ดร.พรศิริ จงกล)
คณบดีสำนักวิชาวิศวกรรมศาสตร์

ผกาสินี สิงห์เจริญกิจ : การควบคุมตำแหน่งและแนวการวางตัวของแขนหุ่นยนต์ 2 แกน บนพาหนะขณะเคลื่อนที่ (POSITION AND ORIENTATION CONTROL OF ROBOT ARMS 2 AXIS ON MOVING VEHICLE) อาจารย์ที่ปรึกษา : รองศาสตราจารย์ เรืออากาศเอก ดร.กนต์ธร ชำนิประศาสน์, 69 หน้า.

งานวิจัยนี้มีวัตถุประสงค์เกี่ยวกับการออกแบบการควบคุมตำแหน่งและแนวการวางตัวของแขนหุ่นยนต์ 2 แกน บนพาหนะขณะเคลื่อนที่ ในงานวิจัยนี้ผู้วิจัยได้ทำการออกแบบและสร้างระบบควบคุมตำแหน่งและแนวการวางตัวของแขนหุ่นยนต์ 2 แกนบนหุ่นยนต์เคลื่อนที่ แขนของหุ่นยนต์ประกอบด้วย Stepper Motor ที่สามารถเคลื่อนที่ในระยะ 0 – 359 องศา และแกนที่ 2 ถูกควบคุมด้วย Servo Motor ที่สามารถเคลื่อนที่ขึ้นลง ในช่วงมุม 0 – 90 องศา ที่ปลายของแขนหุ่นยนต์จะมีกล้อง Raspberry Pi เวอร์ชัน 2 ติดอยู่เพื่อใช้สำหรับตรวจจับภาพของวัตถุ แล้วนำไปประมวลผลบนคอมพิวเตอร์ขนาดเล็ก Raspberry Pi4 โมเดล B สำหรับการตรวจจับวัตถุ ผู้วิจัยได้นำการเรียนรู้เชิงลึกมาประยุกต์ใช้หลังจากสามารถระบุวัตถุที่สนใจได้แล้ว เมื่อวัตถุมีการเคลื่อนไหว หุ่นยนต์จะปรับแขนกลติดตามเพื่อให้วัตถุอยู่ตรงกลางของภาพจากกล้องเสมอ วิธีที่ใช้ประเมินผลและประสิทธิภาพการติดตามวัตถุคือการหาค่าการทับซ้อนของพื้นที่ (IoU) และค่าความมั่นใจ (Confidence level) ของภาพที่ตรวจจับได้ จากผลการทดสอบพบว่า ค่าเฉลี่ยของ IoU และค่าความมั่นใจเป็น 0.542 และ 0.942 ตามลำดับ ในขนาดภาพ 205 × 154 พิกเซล

สาขาวิชา วิศวกรรมเมคคาทรอนิกส์ ลายมือชื่อนักศึกษา _____

ปีการศึกษา 2563 ลายมือชื่ออาจารย์ที่ปรึกษา _____

PHAKASINEE SINGCHAROENKIT : POSITION AND ORIENTATION CONTROL OF ROBOT ARMS 2 AXIS ON MOVING VEHICLE. THESIS ADVISOR : ASSOC. PROF. FLT. LT. KONTORN CHAMNIPRASART, Ph.D., 69 PP.

TEACHER TRACKING/OBJECT TRACKING/PAN AND TILT

This research aims to design mechanical components to control the position and orientation of the mobile manipulator arm. Mechanism and control systems are designed for target acquisition tasks. The manipulator consists of two degrees of freedom which are located on the four-wheel mobile robot. A robot can turn its pan-angle with a range of 0 – 359 degrees using a stepper motor and for the tilt-angle with a range of 0 – 90 degrees by DC servo motor. Pi-camera V2 is integrated at the end of the end-link of a manipulator. The target image will be capture and will be sent to Raspberry Pi 4 Model B. Deep learning was selected to identify the desired target. The result from the image processing unit shows the pan and tilt angle for manipulator moves. The mission is to center the desired target at the center at all times. Intersection Over Union (IoU) and confidence level are chosen to determine the mathematical efficiency of this robot. The average IoU is 0.542 with a confidence level of 0.942 at a resolution of 205×154 pixels.

School of Mechatronics Engineering Student's Signature

Academic year 2020 Advisor's Signature

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมเมคคาทรอนิกส์ ผู้จัดทำได้รับการอนุเคราะห์จากบุคคลหลายฝ่ายที่ให้คำปรึกษา การแนะนำแนวคิดการทำวิจัยและการช่วยเหลืออย่างดี ดังนี้

รองศาสตราจารย์ เรืออากาศเอก ดร.กนต์ธร ชำนิประศาสน์ รองอธิการบดีฝ่ายวิชาการ และพัฒนาความเป็นสากล และยังเป็นอาจารย์ที่ปรึกษา ที่ได้มอบโอกาสทางการศึกษาระดับ บัณฑิตศึกษา อีกทั้งยังให้ความรู้ด้านวิชาการ คอยให้คำแนะนำและการสนับสนุนในการทำงานมา โดยตลอด

ผู้ช่วยศาสตราจารย์ ดร.โสรฎา แข็งการ, อาจารย์ ดร.พิจิตรา เอื้องไพโรจน์ และ ดร.สุพัฒน์ กลิ่นเขียว ที่ได้เสียสละเวลาให้ความรู้และให้คำปรึกษาแนะนำแนวทางการทำวิจัย รวมทั้ง ตรวจสอบแก้ไขให้งานวิจัยมีความถูกต้องและสมบูรณ์มากยิ่งขึ้น ทำให้งานวิจัยสำเร็จลุล่วง ตามวัตถุประสงค์ทุกประการ

อาจารย์ ดร.จิตติมา วระกุล, อาจารย์อภิลักษณ์ หล่อนกลาง, นายแสณภูมิ ทรงไตรย์ และ นายภูวนาถ เผือกทอง ที่ได้ช่วยแนะนำการเขียนโปรแกรมควบคุม การออกแบบโปรแกรมและ ออกแบบตัวหุ่นยนต์ เพื่อให้ได้ตามเป้าหมายที่กำหนดไว้

ขอขอบคุณ สถาบันวิจัยแสงซินโครตรอน (องค์การมหาชน) ที่มอบทุนการศึกษาระดับ บัณฑิตศึกษา และบุคลากรทุกท่านที่ให้ความรู้และคำแนะนำในการทำวิจัย และบุคคลอื่น ๆ ที่ไม่ได้กล่าวนามทุกท่านที่ให้คำแนะนำ และช่วยเหลือในเรื่องต่าง ๆ ที่เป็นประโยชน์และสามารถ ทำให้งานวิจัยนี้สำเร็จลุล่วงไปด้วยดี ทำ ให้ได้รับความรู้ ประสบการณ์และทักษะต่าง ๆ ใน การทำงานวิจัย

ขอกราบขอบพระคุณ บิดา มารดา ที่ให้การอบรมเลี้ยงดูสั่งสอน สนับสนุนทางการศึกษา รวมทั้งให้กำลังใจเป็นอย่างดีมาตลอด จนทำให้ผู้วิจัยประสบความสำเร็จเสมอมา

สุดท้ายนี้ ขออาราธนาสิ่งศักดิ์สิทธิ์ทั้งหลาย จงดลบันดาลให้บุคคลทั้งหลายที่ได้กล่าวนาม และไม่ได้กล่าวนาม จงมีแต่ความสุขและความเจริญในชีวิตตลอดไปเรื่อยมา

ผกาสินี สิงห์เจริญกิจ

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

ตารางที่ **หน้า**

สารบัญรูป

รูปที่ **หน้า**

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหาการวิจัย

หุ่นยนต์มีวิวัฒนาการและความก้าวหน้าอย่างรวดเร็วต่อเนื่องมาตลอดหลายปีที่ผ่านมา โดยได้เข้ามามีบทบาทมากขึ้นในชีวิตของมนุษย์ ทั้งในด้านการช่วยเพิ่มผลผลิตในกระบวนการผลิตสินค้า ช่วยดูแลในเรื่องคุณภาพชีวิต ไปจนถึงการสร้างความสะดวกสบายต่างๆ สำหรับการใช้งานหุ่นยนต์และระบบอัตโนมัติในประเทศไทยมีแนวโน้มเพิ่มขึ้นอย่างต่อเนื่อง จากการนำระบบควบคุมไปใช้ในงานอุตสาหกรรม ซึ่งระบบมีความซับซ้อนขึ้นมาก ส่วนใหญ่เป็นระบบที่ไม่มีเสถียรภาพ เช่น การควบคุมเครื่องจักรอัตโนมัติ การควบคุมมอเตอร์ การควบคุมแขนหุ่นยนต์ การควบคุมเครื่องจักรซีเอ็นซี (CNC) ในด้านอุตสาหกรรม ดังนั้นในปัจจุบันจึงเกิดการทฤษฎีการควบคุมขึ้นมากมายเพื่อสามารถนำไปพัฒนาและควบคุมอุปกรณ์ได้อย่างมีประสิทธิภาพ

แขนหุ่นยนต์หรือแขนกล ถูกสร้างขึ้นมาเพื่อทำงานแทนมนุษย์ในงานหลายๆด้าน เช่น งานที่ต้องอยู่ในสภาพแวดล้อมที่อันตราย งานที่ต้องซ้ำๆ เป็นต้น โรงงานอุตสาหกรรมจึงนำแขนกลเข้ามามีบทบาทในการผลิตมากขึ้น ไม่ว่าจะเป็นการเชื่อมชิ้นส่วน (Welding) การพ่นสี (Painting) การประกอบชิ้นส่วน (Assembling) ฯลฯ ดังนั้นอัตราการผลิตของโรงงานเหล่านี้ จึงขึ้นอยู่กับขีดความสามารถของแขนกลด้วย

ด้วยเหตุนี้ผู้วิจัยจึงได้จัดทำงานวิจัยนี้ขึ้นมา โดยจะออกแบบระบบควบคุมตำแหน่งและแนวการวางตัวของแขนหุ่นยนต์ 2 แกน ให้มีความเสถียรภาพในตำแหน่งที่ต้องการ บนพาหนะขณะเคลื่อนที่ ใช้หลักการควบคุมพีไอดี โดยมีเซนเซอร์ติดไว้ที่ปลายแขนของหุ่นยนต์ซึ่งเป็นตัวรับสัญญาณ จากตำแหน่งที่เราต้องการ ขณะที่พาหนะเคลื่อนที่ไม่ว่าจะไปทิศทางไหน ปลายแขนของหุ่นยนต์จะต้องชี้หรืออยู่ในตำแหน่งเดิมที่เราตั้งไว้

1.2 วัตถุประสงค์ของงานวิจัย

1.2.1 เพื่อศึกษาหลักการทำงานและระบบควบคุมของแขนหุ่นยนต์หรือแขนกล

1.2.2 เพื่อออกแบบระบบควบคุมตำแหน่งแนวการวางตัวของแขนหุ่นยนต์ 2 แกน ให้มีความเสถียรภาพในตำแหน่งที่ต้องการ บนพาหนะขณะเคลื่อนที่

1.3 ขอบเขตของงานวิจัย

1.3.1 ออกแบบและสร้างระบบควบคุมตำแหน่งและแนวการวางตัวของแขนหุ่นยนต์ 2 แกน

1.3.2 มีระบบควบคุมการเคลื่อนที่ของหุ่นยนต์ โดยใช้โปรแกรมคอมพิวเตอร์

1.3.3 พาหนะเคลื่อนที่ในระยะ 1.20 m×2.40 m

1.3.4 สามารถควบคุมตำแหน่งและแนวการวางตัวของแขนหุ่นยนต์ 2 แกน บนพาหนะขณะเคลื่อนที่ให้ได้ตำแหน่งที่ต้องการได้

1.4 สถานที่ทำการวิจัย

มหาวิทยาลัยเทคโนโลยีสุรนารี

1.5 วิธีการดำเนินงานวิจัย

1.5.1 ศึกษาค้นคว้าความรู้ ทฤษฎี เอกสารที่เกี่ยวข้อง

1.5.2 รวบรวมข้อมูล และทำการออกแบบระบบ

1.5.3 จัดหาวัสดุและอุปกรณ์ในจัดสร้าง

1.5.4 ออกแบบและสร้างเครื่องต้นแบบ

1.5.5 เขียนโปรแกรมควบคุมการทำงานของระบบควบคุมตำแหน่งและแนวการวางตัวของแขนหุ่นยนต์ 2 แกน

1.5.6 ออกแบบและทดสอบการทดลอง

1.5.7 รวบรวมข้อมูล วิเคราะห์ และสรุปผลการวิจัย

1.5.8 จัดทำรูปเล่มวิทยานิพนธ์

1.5.9 สอบวิทยานิพนธ์

1.6 ประโยชน์ที่คาดว่าจะได้รับ

1.6.1 ทำให้ได้ระบบที่มีความเสถียรภาพ ถูกต้องแม่นยำ สามารถควบคุมตำแหน่งและแนวการวางตัวของแขนหุ่นยนต์ 2 แกน บนพาหนะขณะเคลื่อนที่

1.6.2 สามารถนำระบบที่ได้มาไปประยุกต์ใช้ให้เกิดประโยชน์ในด้านอุตสาหกรรมต่อไปในอนาคตได้

บทที่ 2
ปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง

2.1 กล่าวนำ

งานวิจัยนี้เป็นงานวิจัยที่ต้องการออกแบบการควบคุมตำแหน่งและแนวการวางตัวของแขนหุ่นยนต์ 2 แกน บนพาหนะขณะเคลื่อนที่ เพื่อนำไปประยุกต์ใช้ในการพัฒนาแขนหุ่นยนต์ในภาคอุตสาหกรรม ในส่วนแรกผู้วิจัยได้ทำการค้นคว้าเอกสาร ผลงานวิจัย วิทยานิพนธ์ต่าง ๆ ที่เกี่ยวข้องกับงานวิจัยที่ได้ดำเนินการอยู่พบว่า มีบทความและเอกสารต่าง ๆ ที่เกี่ยวข้องเป็นจำนวนมาก ดังนั้นผู้วิจัยจึงเลือกนำเสนองานวิจัยที่สอดคล้องเท่านั้น

2.2 ปริทัศร์วรรณกรรม

Kung-Ye, Ming-Yang และ Mi-Ching (2002) ได้พัฒนาระบบติดตามภาพที่ติดตั้งบนแขนหุ่นยนต์แบบ Pan – Tilt แบบเรียลไทม์ สำหรับงานด้านความปลอดภัย และได้นำเสนอการทดสอบเพื่อเปรียบเทียบประสิทธิภาพของโมเดล SSD ที่มีการปรับปรุงแก้ไขแล้วร่วมกับวิธีพลังงานการเคลื่อนไหว ผลการทดลองแสดงว่าวิธี SSD รวมกับ 3SHS ลดเวลาในการคำนวณได้อย่างมีประสิทธิภาพจนสามารถดำเนินการตามเวลาจริง นอกจากนี้วิธี SSD มีประสิทธิภาพมากขึ้นและส่งผลให้การติดตามดีขึ้นแม่นยำกว่าวิธีพลังงานเคลื่อนที่ ในทางตรงกันข้าม วิธีพลังงานการเคลื่อนไหวต้องการภาระในการคำนวณน้อยลงกว่าวิธี SSD นอกจากนี้ วิธีพลังงานเคลื่อนที่สามารถตรวจจับเป้าหมายที่เคลื่อนที่ได้สำเร็จตราบเท่าที่เป้าหมายอยู่ในระนาบภาพ ข้อเท็จจริงนี้แสดงให้เห็นว่ามีความแม่นยำขึ้นเหมาะสำหรับการตรวจจับเป้าหมายที่เคลื่อนที่เร็วขึ้น แต่ถึงอย่างไร ข้อเสียเปรียบหลักของวิธีพลังงานเคลื่อนที่คือมันถูกจำกัดให้จัดการกับกรณีของวัตถุเคลื่อนที่ชิ้นเดียว

Gardel, Lazaro, Lavest and Vazquez (2002) นำเสนอการพัฒนาและปรับปรุง ระบบที่เพิ่มทัศนวิสัยได้และตรวจจับยานพาหนะเคลื่อนที่หรือสิ่งกีดขวาง จากการจับภาพแล้วนำลำดับภาพจากกล้องมาวิเคราะห์มีการใช้กล้องซูมเพื่อปรับปรุงความปลอดภัยของระบบขนส่งโดยบันทึกภาพจากกล้องซูมแล้วนำมาใช้วัดระยะของวัตถุ นอกจากนี้ยังเพิ่มความแม่นยำและความน่าเชื่อถือของระบบมากขึ้น

Hiroyuki Ukida (2010) ได้พัฒนาระบบติดตามวัตถุโดยใช้หุ่นยนต์แขนกลและกล้องที่สามารถเคลื่อนที่แบบ Pan-tilt ได้ แขนหุ่นยนต์สามารถหมุนได้ในมุมกว้างแต่จะเคลื่อนที่ได้ช้า ในทางกลับกัน กล้อง Pan-tilt จะเคลื่อนที่อย่างรวดเร็ว แต่ระยะการหมุนจะถูกจำกัด และจากการรวมกันของอุปกรณ์เหล่านี้ บทความนี้จึงเสนอวิธีการติดตามวัตถุช่วงกว้าง เพื่อที่จะตรวจจับวัตถุในภาพกล้องอย่างรวดเร็ว ซึ่งใช้ข้อมูลสีและวิธีการ condensation ซึ่งอธิบายตำแหน่งของวัตถุมาประยุกต์ใช้

Robin, Carmadi and Egi (2017) บทความนี้นำเสนอการออกแบบและการติดตามใบหน้าโดยใช้ Pan-Tilt ที่ควบคุมด้วยเซอร์โวมอเตอร์ เพื่อรักษาตำแหน่งของใบหน้าบนหน้าจอ อัลกอริทึม Viola-Jones และ template ถูกนำมาผสมผสานกันในการตรวจจับใบหน้าการติดตามใบหน้าถูกควบคุมโดยอัตโนมัติด้วยไมโครคอนโทรลเลอร์ร่วมกับ Lead-Lag controller และ PID เพื่อควบคุมการตอบสนองของมอเตอร์

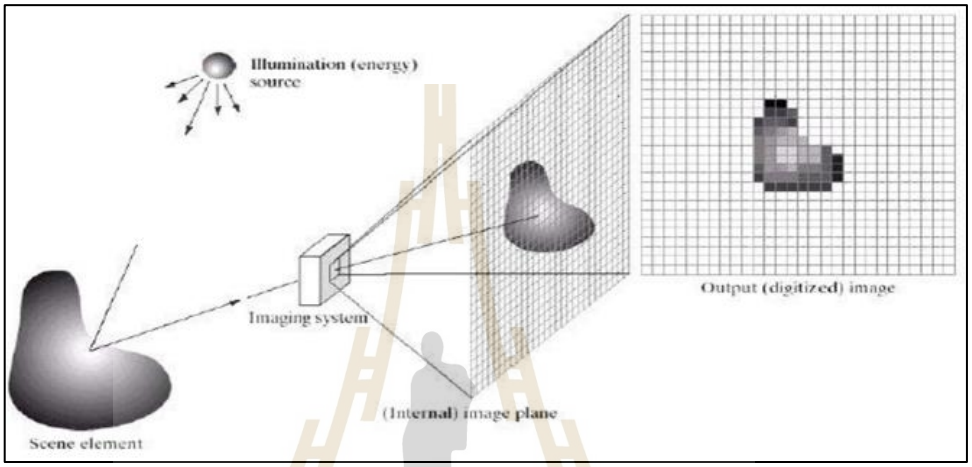
2.3 การประมวลผลภาพ

การประมวลผลภาพเป็นกระบวนการจัดการและวิเคราะห์รูปภาพให้เป็นข้อมูลในแบบดิจิทัล โดยใช้คอมพิวเตอร์ในการประมวลผลโดยวิธีการต่าง ๆ เพื่อให้ได้ภาพที่มีคุณสมบัติตามความต้องการทั้งในเชิงคุณภาพและปริมาณ มีหลากหลายรูปแบบซึ่งเราเรียกโดยรวมว่า “ปรับปรุงคุณภาพของภาพ (Image enhancement)” การปรับเปลี่ยนหรือแปลงรูปภาพทั้งขนาดและรูปร่าง (Image transformation) การกรองภาพหรือการกำจัดสัญญาณรบกวนออกจากภาพ (Image filters) การซ้อนทับภาพ (Image registration) การคืนสภาพของภาพ (Image restoration) การตัดแบ่งภาพหรือคัดเลือกส่วนที่ต้องการและการหาขอบภาพในวัตถุ (Image segmentation and Detection) การบีบอัดภาพ (Image compression) การสร้างภาพ 3 มิติ (3D Image reconstruction) เป็นต้น

2.4 ภาพดิจิตอล (Digital Image)

ในทางการทฤษฎีนั้น ภาพหนึ่งภาพสามารถที่จะกำหนดได้ด้วย ฟังก์ชั่น 2 มิติ ซึ่งเป็นฟังก์ชันที่ใช้แสดงระดับของปริมาณความหนาแน่นของแสง (Light intensity) และเมื่อ f คือฟังก์ชันของความสว่างแล้วนั้นค่า x และy นั้นจะแสดงถึงพิกัดที่ระบุตำแหน่งทางแนวนอนและแนวตั้งตามลำดับ ซึ่งอยู่ในพิกัดแกนของระบบภาพหรือเรียกวา “Spatial coordinate” หรือเรียกพิกัด x-y อีกอย่างว่า “จุดภาพ (Pixel)” โดยปกติแล้วนั้น Spatial coordinate นั้น จะเริ่มต้น ด้วยพิกัดและจะเริ่มที่ตำแหน่งบนสุดและซ้ายสุดของภาพโดยในแกนจะมีค่าเพิ่มขึ้นจากซ้ายไปขวาและในแกนจะมีค่า การกำเนิดภาพนั้นเกิดขึ้นจากอุปกรณ์สำหรับถ่ายภาพได้แปลงสัญญาณทาง

อิเล็กทรอนิกส์มาเป็นปริมาณทางดิจิตอล ซึ่งปริมาณนี้จะขึ้นอยู่กับระดับความละเอียดทางดิจิตอลว่าจะมีกี่บิต โดยในส่วนใหญ่แล้วจะได้ภาพที่มีระดับ 8 บิต โดยระดับความแตกต่างที่ได้ สามารถหาได้จากปริมาณการแยกแยะของจำนวนบิตมาจาก 2n เมื่อคือจำนวนบิตที่ใช้ ดังนั้นจะสามารถแยกแยะระดับได้ $2^8 = 256$ ระดับ ซึ่งในการกำหนดนั้น จะเริ่มที่ 0 - 255 ระดับ ซึ่งเรียกระดับที่แบ่งแยกได้นี้ว่า “ระดับเทา level” โดยลักษณะการเกิดภาพดิจิตอลสามารถแสดงได้ดังรูปที่ 2.1



รูปที่ 2.1 ลักษณะของการเกิดภาพทางดิจิตอล

คุณสมบัติของภาพดิจิตอล (Properties of Digitized Image)

- ความละเอียดของภาพ (Image resolution) คือ จำนวนของจุดภาพในแนวแกน x-y ซึ่งขึ้นอยู่กับความสามารถของอุปกรณ์ถ่ายภาพโดยยิ่งมีค่ามากยิ่งมีความละเอียดมาก
- ความชัดของภาพ (Image Definition) คือ ระดับความสามารถที่ใช้แยกแยะปริมาณความแตกต่างของค่าที่ได้ในจุดภาพ เช่นภาพ 8 บิตนั้นก็จะมี 256 ระดับแตกต่าง
- จำนวนของระนาบของภาพ (Number of Planes) เป็นจำนวนที่บอกให้ทราบถึงลำดับชั้นของจุดภาพ เช่น ในระบบภาพขาวดำหรือสองสีนั้นเรียกว่าเป็น “ภาพ 1 ระนาบ” และถ้าในกรณีภาพสีนั้นประกอบไปด้วย แม่สีแดง เขียว และน้ำเงิน จะเรียกว่า “3 ระนาบ”

2.5 โมเดลประมวลผลภาพสำหรับการติดตามวัตถุ

ในการประมวลผลภาพสำหรับการติดตามวัตถุนั้น มีหลักการเบื้องต้นคือ การจับคู่ภาพ (Matching) เมื่อภาพมีการเคลื่อนที่ไปในแต่ละเฟรมของภาพ หลังจากทำการจับคู่ได้แล้วภาพที่มีความคล้ายคลึงกันมากที่สุดระหว่างเฟรมถัดไปกับเฟรมก่อนหน้าจะถูกอัพเดต หากมีภาพ

ที่คล้ายคลึงกันหลายจุดและมีค่าความคล้ายใกล้เคียงกัน การหาระยะทางระหว่างวัตถุในเฟรมก่อนหน้ากับเฟรมถัดไปจะถูกนำมาใช้หาการเคลื่อนที่ของวัตถุจริง โดยวัตถุในเฟรมถัดไปที่มีระยะทางใกล้ที่สุดกับเฟรมก่อนหน้าจะถูกตัดสินว่าเป็นวัตถุเดียวกันที่มีการเคลื่อนที่ไป ในการเขียนโปรแกรมสำหรับการติดตามวัตถุในไลบรารีของโอเพนซีวี (OpenCV) จะมีโมเดลในการจับคู่การเคลื่อนไหวของภาพอยู่ โดยโมเดลที่นำมาทดลองมีดังนี้

2.5.1 โมเดลตัวกรองความสัมพันธ์โดยการใช้เคอร์เนล (Kernelized Correlation Filter)

Henriques, et al (2014) โมเดลตัวกรองความสัมพันธ์โดยการใช้เคอร์เนลเป็นอัลกิริทึมในการจับคู่วัตถุในแต่ละเฟรมของภาพที่มีการเคลื่อนที่เปลี่ยนไป โดยอาศัยค่าของความสัมพันธ์ (Correlation value) ของภาพของวัตถุในแต่ละเฟรม โดยหากส่วนใดของภาพที่มีค่าความสัมพันธ์สูงที่สุดจะถูกพิจารณาว่าเป็นวัตถุที่กำลังติดตาม ค่าของ Correlation Filter จะถูกคิดออกมาในรูปของโดเมนของฟูริเยและการแปลงฟาสต์ฟูริเย (FFT) โดยมีสมการรูปแบบเป็นดังสมการที่ 2.1

$$G = F \cdot H^* \tag{2.1}$$

เมื่อ F คือ การแปลงแบบฟูริเยของภาพ 2 มิติที่รับเข้าไป โดย F= *F(f)*

H คือ การแปลงแบบฟูริเยสองมิติของตัวกรอง โดย H=*F(h)*

โดยเอาท์พุทของค่าความสัมพันธ์จะถูกแปลงกลับไปเป็นสปาเทียลโดเมน (Spatial domain) ด้วยการแปลงฟาสต์ฟูริเยผกผัน (Inverse FFT)

2.5.2 โมเดลแบบการใช้ค่าขั้นต่ำของผลรวมความผิดพลาดของเอาท์พุต (Minimum Output Sum of Squared Error: MOSSE)

Bolme, Beveridge, Draper and Lui. (2010) โมเดลแบบการใช้ค่าขั้นต่ำของผลรวมความผิดพลาดของเอาท์พุต เป็นโมเดลการติดตามวัตถุจะมีหลักการทำงานโดยใช้ค่าความสัมพันธ์ (Correlation value) คล้ายกับโมเดลแบบตัวกรองความสัมพันธ์โดยการใช้เคอร์เนล และจะมีการทำออพติไมเซชันเพื่อหาตัวกรอง (*H*) ที่จะสามารถจับคู่ภาพที่อินพุทเข้าไปกับภาพที่ทำการติดตั้งได้หลังจากการทำคอนโวลูทชัน (Convolution) โดยการหาค่าความผิดพลาด (Error) ที่ต่ำที่สุดจะสามารถเขียนได้ในรูปสมการที่ 2.2

$$\min_{H^*} \sum_i \left| F_i \cdot H^* - G_i \right|^2 \tag{2.2}$$

โมเดลแบบ MOSSE จะทำงานได้ดีแม้วัตถุจะมีการหมุน มีแสงสว่างในสภาวะต่าง ๆ และมีขนาดของวัตถุที่เปลี่ยนแปลงไป โดยโมเดลนี้จะให้ความแม่นยำที่น้อยว่าโมเดล KCF และ CSRT เล็กน้อยแต่ใช้การประมวลผลของคอมพิวเตอร์ที่ต่ำจึงทำงานได้เร็วกว่าโมเดลอีกสองตัวที่นำมาทดลอง

2.5.3 โมเดลแบบความน่าเชื่อถือของช่องและพื้นที่ (Channel and Spatial Reliability Tracker: CSRT)

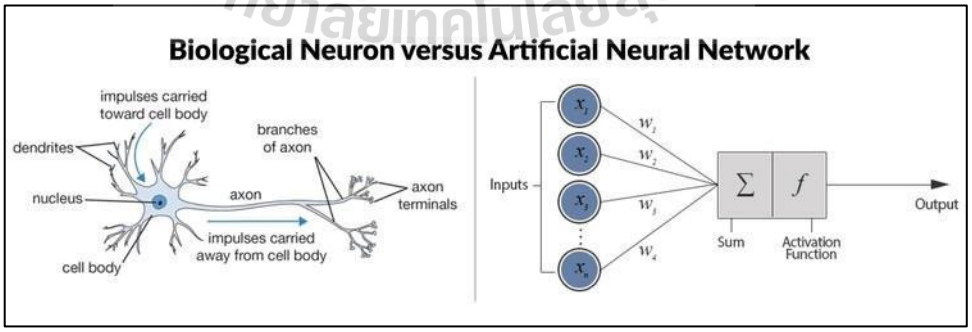
Lukezic, et al., (2019) โมเดลแบบความน่าเชื่อถือของช่องและพื้นที่ เป็นโมเดลที่ใช้ค่าความสัมพันธ์ในการติดตามวัตถุอีกตัวหนึ่ง แต่สามารถปรับขนาดของตัวกรองได้ ซึ่งจะทำให้โมเดลนี้สามารถตรวจจับวัตถุที่ไม่ใช่รูปทรงสี่เหลี่ยมได้ดี โมเดลในการติดตามวัตถุตัวนี้จะให้ค่าความแม่นยำที่มากที่สุดเมื่อเทียบกับโมเดลเคซีเอฟและโมเดลมอซซี แต่ว่าจะใช้ทรัพยากรในการทำงานที่มากที่สุด ส่งผลให้การทำงานช้ากว่าโมเดลตัวอื่น

2.6 การเรียนรู้เชิงลึก

การเรียนรู้เชิงลึก (Deep learning: DL) เป็นสาขาหนึ่งของศาสตร์ด้านการเรียนรู้ของเครื่องจักร (Machine Learning: ML) โดยมุ่งเน้นไปที่การสร้างแบบจำลองโครงข่ายประสาท (Neural Network) ของมนุษย์ให้กับคอมพิวเตอร์ เพื่อที่จะตัดสินใจหรือทำนายสิ่งต่าง ๆ จากข้อมูลที่ได้รับมา

1. การเลียนแบบเซลล์ประสาทของมนุษย์ในโครงข่ายประสาทเทียม

โครงข่ายประสาทเทียม (Artificial neural network) เกิดขึ้นจากการเลียนแบบเซลล์ประสาทของมนุษย์



รูปที่ 2.2 การเลียนแบบระบบประสาทของมนุษย์กับโครงข่ายประสาทเทียม
(ที่มา: https://phusitsom.medium.com)

พิจารณาระบบการทำงานของเซลล์ประสาทของมนุษย์ จะมีการทำงานแยกออกได้เป็น 4 ส่วนดังนี้

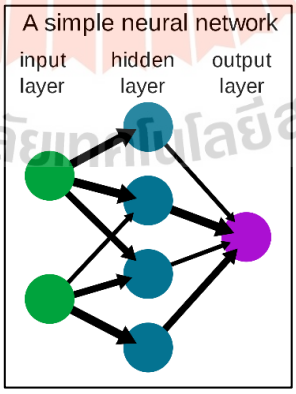
1) Cell boy หรือส่วนกลางของ Neuron ในทางชีววิทยาจะเป็นส่วนรวบรวมและประมวลผลข้อมูลทั้งหมดในโครงข่ายประสาทเทียมจะเปรียบเสมือน Node แต่ลำ node ที่มีหน้าที่รวม Input และ Weight ที่ได้จากการสอนโมเดลไปประมวลผล อาจจะต้องมีการแปรสภาพก่อน ด้วย Activation function ก่อนจะส่งค่าที่ได้ไปที่ Node อื่นต่อไป

2) เดนไดรท์ (Dendrite) เป็นส่วนที่รับสัญญาณขาเข้าของเซลล์ประสาท ในโครงข่ายประสาทเทียมเปรียบเสมือนส่วนที่รับค่าข้อมูลเข้ามา

3) แอ็กซอน (Axon) เป็นส่วนที่ส่งสัญญาณขาออกของเซลล์ประสาท 1 ตัว ไปยังเซลล์อื่น ๆ ในโครงข่ายประสาทเทียมเปรียบเสมือนส่วนที่ส่งข้อมูลออกไป

4) ไซแนป (Synapse) เป็นส่วนที่ แอ็กซอนของเซลล์หนึ่งกับเดนไดรท์ของเซลล์ถัดไปเชื่อมต่อกันเป็นจุดผ่านสำหรับการส่งข้อมูลระหว่างกัน และต้องการแรงในการส่งสัญญาณที่แตกต่างกันในแต่ละกรณี ขึ้นอยู่กับความแข็งแรงของไซแนปส์นั้น ๆ ยิ่งมีกระบวนการเรียนรู้เกิดขึ้นมากไซแนปส์จะยิ่งแข็งแรง ในโครงข่ายประสาทเทียมนั้นไซแนปส์เปรียบเสมือนค่าน้ำหนัก (Weight) ที่ได้จากการเทรนโมเดล โดยยิ่งโมเดลมีการเทรนเยอะขึ้นค่าน้ำหนักที่ได้จะยิ่งแข็งแรงและทำให้ผลการทำนายมีความแม่นยำมากขึ้น

จากส่วนประกอบที่กล่าวมาข้างต้นหากจำลองการทำงานของ Neuron ออกมาเป็นโมเดลคณิตศาสตร์ ก็จะได้รูปแบบการทำงานของโครงข่ายประสาทเทียมอย่างง่ายดังในรูปที่ 2.3



รูปที่ 2.3 Simple Neural Network Layers

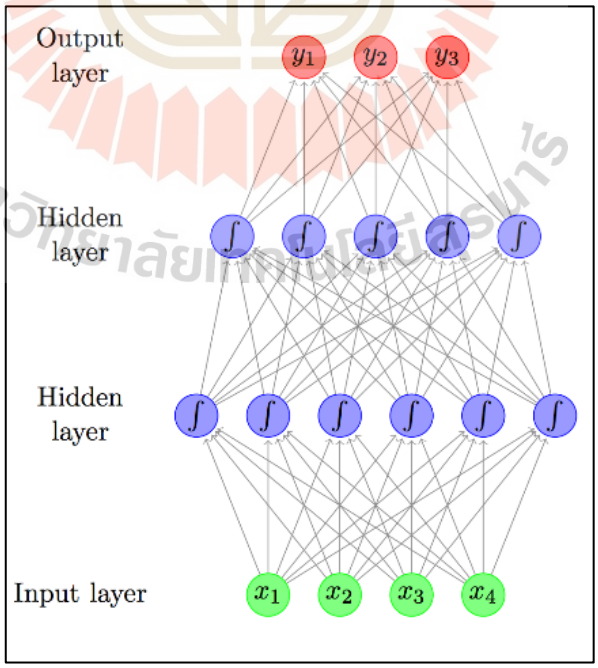
การทำงานของโครงข่ายประสาทเทียมจะถูกแบ่งเป็นชั้น ๆ (Layers) โดยมีชื่อเรียกชั้นการทำงานหลักดังนี้

1) ชั้นข้อมูลเข้า (Input layer) คือ ชั้นที่มีโหนดที่ใช้ในการรับข้อมูลเข้ามาโหนดในชั้นนี้จะมากหรือน้อยขึ้นอยู่กับปริมาณข้อมูลที่เข้ามา เช่น หากข้อมูลที่เข้ามามีเพียงพิกัด x,y นั้น โหนดในชั้นข้อมูลเข้า 2 โหนดก็เพียงพอแล้ว

2) ชั้นที่ถูกซ่อน (Hidden layer) คือ ชั้นที่ทำหน้าที่ส่งต่อข้อมูล โดยในชั้นที่ถูกซ่อนนี้ค่าน้ำหนักที่ได้จากการเทรนจะถูกทำให้มีการปรับเข้ากับค่าข้อมูลมากขึ้นเรื่อย ๆ ตามจำนวนรอบการเทรนโมเดล จนในที่สุดจะมีค่าความแม่นยำอยู่ในระดับที่สามารถนำไปใช้งานได้อย่างแม่นยำ

3) ชั้นข้อมูลออก (Output layer) คือ ชั้นที่ส่งข้อมูลออกมาเป็นชั้นสุดท้ายของโครงข่ายประสาทเทียม เช่น หากข้อมูลออกที่เราต้องการ คือ ข้อมูลประเภทของข้อมูลเข้า เช่น ข้อมูลที่เข้าไปเป็นประเภท A หรือ B ชั้นข้อมูลออกก็จะส่งค่าเหล่านั้นออกมาเป็นระดับความน่าหรือค่าข้อมูลหรือระดับความมั่นใจ ขึ้นอยู่กับปัญหาที่เราทำการวิเคราะห์

จากรูปที่ 2.3 ของโครงข่ายประสาทเทียมนี้เป็นรูปแบบที่ง่ายที่สุด คือ มีชั้นการทำงานแต่ละแบบ อย่างละ 1 ชั้น ซึ่งในโจทย์ปัญหาบางโจทย์ เช่นการแยกแยะ (Classification) วัตถุที่เห็นจากรูปภาพ อาจจำเป็นที่จะต้องใช้ชั้นการทำงานมากกว่านั้น โดยชั้นที่ถูกซ่อน (Hidden layer) จะถูกเพิ่มจำนวนขึ้นมากกว่า 1 ชั้น ขึ้นอยู่กับปัญหา ซึ่งเมื่อใดก็ตามที่ชั้นที่ถูกซ่อนมีมากกว่า 1 ชั้น หรือตั้งแต่ 2 ชั้นขึ้นไป ระบบโครงข่ายประสาทเทียมนั้นจะถูกเรียกว่า “การเรียนรู้เชิงลึก (Deep learning)”



รูปที่ 2.4 Deep learning layers (ที่มา: https://lloli.medium.com)

2. โครงข่ายประสาทเทียมแบบคอนโวลูทชั่น (Convolution Neural Network)

โครงข่ายประสาทเทียมแบบคอนโวลูทชั่น เป็นโครงข่ายประสาทเทียมที่ใช้กับปัญหาการประมวลผลของรูปภาพ เช่น การแยกแยะภาพ (Classification) โดยในโครงข่ายประสาทเทียมแบบนี้จะใช้วิธีที่เรียกว่า “การทำคอนโวลูทชั่น (Convolution)” ในการแยกส่วนประกอบของรูปภาพ (Feature) ออกมาว่าในแต่ละส่วนย่อยของรูปภาพนั้น รูปภาพมีเส้น มีขอบ มุม หรือรายละเอียดต่าง ๆ อย่างไรบ้าง แล้วแยกแยะออกมา (Feature extraction) ในขั้นตอนนี้ จำเป็นต้องทำการสร้างตัวกรอง(Filter)ออกมา หรืออีกชื่อเรียกว่า “เคอร์เนล (Kernel)” โดยเคอร์เนลแต่ละตัวจะสามารถดึงคุณสมบัติออกมาได้แตกต่างกันและเฉพาะเจาะจง ดังนั้น เราอาจจะต้องใช้เคอร์เนลหลายตัวในการแยกคุณสมบัติของรูปภาพออกมา ตัวอย่างของเคอร์เนลขนาด 3×3 เป็นดังรูปต่อไปนี้

1	-1	-1
-1	1	-1
-1	-1	1

รูปที่ 2.5 เคอร์เนลขนาด 3×3

เมื่อนำเคอร์เนลเข้าไปคูณกับค่าสีของพิกเซลของรูปภาพแล้วทำการคอนโวลูทชันจะทำให้เราได้ผังคุณลักษณะ(Feature map) ออกมา ในการทำคอนโวลูทชันนั้น คำศัพท์ที่สำคัญคือ Stride, Padding และ Max pooling โดย Stride คือ สเต็ปในการเลื่อนเคอร์เนลว่าเราต้องการเลื่อนเคอร์เนลไปทีละเท่าไร ยิ่งค่า Stride มีค่ามากจะทำให้ผังคุณลักษณะมีขนาดเล็กลง ส่วน Padding คือ การเพิ่มพื้นที่พิกเซลของรูปภาพเข้าไปเพื่อให้ขนาดของผลลัพธ์จาก คอนโวลูทชันนั้นเท่ากับขนาดภาพที่เข้ามา โดยส่วนใหญ่นิยมใส่ค่าเป็น 0 และถูกเรียกว่า “Zero padding” และ Max pooling คือ การหาค่าสูงสุดของค่าในช่วงที่เคอร์เนลอยู่ เพื่อลดความละเอียดของภาพลงแต่ยังคงไว้ซึ่งจุดสำคัญ

3. การฝึกโมเดล (Model training)

ความแตกต่างของการเขียนโปรแกรมทั่วไปกับการเรียนรู้เชิงลึกหรือการเรียนรู้ของเครื่องในแบบอื่น ๆ คือ ในโปรแกรมทั่วไปเราต้องใส่อัลกอริทึมเข้าไป แต่ในการเรียนรู้เชิงลึกหรือการเรียนรู้ของเครื่องแบบอื่น ๆ คือ เราต้องการหาอัลกอริทึมโดยการฝึกระบบด้วยผลลัพธ์ที่มีก่อนหน้านี้เพื่อเก็บอัลกอริทึมตัวนั้นไว้สำหรับทำนายผลลัพธ์ในอนาคต ดังนั้น เราจะต้องทำการฝึกโมเดลสำหรับการเรียนรู้เชิงลึกก่อน

วิธีการฝึกโมเดลการเรียนเชิงลึกเปรียบเสมือนการสอนให้คน 1 คนรู้จักสิ่งมีชีวิต 1 ตัว เราต้องนำข้อมูลของสิ่งมีชีวิตนั้นใส่เข้าไปในโมเดล เช่น รูปร่าง ลักษณะ รูปทรง เป็นต้น แล้วบอกให้โมเดลรู้ว่าสิ่งนั้นคืออะไร ซึ่งเราจะทำเสมือนเขียนป้ายชื่อ (Label) ติดไว้ แล้วใส่ข้อมูลพร้อมป้ายชื่อนั้นลงไปในโมเดล จากนั้นโมเดลก็จะผ่านกระบวนการต่าง ๆ ผ่านการคำนวณหา Weight จากข้อมูลในแต่ละชั้นของชั้นที่ถูกซ่อนจนค่าความผิดพลาด (Error) มีค่าน้อยที่สุด แล้วจะได้เป็นอัลกอริทึมออกมา

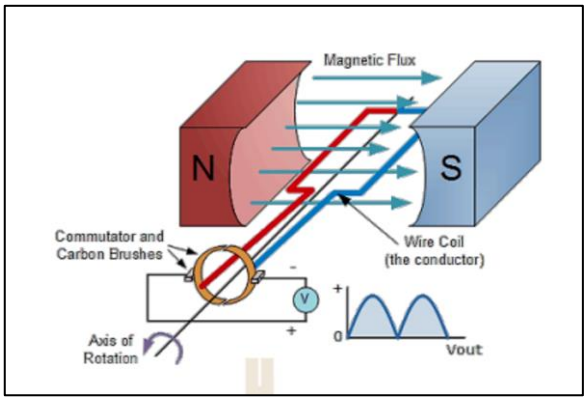
ดังนั้น การเตรียมข้อมูลสำหรับฝึกโมเดลจึงเป็นสิ่งสำคัญ ข้อมูลที่คุณภาพจะให้ผลลัพธ์ที่มีคุณภาพออกมา แต่หลังจากที่เราได้อัลกอริทึมออกมา เราต้องทำการประเมินผลด้วยการนำอัลกอริทึมที่ได้มาไปทดสอบกับข้อมูลที่ไม่เคยเห็นมาก่อน แสดงว่าต้องมีการเตรียมข้อมูลส่วนนี้ไว้ด้วย ดังนั้นในการเก็บข้อมูลนั้น ข้อมูลที่เก็บมาต้องถูกแบ่งออกเป็น 2 ส่วน คือ ส่วนสำหรับฝึก (Train) โมเดล และส่วนสำหรับการทดสอบโมเดล (Test model) ซึ่งอัตราส่วนควรจะอยู่ที่ข้อมูลสำหรับการฝึกร้อยละ 80 และข้อมูลส่วนสำหรับการทดสอบร้อยละ 20

2.7 มอเตอร์กระแสตรง

มอเตอร์ไฟฟ้ากระแสตรง (Direct current motor หรือ DC Motor) คือ มอเตอร์ที่ป้อนไฟฟ้ากระแสตรงที่เปลี่ยนพลังงานไฟฟ้าเป็นพลังงานกล มีทั้งชนิดกระตุ้นฟิลด์จากภายนอก (Separated excited motor) และชนิดกระตุ้นฟิลด์ด้วยตัวเอง (Self-excited motor) มอเตอร์ไฟฟ้ากระแสตรงแบ่งออกเป็น 3 ชนิด ได้แก่

1. มอเตอร์แบบอนุกรมหรือเรียกว่าซีรีส์มอเตอร์ (Series motor)
2. มอเตอร์แบบอนุขนานหรือเรียกว่าชันท์มอเตอร์ (Shunt motor)
3. มอเตอร์ไฟฟ้าแบบผสมหรือเรียกว่าคอมเปาวด์มอเตอร์ (Compound motor)

หลักการทำงานของมอเตอร์ไฟฟ้ากระแสตรง (Direct current motor) เมื่อแรงดันไฟฟ้ากระแสตรงเข้าไปในมอเตอร์ ส่วนหนึ่งจะแปรงถ่านผ่านคอมมิวเตเตอร์เข้าไปในขดลวดอาร์มาเจอร์สร้างสนามแม่เหล็กขึ้น และกระแสไฟฟ้าอีกส่วนหนึ่งจะไหลเข้าไปในขดลวดสนามแม่เหล็ก (Field coil) สร้างขั้วเหนือ-ใต้ขึ้น จนเกิดสนามแม่เหล็ก 2 สนาม ในขณะเดียวกันตามคุณสมบัติของเส้นแรงแม่เหล็กจะไม่ตัดกัน ทิศทางตรงข้ามจะหักล้างกันและทิศทางเดียวจะเสริมแรงกัน ทำให้เกิดแรงบิดในตัวอาร์มาเจอร์ ทำให้อาร์มาเจอร์นี้หมุนได้ อาร์เมจอร์ที่หมุนนี้เรียกว่า “โรเตอร์ (Rotor)”



รูปที่ 2.6 การทำงานของมอเตอร์กระแสตรง (ที่มา: http://xuperb.blogspot.com)

2.8 สเต็ปเปอร์มอเตอร์ (Stepper motor)

สเต็ปเปอร์มอเตอร์เป็นมอเตอร์ไฟฟ้าที่ขับเคลื่อนด้วยสัญญาณพัลส์ โดยโครงสร้างภายในนั้นจะประกอบไปด้วยขั้วแม่เหล็กบนสเตเตอร์ (Stator) ทำมาจากแผ่นเหล็กวงแหวนจะมีซี่ยื่นออกมาประกอบกันเป็นชั้น ๆ โดยแต่ละซี่ที่ยื่นออกมานั้นจะมีขดลวด (Coil) พันอยู่เมื่อมีกระแสผ่านคอยล์จะเกิดสนามแม่เหล็กไฟฟ้าขึ้น ในการทำงานของ Stepper Motor นั้นจะไม่สามารถขับเคลื่อนหรือทำงานเองได้ จำเป็นต้องมีวงจร อิเล็กทรอนิกส์ที่ใช้ในการสร้างสัญญาณหรือจ่ายพัลส์ไปให้วงจรขับ Stepper driver การสร้างสัญญาณนั้นจำเป็นจะต้องสร้างและเรียงลำดับของสัญญาณ และอีกสิ่งที่สำคัญคือการดูตำแหน่งของสายที่ทำการต่อเข้ากับตัวสเต็ปเปอร์มอเตอร์เพื่อให้การควบคุมทิศทางการหมุนของมอเตอร์ถูกต้อง

2.8.1 Stepper motor NEMA17

สเต็ปเปอร์มอเตอร์ NEMA 17 Stepper 57mm 2.8A แสดงดังรูปที่ 2.7

รูปที่ 2.7 Stepper Motor NEMA 17 (ที่มา: https://www.reichelt.com)

ตารางที่ 2.1 รายละเอียด Stepper motor NEMA 17

Model	**SL42STH40-1684A**
วัสดุ	โลหะ
ขนาด	42.3 mm×42.3 mm×40 mm
มุม step	1.8° (Full Step Mode)
แรงดันไฟฟ้า	2.8 V
กระแสไฟฟ้า	1.68 A
ความต้านทาน / เฟส	1.65 Ω
ความเหนี่ยวนำ / เฟส	3.2 mH
แรงบิด	0.4 Nm

สมการทางคณิตศาสตร์ของสเต็ปเปอร์มอเตอร์แบ่งได้เป็น 2 แบบ คือ แบบจำลองไฟฟ้าและทางกล

1. แบบจำลองไฟฟ้า (Electrical model) เฟสไฟฟ้าทั้งสองเฟสของสเต็ปเปอร์มอเตอร์สามารถจำลองได้ที่ความถี่และกระแสที่ค่อนข้างต่ำเป็นวงจร RL บวกกับอิเล็กโทรโมทีฟด้านหลังแรง (emf) วงจรนี้อธิบายโดยสมการต่อไปนี้:

$$L_w = \frac{di_j(t)}{dt} = R_w i_j(t) - e_j(t) + u_j(t) \tag{2.3}$$

โดยที่ R_w คือ ความต้านทานเฟส

L_w คือ ตัวเหนี่ยวนำเฟสในกระแสเฟส

i_j คือ เฟสปัจจุบัน

u_j คือ แรงดันขั้วและแรงดันไฟย้อนกลับ

ซึ่งสามารถอธิบายโดยสมการดังต่อไปนี้:

$$e_A(t) = -K_m w_m \sin p\theta_m \tag{2.4}$$

$$e_b(t) = K_m w_m \cos p\theta_m \tag{2.5}$$

โดยที่ K_m คือ เป็นค่าคงที่ของมอเตอร์

P คือ จำนวนคู่ขั้วของมอเตอร์

w_m คือ ความเร็วเชิงมุมของโรเตอร์

θ_m คือ มุมกลของมอเตอร์

เมื่อทำการแปลงลาปลาซได้ดังสมการที่ 2.6

$$I_j = \frac{1}{Z_{mot}} (U_j(s) - E_j(s)) \qquad \text{for } j = A, B \tag{2.6}$$

และสมการอิมพีแดนซ์ไฟฟ้าของมอเตอร์แสดงดังสมการที่ 2.7

$$Z_{mot}(s) = L_W s + R_w \tag{2.7}$$

2. แบบทางกล (Mechanical model) ชิ้นส่วนทางกลของมอเตอร์ถูกออกแบบให้มีลักษณะแข็งตามแรงบิดต่าง ๆ

$$J\frac{dw_m}{dt} = \tau_{em} - Bw_m - \tau_1 \tag{2.8}$$

แรงบิดแม่เหล็กไฟฟ้าของมอเตอร์แสดงดังสมการที่ 2.9

$$\tau_{em} = K_m(-i_{mot_A} \sin p\theta_m + i_{mot_B} \cos p\theta_m) \tag{2.9}$$

แรงบิดกักขังแสดงดังสมการที่ 2.10

$$\tau_{dm} = T_{dm} \sin(2p\theta_m + \phi) \tag{2.10}$$

โดยที่ T_{dm} คือ แอมพลิจูดแรงบิดกักขัง

ϕ คือ การเลื่อนเฟสที่เกี่ยวข้องกับ τ_{dm} และ τ_1 แรงบิดโหลดภายนอก

2.9 เซอร์โวมอเตอร์ (Servo motor)

อภิลักษณ์ (2015) หลักการพื้นฐานของ Servo Motor และระบบควบคุมในปัจจุบันการนำเอาระบบ Automation เข้ามาใช้ในระบบงานการผลิต การลำเลียงจัดส่งการจัดเก็บในโรงงานอุตสาหกรรมมากขึ้นตลอดจนมีการพัฒนาปรับปรุงระบบแบบเดิมที่ไม่มีใช้มาใช้ มากขึ้นด้วยหนึ่งในระบบ Automation System ที่มีการนำมาใช้นอกเหนือไปจากพวกระบบPneumatic และ Hydraulic คือ Servo motor & Control ซึ่งเป็นระบบที่นำมาใช้อย่างกว้างขวางมากในแทบทุกอุตสาหกรรม เช่น Packaging machinery, Automate manufacturing, Printing, Labeling, Textile machinery, Food processing, Plastic machinery, Metal forming รวมถึงงานที่มีการประยุกต์ใช้ เช่น Pick and Place, Flying cutoffs, X-Y-Z Table, Synchronized speed

การที่เซอร์โวมอเตอร์มีการใช้อย่างกว้างขวางก็เพราะว่ามีความยืดหยุ่นในการใช้งานสูง สามารถควบได้คุมได้ทั้งความเร็ว (Speed), ตำแหน่ง (Position) และแรงบิด (Torque) ซึ่งให้ประสิทธิภาพสูงสุดในกระบวนการควบคุมมอเตอร์ เมื่อเปรียบเทียบกับมอเตอร์ธรรมดา (AC MOTOR) หรือมอเตอร์กระแสตรง (DC MOTOR) การใช้ Servo Motor & Control

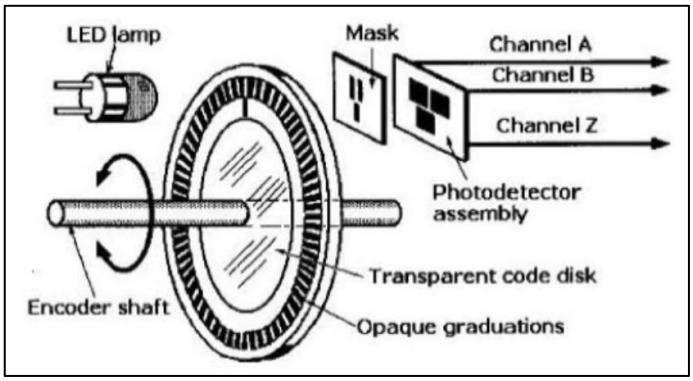
เซอร์โวมอเตอร์มีขนาดตั้งแต่ 30 W ขึ้นไปจนถึงกว่า 370 kW โครงสร้างภายในจะแตกต่างจากมอเตอร์เหนี่ยวนำ โดยที่โรเตอร์จะเป็นแม่เหล็กถาวร การออกแบบลดแรงเฉื่อยให้มากที่สุดจะเห็นจากรูปทรงเหลี่ยมและยาว ซึ่งมีผลทำให้การตอบสนองต่ออัตราเร่ง (Accelerate), ความเร็ว (Speed) และอัตราหน่วง (Decelerate) ทำได้ดีเมื่อเทียบกับมอเตอร์เหนี่ยวนำหรือมอเตอร์ทั่วไป การออกแบบจะมีสัญญาณป้อนกลับทั้งความเร็วและตำแหน่งที่ท้ายมอเตอร์ (ซึ่งอาจจะเป็น Encoder, Resolver หรือประเภทอื่น ๆ ก็ได้)

2.10 เอ็นโค้ดเดอร์ (Encoder)

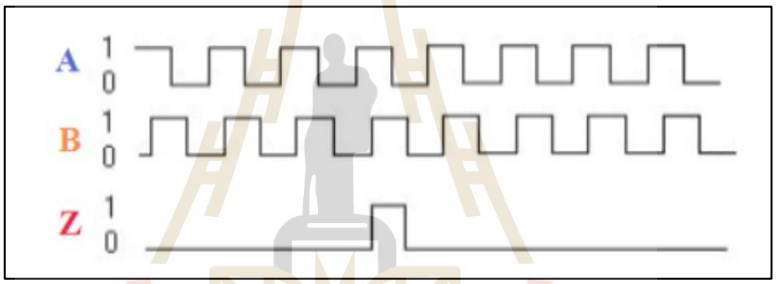
เอ็นโค้ดเดอร์ (Encoder) เป็นเซ็นเซอร์สำหรับวัดระยะทาง (Distance Sensor), ความเร็ว (Speed), ทิศทางการหมุนของมอเตอร์ (Direction of Rotation), ตำแหน่งหรือมุม ที่ใช้ในงานอุตสาหกรรม ซึ่งสามารถแบ่งได้ 2 ประเภท ตามลักษณะของสัญญาณเอาต์พุต (Output signal) ได้ดังนี้

1. Encoder แบบ Increment หรือที่เรียกว่า “Increment Encoder/Rotary Encoder”

Incremental encoder หรือ Incremental Rotary Encoder (เอ็นโค้ดเดอร์แบบหมุน) โครงสร้าง (ดังรูปที่ 2.3) จะประกอบด้วย จานหมุน และอุปกรณ์ตรวจจับ โดยจานหมุนจะมีช่องเล็ก ๆ เมื่อเพลาของมอเตอร์หมุนจะทำให้จานหมุนไปตัดลำแสงของเซ็นเซอร์ (Sensor) ทำให้ชุดรับแสงได้รับสัญญาณเป็นช่วง ๆ จึงทำให้สัญญาณเอาต์พุตออกมาเป็นสัญญาณพัลส์ต่อรอบ (PPR) ดังรูปที่ 2.4



รูปที่ 2.8 แสดงส่วนประกอบของ Incremental encoder/Rotary encoder
(ที่มา: http://eng.sut.ac.th)



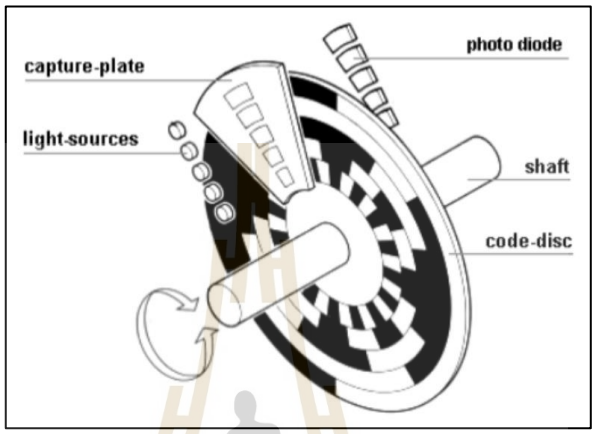
รูปที่ 2.9 ตัวอย่างสัญญาณพัลส์เอาต์พุตของ Incremental encoder
(ที่มา: https://www.primusthai.com)

Incremental encoder เป็นเอ็นโค้ดเดอร์ (Encoder) ที่ใช้หลักการเมื่อมีการหมุนของแกนเพลา จะทำให้มีสัญญาณเอาต์พุตที่เป็นสัญญาณลูกคลื่นพัลส์สี่เหลี่ยม (Square wave) มี 3 แทรค (Tracks) คือ A, B, Z โดยจะสัมพันธ์กับระยะการเคลื่อนที่และตำแหน่งสัญญาณเอาต์พุตของ Encoder A และ B มีมุมที่ห่างกัน 90 องศา ทางไฟฟ้า ส่วน Z จะมีสัญญาณ 1 พัลส์ ต่อ 1 รอบ หรือบางตัวจะเป็นพัลส์แบบ Invert เช่น A-, B-, Z- ซึ่งเป็นสัญญาณที่กลับเฟสกัน 90 องศา เพื่อเช็คทิศทางการหมุนของมอเตอร์ เป็นต้น

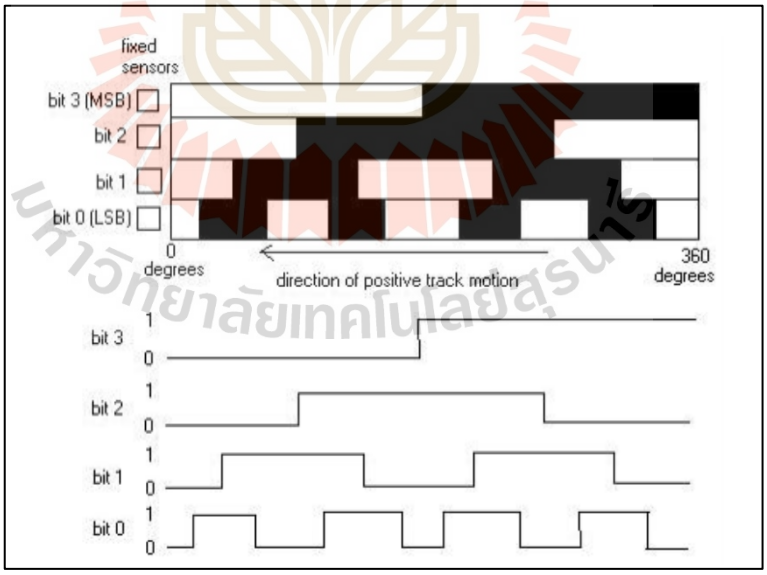
Incremental Encoder แบบนี้จะมีข้อด้อยในกรณีหากมีการถอดสายสัญญาณออกชั่วขณะหรือแหล่งจ่ายไฟฟ้าดับข้อมูลของการเคลื่อนที่ก็จะหายไปหมด ไม่สามารถระบุตำแหน่งพัลส์หรือตำแหน่งองศาได้ ทำให้ต้องมีการปรับที่จุดอ้างอิงใหม่อยู่ตลอดเวลา กรณีนี้อาจจำเป็นต้องใช้เครื่องนับจำนวนแบบตัวเลข (Digital counter) เชื่อมต่อกับคอมพิวเตอร์เพื่อช่วยบันทึกข้อมูลได้

2. Encoder แบบ Absolute หรือที่เรียกว่า “Absolute encoder”

โครงสร้าง (ดังรูปที่ 2.5) จะมีหัวอ่านหลายชุดเท่ากับจำนวนบิตเอาต์พุต การเจาะรูบนแผ่นแต่ละชุดก็จะมีระยะห่างเป็นทวีคูณทำให้สามารถทราบตำแหน่งของการหมุนจึงทำให้สัญญาณออกมาในรูปแบบของรหัสโค้ด เช่น Binary, Gray Code เป็นต้น (ดังรูปที่ 2.6)



รูปที่ 2.10 แสดงส่วนประกอบของ Absolute encoder (ที่มา: http://eng.sut.ac.th)



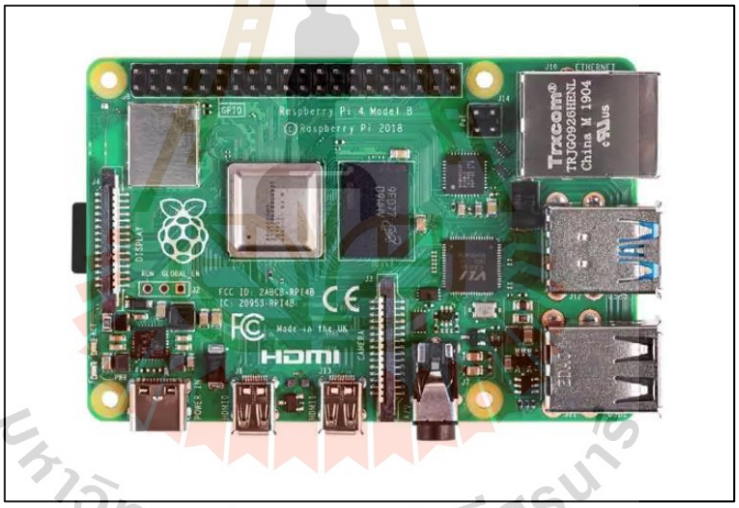
รูปที่ 2.11 แสดงส่วนประกอบของ Absolute encoder (ที่มา: http://www.9engineer.com)

Absolute Encoder เป็นเอ็นโค้ดเดอร์ (Encoder) ที่ออกแบบมาให้มีรูปแบบสัญญาณเอาต์พุตที่เป็นลักษณะของการเข้ารหัส โดยการใช้รหัสแทนสัญญาณพัลส์ เช่น Binary, Gray code

เป็นต้น เพื่อระบุตำแหน่งการเคลื่อนที่และองศาของแกนเอ็นโค้ดเดอร์ได้มีตำแหน่งที่ถูกต้องและแม่นยำมากที่สุด กรณีแหล่งไฟฟ้าหยุดและทำการจ่ายไฟเข้าไปใหม่ข้อมูลก็ยังอยู่ที่ตำแหน่งเดิมและบ่งบอกได้ว่าตำแหน่งองศาที่อยู่นั้นคือเท่าใด แต่โดยทั่วไป Absolute encoder จะมีราคาที่แพงกว่าแบบ Incremental encoder ดังนั้นผู้ใช้งานสามารถเลือกตามความเหมาะสม

2.11 บอร์ดราสเบอร์รี่พาย 4 โมเดล บี (Raspberry Pi4 Model B)

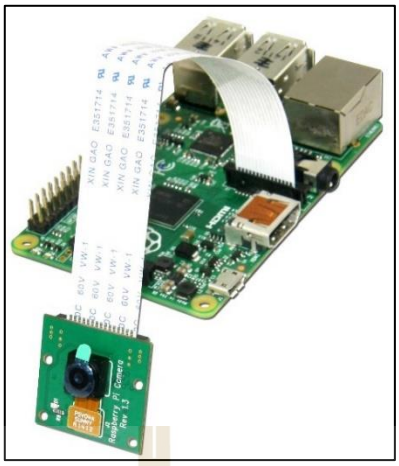
บอร์ดราสเบอร์รี่พายสี่ โมเดลบี คือ เครื่องคอมพิวเตอร์ขนาดเล็ก สามารถทำงานพื้นฐานได้เหมือนคอมพิวเตอร์ทั่วไป ใช้หน่วยประมวลผล Bradcom BCM2711 แบบ Quad-Core ARM Cortex-A72 @ 1.5GHz มีพอร์ตจีพีไอโอ 40 พอร์ต ใช้แรงดันไฟฟ้า 5 VDC ในการทำงาน โดยบอร์ดราสเบอรี่พายสี่ โมเดลบี สามารถเชื่อมต่อเครือข่าย WIFI ได้ และมีความเร็วเพิ่มขึ้นเมื่อเทียบกับรุ่นก่อนหน้า นอกจากนี้ยังทำหน้าที่ติดต่อสื่อสารกับบอร์ดอาดุยโนเมกะ 2560



รูปที่ 2.12 บอร์ดราสเบอร์รี่พายสี่ โมเดลบี Raspberry Pi 4 Model B 4 GB

2.12 กล้องราสเบอร์รี่พาย เวอร์ชัน2 (Raspberry Pi Camera V2 8MP)

Raspberry Pi Camera V2 เป็นอุปกรณ์โมดูลกล้องสำหรับบอร์ด Raspberry Pi นี้มีขนาดเล็กกะทัดรัดเพียง 23.86×25×9 มม. และสามารถเชื่อมต่อใช้งานได้โดยตรงกับบอร์ด Raspberry Pi โดยใช้สายแพร์ด้วยบัส CSI (Common system interface) ดังแสดงในรูปที่ 2.12 ซึ่งเป็นการเชื่อมต่อแบบ Point to Point บัส CSI นี้ถูกพัฒนาโดย Intel ซึ่งออกแบบมาเพื่อการรับส่งข้อมูลความเร็วสูง 12-16 GB/s ด้วยการใช้เทคนิค Low-voltage differential signaling ใช้กระแสไฟฟ้าต่างเหมาะกับอุปกรณ์กล้องที่ต้องถ่ายข้อมูลจำนวนมากอย่างรวดเร็ว



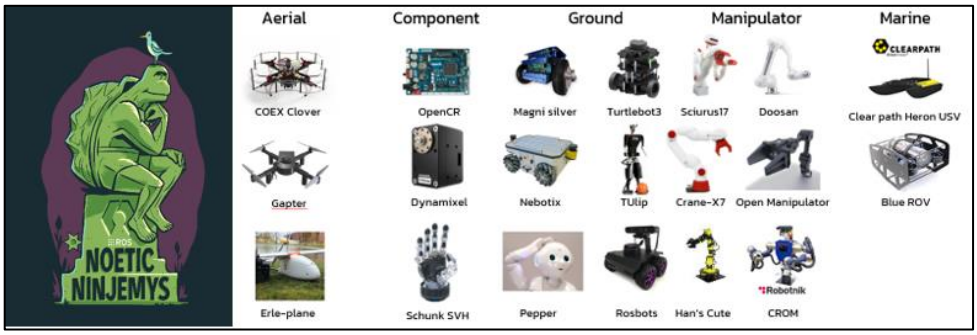
รูปที่ 2.13 แสดงการเชื่อมต่อโมดูลกล้องกับบอร์ด Raspberry Pi โดยผ่านพอร์ต CSI
(ที่มา: http://www.etteam.com)

คุณสมบัติทางเทคนิคของ Raspberry Pi Camera V2 8MP

- ความละเอียดสูง 8 ล้านพิกเซล
- ถ่ายวิดีโอคุณภาพระดับ HD ความคมชัด 1080p, 720p และ 640×480 ด้วยอัตราแสดงผล 30 (1080p), 60 (720p และ 640×480) และ 90 (640×480) เฟรมต่อวินาที
- ขนาดของโมดูลกล้อง 23.86×25×9 ม.ม. และน้ำหนัก 3 กรัม
- การเชื่อมต่อกับบอร์ด Raspberry Pi ด้วยบัส CSI (Common System Interface)
- ภาพมีความคมชัดสูงเมื่อถ่ายในที่มีแสงเหมาะสมและระยะ 1.5 เมตรขึ้นไป

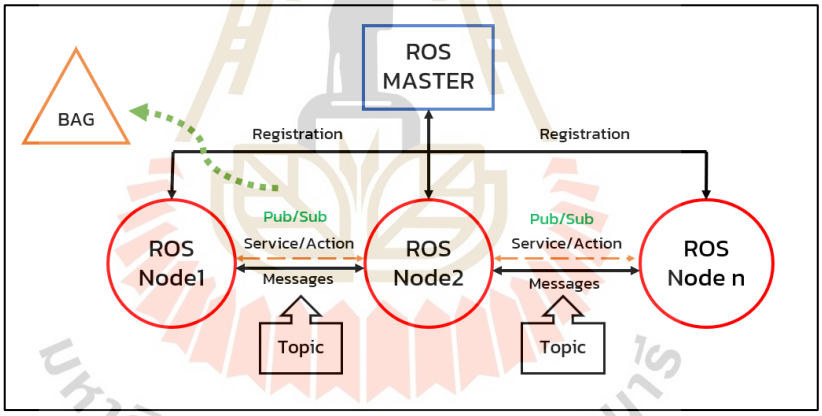
2.13 ระบบปฏิบัติการหุ่นยนต์ หรือ Robot Operating System: ROS

ระบบปฏิบัติการหุ่นยนต์ หรือ Robot Operating System : ROS เป็น framework สำหรับการเขียนซอฟท์แวร์ในการพัฒนาหุ่นยนต์ให้ง่าย มีระบบและพัฒนาหุ่นยนต์ที่มีความซับซ้อนได้ โดยระบบปฏิบัติการหุ่นยนต์เป็น Open-source software เปิดให้ใช้ฟรี ในตัวของ ROS นั้นจะประกอบด้วย เครื่องมือ, library และอุปกรณ์ต่าง ๆ ที่ช่วยในการพัฒนาหุ่นยนต์ในหลาย ๆ แพลตฟอร์ม และยังมีระบบนิเวศของผู้ที่ใช้ ROS อยู่ทั่วโลกอย่างกว้างขวางอีกด้วย ROS ถูกนำไปใช้ในหุ่นยนต์หลายประเภท ทั้งโดรน, หุ่นยนต์เคลื่อนที่ (Mobile robot), แขนกล (Manipulator) และเรือ ROS นั้นเนื่องจากเป็น Open source software ดังนั้นจะเน้นการพัฒนาให้ใช้งานบนระบบปฏิบัติการยูนิกซ์ (Unix) หรือลินุกซ์ (Linux) แต่ในปัจจุบันก็มีการทำให้มีการใช้งานได้บนวินโดว์อย่างต่อเนื่องแต่ยังไม่สมบูรณ์



รูปที่ 2.14 ตราสัญลักษณ์ของ ROS เวอร์ชันล่าสุด (ซ้าย) ตัวอย่างหุ่นยนต์ที่ใช้ ROS ในการพัฒนา (ขวา) (ที่มา : ROS.org)

จุดเด่นของ ROS คือ การจัดระบบให้หุ่นยนต์แบ่งการทำงานเป็น Node แต่ละ Node แล้วมาเชื่อมต่อกันผ่าน Master 1 ตัว มีระบบการส่งข้อมูลแบบ Publish – Subscribe ดังนี้



รูปที่ 2.15 แผนผังหลักการทำงานของระบบปฏิบัติการหุ่นยนต์

ตัวอย่างการทำงาน เช่น Node 1 ควบคุมมอเตอร์ของหุ่นยนต์ Node 2 ควบคุมการทำงานของกล้อง แต่ละ Node ก็จะทำงานแยกกันไป และเขียนโปรแกรมแยกกัน แล้ว Node แต่ละ Node จะส่งข้อความมาติดต่อสื่อสารกันผ่าน ROS master ซึ่งทำหน้าที่เป็นนายทะเบียนข้อมูลที่ถูกส่งหากันเรียกว่า “Message” โดย Message จะมีหัวข้อของตัวเองเรียกว่า “Topic” ด้วยการเขียนโปรแกรมแบบนี้จะทำให้โปรแกรมที่ซับซ้อนสามารถแยกออกไปหลาย ๆ ส่วนได้ง่าย สามารถเขียนโค้ดได้หลายคนแล้วนำมารวมกันได้ง่าย เพราะจะมีมาตรฐานในการส่ง Message กำกับอยู่ด้วย นอกจากนี้การที่ Node แต่ละ Node ถกเขียนโปรแกรมแยกกันออกไปทำให้เราสามารถ

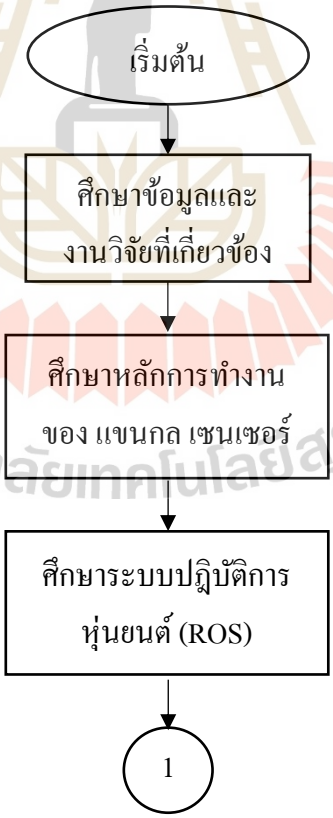
แก้ไขโปรแกรมได้ง่าย ๆ โดยที่ไม่กระทบกับโปรแกรมในส่วนของ Node อื่น โดยสามารถเขียนด้วยด้วยภาษา Python หรือ CPP ก็ได้และอื่น ๆ นอกจากนี้เครื่องมือต่าง ๆ ที่ถูกพัฒนาขึ้นมาใช้กับ ROS มีเยอะมาก ทั้งในด้านการทำ simulation, การควบคุมพฤติกรรมของหุ่นยนต์ และอื่น ๆ เช่น โปรแกรม Gazebo simulation, FlexBe Behavior Engine และ Move It Path Planning เป็นต้น

ระบบปฏิบัติการหุ่นยนต์จะมี package สำหรับการทำงานเบื้องต้นให้ เช่น Package สำหรับการเชื่อมต่ออุปกรณ์ผ่านพอร์ตอนุกรม (Serial port) ซึ่งมีชื่อว่า “Rosserial package” และ “Package” สำหรับแปลงภาพที่รับเข้ามาจากระบบปฏิบัติการหุ่นยนต์ให้เป็นภาพที่สามารถใช้งานกับไลบราลีของ OpenCV ได้ เมื่อว่า Ros CV_bridge เป็นต้น นอกจากนี้ ROS ยังมีเครื่องมือด้านระบบเครือข่ายเพื่อส่งข้อมูลไปมาระหว่างหลาย ๆ อุปกรณ์ได้อีกด้วย เรียกว่า ROS network หรือระบบเครือข่ายของระบบปฏิบัติการหุ่นยนต์ ที่ช่วยให้อุปกรณ์ที่อยู่บนเครือข่ายอินเทอร์เน็ตเดียวกัน สามารถส่งข้อมูลไปมาหากันได้ เช่น ส่งข้อมูลจากบอร์ดราสเบอร์รี่พายไปพีซีคอมพิวเตอร์เพื่อทำการประมวลผลก่อนแล้วจึงส่งกลับไปที่บอร์ดราสเบอร์รี่พายอีกครั้งเพื่อสั่งการอุปกรณ์มอเตอร์เป็นต้น

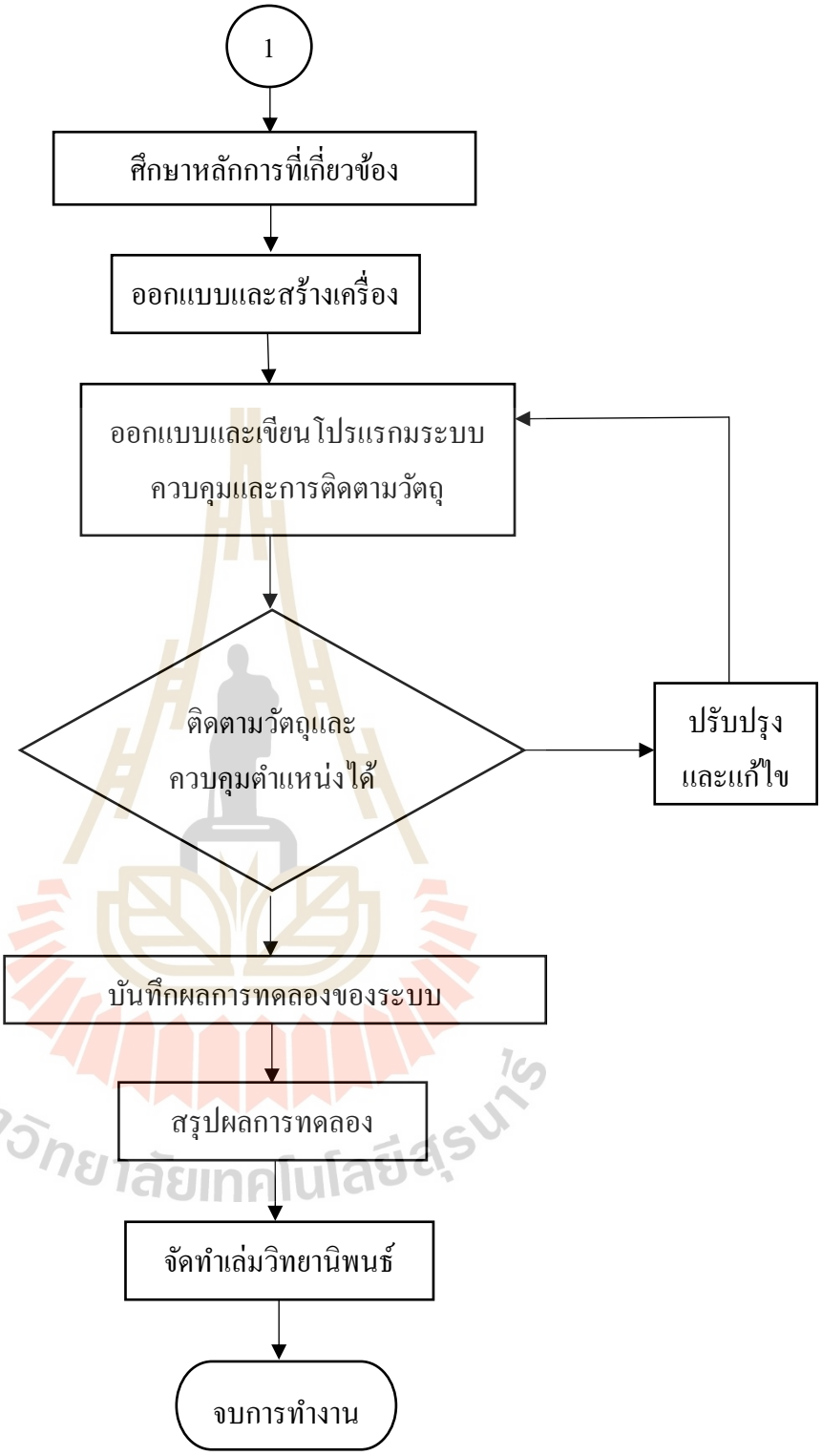
บทที่ 3
วิธีดำเนินการวิจัย

3.1 กล่าวนำ

งานวิจัยนี้ได้ทำการออกแบบและสร้างระบบควบคุมตำแหน่งและแนวการวางตัวของแขนหุ่นยนต์ 2 แกน ให้มีเสถียรภาพในตำแหน่งที่ต้องการ บนพาหนะขณะเคลื่อนที่ เพื่อนำไปประยุกต์ใช้ในการพัฒนาแขนหุ่นยนต์ในภาคอุตสาหกรรม โดยทำการศึกษาและหาข้อมูลการทำงานของ แขนกล เซนเซอร์ กล้อง และไมโครคอนโทรลเลอร์ ออกแบบและคำนวณหาสมการทางคณิตศาสตร์นำไปหาพารามิเตอร์ของระบบ เพื่อความเหมาะสมในการออกแบบและสร้างระบบควบคุม



รูปที่ 3.1 ขั้นตอนการทำงานวิจัย (ต่อ)



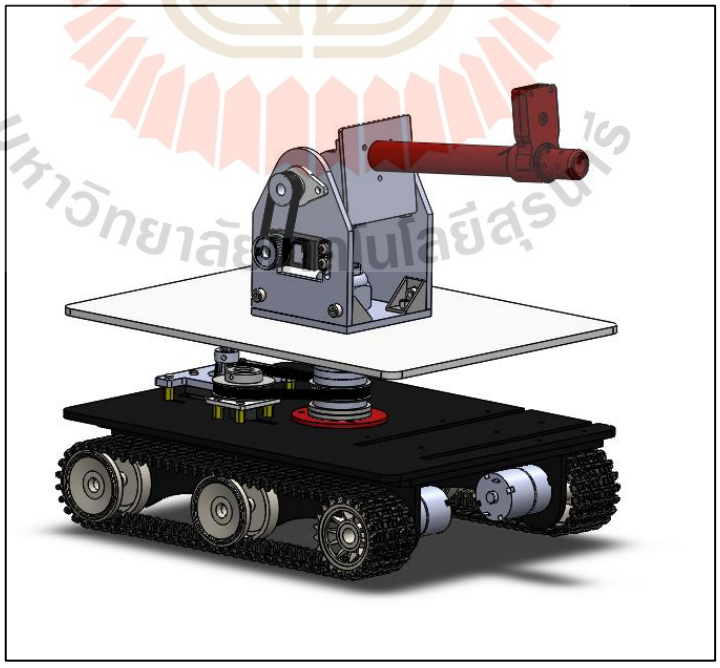
รูปที่ 3.2 ขั้นตอนการทำงานวิจัย

3.2 การออกแบบและสร้างหุ่นยนต์ต้นแบบ

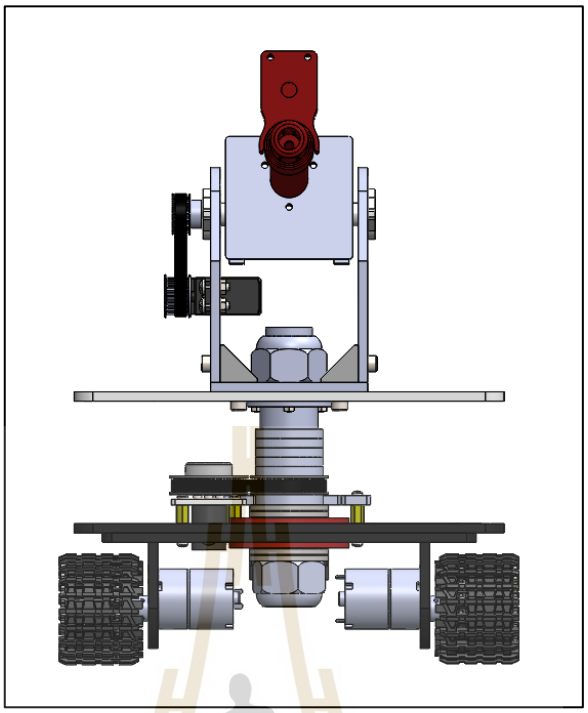
เนื่องจากในงานวิจัยนี้ต้องการสร้างหุ่นยนต์ต้นแบบ เพื่อนำมาใช้กับโปรแกรมประมวลผลภาพที่ออกแบบขึ้น โดยในส่วนนี้ผู้วิจัยได้สร้างหุ่นยนต์บนพาหนะเคลื่อนที่ ซึ่งแขนของหุ่นยนต์สามารถเคลื่อนที่ได้ 2 แกน โดยแกนที่ 1 สามารถเคลื่อนที่ได้ 360 องศาในแนวระดับ และแกนที่ 2 เคลื่อนที่ได้ 90 องศาในแนวดิ่ง โดยแขนหุ่นยนต์จะเคลื่อนที่ได้จากการรับค่าสัญญาณจาก Arduino MEGA2560 ซึ่งได้รับการสั่งการตำแหน่งจากการประมวลผลภาพมาจาก Raspberry Pi อีกที

3.2.1 การออกแบบระบบเชิงกล

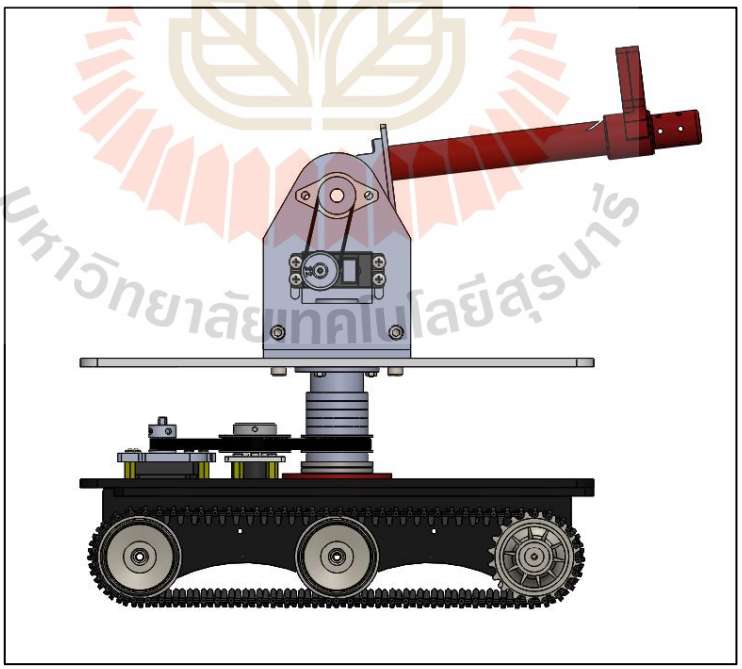
ในการออกแบบหุ่นยนต์เชิงกล ผู้วิจัยต้องออกแบบหุ่นยนต์ที่มีแขนหุ่นยนต์ 2 แกน บนพาหนะเคลื่อนที่ โดยที่เมื่อระบุเป้าหรือภาพที่ต้องการแล้ว เมื่อพาหนะเคลื่อนที่ไปทิศทางใดหรือระยะใด แขนของหุ่นยนต์ทั้ง 2 แกน จะต้องเคลื่อนที่ไปยังเป้าหรือภาพนั้น โดยมีกล้อง Raspberry Pi Camera V2 และเลเซอร์สีแดง ติดที่อยู่ปลายแขนของหุ่นยนต์ในแกน 2 ซึ่งกล้อง Raspberry Pi Camera V2 ทำหน้าที่รับภาพ เพื่อส่งให้ Raspberry Pi 4 model B ประมวลผลภาพ และส่งค่าไปยังบอร์ด Arduino MEGA 2560 เพื่อคำนวณค่าคลาดเคลื่อนและควบคุมแขนหุ่นยนต์ทั้ง 2 แกน โดยแกนที่ 1 ควบคุม Stepper Motor ให้เคลื่อนที่ 0-360 องศา และ แกนที่ 2 ควบคุม Servo Motor ให้เคลื่อนที่ขึ้นลงได้ 0-90 องศา มีการออกแบบโดยใช้โปรแกรม SolidWorks ดังรูปที่ 3.1



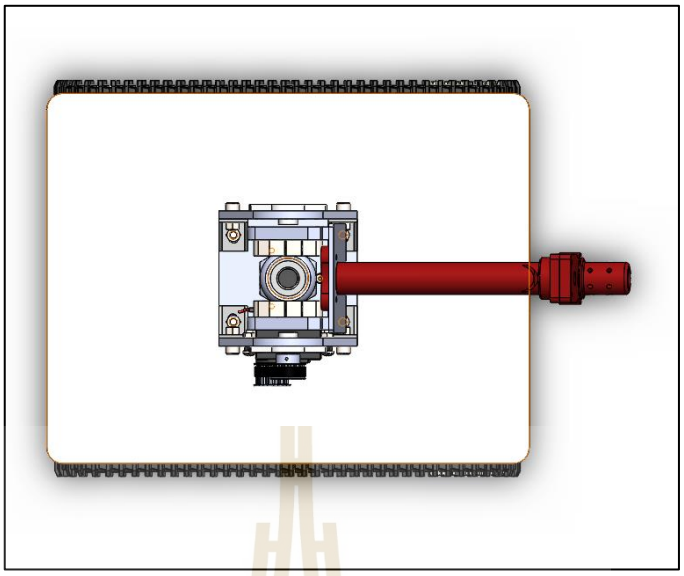
รูปที่ 3.3 รูป 3 มิติ ของหุ่นยนต์ในโปรแกรม SolidWorks



รูปที่ 3.4 รูป 3 มิติ ของหุ่นยนต์ในโปรแกรม SolidWorks (ด้านหน้า)



รูปที่ 3.5 รูป 3 มิติ ของหุ่นยนต์ในโปรแกรม SolidWorks (ด้านข้างขวา)

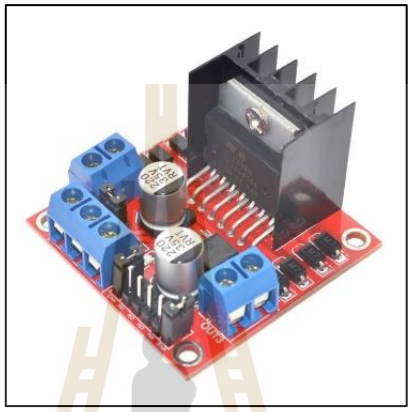


รูปที่ 3.6 รูป 3 มิติ ของหุ่นยนต์ในโปรแกรม SolidWorks (ด้านบน)

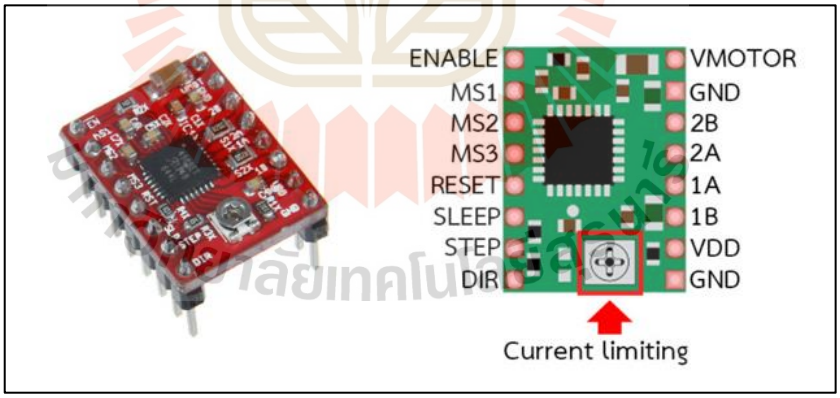
รูปที่ 3.7 หุ่นยนต์ 2 แกน บนพาหนะเคลื่อนที่ที่ประกอบและต่อวงจร

เมื่อทำการสร้างและติดตั้งระบบเชิงกล จำเป็นต้องมีวงจรไฟฟ้าสำหรับใช้ควบคุม โดยมีบอร์ด Microcontroller Arduino เป็นตัวสั่งการ ซึ่งมีข้อจำกัดของบอร์ด Arduino ไม่สามารถจ่ายแรงดันไฟฟ้าได้สูงพอที่จะขับ DC Motor ที่ 12 โวลต์ และ Stepper motor ที่ 24 โวลต์ได้

โดยสามารถจ่ายไฟได้สูงสุด 5 โวลต์ จึงจำเป็นที่จะต้องมีบอร์ดขยายแรงดัน โดยจะควบคุมสัญญาณ analog PWM ที่มีค่าระหว่าง 0 ถึง 255 โดยเลือกใช้บอร์ด L298N สำหรับ DC Motor 12 โวลต์ ซึ่งสามารถรับไฟเข้า 7-35 โวลต์ ขับกระแสสูงสุดได้ 2 A และใช้บอร์ดไดร์ฟ Stepper motor 24 โวลต์ เป็น บอร์ด A4988 มีแรงดันสูงสุด 35 โวลต์ สามารถรองรับการทำงานใน Step Mode ได้อย่างครบถ้วนทั้ง 1/1(Full Step), 1/2(Half Step), 1/4, 1/8, 1/16



รูปที่ 3.8 บอร์ดไดร์มอเตอร์ L298N



รูปที่ 3.9 บอร์ดไดร์สเต็ปเปอร์มอเตอร์ A4988

ในการใช้งานโมดูล A4988 นั้นจะต้องมีการจ่ายไฟ 2 ชุด ชุดแรกเป็นไฟแรงต่ำต่อเข้ายัง VCC และ GND (ที่มุมล่างขวา) เป็นไฟที่ใช้สำหรับเลี้ยงไอซีและวงจร ชุดนี้จะใช้ไฟ 3-5 โวลต์ ส่วนชุดที่สองเป็นไฟสำหรับขับมอเตอร์ เป็นชุดไฟแรงสูงตามแรงดันของมอเตอร์ที่เราใช้งาน หากเราใช้มอเตอร์ 24 โวลต์ก็ต่อไฟ 24 โวลต์เข้าไปที่ไฟชุดนี้ (ที่มุมบนขวา)

การตั้งค่า Step Mode

A4988 นั้นสามารถตั้งค่า Step Mode ได้หลายแบบ โมดูล A4988 ก็ทำมาให้เราสามารถตั้งค่าได้โดยการใช้ขา M1, M2 และ M3 โดยการต่อขาต่าง ๆ เข้ากับแรงดันลอจิกต่าง ๆ ดังนี้

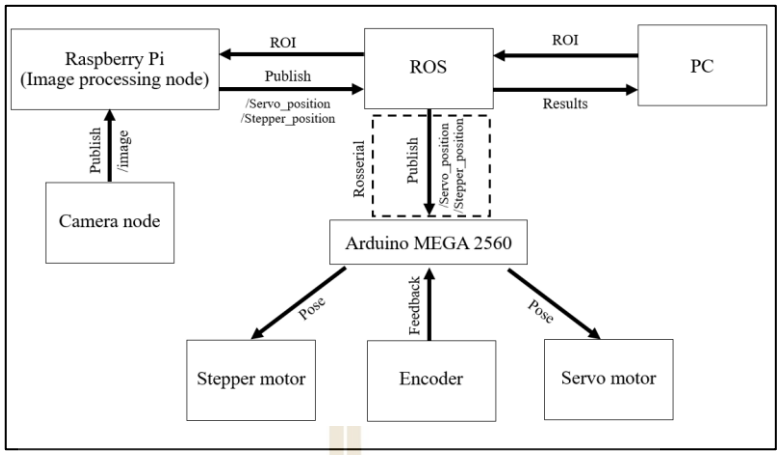
ตารางที่ 3.1 ลอจิกต่าง ๆ ในการควบคุมสเต็ปเปอร์มอเตอร์

MS1	MS2	MS3	Resolution
LOW	LOW	LOW	1/1 Full
HIGH	LOW	LOW	1/2 Half
LOW	HIGH	LOW	1/4 Quarter
HIGH	HIGH	LOW	1/8 Eighth
HIGH	HIGH	HIGH	1/16 Sixteenth

3.2.2 การออกแบบระบบซอฟต์แวร์

การออกแบบระบบซอฟต์แวร์ใช้ระบบปฏิบัติปฏิบัติการหุ่นยนต์ (ROS Version Noetic) เป็น Framework ในการพัฒนาซอฟต์แวร์และสื่อสารข้อมูลระหว่างอุปกรณ์ส่วนต่าง ๆ โดย ROS Noetic ถูกติดตั้งบน Ubuntu 20.04 focal fossa ซึ่งใช้ภาษา Python เป็น Python 3 ส่วนซอฟต์แวร์ที่ใช้ทำการประมวลผลภาพ คือ OpenCV เวอร์ชั่น 4.3 ซอฟต์แวร์ที่ใช้เขียนโปรแกรมลงใน Microcontroller ใช้โปรแกรม Arduino IDE เขียนควบคุมการทำงาน Stepper Motor กับ Encoder และ Servo Motor

โปรแกรมที่เขียนลงใน Microcontroller ประกอบด้วย ฟังก์ชั่นที่ใช้ควบคุมตำแหน่งของ Stepper Motor และรับ feedback จาก Encoder ตัว controller ที่ใช้เป็นแบบ P Controller Orientation ของ Stepper Motor ถูกจำกัดในช่วง 0 ถึง 360 องศา ฟังก์ชั่นถัดไป ใช้ควบคุมตำแหน่งของ Servo Motor โดยมุมถูกจำกัดในช่วง 0 ถึง 90 องศา นอกจากนี้จะมีฟังก์ชันสำหรับการสร้าง ROS Node เพื่อสื่อสารกับROS ผ่าน ROS Serial Library และรอรับ Input ที่ส่งมาจากการคำนวณของอัลกอริธึมที่ใช้ในการติดตามวัตถุ ตั้งแต่ข้อมูลรูปภาพถูกรับเข้ามาจนกระทั่งมีคำสั่งออกไปให้ Actuator ทำงาน เพื่อติดตามวัตถุ

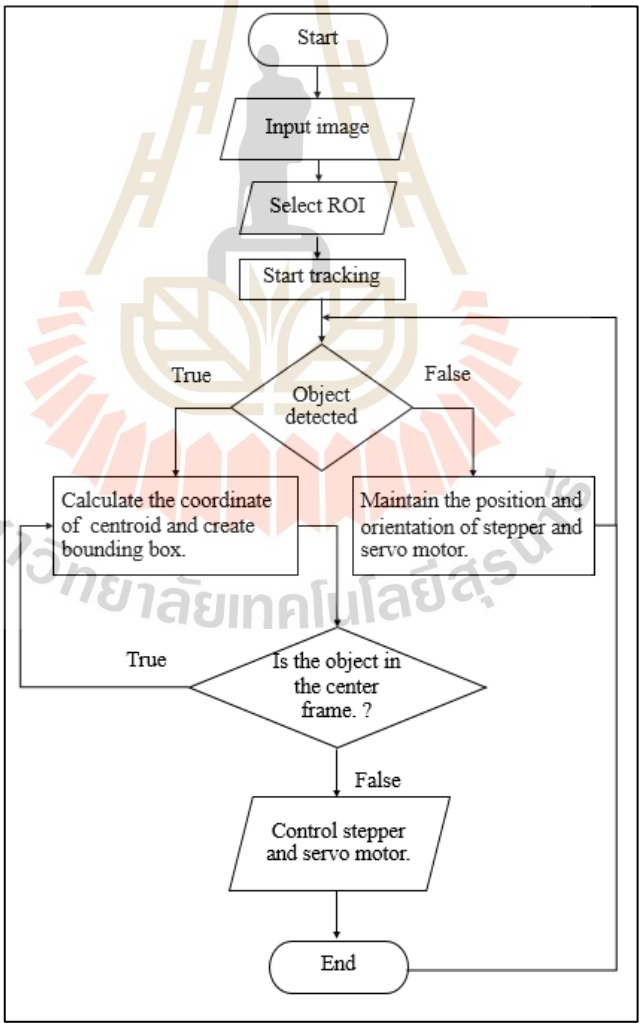


รูปที่ 3.10 แผนผังการส่งข้อมูลของอุปกรณ์

จากรูปที่ 3.10 แผนผังการส่งข้อมูลของอุปกรณ์ การทำงานจะเริ่มจากการรับภาพจากกล้องเข้ามาในขนาด Pixel ที่ต้องการ โดยหากเลือก Pixel ที่มีขนาดใหญ่เกินไปจะทำให้การส่งข้อมูลล่าช้า แต่ถ้าหาก Pixel น้อย จะทำให้ความเร็วการส่งข้อมูลมีมากและใช้ Computational cost ในการประมวลผลที่น้อย แต่จะทำให้รายละเอียดของภาพบางส่วนตกหล่นไปด้วย ซึ่ง Pixel ที่เลือกใช้ในงานวิจัย นี้จะมีขนาด Pixel ไม่เกิน 308×210 Pixel สำหรับกรณีที่ใช้โมเดลการติดตามวัตถุแบบ KCF, CSRT และ MOSSE แต่ในการใช้งานร่วมกับการเรียนรู้เชิงลึกจะใช้ขนาดไม่เกิน 205×154 Pixel ในส่วนของการส่งข้อมูลรูปภาพ เราจะสร้าง ROS camera node ขึ้นมาสำหรับการรับภาพโดยเฉพาะแล้วส่งข้อมูลไปยัง Node การทำงานอื่นผ่าน ROS Master

รูปที่ 3.11 เป้าหมายที่เลือกและมีการ Tracking

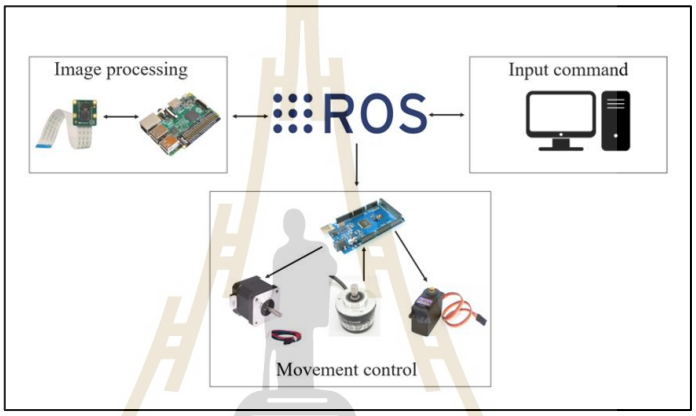
ภาพจาก Camera node จะถูกส่งไปที่ Processing node โดย Processing node จะรอรับ ROI (Region of interest) จากผู้ใช้ก่อน สำหรับงานวิจัยนี้อัลกอริทึมในการติดตามวัตถุ (Deep learning) จะทำการติดตามวัตถุตลอดเวลาเมื่อวัตถุมีการเคลื่อนไหว และสร้างกรอบขอบเขตรอบวัตถุเอาไว้ จากนั้นจะใช้สมการทางคณิตศาสตร์คำนวณหาค่า Centroid ของวัตถุหรือภาพที่ติดตามทั้งในทิศทางแนวตั้งและแนวนอน เพื่อที่จะนำพิกัดนั้นไปเป็นจุดอ้างอิงสำหรับการเคลื่อนที่ของวัตถุ ทำการหาจุดกลางขอบเขตของกรอบ เมื่อวัตถุหลุดจากขอบเขตที่กำหนดไว้ Actuator จะทำงาน เพื่อรักษาตำแหน่ง Centroid ของวัตถุหรือภาพ ให้อยู่ในจุดกึ่งกลางของเฟรม โดย Processing node จะทำการส่งตำแหน่งของเซอร์โวและสเต็ปเปอร์มอเตอร์ไปที่ Arduino ผ่าน ROS Serial Arduino port ซึ่งเมื่อ Arduino board รับตำแหน่งมาแล้ว จะนำไปควบคุมสั่งการทำงานของสเต็ปเปอร์ และมอเตอร์เซอร์โวให้อยู่ในตำแหน่งที่ต้องการ



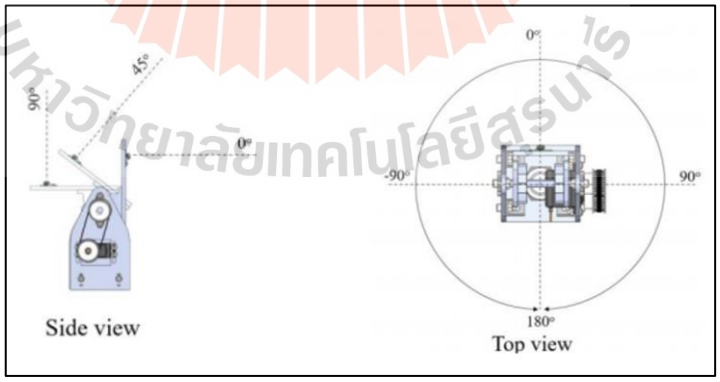
รูปที่ 3.12 Algorithm Work Flow

3.3 องค์ประกอบของระบบ

Tilt and pan robotics arms สามารถเคลื่อนที่ได้ 2 DOF ประกอบด้วย base ที่สามารถหมุนได้ 0 - 360 องศา และแขนของหุ่นยนต์ที่เคลื่อนที่ขึ้นลง 0 - 90 องศา ดังในรูปที่ 3.14 อุปกรณ์ที่ใช้ในการควบคุมประกอบได้ 3 ส่วนหลักอุปกรณ์ของระบบจะประกอบด้วย 3 ส่วนหลัก ดังแสดงในรูปที่ 3.13 ได้แก่ ส่วนของ Image processing , Movement control และส่วนรับ Input command จากผู้ใช้ทั้งสามระบบทำงานเชื่อมต่อกันเป็นเครือข่าย โดยมีระบบปฏิบัติการหุ่นยนต์ (Robot operating system: ROS) เป็นส่วนกลางในการสื่อสารข้อมูลระหว่างทั้งสามส่วน



รูปที่ 3.13 ส่วนประกอบฮาร์ดแวร์ของระบบ



รูปที่ 3.14 การทำงานของ Stepper และ Servo

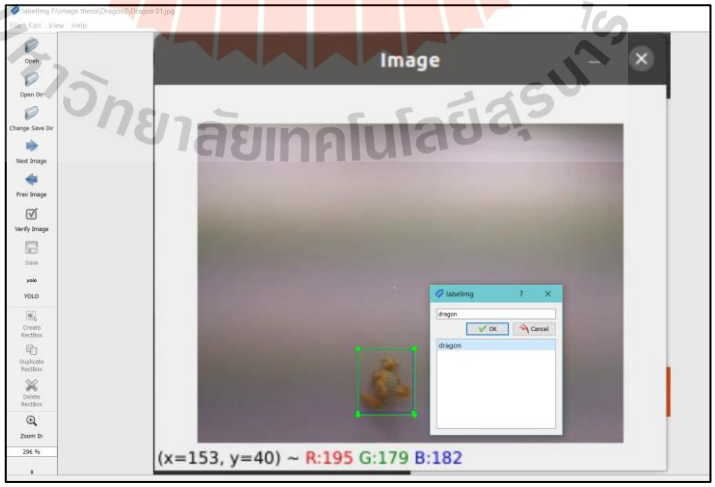
ส่วนของ Input command จะเป็นส่วนรับ ROI (Region of Interest) จากผู้ใช้ว่าต้องการติดตามเป้าหมายใด แล้วจะทำการเทรนโมเดลให้รู้จักกับวัตถุนั้นเอาไว้ก่อน จากนั้นข้อมูลของ

เป้าหมายจะถูกส่งไปยังส่วนของ Image processing ให้ทำงานต่อ ส่วนของ Image processing ประกอบด้วย Raspberry Pi4 และ Pi camera V2 อุปกรณ์ส่วนนี้จะใช้สำหรับรับภาพเข้ามาและทำการประมวลผลภาพ ส่วนสุดท้าย ส่วนของ Movement control ประกอบด้วย Actuators คือ Servo motor และ Stepper motor พร้อมทั้ง Encoder sensor นอกจากนี้ยังมี Microcontroller Arduino MEGA 256 อยู่ด้วย ส่วนของ Movement control นี้จะเป็นส่วนปลายน้ำของระบบ Arduino mega จะคอยรับค่า Output command จาก ส่วน Image processing แล้วสั่งการ Actuator อีกที

3.4 การดำเนินการทดลอง

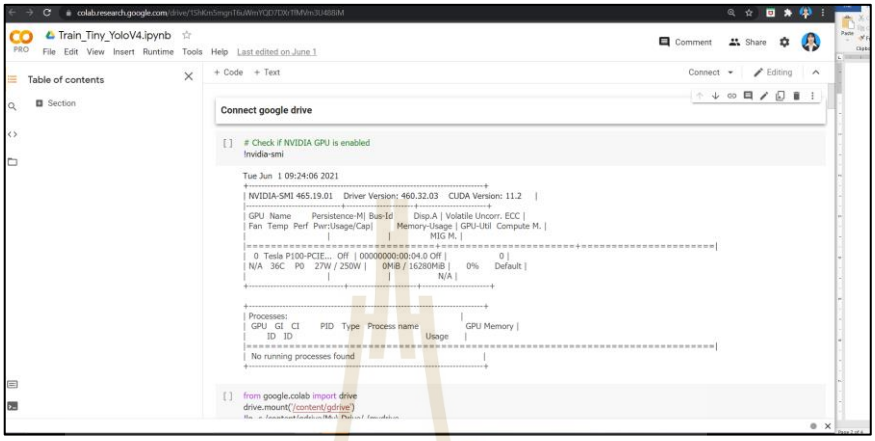
ในตอนเริ่มต้น ผู้วิจัยได้ทำการทดลองเขียนโปรแกรมภาษา Python ที่ใช้อัลกอริธึมในการติดตามวัตถุของ OpenCV โดยได้เลือกใช้โมเดล KCF , CSRT และ MOSSE เมื่อทำการทดสอบโปรแกรม โดยเลือกเป้าวัตถุหรือภาพที่เราต้องการ ผลที่ได้คือสามารถติดตามวัตถุได้ แต่เมื่อเคลื่อนที่เป้าหรือเคลื่อนที่ตัวพาหนะออกจากเป้าแล้วนำกลับมาใหม่จะสามารถกลับมาติดตามอีกครั้งได้ยาก ผู้วิจัยจึงได้ดำเนินการหาวิธีการเพิ่มเติมโดยการใช้การเรียนรู้เชิงลึก (Deep Learning) มาใช้ในงานวิจัยโดยมีขั้นตอนดังนี้

1) เก็บภาพเป้าหมายในมุมต่าง ๆ ใหมากที่สุดเพื่อนำมาฝึก (Train) โมเดล โดยก่อนจะนำไปฝึกนั้น ต้องทำการติด Label ให้กับภาพหรือวัตถุก่อน ซึ่งภาพที่นำมาเป็นภาพที่ได้จากกล้อง Raspberry Pi ที่ติดตั้งบนแขนหุ่นยนต์ ตัวอย่างการติดลาเบลของภาพดังในรูปที่ 3.15 โดยในภาพตัวอย่าง ลาเบลที่ติดมีชื่อว่า “Dragon”

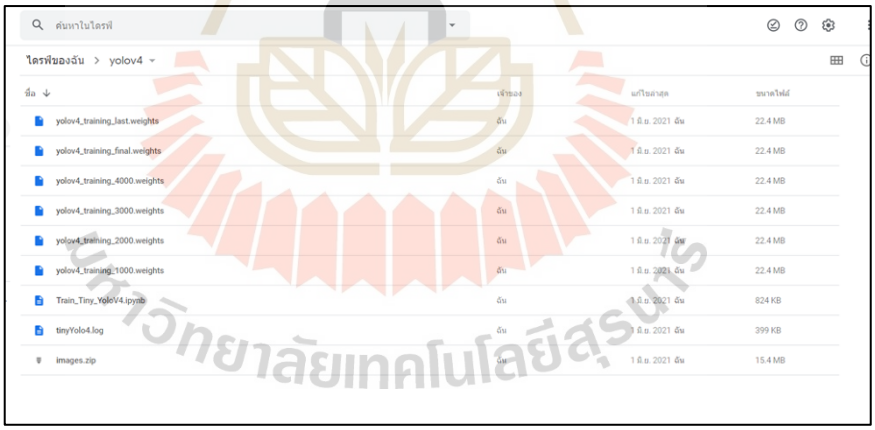


รูปที่ 3.15 ภาพวัตถุหรือเป้าที่ทำการติดลาเบล

2) นำภาพที่ติดลาเบลเรียบร้อยแล้ว ไปฝึกโมเดล โดยใช้ Google Collab เป็นเครื่องมือในการฝึกและการประมวลผล เนื่องจากหากทำการฝึกในคอมพิวเตอร์ปกติจะใช้เวลานาน ส่วน Google Collab จะเป็นบริการคลาวด์คอมพิวเตอร์ มีหน่วยประมวลผลประสิทธิภาพสูง



รูปที่ 3.16 การฝึกโมเดล บน Google Collab

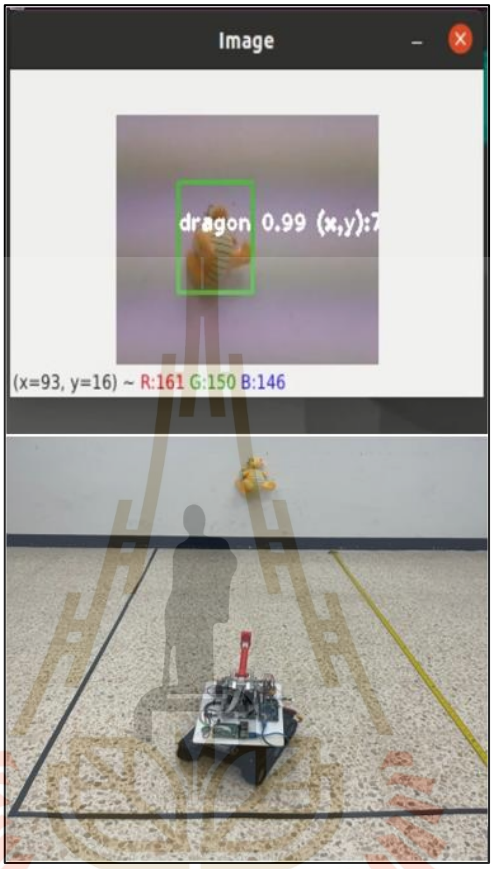


รูปที่ 3.17 ไฟล์ที่ได้หลังจากการฝึกโมเดล

3) เมื่อการฝึกโมเดลเสร็จสิ้น สิ่งที่ได้ออกมาคือ น้ำหนัก (Weight) ของโมเดล น้ำหนักนี้เราจะนำไปใส่ในโค้ดโปรแกรมเพื่อเรียกใช้งานโมเดลการเรียนรู้เชิงลึกที่เราได้ทำการฝึกเอาไว้

4) ทำการเขียนโปรแกรมเพื่อดึงโมเดลการเรียนรู้เชิงลึกที่ฝึกไว้มาใช้ ผ่านโมดูลโครงข่ายประสาทเชิงลึก (Deep Neural Network) ของ OpenCV แล้วนำไปทดสอบการทำงานร่วมกัน

การควบคุมของมอเตอร์ โดยทำการติดตามเป้าหมายที่ระยะแตกต่างกันทั้งแบบรถอยู่กับที่ (Static) และรถเคลื่อนที่ (Dynamics)



รูปที่ 3.18 ภาพที่ได้หลักจากการรันโปรแกรม

5) ทำการบันทึกผลการทดลอง

3.5 การประเมินผล

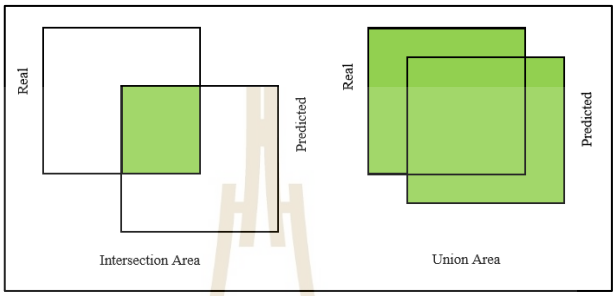
ในการประเมิลผลการตรวจจับและติดตามวัตถุด้วยคอมพิวเตอร์ด้วยการเรียนรู้เชิงลึกนั้น วิธีที่นิยมใช้จะมีอยู่สองวิธีคือ การหาค่าพื้นที่ทับซ้อน (Intersection over Union : IoU) และวิธีการวัดค่าความมั่นใจ (Confidence level) ของโมเดลการเรียนรู้เชิงลึก

3.5.1 Intersection over Union value (IoU)

ในการการวัดผลความแม่นยำของการตรวจจับวัตถุจะใช้การวัดค่า Intersection over Union (IoU) ของการติดตามวัตถุ ซึ่งเป็นการคำนวณพื้นที่ที่ทับซ้อนกันระหว่างวัตถุใน

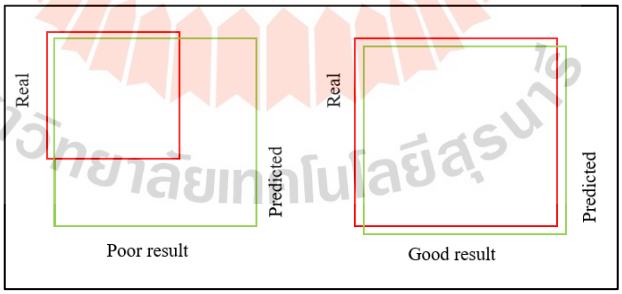
ความเป็นจริงกับสิ่งที่คอมพิวเตอร์สามารถวัดออกมาและสร้างกรอบขอบเขตออกมาได้ โดยสมการที่ใช้คำนวณค่าของ IoU นั้นเป็นดังต่อไปนี้

$$\text{Intersection over union (IoU)} = \frac{\text{Intersection area}}{\text{union area}} \tag{3.1}$$



รูปที่ 3.19 Intersection over Union (IoU)

ค่า IoU ที่มีค่าสูงหมายความว่า พื้นที่ของ Bounding box ที่คอมพิวเตอร์ สร้างนั้นมีค่าใกล้เคียงกับพื้นที่ของวัตถุในความเป็นจริงมาก ดังนั้นยิ่ง IoU มีค่ามาก หมายความว่าความแม่นยำในการตรวจจับและติดตามวัตถุ นั้นมีค่าความแม่นยำสูง



รูปที่ 3.20 (ซ้าย) ผลการติดตามไม่ดี (ขวา) ผลการติดตามที่ดี

3.5.2 Confidence Level

เป็นค่าความมั่นใจในการตรวจจับของโมเดลการเรียนรู้เชิงลึก ค่ามากที่สุดมีค่าเท่ากับ 1 ค่า Confidence level จะแสดงขึ้นมาตอนตรวจจับ ด้านบนของกรอบตรวจจับ ซึ่งแสดงออกมาโดยการเขียนโค้ดภาษา Python ดึงออกมาแสดง

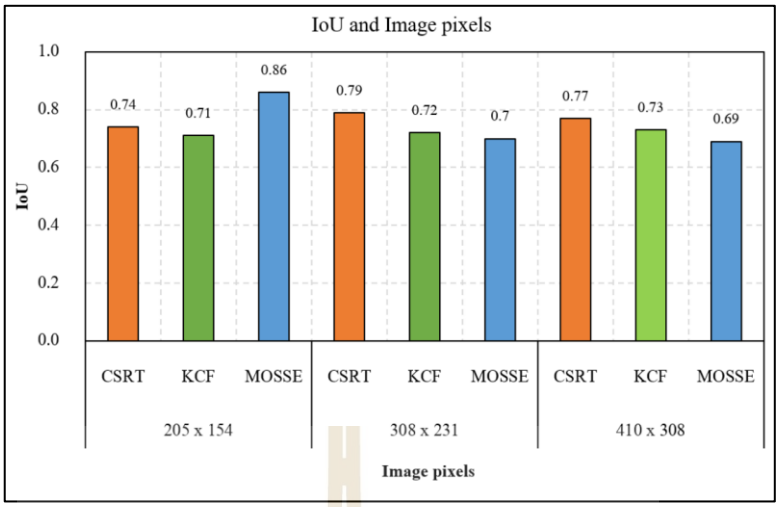
บทที่ 4
ผลการทดลองและวิเคราะห์ผล

4.1 ค่าเฉลี่ยของค่า IoU ของแต่ละโมเดล ในภาพขนาดต่าง ๆ

จากการทดสอบการทำงานของ สเต็ปเปอร์มอเตอร์และเซอร์โวมอเตอร์ จะทำการทดสอบโดยทำการ Track object โดย algorithm ทั้งสามคือ KCF, CSRT และ MOSSE algorithm โดยทำการทดสอบเพื่อวัดความแม่นยำของทั้งสามโมเดล ในภาพที่มีความละเอียดแตกต่างกัน 3 ระดับ คือ 205×154, 308×231 and 410×308 Pixels ตามลำดับ โดยเป้าหมายที่จะให้ทำการ Tracking ในแต่ละกรณีจะเป็นเป้าหมายที่มีการเคลื่อนที่ในรูปแบบเดียวกันและตำแหน่งเดียวกัน สำหรับวิธีการวัดผลความแม่นยำจะใช้การวัด Intersection over Union value (IoU) ของการ Tracking ซึ่งเป็นการคำนวณพื้นที่ที่ทับซ้อนกันระหว่างวัตถุในความเป็นจริงกับสิ่งที่ Object tacking model สามารถวัดออกมาและสร้าง Bounding box ออกมาได้

ตารางที่ 4.1 ค่าเฉลี่ยของค่า IoU ของแต่ละโมเดล ในภาพขนาดต่าง ๆ

Image Pixel	**Model**	**(IoU)**
205×154	CSRT	0.74
	KCF	0.71
	MOSSE	0.86
308×231	CSRT	0.79
	KCF	0.72
	MOSSE	0.70
410×308	CSRT	0.77
	KCF	0.73
	MOSSE	0.69

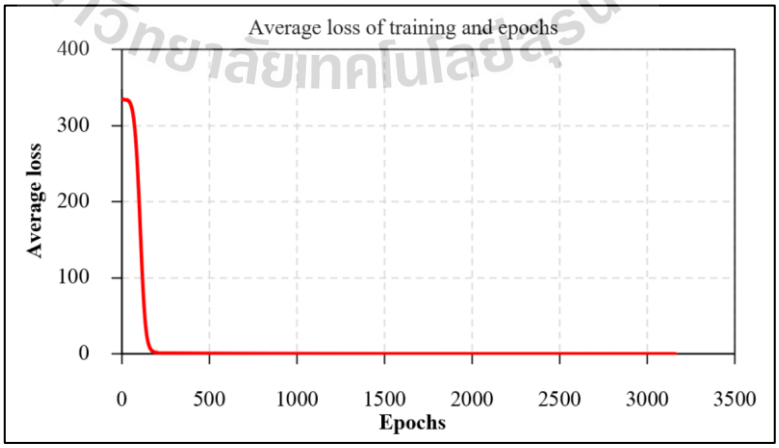


รูปที่ 4.1 กราฟแสดงความสัมพันธ์ระหว่าง IoU และขนาดภาพ

จากรูปที่ 4.1 ขนาดภาพในแต่ละ Pixel ไม่มีผลต่อความแม่นยำในการตรวจจับของโมเดล CSRT และ KCF อย่างมีนัยสำคัญ แต่มีผลกับโมเดล MOSSE

4.2 ผลการฝึก Model การตรวจจับวัตถุด้วย Deep learning

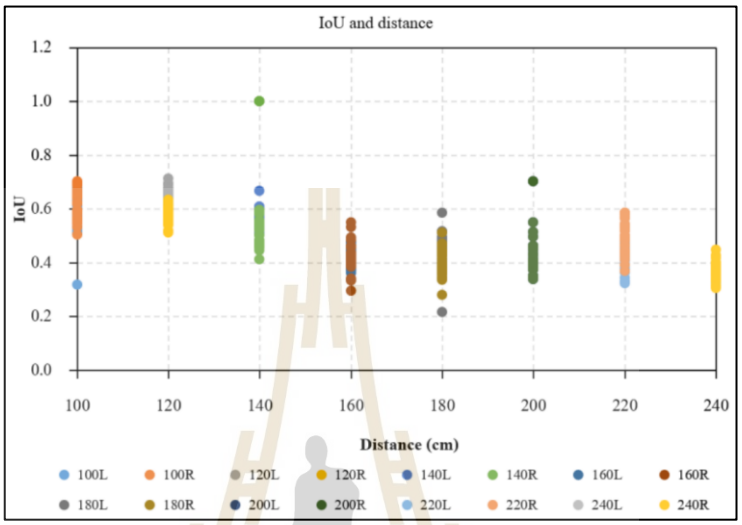
จากการฝึกโมเดล (train model) เมื่อรอบการคำนวณ (Epochs) มีค่าเพิ่มมากขึ้น ค่า Average loss ของการฝึกโมเดลมีค่าน้อยลงจนน้อยกว่า 1 ยิ่ง Average loss มีค่าน้อยแสดงว่าโมเดลที่เทรนนั้นมีความเป็นไปได้ที่จะมีความแม่นยำในการตราวจจับสูง



รูปที่ 4.2 กราฟแสดงผล Average loss of training and epochs

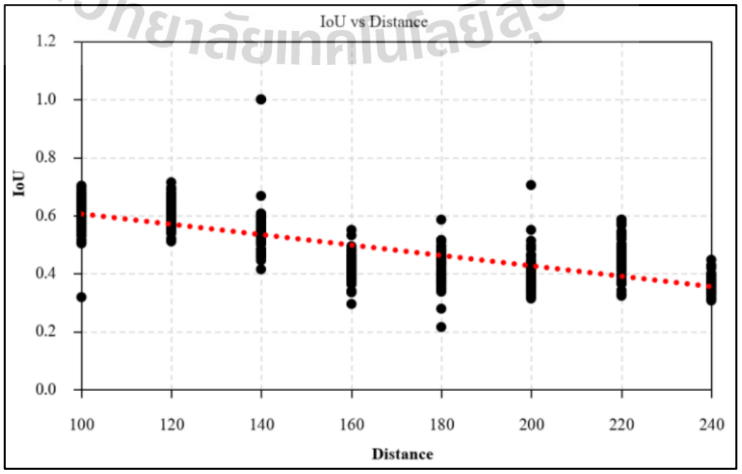
4.3 ผลการทดสอบการติดตามวัตถุด้วย Deep learning แบบ Static test

จากทดสอบการติดตามวัตถุด้วย Deep Learning แบบ static test ความละเอียดของภาพจาก Pi Camera เป็น 205 × 154 Pixels โดยแบ่งเป็นซ้าย ขวา ในระยะ100 cm – 240 cm ได้กราฟดังรูปที่ 4.2



รูปที่ 4.3 กราฟแสดงผล IOU and distance

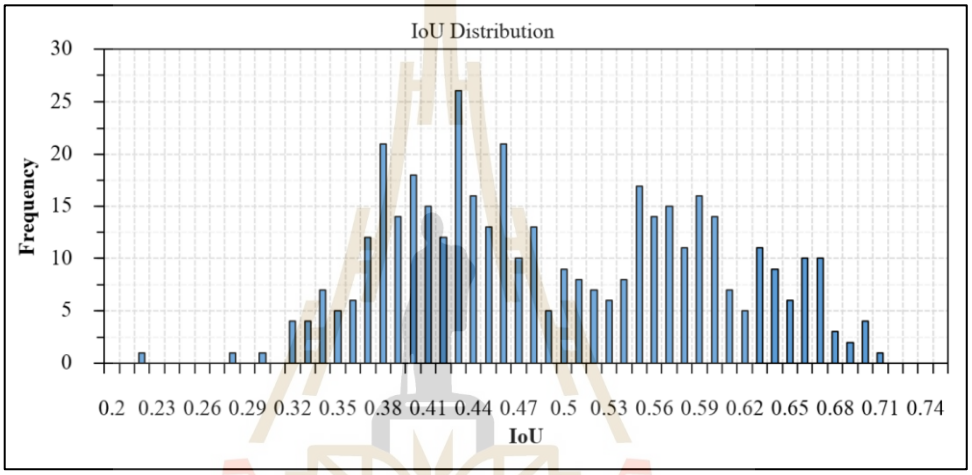
เมื่อนำข้อมูลมาพลอตเส้นแนวโน้มเป็นเส้นตรงโดยวิธี Linear regression จะได้ว่า เมื่อหุ่นยนต์ Track เป้าหรือภาพระยะใกล้ ๆ ค่า IoU จะมีค่ามาก แต่เมื่อระยะเริ่มไกลขึ้น ค่า IoU จึงเริ่มมีค่าลดลง



รูปที่ 4.4 กราฟแสดงผล IoU and distance โดยมีเส้นตรง Linear regression

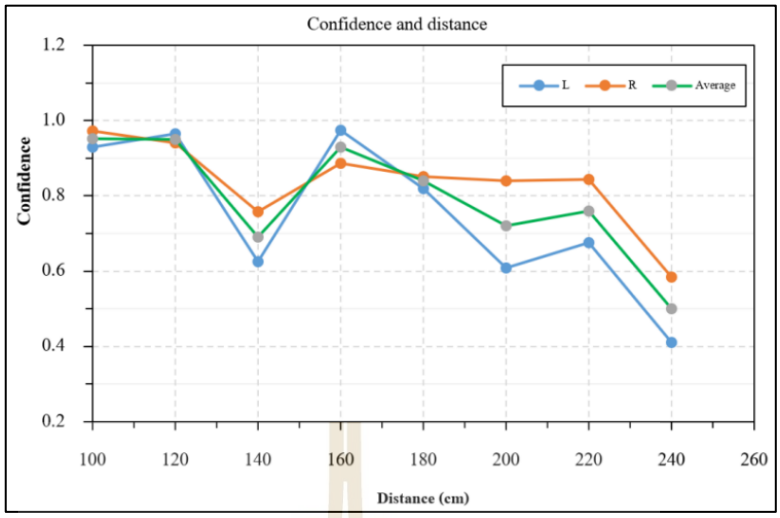
นอกจากนี้เมื่อนำค่า IOU จากข้อมูลทั้งหมดมาสร้างเป็นกราฟแจงแจกความถี่จะได้กราฟดังรูปที่ 4.4 ค่าเฉลี่ยของ IOU ทั้งหมดเป็น 0.498 ส่วนเบี่ยงเบนมาตรฐานเป็น 0.108 ค่าความแปรปรวนเป็น 0.012 ค่า IoU มีการเกาะกลุ่มมากที่สุดในช่วงค่า IoU เป็น 0.38 – 0.48

นอกจากนี้กราฟที่ได้ยังมีลักษณะเป็นการแจกแจงปกติแบบสองยอด แต่ยอดของการแจกแจงปกติที่ IoU สูงกว่าจะอยู่ฝั่งด้านขวามือ ซึ่งมีค่าอยู่ในช่วง 0.54 – 0.61 ซึ่งค่าระยะที่ให้ IoU อยู่ในช่วงนี้คือ ระยะที่น้อยกว่า 150 เซนติเมตร ดังนั้นหากจะให้ผลการตรวจจับสามารถตรวจจับได้โดยไม่หลุดการตรวจจับที่สุดควรให้หุ่นยนต์กับวัตถุอยู่ห่างกันไม่เกิน 150 เซนติเมตร



รูปที่ 4.5 กราฟแสดงผล IoU and distance โดยมีเส้นตรง linear regression

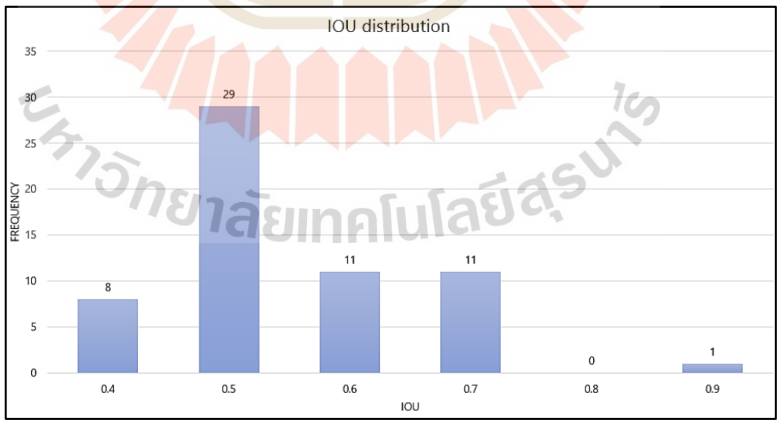
นอกจากนี้ค่า Confidence level ที่ได้จากการทดสอบแบบ Static ในระยะต่าง ๆ ยังแสดงดังในกราฟรูปที่ 4.5 ค่าเฉลี่ยของ Confidence level ทั้งหมดเป็น 0.820 ส่วนเบี่ยงเบนมาตรฐานเป็น 0.158 ค่าความแปรปรวนเป็น 0.025 จะเห็นว่ายิ่งหุ่นยนต์เข้าใกล้เป้าหมายค่า Confidence level เฉลี่ยย่อมมากขึ้น และมีแนวโน้มแบบเดียวกันทั้งด้านซ้ายและด้านขวา แต่จุดที่น่าสังเกตคือที่ระยะ 140 mm ห่างจากเป้าหมายพบว่าค่า Confidence level ตกลงไปทั้งซ้ายและขวาอาจจะเกิดจากมุมมองที่ขาดไปในการเทรนภาพของโมเดล จำเป็นต้องเทรนภาพในมุมนี้ให้มากขึ้น



รูปที่ 4.6 กราฟแสดงผล Confidence and distance

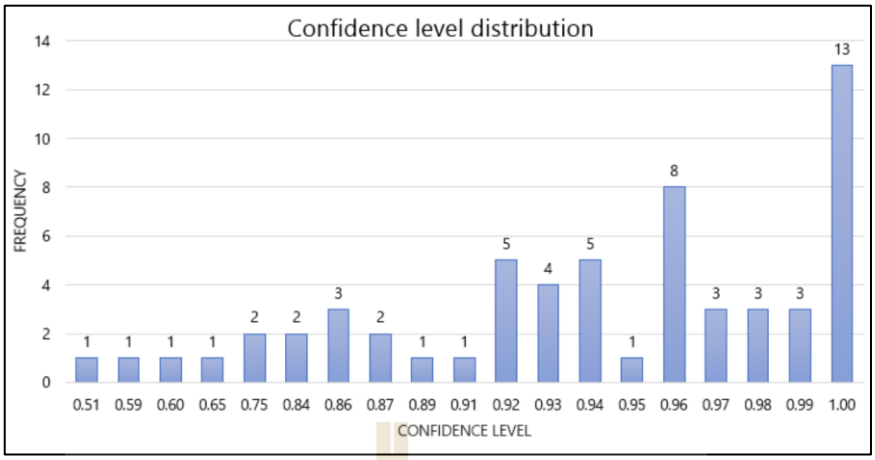
4.4 ผลการทดสอบการติดตามวัตถุด้วย Deep learning แบบ Dynamics test

เมื่อนำข้อมูลที่เก็บได้จากการที่ให้หุ่นยนต์เคลื่อนที่บนพาหนะมาสร้างกราฟแจกแจงความถี่กับค่า IoU จะได้กราฟดังรูปที่ 4.6 ค่าเฉลี่ยของ IOU ทั้งหมดเป็น 0.542 ส่วนเบี่ยงเบนมาตรฐานเป็น 0.100 ค่าความแปรปรวนเป็น 0.010



รูปที่ 4.7 กราฟแสดงผล IoU distribution

ค่า Confidence level ที่ได้จากการทดสอบแบบ Dynamics สามารถนำมาสร้างเป็นกราฟแจกแจงความถี่ได้ดังรูปที่ 4.7 ค่าเฉลี่ยของ Confidence level ทั้งหมดเป็น 0.92 ส่วนเบี่ยงเบนมาตรฐานเป็น 0.107 ค่าความแปรปรวนเป็น 0.0114



รูปที่ 4.8 กราฟแสดงผล Confidence level distribution

จากผลการทดสอบทั้งหมดพบว่า หากใช้โมเดล KCF, MOSSE และ CSRT จะให้ความแม่นยำที่สูงกว่า และสามารถทำงานได้กับความละเอียดภาพที่สูงกว่าการใช้ Deep learning model โดยความละเอียดภาพสูงสุดที่ทำได้คือ 410×308 พิกเซล โดยมีค่า IOU มากที่สุดเป็น 0.77 ในโมเดลแบบ CSRT แต่ว่าเมื่อวัตถุมีการหายไปจากหน้าจอแล้วกลับมาจะไม่สามารถกลับไปตรวจจับได้อีก ขาดความต่อเนื่องในการตรวจจับ แต่หากใช้ Deep learning model จะทำให้เกิดความต่อเนื่องในการตรวจจับ ไม่ว่าวัตถุจะหายไปจากหน้าจอกี่ครั้งแล้วกลับมาก็สามารถตรวจจับได้ แต่จะให้ความแม่นยำที่น้อยกว่า โดยค่า IoU และ Confidence level เฉลี่ยคือ 0.542 และ 0.942 ที่ความละเอียดภาพสูงสุดเป็น 205×154 พิกเซล

บทที่ 5
บทสรุปและข้อเสนอแนะ

5.1 สรุปผลงานวิจัย

งานวิจัยนี้เป็นการออกแบบการควบคุมตำแหน่งและแนวการวางตัวของแขนหุ่นยนต์ 2 แกน บนพาหนะขณะเคลื่อนที่ได้แสดงให้เห็นว่า โดยผู้วิจัยออกแบบการควบคุมให้สามารถใช้เป้าหมายเป็นประเภทใดก็ได้ เป็นรูปภาพ 2 มิติ หรือ วัตถุที่เป็น 3 มิติ ซึ่งเป็นรูปแบบที่แตกต่างกันเพื่อใช้แนวคิดหรือใช้ปรับปรุงการทำงานของระบบควบคุมให้มีประสิทธิภาพมากขึ้นจะแบ่งเป็น 2 ระบบใหญ่ คือ

1. ระบบเชิงกล โดยส่วนของแขนหุ่นยนต์ทั้ง 2 แกนจะใช้อุปกรณ์ดังนี้ แกนที่ 1 ใช้ Stepper Motor Nema17 ในการเคลื่อนที่ 0-360 องศา และ แกนที่ 2 ใช้ Servo motor เคลื่อนที่ขึ้นลง 0-90 องศา โดยควบคุมผ่าน บอร์ด Arduino Nano

2. ระบบซอฟแวร์ ใช้ระบบปฏิบัติปฏิบัติการหุ่นยนต์ (ROS Version Noetic) ที่ถูกติดตั้งบน Ubuntu 20.04 focal fossa ซึ่งใช้ภาษา Python และในการเขียนโปรแกรมเพื่อประมวลผลใช้ Model Tracking มาช่วยในการติดตาม โดยใช้ Model KCF, MOSSE และ CSRT ที่ขนาดรูปภาพ 205×154 Pixel, 308×231 Pixel และ 410×308 Pixel โดยผลที่ได้ Model สามารถติดตามเป้าหมายได้ แต่ในกรณีที่เป้าหมายหายไปจากกล้อง และเข้ามาใหม่ จะมีการติดตามเป็นบางครั้งแต่ไม่ทุกครั้ง ซึ่ง Model ที่เหมาะสมที่สุด คือ Model CSRT ที่ขนาดรูปภาพ 308×231 Pixel มีค่า IoU เฉลี่ย 0.79

จากการใช้ Model ในกรณีที่เป้าหมายหายไปจากกล้อง และเข้ามาใหม่ จะมีการติดตามเป็นบางครั้งแต่ไม่ทุกครั้ง จึงได้นำการติดตามเป้าหมาย ด้วย Deep learning โดยการนำภาพเป้าหมายหรือเป้าหมายมาเทรนในลักษณะต่าง ๆ เมื่อเทรนภาพแล้ว ผลการติดตามเป้าหมายสามารถติดตามได้ และเมื่อนำเป้าหมายออกจากกล้องและนำเข้ามาใหม่ ก็ยังมีการติดตามเป้าหมายตลอด โดยแบ่งการทดสอบเป็น 2 แบบ

1) แบบ Static Test ความละเอียดของภาพจากกล้อง Pi Camera เป็น 205×154 Pixels โดยแบ่งเป็นซ้าย ขวา ในระยะ 100 cm – 240 cm ผลที่ได้ในระยะที่เข้าใกล้เป้าหมาย ค่า IoU จะมีค่ามาก และเมื่อระยะเริ่มไกลขึ้นค่า IoU จะมีค่าลดลง ซึ่งค่าเฉลี่ยของ IoU เป็น 0.498 และมีค่าระดับความมั่นใจเฉลี่ย 0.820

2) แบบ Dynamics test หุ่นยนต์เคลื่อนที่บนพาหนะ ความละเอียดของภาพจากกล้อง Pi Camera เป็น 205×154 pixels ผลที่ได้มีค่าเฉลี่ย IoU 0.542 และมีค่าระดับความมั่นใจเฉลี่ย 0.942

จากการออกแบบและทดสอบระบบของซอฟแวร์ ผู้วิจัยพบว่า ควรใช้การประมวลผลแบบ Deep learning ที่จะสามารถมีการติดตามเป้าหมายหรือวัตถุตามที่เราต้องการได้ ซึ่งขึ้นอยู่กับกับการเลือกใช้วัสดุอุปกรณ์ที่มีประสิทธิภาพ การเลือกใช้อุปกรณ์สำหรับติดตั้งชุดขับเคลื่อนต่าง ๆ เมื่อภาพเป้าหมายหรือวัตถุมีการหายไปจากกล้องแล้ว ซึ่งกลับมาใหม่จะสามารถติดตามเป้าหมายได้อีกครั้ง โดยค่า IoU เฉลี่ยอยู่ที่ 0.49 – 0.54 ส่วนค่าระดับความมั่นใจเฉลี่ยอยู่ที่ 0.82 – 0.94

5.2 ข้อเสนอแนะ

1. ระบบเชิงกลด้วยการออกแบบ มีผลต่อการประมวลผล ด้วยการขับ Stepper motor และมี Feedback ตำแหน่งมุมองศาที่วัดจาก Encoder โดยการขับมอเตอร์ผ่านด้วย สายพานและ Pulley ในการสวมแกนเพลาของ Encoder ที่มีขนาดไม่พอดีกับตัวแกนเพลาซึ่งได้ทำการกลึง Pulley เอง อาจส่งผลให้การประมวลผลในเชิงลึกไม่สามารถการติดตามเป้าหมาย และเก็บรายละเอียดได้ไม่ดีเท่าที่ควร วิธีแก้ไขเปลี่ยน Encoder ที่มีแกนเพลาที่พอดีกับ Pulley

2. เนื่องด้วยการประมวลผลภาพ ด้วย Deep learning จะติดตามเป้าหมายได้ดีในขนาด 205×154 Pixel ซึ่งมีขนาดเล็กเป็นเพราะตัวกล้อง Pi camera V2 มีความละเอียดภาพไม่ได้สูง เมื่อปรับขนาดภาพที่ประมวลผลให้มีขนาดใหญ่ขึ้นทำให้ภาพความละเอียดแตก ส่งผลให้กับการติดตามเป้าหมายและค่าระดับความมั่นใจ วิธีแก้ไขจะต้องเปลี่ยนกล้องที่มีความละเอียดสูงเพิ่มขึ้นหรือมีเลนส์ซูมให้กับตัวกล้อง ทำให้สามารถเพิ่มละเอียดภาพในการประมวลผลได้

3. ภาพเป้าหมายหรือวัตถุเป้าหมาย ที่มีสีหรือลักษณะใกล้เคียงกัน เมื่อนำมาสั่งการให้เกิดการติดตาม ผลที่ได้ยังมีการติดตามวัตถุอื่นบ้างเล็กน้อยแม้จะด้วยระดับความมั่นใจที่ต่ำก็ตาม ซึ่งเป็นจุดอ่อนของการเรียนรู้เชิงลึก ซึ่งส่วนที่จะแก้ไขได้ คือ การเทรนภาพเป้าหมายและวัตถุเพิ่มเติมให้มากพอ และใช้ฮาร์ดแวร์ที่มีกำลังประมวลผลสูง ใช้ภาพที่มีขนาดใหญ่มากขึ้นเก็บรายละเอียดได้มากขึ้นจะทำให้การตรวจจับและติดตามมีความแม่นยำมากขึ้น

4. การประมวลผลของซอฟแวร์ ผู้วิจัยนำ บอร์ด Raspberry Pi 4 Model B มาประมวลผลบนคอมพิวเตอร์เพราะถ้าประมวลผลบน บอร์ด Raspberry Pi 4 Model B เลยจะประมวลผลไม่เพียงพอจะเกิดการ Delay ในขณะที่ส่งผลค่าข้อมูลที่ประมวลผลเสร็จแล้วไปยังไมโครคอนโทรลเลอร์ บอร์ด Arduino MEGA2560 ทำให้มีผลต่อความแม่นยำ และการเคลื่อนที่ของอุปกรณ์ ซึ่งแนวทางการแก้ไข ผู้วิจัยต้องเปลี่ยนบอร์ดประมวลผลที่มีความสามารถในการประมวลผลสูงมากกว่านี้

ซึ่งบอร์ดที่มีความมากสูงก็จะมีราคาที่แพงขึ้น และต้องเขียนโปรแกรมควบคุมสำหรับซอฟแวร์ใหม่ให้เหมาะสมกับบอร์ดประมวลผล

5. แขนหุ่นยนต์ยังขาดอัลกอริทึมและโปรแกรมสำหรับทำการค้นหาเป้าหมายได้เองอัตโนมัติ เมื่อเป้าหมายออกจากเฟรมไป ผู้ที่สนใจต่อยอดสามารถพัฒนาตัวโปรแกรมในส่วนนี้เพิ่มได้ นอกจากนี้ยังสามารถติด Lidar เพิ่มได้เพื่อทำให้มีระบบหลบหลีกสิ่งกีดขวางอัตโนมัติในตัว

รายการอ้างอิง

อภิลักษณ์ หล่อนกลาง. (2558), การพัฒนาเครื่องฝึกบินจำลองชนิดสามองศาอิสระ. **มหาวิทยาลัยเทคโนโลยีสุรนารี**. นครราชสีมา.

Bolme, Beveridge, Draper and Lui. (2010). Visual Object Tracking using Adaptive Correlation Filters. **Computer Science Department, Colorado State University**.

Gardel, Lazaro, Lavest and Vazquez. (2002). Detection and Tracking Vehicles Using a Zoom Camera Over a Pan & Tilt Unit**. University of Alcala**.

Henriques, et al. (2014). High-Speed Tracking with Kernelized Correlation Filters. **2014 IEEE Transactions on Pattern Analysis and Machine Intelligence**.

Hiroyuki Ukida. (2010). Object Tracking System by Pan-Tilt Moving Cameras and Robot Using Condensation Method. **Mechanical Engineering, Faculty of Engineering, The University of Tokushima**.

Kung-Ye, Ming-Yang and Mi-Ching. (2002). Design and Implementation of a Real-Time Pan-Tilt Visual Tracking System. **National Cheng Kung University**.

Lukezic, et al. (2019). Discriminative Correlation Filter Tracker with Channel and Spatial Reliability. **Faculty of Computer and Information Science, University of Ljubljana**.

Robin, Carmadi and Egi. (2017). Design and Implementation of Pan-Tilt Control for Face Tracking. **School of Electrical Engineering and Informatics, Bandung Institute of Technology**.

ภาคผนวก ก

โค้ดโปรแกรมสำหรับการควบคุมหุ่นยนต์ด้วย Arduino

```
#include <ros.h>
#include <std_msgs/UInt16.h> //
ros::NodeHandle  nh; //
#include <Servo.h>

#define pulse 3 //stepper
#define dir 2 //stepper
//step 1/2 H L L
bool state1, state2 = 0, CR = 0;
//encoder
int CLK = 10; // Blu Digital Pin 10
int DO = 11; // Grn Digital Pin 11
int CSn = 12; // Ylw Digital Pin 12
int Data, Degree;
int input = 0, desired = 0;
unsigned long time_now = 0;
int speed_stepper = 500; //หน่วยmicro
int TT = 200;
int diff = 0;
Servo tiltCam;
void StepperCb( const std_msgs::UInt16 &Stepper_msg) {
  input = Stepper_msg.data;   // blink the led
  Data = ReadSSI();
  Degree = map(Data, 0, 4095, 0, 359);
  desired = input;
  if (desired > 359) {
    desired = desired % 360;
  }
  //ROS
}
void ServoCb(const std_msgs::UInt16 &Servo_msg) {
```

```
  int tilt = Servo_msg.data;
  tiltCam.write(tilt);
}
ros::Subscriber<std_msgs::UInt16> subStep("Stepper", StepperCb );
ros::Subscriber<std_msgs::UInt16> subServ("Servo", ServoCb );
void setup() {
  //ROS
  nh.initNode();
  nh.subscribe(subStep); //subscriber = รับค่าจากROS
  nh.subscribe(subServ);
  pinMode(pulse, OUTPUT);
  pinMode(dir, OUTPUT);
  //encoder
  pinMode(CSn, OUTPUT);// Chip select
  pinMode(CLK, OUTPUT);// Serial clock
  pinMode(DO, INPUT_PULLUP);// Serial data IN/OUT
  digitalWrite(CSn, HIGH);
  digitalWrite(CLK, HIGH);
  tiltCam.attach(30);
  Data = ReadSSI();
  Degree = map(Data, 0, 4095, 0, 359);
  desired = Degree;
  Serial.begin(57600);
}

void loop() {
  //อ่านค่ามุมจาก encoder
  nh.spinOnce();
  Data = ReadSSI();
  Degree = map(Data, 0, 4095, 0, 359);
  diff = desired - Degree;
```

```
  //Serial.println("Error =" + String(diff) + "\t" + String(desired));
  if (diff > 0) {
    if (desired <= 180) {
      CW_step();
    }
    else if (desired > 180 && Degree < 180)
    {
      CCW_step();
    }
    else if (Degree > 180 && desired > 180)
      CW_step();
  }
  else if (diff < 0) {
    if (desired == 0 && Degree < 180)
      CCW_step();
    else if (desired == 0 && Degree > 180)
      CW_step();
    else
      CCW_step();
  }
  else
    Break();
  delay(2);
}
int ReadSSI(void)
{ int i, dReading;
  char Resolution = 12;
  unsigned int bitStart = 0x0800;
  dReading = 0;
  digitalWrite(CSn, LOW);
  digitalWrite(CLK, LOW);
```

```
  for (i = (Resolution - 1); i >= 0; i--)
  { digitalWrite(CLK, HIGH);
    if (digitalRead(DO)) dReading |= bitStart;
    digitalWrite(CLK, LOW);
    bitStart = bitStart >> 1;
    if (i == 0)
    { digitalWrite(CLK, HIGH);
      if (digitalRead(DO)) dReading |= bitStart;
    }  }
  digitalWrite(CSn, HIGH);
  return dReading;
}
void CW() { //
  digitalWrite(dir, LOW); //dir
  if (micros() > time_now + speed_stepper ) {
    digitalWrite(pulse, HIGH);
    time_now = micros();
  }
  digitalWrite(pulse, LOW); //speed
}
void CW_step() { //
  digitalWrite(dir, LOW); //dir
  digitalWrite(pulse, HIGH);
  delayMicroseconds(TT);
  digitalWrite(pulse, LOW); //speed
  delayMicroseconds(TT);
}
void CCW_step() { //
  digitalWrite(dir, HIGH); //dir
  digitalWrite(pulse, HIGH);
  delayMicroseconds(TT);
```

```
  digitalWrite(pulse, LOW); //speed
  delayMicroseconds(TT);
}
void CCW() { //
  digitalWrite(dir, HIGH); //dir
  if (micros() > time_now + speed_stepper ) {
    digitalWrite(pulse, HIGH);
    time_now = micros();
  }
  digitalWrite(pulse, LOW); //speed
}
void Break() { //
  digitalWrite(pulse, LOW); //speed
}
```

ภาคผนวก ข

โค้ดโปรแกรมสำหรับการติดตามวัตถุภาษา Python

Deep learning (DL) code

```
#!/usr/bin/env python3
import roslib
import sys
import rospy
import numpy as np
import cv2
import time

from cv_bridge import CvBridge, CvBridgeError
from std_msgs.msg import UInt16
from sensor_msgs.msg import CompressedImage
class BallTrack(object):
        def __init__(self):
self.bridge = CvBridge()
self.image_sub = rospy.Subscriber("/raspicam_node/image/compressed",
CompressedImage, self.camera_callback)
                print(" << Subscribe image from camera")
self.stepper_pub = rospy.Publisher("/Stepper", UInt16, queue_size = 5)
print(" >> Publish the stepper position ")
self.servo_pub = rospy.Publisher("/Servo", UInt16, queue_size = 5)
print(" >> Publish the servo position ")
self.stepper= UInt16()
self.servo= UInt16()
self.pan_angle = 0
self.tilt_angle = 0
self.frame = None
self.fps = None
self.prev_frame_time = 0
self.new_frame_time = 0
```

```
(major, minor) = cv2.__version__.split(".")[:2]
print("openCV version : {}.{}".format(major,minor))
def stop(self):
self.stepper.data = 0
self.servo.data = 0
        def tilt(self,servo_angle):
self.servo.data = servo_angle
        def pan(self,pan_angle) :
                self.stepper.data = pan_angle
        def camera_callback(self,data):
#yolo setup
                net = cv2.dnn.readNet("/home/aexotic/tank_ws/src/tank_power/src/
yolov4_tiny_training_final.weights"\,"/home/aexotic/tank_ws/src/tank_power/src/yolov4_tiny_tra
        ining.cfg") # Original yolov3
                classes = ["dragon"]
        layer_names = net.getLayerNames()
        outputlayers = [layer_names[i[0] - 1] for i in net.getUnconnected
OutLayers()]
                colors= np.random.uniform(0,255,size=(len(classes),3))
                try:
cv_image=self.bridge.compressed_imgmsg_to_cv2(data,"bgr8")
        self.stepper_pub.publish(self.stepper)
        self.servo_pub.publish(self.servo)
                except CvBridgeError as e:
                        print(e)
    #tracker config
    #print("[INFO] starting video stream...")
                #cap = cv2.VideoCapture()
font = cv2.FONT_HERSHEY_PLAIN
starting_time= time.time()
frame_id = 0
```

```
self.frame = cv_image
#_, frame = cap.read()
#self.frame = imutils.resize(self.frame, width=210)
centroid_x = 0
centroid_y = 0
(H, W) = self.frame.shape[:2]
                blob = cv2.dnn.blobFromImage(self.frame,0.00392,(210,210),(0,0,0),True,
crop=False) #reduce 416 to 320
net.setInput(blob)
outs = net.forward(outputlayers)
#print(outs[1])
#Showing info on screen/ get confidence score of algorithm in detecting an object in blob
class_ids=[]\
confidences=[]
boxes=[]
for out in outs:
for detection in out:
scores = detection[5:]
class_id = np.argmax(scores)
confidence = scores[class_id]
if confidence > 0.1:
          #onject detected
        centroid_x= int(detection[0]*W)
        centroid_y= int(detection[1]*H)
        w = int(detection[2]*W)
        h = int(detection[3]*H)
          #cv2.circle(img,(center_x,center_y),10,(0,255,0),2)
          #rectangle co-ordinaters
        x=int(centroid_x - w/2)
        y=int(centroid_y - h/2)
          #cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),2)
```

```
boxes.append([x,y,w,h]) #put all rectangle areas
confidences.append(float(confidence)) #how confidence was that object detected and
show that percentag
class_ids.append(class_id) #name of the object tha was detected
indexes = cv2.dnn.NMSBoxes(boxes,confidences,0.4,0.6)
for i in range(len(boxes)):
                if i in indexes:
                        x,y,w,h = boxes[i]
                        label = str(classes[class_ids[i]])
                        confidence= confidences[i]
                        print(confidence)
                        color = colors[class_ids[i]]
                        cv2.rectangle(self.frame,(x,y),(x+w,y+h),color,2)
                        cv2.putText(self.frame,label+" "+str(round(confidence,2))+"
(x,y):"+str(centroid_x)+","+str(centroid_y),(x,y+30),font,1,(255,255,255),2)
elapsed_time = time.time() - starting_time
fps=frame_id/elapsed_time
#cv2.putText(self.frame,"FPS:"+str(round(fps,2)),(10,50),font,2,(0,0,0),1)
cv2.imshow("Image",self.frame)
key = cv2.waitKey(1) #wait 1ms the loop will start again and we will process the next
frame
#print("fps: {}".format(self.fps))
center = None
rows = cv_image.shape[0]
cols = cv_image.shape[1]
size = min([rows,cols])
center_x = int(cols/2.0)
center_y = int(rows/2.0)

threshold = 20
left_bound = int(center_x-threshold)
```

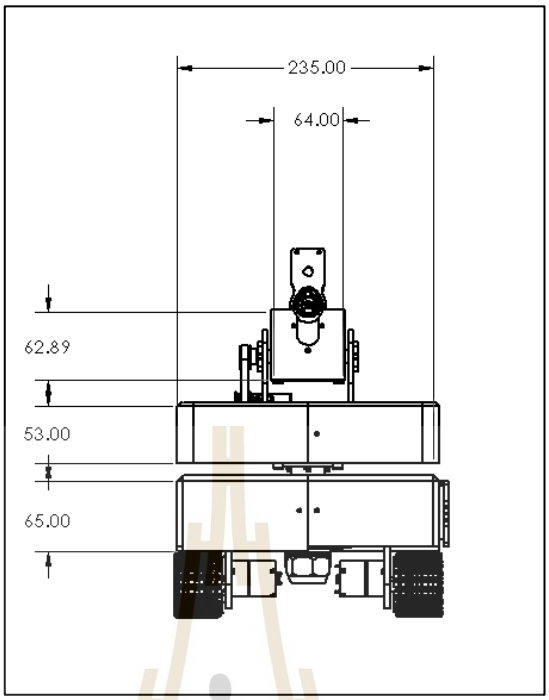
```
            right_bound = int(center_x+threshold)
            upper_bound = int(center_y-threshold)
            lower_bound = int(center_y+threshold)
        #Tracking
#pan
                        if(centroid_x==0):
if (self.pan_angle<0):
reverse_pan_angle = 360+self.pan_angle  self.pan(reverse_pan_angle)
else:
self.pan(self.pan_angle)
                        elif (centroid_x < left_bound):
if self.pan_angle>-180:
self.pan_angle-=1

            if self.pan_angle<0:
                                reverse_pan_angle = 360+self.pan_angle
                                self.pan(reverse_pan_angle)
            else:
                        self.pan(self.pan_angle)
                        elif (centroid_x> right_bound):
if self.pan_angle<180:
self.pan_angle+=1
 if self.pan_angle<0:
reverse_pan_angle = 360+self.pan_angle  self.pan(reverse_pan_angle)
            else:
                        self.pan(self.pan_angle)
#tilt
                        if centroid_y == 0:
self.tilt(self.tilt_angle)
elif (centroid_y > lower_bound):
if self.tilt_angle<1:
```

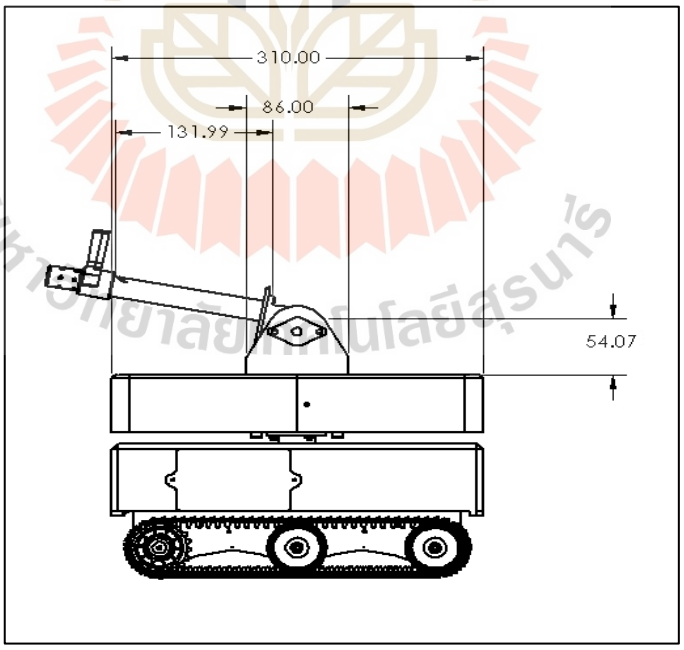
```
self.tilt_angle = 0
self.tilt(self.tilt_angle)
else:
self.tilt_angle-=1
self.tilt(self.tilt_angle)
        #       self.tilt(self.tilt_angle)
                        elif (centroid_y < upper_bound):
if (centroid_y < upper_bound):
self.tilt_angle+=1
if self.tilt_angle<90:
self.tilt(self.tilt_angle)
else:
self.tilt_angle = 90
self.tilt(self.tilt_angle)
print("pan angle{}".format(self.pan_angle))
#cv2.waitKey(1)
#cap.release()
def main():
            rospy.init_node("line_track_node", anonymous=True)
ball_tracking_object=BallTrack()
try:
rospy.spin()
except KeyboardInterrupt:
print("Shutting down")
cv2.destroyAllWindows()
if __name__ =='__main__':
        main()
```

ภาคผนวก ค

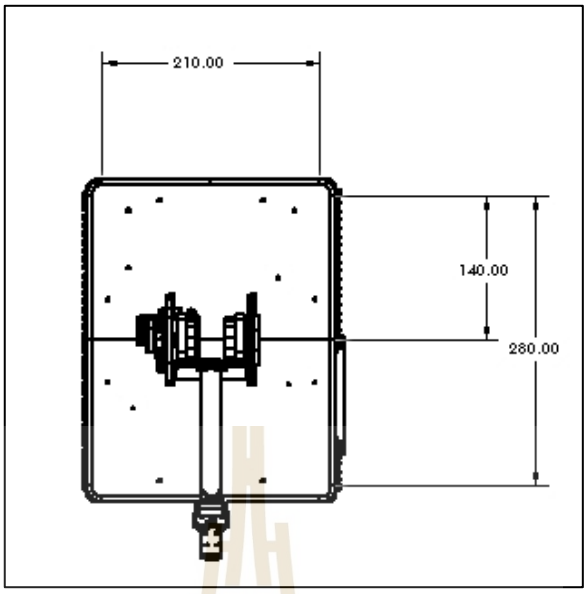
Drawing หุ่นยนต์



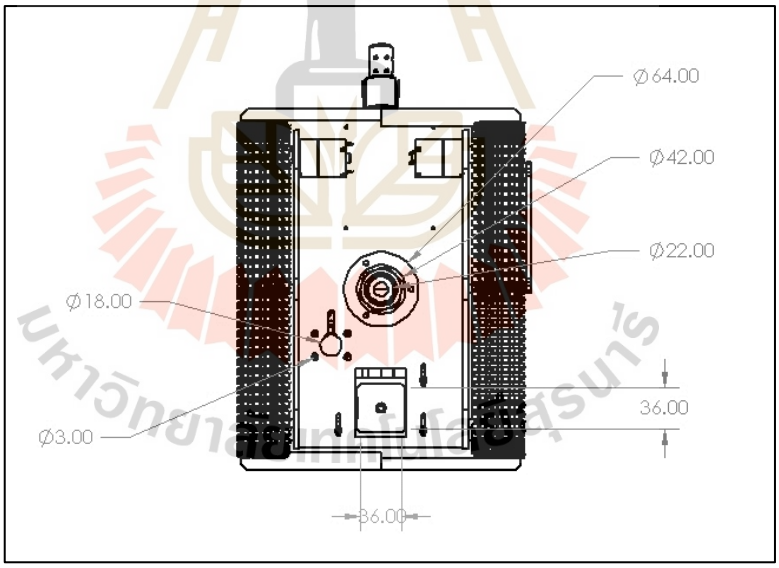
รูปที่ ค.1 Drawing หุ่นยนต์และขนาดในมุมมองด้านหน้า



รูปที่ ค.2 Drawing หุ่นยนต์และขนาดในมุมมองด้านขวา



รูปที่ ค.3 Drawing หุ่นยนต์และขนาดในมุมมองข้างบน



รูปที่ ค.4 Drawing หุ่นยนต์และขนาดในมุมมองด้านล่าง

ภาคผนวก ง

บทความวิชาการที่ได้รับการตีพิมพ์เผยแพร่ในระหว่างการศึกษา

บทความวิชาการที่ได้รับการตีพิมพ์เผยแพร่ในระหว่างการศึกษา

Prakasinee Singcharoenkit, Phuwanat Phueakthong, Aphilak Lonklang, Jittima Varagul and Kontorn Chamniprasart. (2021). An Implementation of Object Tracking Methods on Pan and Tilt Manipulator for Teacher Tracking in Hybrid Classroom. **In the International Conference on Mechanical & Production Engineering (ICMPE)**. Phuket, Thailand, 8th May, 2021.

Paper ID:WR-ICMPE-PHUK-080521-6091

WRFER WORLD RESEARCH FORUM FOR ENGINEERS AND RESEARCHERS

WORLD RESEARCH FORUM FOR ENGINEERS AND RESEARCHERS

International Conference on
Mechanical & Production Engineering

Certificate

This is to certify that Phakasinee Singcharoenkit has presented a paper entitled "An Implementation of Object Tracking Methods on Panand Tilt Manipulatorfot Teacher Tracking in Hybrid Classroom" at the International Conference on Mechanical & Production Engineering (ICMPE) held in Phuket, Thailand on 08th May, 2021.

World Research Forum for Engineers and Researchers
WRFER

Chairman
World Research Forum for Engineers and Researchers

AN IMPLEMENTATION OF OBJECT TRACKING METHODS ON PAN AND TILT MANIPULATOR FOR TEACHER TRACKING IN HYBRID CLASSROOM

[1]PHAKASINEE SINGCHAROENKIT, PHUWANAT PHUEAKTHONG, APHILAK LONKLANG, [2]JITTIMA VARAGUL and [3]KONTORN CHAMNIPRASART

[1,2,3]School of Mechanical Engineering, Suranaree University of Technology,
111 Suranaree Mueang Nakhon Ratchasima Thailand 30000

Email: [1]phakasinee.s@gmail.com, [2]jittima@sut.ac.th, [3]kontorn@sut.ac.th
Contact: [1,2,3]0-4422-3000 Ext. 4678

Abstract: Hybrid Classroom due to the pandemic of COVID-19, the key to success for a digital classroom of the Suranaree University of Technology is broadcasting the teaching and learning activities onsite classroom via zoom application. One problem about the students who take this classroom online can not meet their teacher because the integrated camera on the classroom computer is stacked on the table in front of the class. The reason why there can not sense to teachers acts in the classroom. This paper aims to implement the object tracking method to the pan and tilt manipulator, a camera integrated. The results show that the three selected methods can achieve this task. The best accuracy for teacher tracking is the CSRT method with an IoU of 0.77 at 410 x 308 pixels.

Index terms: Teacher Tracking, Object Tracking, Camera Tracking, Pan and Tilt

I. INTRODUCTION

Due to the pandemic of COVID-19, physical distancing is the critical rule for university teaching and learning activities. Laboratories are essentials classes for the engineering education field. Online classrooms are not the key to success for these cases. A hybrid classroom was selected. They are focusing on large size industrial robot laboratories. These are the essential laboratory for undergraduate students who are majoring in a mechatronics engineering curriculum. The teacher assistants need to use the monitoring camera to present the movement of robots and robot teaching situations. Sometimes the size of the robot is a large size, consequenctly the detail of robot cound not be collected.

Due to the above problem, the aim of this research is to develop the tracking base for teacher monitoring camera. Real-time video will be used for image processing to achive the tracking task and will be recorded for e-courseware stuff. By implementing the three onshelf tracking algoritm to the pan and tilt angle base which paning by DC stepper and tilting by servo motor.

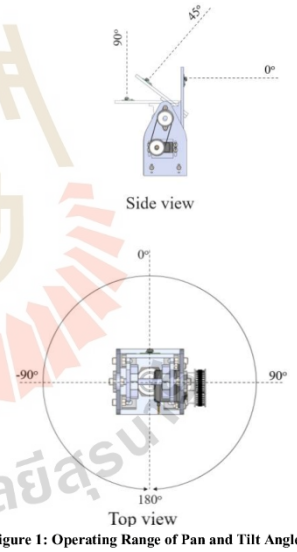


Figure 1: Operating Range of Pan and Tilt Angles

II. System Description

A. Hardware configuration

The range of the pan algle is from 0 – 360 degrees left to right and the tilt angle range is from 0 – 90 degrees from ground to air as in Fig.1.

There are three main components as in Fig.2. Using the robot operating system (ROS) network to be a server for data communication. Firstly, the image processing module are included with raspberry pi and pi-camera. The second is the input command, this

module will help the user set the parameter for tracking algorithm and starting the system.

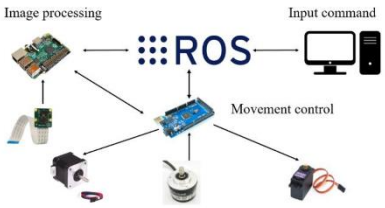


Figure 2: Hardware Overview

Input Command will be sent to the image processing module via ROS network. The tracking data is a region of interest (ROI) from the user. After a user created the ROI from the raw picture, the ROI data will be sent to the image processing module for making the decision. The making decision algorithm was integrated into the Raspberry Pi 4 controllers. Python programming was developed be a computational module. The last section, the movement module, will be received the position command from raspberry pi 4 to activate the pan and tilt base. An Arduino mega controller was selected to control these two motors at the same time. P-controller is integrated into the stepper control algorithm for stability movement.

B. Software descriptions

Robot Operating System(ROS) is commonly used for the robot task framework to develop the software and hardware related to the robots. They were using of ROS Noetic version and Ubuntu 20.04 focal fossa as the system environment. Python language is used for algorithm development. The image processing software is developed based on the OpenCV version 4.3. Moreover, the last section is motors controllers, Arduino IDE, to create the P-controller programming for stepper motor and servo motor controller. The Arduino controller received the orientation from previous software via ROS node and controlled the motor simultaneously. The maximum range of pan limit is -180 degrees and 180 degrees, and the overall pan angle is 360 degrees. P-controller is the optimized controller for use in this position control case because the speed of the stepper motor to achieve the task is still slow.

The system workflow of this system can be presented as the flowchart in Fig.3.

1) The real-time picture, captured from pi-camera, with desired quality of pixel. The more pixels of the picture, the more processing time need to use in the image processing period. A size of 410x308 pixels or lower were selected to deal with. The captured picture will be sent to the next node through the ROS master server.

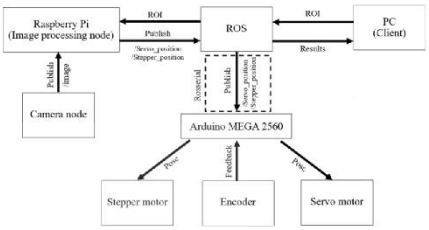


Figure 3: System Work Flow

2) Processing node received the captured picture and compared it with ROI from a user. In this case, we selected the teacher as ROI for the tracking system during the class period by marking a blue square bounding box. Three methods of tracking system were implemented into this system, KCF, MOSSE, and CSRT. For easy monitoring of the operation of the image processing module, a square bounding box is marked by green bounding box as in Fig.8.

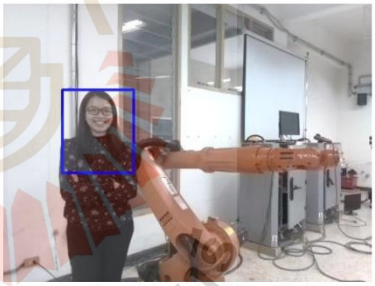


Figure 4: Example of Target Selection

3) Using the mathematical equation to compute [6] the centroid of the tracking object in vertical and horizontal directions. Coordinate from this equation will be the centroid of a tracking box.

4) If the center of the tracking box is not located in the center of the camera frame, the pan and tilt commands will send the movement angle to the movement controller to respond to this situation. The

automatic control concept is to center the teacher in the middle of the camera frame.

5) Movement Controller receives the angle data from the ROS node and takes action to the pan and tilt angle with each motor controller.

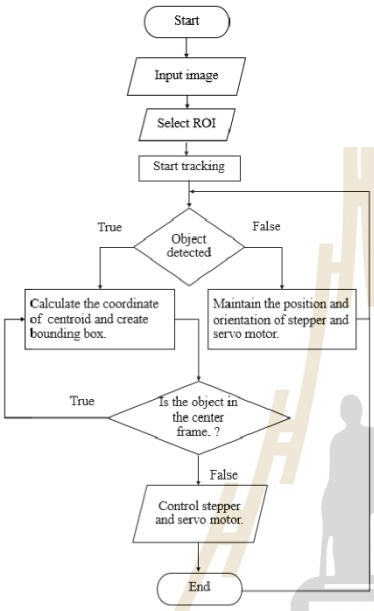


Figure 5: Algorithm Work Flow

C. Object tracking method

The solution of the teacher tracking method has to use the object tracking method. In this research, we implement on-shelf object tracking algorithms to perform this task. KCF, MOSSE, and CSRT were selected for the accuracy test. This group of algorithms uses a low level of computational cost.

KCF [1],[2] (Kernelized Correlation Filters) is the object tracking algorithm that uses the correlation value to match the sample. In object tracking, the correlation value between the ROI patch in the future frame and the original translated patch will be the highest. The KCF algorithm tends to be more accurate than the MOSSE algorithm.

MOSSE [5] (Minimum Output Sum of Squared Error) is the algorithm that used the MOSSE filter, which can discriminate between the ROI and the background image. This algorithm performs well in the change of rotation, light, brightness, and object scale. The MOSSE algorithm tends to much faster than KCF and CSRT, the accuracy less than KCF and CSRT algorithm.

CSRT[3],[4](Channel and Spatial Reliability Tracker) is the object tracking model which improved the Discriminative Correlation Filter (DCF) algorithm with spatial and channel reliability. The spatial reliability map makes the CSRT can adjust the filter size, which makes the CSRT model better than the DCF algorithm and can handle non-rectangular shape ROI. The CSRT algorithm tends to be more accurate than the KCF algorithm but slightly slower than KCF.

III. Experiment Results

A. Evaluation method

For the method of testing the tracking method, accuracy on pan and tilt angle manipulator. By using the three algorithms mentioned in the previous chapter. We divide the captured picture into three parts of the experiment, including 205x154 308x231, and 410 x308 pixels, respectively. The target moves at the same track, and time is the controlled parameter of this experiment. Then measuring the [7] Intersection over Union (IoU) of each track is used to determine the first accuracy of the tracking method. IoU is one measuring value to measure the intersection of area between the ideal frame and the actual frame. The output value of IoU falls into the range of 0 to 1 (equation 1).

$$\text{Intersection over union (IOU)} = \frac{\text{Intersect area}}{\text{Union area}} \quad Eq.1$$

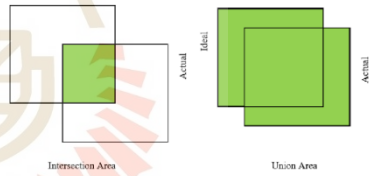


Figure 6: Intersection over Union (IoU)

If the value of IoU is higher and close to 1 is a sign that the actual and ideal area is located in the same frame. If the value of IoU is one shows that the tracking area and the ROI area are located in the same frame.

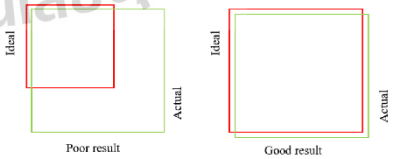


Figure 7: Result of the Value of IoU vs. Accuracy

In the second accuracy test, the difference between the actual frame and ideal is compared. In this test, the Euclidean Distance Expression in Eq.2 for measuring the distance between two centroids was used.

$$d(x,y) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad Eq.2$$

On the other hand, compared with the IoU value, the value of the Euclidean distance must be closed to Zero, the accuracy of the tracking method will be better than others. For easy understanding of this value, the percent of centroid error (C.E.) is calculated by Eq.3.

$$C.E. = \left| \frac{d_{actual} - d_{ideal}}{d_{actual}} \right| X\ 100\% \quad Eq.3$$

B. Results

The result of IoU and C.E. of the experiment was shown in Tables 1 and 2. The example of the tracking method is shown in Fig.8. The red bounding box represents the ideal ROI, and the green one represents the actual one.

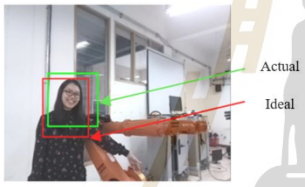


Figure 8: Intersection over Union of the Actual and Ideal Frame

From Table 1, The result shows that if the number of pixels is increasing, the more accuracy in IoU. MOSSE algorithm returns the highest value of IoU in 205x154 pixels condition. On the other hand, CSRT returns the highest value of the IoU.

Table 1: Intersection over Union (IoU) Results

Image Pixel	Model	(IOU)
205x154	CSRT	0.74
	KCF	0.71
	MOSSE	0.86
308x231	CSRT	0.79
	KCF	0.72
	MOSSE	0.70
410x308	CSRT	0.77
	KCF	0.73
	MOSSE	0.69

From Table 2, the result shows that the centroid error of the tracking methods is in the same way—the overall centroid error of each tracking condition not over than 2% error. KCF returns the highest centroid error in all conditions of image pixels.

Table 2: Centroid error (C.E) Results

Image Pixel	Model	C.E. [%]
205x154	CSRT	1.79
	KCF	1.59
	MOSSE	0.30
308x231	CSRT	1.33
	KCF	1.72
	MOSSE	1.64
410x308	CSRT	0.88
	KCF	1.60
	MOSSE	1.16

IV. CONCLUSION

This paper presents the accuracy testing result from the three on-shelf tracking object algorithms to the teacher tracking task in the robot laboratory class in university. The three selected algorithms are KCF, MOSSE, and CSRT. By implementing the tracking algorithm results to the pan and tilt manipulator, the camera is integrated to track the teacher in a laboratory. Stepper motor and servo motor are integrated to control pan and tilt motions for centering the ROI in the middle of the frame. The result shows that the accuracy of the CSRT both in IoU and centroid error returns the accuracy of IoU value of 0.77 and centroid error of 0.88 in a condition of 410x380 pixels. In conclusion, the CSRT algorithm is the best choice in three selected algorithms to deal with teacher tracking in the laboratory classroom.

V. REFERENCES

[1] Henriques, et al., "High-Speed Tracking with Kernelized Correlation Filters", 2014.
[2] George, Jose and Mathew, "Performance Evaluation of KCF based Trackers using VOT Dataset" , 2018.
[3] Lukezic, et al.,, "Discriminative Correlation Filter Tracker with Channel and Spatial Reliability" , 2019.
[4] Wei Mi and Tsuen Yang, "Comparison of Tracking Techniques on 360-Degree Videos" , 2019.
[5] Bolme, et al., "Visual Object Tracking using Adaptive Correlation Filters", 2010.
[6] George, Jose and Mathew, "Performance Evaluation of KCF based Trackers using VOT Dataset" , 2018.
[7] Adrian Rosebrock, "Intersection over Union (IoU) for object detection", 2018, from https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

««««

ประวัติผู้เขียน

นางสาวผกาสินี สิงห์เจริญกิจ เกิดเมื่อวันที่ 10 ธันวาคม พ.ศ. 2536 ณ อำเภอเมือง จังหวัดสุรินทร์ เริ่มการศึกษาชั้นประถมศึกษาที่โรงเรียนเทศบาล 2 “วิภัชศึกษา” มัธยมศึกษาตอนต้นและตอนปลายที่โรงเรียนวีรวัฒน์โยธิน สำเร็จการศึกษาหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมเครื่องกล หลักสูตรวิศวกรรมเมคคาทรอนิกส์ สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี อำเภอเมือง จังหวัดนครราชสีมา เมื่อปี พ.ศ. 2560 ได้เข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมเมคคาทรอนิกส์ ณ มหาวิทยาลัยแห่งเดิม โดยได้รับทุนสนับสนุนการศึกษาจากสถาบันวิจัยแสงซินโครตรอน (องค์การมหาชน) จังหวัดนครราชสีมา ภายใต้โครงการพัฒนาบุคลากรทางด้านเครื่องเร่งอนุภาคและเครื่องกำเนิดแสงซินโครตรอนในระหว่างการศึกษาได้ปฏิบัติหน้าที่เป็นผู้ช่วยสอนและวิจัยประจำสาขาวิชาวิศวกรรมเมคคาทรอนิกส์ และได้รับมอบหมายให้เป็นผู้สอนปฏิบัติการประจำสาขาวิชาวิศวกรรมเครื่องกล สาขาวิชาวิศวกรรมการผลิต และสาขาวิชาวิศวกรรมเมคคาทรอนิกส์ ดังนี้

1. ระบบอัตโนมัติอุตสาหกรรม
2. ปฏิบัติการระบบควบคุมและอัตโนมัติ
3. ปฏิบัติการวิศวกรรมเมคคาทรอนิกส์ 1
4. ปฏิบัติการวิศวกรรมเมคคาทรอนิกส์ 2
5. ปฏิบัติการวิสวกรรมเมคคาทรอนิกส์ 3
6. ปฏิบัติการวิสวกรรมเมคคาทรอนิกส์ 4
7. ปฏิบัติการฟิสิกส์สำหรับวิศวกร 1
8. หุ่นยนต์เบื้องต้น