ระบบคัดแยกและเลือกสิ่งของด้วยการเรียนรู้ของเครื่อง

SORTING AND PICKING SYSTEMS VIA MACHINE LEARNING

ธาม คุณากรกุล

พิชญา ลากูล

วริทธิ์ธร สัมพันธกิจ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2562

ระบบคัดแยกและเลือกสิ่งของด้วยการเรียนรู้ของเครื่อง ปีการศึกษา 2562

ปริญญานิพนธ์ปีการศึกษา 2562

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบคัดแยกและเลือกสิ่งของด้วยการเรียนรู้ของเครื่อง

SORTING AND PICKING SYSTEMS VIA MACHINE LEARNING

ผู้จัดทำ

1. นายธาม คุณากรกุล รหัสนักศึกษา 59010649
2. นางสาวพิชญา ลากูล รหัสนักศึกษา 59010964
3. นายวริทธิ์ธร สัมพันธกิจ รหัสนักศึกษา 59011194

อาจารย์ที่ปรึกษา

(อาจารย์จิระศักดิ์ สิทธิกร)

ระบบคัดแยกและเลือกสิ่งของด้วยการเรียนรู้ของเครื่อง

นายธาม	คุณากรกุล	59010649
นางสาวพิชญา	ลากูล	59010964
นายวริทธิ์ธร	สัมพันธกิจ	59011194
อาจารย์จิระศักดิ์	สิทธิกร	อาจารย์ที่ปรึกษา

ปีการศึกษา 2562

บทคัดย่อ

ปัจจุบันกระบวนการคัดแยกสิ่งของและนับจำนวนนั้นมีความสำคัญต่อกระบวนการผลิตและกระบวนการบรรจุภัณฑ์ในระดับอุตสาหกรรม เนื่องจากเป็นกระบวนการที่ต้องการความรวดเร็ว และความถูกต้องแม่นยำสูง โครงงานนี้จึงนำเสนอวิธีการคัดแยกและเลือกสิ่งของโดยใช้การเรียนรู้เครื่อง (Machine Learning) เข้ากับการทำงานของหุ่นยนต์ Delta Robot ที่ใช้ในการหยิบสิ่งของไปยังทิศทางที่กำหนด ซึ่งควบคุมกระบวนการผ่านเว็บแอปพลิเคชันที่พัฒนาขึ้นมา โดยใช้กล้องต่อเข้ากับบอร์ด Raspberry Pi ในการประมวลผลภาพสิ่งของจากนั้นจึงสั่งงานแขนกลให้เคลื่อนไปตามตำแหน่งที่ต้องการด้วยการควบคุมผ่าน Stepper Motor ทั้งสามตัวให้ทำงานพร้อมกันและใช้การคำนวณจากหลักการ Delta Robot Kinematic มีการเก็บข้อมูลภาพที่จากกระบวการไว้ในระบบ เพื่อให้ผู้ใช้งานที่มีสิทธิ์ในการเข้าใช้งานอุปกรณ์สามารถดูขั้นตอนกระบวนการทำงานของเครื่องได้ผ่านเว็บแอปพลิเคชันที่พัฒนาขึ้นมา

Sorting And Picking Systems Via Machine Learning

Mr. Time	Kunakornkul	59010649
Ms. Pitchaya	Lakool	59010964
Mr. Waritthon	Sampantakit	59011194
Mr. Jirasak	Sittigorn	Advisor

Academic Year 2019

ABSTRACT

Currently, The sorting and counting process is important to the production and packaging processes at the industrial that requires speed and high accuracy. This project presents a method for sorting and selecting objects by using machine learning and the Delta robot that is used to pick things in a specified direction. Which controls the process through the developed web application. By using the camera connected to the Raspberry Pi to process images from objects, then the robot arms can be moved to the desired position by controlling all three stepper motors to work at the same time and use the calculations from the Delta robot kinematic principle. The images from the process are stored in the system. So that users with access rights can view the process of the machine through the developed web application.

กิตติกรรมประกาศ

ปริญญานิพนธ์นี้สามารถลุล่วงไปได้ด้วยดี ซึ่งขอขอบพระคุณอาจารย์จิระศักดิ์ สิทธิกรที่ให้ความรู้ ให้คำปรึกษาและคำแนะนำที่เป็นประโยชน์อย่างมากในการศึกษาและการแก้ปัญหาต่าง ๆ ระหว่างทำงานจนกระทั่งปริญญานิพนธ์นี้เสร็จสมบูรณ์

ขอขอบคุณผู้ที่เกี่ยวข้องกับโครงงานนี้ทุกท่าน ที่ไม่ได้เอ่ยนามในที่นี้ สุดท้ายนี้ขอขอบพระคุณบิดา มารดาและเพื่อน ๆ ที่ให้กำลังใจในการทำงานวิจัยนี้ประโยชน์อันใดที่เกิดจากโครงงานนี้ย่อมเป็นผลมาจากความกรุณาของท่านดังกล่าวข้างต้น ขอขอบพระคุณมา ณ โอกาสนี้

ธาม คุณากรกุล
พิชญา ลากูล
วริทธิ์ธร สัมพันธกิจ

สารบัญ

หน้า

บทคัดย่อภาษาไทย I
บทคัดย่อภาษาอังกฤษ II
สารบัญ IV
สารบัญตาราง X
สารบัญรูป XII

บทที่ 1 บทนำ 1
1.1 ความเป็นมาของปัญหา 1
1.2 วัตถุประสงค์ 1
1.3 ขอบเขตของโครงงาน 1
1.4 ประโยชน์ที่คาดว่าจะได้รับ 2

บทที่ 2 ทฤษฎีและเครื่องมือที่เกี่ยวข้อง 3
2.1 Image Processing 3
2.1.1 Thresholding 3
2.1.2 Morphological Transformations 4
2.1.3 Edge Detection 6
2.1.4 Hough Transform 8
2.1.5 Segmentation 10
2.2 Machine Learning 11
2.2.1 XGBoost 11
2.2.2 Deep Learning 11

สารบัญ (ต่อ)

หน้า

2.3 Software Tool 12

2.3.1 Visual Studio Code 12

2.3.2 Adobe XD 12

2.3.3 GitKraken 12

2.4 Runtime Environment 13

2.4.1 Node.js 13

2.5 Software Library 13

2.5.1 OpenCV 13

2.5.2 Scikit-image 13

2.5.3 Scikit-learn 13

2.5.4 Tensorflow 13

2.5.5 Socket.IO 14

2.6 Raspberry Pi 14

2.7 Delta Robot Kinematic 15

2.7.1 จลนศาสตร์ไปข้างหน้า (Forward Kinematics) 17

2.7.2 จลนศาสตร์แบบผกผัน (Inverse Kinematics) 17

บทที่ 3 การออกแบบและพัฒนาระบบ 19

3.1 ภาพรวมระบบ 19

3.1.1 เว็บแอปพลิเคชันสำหรับผู้ใช้งาน (Web Application for User) 19

3.1.2 เว็บแอปพลิเคชันสำหรับผู้ดูแลระบบ (Web Application For Admin) 20

3.1.3 ส่วนของเซิร์ฟเวอร์และฐานข้อมูล 20

3.2 แผนภาพยูสเคส (Use Case Diagram) 20

สารบัญ (ต่อ)

หน้า

3.3 ส่วนต่อประสานกับผู้ใช้ (User Interface) 30
3.3.1 Web Application ส่วนของผู้ใช้งาน (User) 30
3.3.2 Web Application ส่วนของผู้ดูแลระบบ (Admin) 45
3.4 ภาพรวมของฮาร์ดแวร์ 49
3.4.1 ฐานสำหรับวางอุปกรณ์ 49
3.4.2 การต่อวงจร 50
3.4.3 การออกแบบวงจรพิมพ์อิเล็กทรอนิกส์ 52
3.4.4 การตั้งข้อกำหนดในการออกแบบวงจรพิมพ์อิเล็กทรอนิกส์ 53
3.5 โครงสร้างหลัก 54
3.6 ออกแบบเฟืองเพื่อช่วยในการเพิ่มกำลังให้กับ Stepper Motor 56
3.7 ขั้นตอนการเพิ่ม Dataset เพื่อเทรนโมเดล 58
3.7.1 ถ่ายภาพ 58
3.7.2 ประมวลผลภาพเบื้องต้น (Image Preprocessing) 58
3.7.3 เทรนโมเดล 58
3.8 ขั้นตอนการแยกวัตถุ 59
3.8.1 ถ่ายภาพ 59
3.8.2 ประมวลผลภาพเบื้องต้น (Image Preprocessing) 59
3.8.3 แยกชนิดวัตถุ 59
3.8.4 ส่งต่อตำแหน่งของแต่ละวัตถุ 59

สารบัญ (ต่อ)

หน้า

บทที่ 4 การทดลองและผลการทดลอง 60

4.1 ทดลองถ่ายภาพด้วย Raspberry Pi 60

4.1.1 จุดประสงค์การทดลอง 60

4.1.2 วิธีการทดลอง 60

4.1.3 ผลการทดลอง 61

4.1.4 สรุปผลการทดลอง 61

4.2 ทดลองเปรียบเทียบความแตกต่างของจำนวนพิกเซลวัตถุชิ้นเดียวกัน ก่อนและหลังการทำ Perspective Transform 61

4.2.1 จุดประสงค์การทดลอง 61

4.2.2 วิธีการทดลอง 61

4.2.3 ผลการทดลอง 67

4.2.4 สรุปผลการทดลอง 67

4.3 ทดลองเปรียบเทียบการ Segment วัตถุที่อยู่ติดกันระหว่าง Tiny Yolo และ Yolo 67

4.3.1 จุดประสงค์การทดลอง 67

4.3.2 วิธีการทดลอง 67

4.3.3 ผลการทดลอง 68

4.3.4 สรุปผลการทดลอง 68

4.4 ทดลองเทรนโมเดล ML Classifiers แต่ละโมเดลเพื่อเปรียบเทียบความแม่นยำ 69

4.4.1 จุดประสงค์การทดลอง 69

4.4.2 วิธีการทดลอง 69

4.4.3 ผลการทดลอง 69

4.4.4 สรุปผลการทดลอง 69

สารบัญ (ต่อ)

หน้า

4.5 ทดลองเปรียบเทียบความแม่นยำระหว่าง XGBoost และ Xception (Convnet) 70
4.5.1 จุดประสงค์การทดลอง 70
4.5.2 วิธีการทดลอง 70
4.5.3 ผลการทดลอง 70
4.5.4 สรุปผลการทดลอง 70
4.6 การทดลองรัศมีของแขนกล โดยการเปลี่ยนความยาวของแขนกลแต่ละส่วน 71
4.6.1 จุดประสงค์การทดลอง 71
4.6.2 วิธีการทดลอง 71
4.6.3 ผลการทดลอง 72
4.6.4 สรุปผลการทดลอง 73
4.7 การทดลองควบคุม Stepper Motor โดยใช้บอร์ด Raspberry Pi 74
4.7.1 จุดประสงค์การทดลอง 74
4.7.2 วิธีการทดลอง 74
4.7.3 ผลการทดลอง 76
4.7.4 สรุปผลการทดลอง 76
4.8 การทดลองควบคุม Servo Motor โดยใช้บอร์ด Raspberry Pi 76
4.8.1 จุดประสงค์การทดลอง 76
4.8.2 วิธีการทดลอง 76
4.8.3 ผลการทดลอง 77
4.8.4 สรุปผลการทดลอง 77

สารบัญ (ต่อ)

หน้า

4.9 การทดลองรับค่า Sensor ผ่าน IC MCP3208 โดยใช้บอร์ด Raspberry Pi 77

4.9.1 จุดประสงค์การทดลอง 77

4.9.2 วิธีการทดลอง 78

4.9.3 ผลการทดลอง 78

4.9.4 สรุปผลการทดลอง 78

4.10 ทดลองใช้ฟังก์ชัน Inverse Kinematic เพื่อเปลี่ยนองศาของมอเตอร์ไปยังตำแหน่งที่กำหนดตาม x0,y0,z0 79

4.10.1 จุดประสงค์การทดลอง 79

4.10.2 วิธีการทดลอง 79

4.10.3 ผลการทดลอง 81

4.10.4 สรุปผลการทดลอง 81

บทที่ 5 บทสรุปและข้อเสนอแนะ 82

5.1 บทสรุป 82

5.1.1 การทำงานของระบบ 82

5.1.2 ความรู้ที่ได้รับ 82

5.2 ปัญหาและแนวทางแก้ไข 82

5.3 แนวทางการพัฒนาต่อ 83

บรรณานุกรม 84

สารบัญตาราง

ตารางที่ **หน้า**

3.1 ตารางแสดงยูสเคสที่ 1 (Register) 21

3.2 ตารางแสดงยูสเคสที่ 2 (Login) 21

3.3 ตารางแสดงยูสเคสที่ 3 (Forget Password) 21

3.4 ตารางแสดงยูสเคสที่ 4 (View Profile) 22

3.5 ตารางแสดงยูสเคสที่ 5 (Edit Profile) 22

3.6 ตารางแสดงยูสเคสที่ 6 (Change Password) 22

3.7 ตารางแสดงยูสเคสที่ 7 (Initial View) 23

3.8 ตารางแสดงยูสเคสที่ 8 (Select Pattern) 23

3.9 ตารางแสดงยูสเคสที่ 9 (Classification Process) 23

3.10 ตารางแสดงยูสเคสที่ 10 (Rectify) 24

3.11 ตารางแสดงยูสเคสที่ 11 (Summary) 24

3.12 ตารางแสดงยูสเคสที่ 12 (Edit Pattern) 24

3.13 ตารางแสดงยูสเคสที่ 13 (Counting Process) 25

3.14 ตารางแสดงยูสเคสที่ 14 (Select Hardware) 25

3.15 ตารางแสดงยูสเคสที่ 15 (Learning Process) 25

3.16 ตารางแสดงยูสเคสที่ 16 (Suggest Type) 26

3.17 ตารางแสดงยูสเคสที่ 17 (Add new type) 26

3.18 ตารางแสดงยูสเคสที่ 18 (View Dataset) 26

3.19 ตารางแสดงยูสเคสที่ 19 (Search By Filtering) 27

3.20 ตารางแสดงยูสเคสที่ 20 (View Dataset Detail) 27

3.21 ตารางแสดงยูสเคสที่ 21 (Delete Dataset) 27

3.22 ตารางแสดงยูสเคสที่ 22 (View Hardware) 28

3.23 ตารางแสดงยูสเคสที่ 23 (View Hardware Detail) 28

3.24 ตารางแสดงยูสเคสที่ 24 (Edit Hardware Detail) 28

สารบัญตาราง (ต่อ)

ตารางที่ **หน้า**

3.25 ตารางแสดงยูสเคสที่ 25 (Delete Hardware) 29
3.26 ตารางแสดงยูสเคสที่ 26 (Add Hardware) 29
3.27 ตารางแสดงยูสเคสที่ 27 (Hardware Permission) 29
3.28 ตารางแสดงยูสเคสที่ 28 (View Log) 30
4.1 ตารางแสดงขนาดที่ใช้ในการทดลองเพิ่ม-ลดระยะแขนกล 71
4.2 ตารางแสดงระยะที่ทำได้เมื่อทดลองเปลี่ยนขนาดแขนกลที่ขนาดต่างๆ 72
4.3 ตารางแสดงสรุปผลการทดลองของขนาดแขนกลที่เหมาะสมที่สุด 73

สารบัญรูป

รูปที่ **หน้า**

2.1 การเปรียบเทียบก่อน-หลังการทำ Erosion 4

2.2 การเปรียบเทียบก่อน-หลังการทำ Dilation 5

2.3 การเปรียบเทียบก่อน-หลังการทำ Opening 5

2.4 การเปรียบเทียบก่อน-หลังการทำ Closing 6

2.5 ตัวอย่างการทำ Non-maxima Suppression 7

2.6 ตัวอย่างการทำ Hysteresis Thresholding 8

2.7 กราฟของเส้นตรงที่ผ่านจุด (x_0,y_0) 9

2.8 กราฟของเส้นตรงที่ผ่านจุด (x_0,y_0), (x_1,y_1) และ (x_2,y_2) 10

2.9 บอร์ด Raspberry Pi 14

3.1 ภาพรวมของระบบแสดงความสัมพันธ์ของการทำงานระหว่างอุปกรณ์และเครื่องมือต่างๆ 19

3.2 แผนภาพแสดงความสัมพันธ์การทำงานของผู้ใช้ระบบและระบบ 20

3.2 หน้าแสดงผลการยืนยันตัวตนเข้าสู่ระบบ 30

3.3 หน้าแสดงผลของการสมัครสมาชิก 31

3.4 หน้าแสดงผลกรณีลืมรหัสผ่าน 31

3.5 หน้าแสดงผลการตั้งค่ารหัสผ่านใหม่กรณีลืมรหัสผ่าน 32

3.6 หน้าแสดงผลหลัก 33

3.7 หน้าแสดงผลข้อมูลส่วนตัว 34

3.8 หน้าแสดงผลการเปลี่ยนแปลงข้อมูลส่วนตัว 34

3.9 หน้าแสดงผลชุดข้อมูล 35

3.10 หน้าแสดงรายละเอียดข้อมูลแยกชิ้น 36

3.11 หน้าแสดงผลหลักกระบวนการคัดแยก 37

3.12 หน้าแสดงผลเลือกรูปแบบการคัดแยก 37

3.13 หน้าแสดงผลกระบวนการคัดแยก 38

3.14 หน้าแสดงผลยืนยันความถูกต้องของข้อมูล 38

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

3.15 หน้าแสดงผลสรุปผลข้อมูลที่ได้จากการคัดแยก 39

3.16 หน้าแสดงผลรูปแบบการคัดแยกที่ได้บักทึกไว้ 39

3.17 หน้าแสดงผลหลักกระบวกการนับ 40

3.18 หน้าแสดงผลกระบวนการนับ 41

3.19 หน้าแสดงผลยืนยันความถูกต้องของข้อมูล 41

3.26 หน้าแสดงผลหลัก 46

3.27 หน้าแสดงรายละเอียดกิจกรรม 47

3.28 หน้าแสดงผลคำร้องขอเข้าถึงอุปกรณ์ 48

3.29 หน้าแสดงผลการจัดการอุปกรณ์ 49

3.30 แสดงการจัดเรียงอุปกรณ์จากด้านบน 50

3.31 แสดง GPIO PIN Rasberry Pi Model B 51

3.32 แสดงภาพรวมของระบบทั้งหมด 51

3.33 แสดงการเชื่อมต่อผ่าน SPI ระหว่าง Raspberry Pi และ MCP3208 52

3.34 แสดงแผนผังวงจรไฟฟ้า (Schematic) 52

3.35 แสดงข้อกำหนดในการออกแบบ (Design Rule) 53

3.36 แสดงวงจรพิมพ์อิเล็กทรอนิกส์ (ด้านหน้า) 54

3.37 แสดงวงจรพิมพ์อิเล็กทรอนิกส์ (ด้านหลัง) 54

3.38 โครงสร้างหลักใช้สำหรับวางอุปกรณ์และเป็นจุดติดตั้งมอเตอร์ 55

3.39 ฐานรูปทรงแปดเหลี่ยมที่ใช้สำหรับวางชิ้นส่วนที่ต้องการคัดแยก 55

3.40 แสดงการนำฐานรูปแปดเหลี่ยมมาวางเข้ากับโครงสร้างหลัก 56

3.41 เฟืองขับขนาดเส้นผ่านศูนย์กลาง 14.36 มิลลิเมตร 57

3.42 เฟืองตามขนาดเส้นผ่านศูนย์กลาง 114.96 มิลลิเมตร 57

3.43 ฐานใช้สำหรับยึดชุดเฟืองกับ Stepper Motor Bracket 57

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

3.44 ชิ้นส่วนช่วยรับแรงที่เกิดขึ้นเมื่อใส่แขนกล 58

3.45 ขั้นตอนการเพิ่ม Dataset เพื่อเทรนโมเดล 58

3.46 ขั้นตอนการแยกวัตถุ 59

4.1 โปรแกรมที่ใช้สำหรับสั่งให้ Raspberry Pi ถ่ายรูป 60

4.2 ถูกถ่ายด้วย Webcam 61

4.3 ฐานที่ใช้วางชิ้นส่วนเพื่อทดลองการทำ Perspective Transform 62

4.4 Threshold หาขอบสีส้ม 62

4.5 การ Threshold หาขอบสีเขียว 63

4.6 การหาขอบโดยใช้ Canny edge detection 63

4.7 เส้นตรงที่ได้จาก Probabilistic Hough Line Transform 64

4.8 เส้นตรงขอบขอบสีที่เลือกใช้ 64

4.9 เส้นตรงของขอบนอกของฐาน 65

4.10 จุดตัดของเส้นตรงที่ได้ 65

4.11 รูปก่อน-หลังทำ Perspective Transform 66

4.12 ผลการ Segment ของวัตถุติดกัน 67

4.13 ผลการ Detect วัตถุโดยใช้ Tiny Yolo และ Yolo ตามลำดับ 68

4.14 ผลการเทรนโมเดล ML Classifiers 69

4.15 ผลการทดสอบโมเดล XGBoost 70

4.16 ผลการทดสอบโมเดล Xception 70

4.17 ภาพแสดงการเคลื่อนที่ของปลายแขนกล 71

4.18 วงจรที่ใช้ในการทดสอบการควบคุม Stepper Motor 74

4.19 โปรแกรมสำหรับใช้ในการทดสอบควบคุม Stepper Motor 75

4.20 วงจรที่ใช้ในการทดสอบการควบคุม Servo Motor 76

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

4.21 โปรแกรมสำหรับใช้ในการทดสอบควบคุม Servo Motor 77
4.22 วงจรที่ใช้ในการอ่านค่า Analog จาก IC MCP3208 78
4.23 โปรแกรมสำหรับใช้ในการอ่านค่า Infrared Sensor 78
4.24 แสดงการตั้งค่าขนาดของแขนกล Delta Robot 79
4.25 แสดงการคำนวณ Trigonometric Constants 79
4.26 ฟังก์ชันที่ใช้ในการคำนวณมุมแต่ละมุม 80
4.27 ฟังก์ชันที่ใช้ในการคำนวณค่า Theta1, Theta2, Theta3 ของมอเตอร์แต่ละตัว 80
4.28 ผลลัพธ์เมื่อส่งค่าให้กับฟังก์ชัน inverse คำนวณค่าองศาของแต่ละมุม 81

บทที่ 1

บทนำ

1.1 ความเป็นมาของปัญหา

กระบวนการคัดแยกและนับจำนวนสิ่งของนั้นเป็นกระบวนการทำงานที่ต้องใช้ความแม่นยำและใช้เวลาในการคัดแยกค่อนข้างมาก หลายครั้งการคัดแยกอาจเกิดความผิดพลาดขึ้นทำให้การคัดแยกที่เกิดไม่มีประสิทธิภาพในการทำงาน จึงนำเสนอโครงงานเครื่องต้นแบบคัดแยกและเลือกสิ่งของ โดยอาศัยหลักการเรียนรู้ของเครื่อง ซึ่งได้ใช้เทคโนโลยีด้านวิศวกรรมศาสตร์ การประมวลผลภาพด้วยคอมพิวเตอร์ รวมถึงการเรียนรู้ของเครื่องมาใช้พัฒนาระบบ

โดยเครื่องต้นนั้นแบบประกอบด้วยชุดแขนกลแบบขนานหรือหุ่นยนต์ประเภทเดลต้าเป็นแขนกลที่ทำงานได้อย่างรวดเร็วและแม่นยำ เหมาะสำหรับกระบวนการคัดแยกสิ่งของ และใช้บอร์ด Raspberry Pi สำหรับประมวลผลภาพและควบคุมแขนกลให้ทำตามที่ระบบสั่ง ซึ่งควบคุมกระบวนการผ่านเว็บแอปพลิเคชันที่พัฒนาขึ้นมา ทำให้เครื่องต้นแบบนั้นสามารถทำการนับจำนวนสิ่งของและคัดแยกวัตถุลงกล่องต่างๆ ที่ผู้ใช้กำหนดได้

1.2 วัตถุประสงค์

1) สามารถคัดแยกวัตถุและนับจำนวนวัตถุได้
2) ประหยัดเวลาในการคัดแยกและนับจำนวนวัตถุ

1.3 ขอบเขตของโครงงาน

สามารถคัดแยกวัตถุที่มีลักษณะแตกต่างกันและนับจำนวนชิ้นของวัตถุได้ โดยใช้แขนกล (Delta Robot) เป็นตัวคัดแยกชิ้นวัตถุไปใส่ไว้ในช่องเก็บวัตถุ รวมถึงเครื่องสามารถเรียนรู้ได้ว่าวัตถุนั้นคือวัตถุรูปแบบใดได้เอง (Machine Learning) โดยสามารถแยกวัตถุที่วางไม่ชิดกันได้

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1) อุปกรณ์ต้นแบบที่มีความสามารถในการคัดแยกและเลือกสิ่งของ

2) ซอฟต์แวร์ที่มีความสามารถในการแยกวัตถุด้วยการเรียนรู้ของเครื่อง รวมทั้งควบคุมการทำงานของ Delta Robot

3) ลดเวลาและบุคลากรในการคัดแยกและเลือกสิ่งของ

บทที่ 2
ทฤษฎีและเครื่องมือที่เกี่ยวข้อง

2.1 Image Processing

การนำข้อมูลภาพมาประมวลผลหรือคำนวณด้วยคอมพิวเตอร์ เพื่อให้ได้ผลลัพธ์เป็นไปตามที่ต้องการทั้งในเชิงคุณภาพ และปริมาณ โดยมีหลักการเบื้องต้น คือ การกำจัดสัญญาณรบกวนออกจากภาพ การปรับปรุงคุณภาพของภาพให้ภาพมีความคมชัดมากขึ้น การแบ่งส่วนของวัตถุที่สนใจออกมาจากภาพ เพื่อนำภาพวัตถุที่ได้ไปวิเคราะห์หาข้อมูลเชิงปริมาณ เช่น ขนาด สี รูปร่าง ไปวิเคราะห์ในลำดับต่อไป

2.1.1 **Thresholding**

Thresholding เป็นวิธีการทำ Segmentation ที่เรียบง่ายที่สุด โดยการเปรียบเทียบค่าของพิกเซลแต่ละพิกเซลกับค่า Threshold

2.1.1.1 Global Thresholding

Global Thresholding จะเปรียบเทียบค่าของทุก ๆ พิกเซลกับค่าของ Threshold ค่าเดียวกันหมด

1) **Binary Thresholding**

Binary Thresholding นั้นจะเทียบทุก ๆ พิกเซลกับค่า Threshold ที่กำหนด หากค่าของ Pixel มีค่าน้อยกว่า ก็จะถูกปรับให้เป็น 0 มิฉะนั้นจะถูกปรับให้เป็นค่าสูงสุด

Binary Thresholding สามารถเขียนเป็นสมการได้ดังนี้

$$dst(x,y) = \begin{cases} maxVal & if\ src(x,y) > thresh \\ 0 & otherwise \end{cases}$$

ถ้าหากความเข้มของพิกเซล $src(x,y)$ นั้นมีค่าสูงกว่า $thresh$ ความเข้มของพิกเซลนั้นจะถูกปรับค่าเป็น $maxVal$ มิฉะนั้นจะถูกปรับค่าเป็น 0

2) **Multiband Thresholding**

Multiband Thresholding คือการทำ Threshold กับภาพที่มีหลาย Channel โดยทำ Threshold ทีละ Channel และนำผลลัพธ์ที่ได้รวมกันด้วยการทำ AND operation

2.1.2 Morphological Transformations

Morphological Transformations คือ การดำเนินงานง่าย ๆ บางอย่างซึ่งขึ้นอยู่กับรูปร่างของภาพ โดยปกติแล้วจะถูกใช้งานในภาพแบบ Binary การใช้งานนั้นจำเป็นต้องมี 2 อินพุต หนึ่งคือภาพ สองคือ Structuring Element หรือ Kernel ซึ่งจะเป็นตัวกำหนดพฤติกรรมของการดำเนินงาน โดยมี Operators พื้นฐานที่สำคัญคือ Erosion และ Dilation จากนั้นจึงนำไปใช้งานในหลาย ๆ แบบเช่น Opening Closing Gradient เป็นต้น

2.1.2.1 Erosion

Erosion นั้นจะทำการกัดเซาะขอบของวัตถุ ด้วยการเลื่อน Kernel ผ่านภาพ พิกเซลในภาพต้นฉบับ (ไม่เป็น 1 ก็ 0) นั้นจะถูกตัดสินให้เป็น 1 ก็ต่อเมื่อทุกพิกเซลใน Kernel เดียวกันเป็น 1 หมด มิฉะนั้นจะถูกกัดเซาะไป (ทำให้เป็น 0)

ดังนั้นการ Erosion จะทำให้พิกเซลทั้งหมดที่อยู่ใกล้ขอบถูกทิ้งไปโดยขึ้นอยู่กับขนาดของ Kernel จึงทำให้วัตถุนั้นมีขนาดเล็กลง หรือก็คือการลดขนาดของส่วนสีขาวในภาพ ซึ่งมีประโยชน์ในการลบ Noise สีขาวขนาดเล็ก ๆ หรือการแยกวัตถุสองชิ้นที่ติดกัน เป็นต้น

รูปที่ 2.1 การเปรียบเทียบก่อน-หลังการทำ Erosion

2.1.2.2 Dilation

Dilation คือ วิธีการตรงกันข้ามกับ Erosion โดยพิกเซลจะมีค่าเป็น 1 ถ้าหากว่ามีอย่างน้อย 1 พิกเซลใน Kernel เป็น 1 ดังนั้นจึงทำให้ส่วนสีขาวในภาพหรือขนาดของวัตถุใหญ่ขึ้น โดยปกติแล้วในกรณีที่ลบ Noise ด้วย Erosion นั้นมักจะทำ Dilation ต่อ เพราะการทำ Erosion นั้นนอกจากจะลบ

Noise แล้วยังทำให้วัตถุเล็กลงด้วย จึงทำการ Dilate ภาพ เนื่องจาก Noise นั้นได้หายไปแล้วดังนั้นจึงไม่กลับมาอีก แต่ขนาดของวัตถุนั้นเพิ่มขึ้น นอกจากนี้ Dilation ยังมีประโยชน์ในการเชื่อมส่วนที่ ขาดออกจากกันของวัตถุ

รูปที่ 2.2 การเปรียบเทียบก่อน-หลังการทำ Dilation

2.1.2.3 Opening

Opening คือ การ Erosion แล้วตามด้วย Dilation มีประโยชน์ในการลบ Noise

รูปที่ 2.3 การเปรียบเทียบก่อน-หลังการทำ Opening

2.1.2.4 Closing

Closing คือ วิธีตรงกันข้ามกับ Opening หรือก็คือ Dilation แล้วตามด้วย Erosion มีประโยชน์ในการปิดรูหรือจุดดำเล็ก ๆ ในวัตถุ

รูปที่ 2.4 การเปรียบเทียบก่อน-หลังการทำ Closing

2.1.3 **Edge Detection**

Edge Detection คือ การหาขอบของวัตถุในภาพด้วยการหาความไม่ต่อเนื่องของความสว่าง ซึ่งมักถูกใช้ในการทำ Segmentation หรือการดึงข้อมูลจากภาพ

2.1.3.1 Sobel Operator

Sobel Operator คือตัวดำเนินการเปลี่ยนแปลงแบบไม่ต่อเนื่อง (Discrete Differentiation Operator) โดยจะคำนวณหาค่าประมาณของการไล่ระดับของฟังก์ชันความเข้มของภาพ โดยมีวิธีการคำนวณดังนี้

1) คำนวณหาอนุพันธ์ 2 ค่า

a. การเปลี่ยนแปลงในแนวนอน หาได้โดยทำการคอนโวลูชันระหว่างภาพ (I) กับ Kernel G_x ที่มีขนาดเป็นคี่ เช่นถ้าใช้ Kernel ที่มีขนาดเป็น 3 จะได้ G_x เป็น

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * I$$

b. การเปลี่ยนแปลงในแนวตั้ง หาได้โดยทำการคอนโวลูชันระหว่างภาพ (I) กับ Kernel G_y ที่มีขนาดเป็นคี่ เช่นถ้าใช้ Kernel ที่มีขนาดเป็น 3 จะได้ G_y เป็น

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * I$$

2) คำนวณหาค่าประมาณของการไล่ระดับกับทุก ๆ พิกเซลในรูปแล้วนำผลลัพธ์ทั้งสองจากด้านบน มารวมกันโดย

$$\mathrm{G} = \sqrt{G_x^2 + G_y^2}$$

2.1.3.2 Canny Edge Detector

Canny Edge Detector คือการหาขอบโดยใช้อัลกอริทึมหลายขั้นตอน โดยมีขั้นตอนดังนี้

1) ลด Noise

เนื่องจากการหาขอบนั้นได้รับผลกระทบจาก Noise ได้ง่ายจึงทำการลบ Noise ในภาพออกด้วยการใช้ Gaussian filter ขนาด 5x5

2) หาการไล่ระดับของความเข้มของภาพ

จากนั้นภาพจะถูกนำมากรองด้วย Sobel Kernel ทั้งในแนวนอนและแนวตั้งเพื่อหาอนุพันธ์อันดับ 1 ในแนวนอน (G_x) และในแนวตั้ง (G_y) จากนั้นจึงนำมาหาค่าการไล่ระดับของขอบ (Edge Gradient) และทิศทางของแต่ละพิกเซล ได้ดังนี้

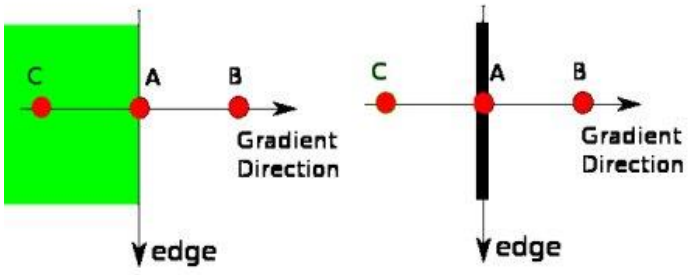
$$Edge\_Gradient\ (G) = \sqrt{G_x^2 + G_y^2}$$

$$Angle\ (\theta) = tan^{-1}\left(\frac{G_y}{G_x}\right)$$

ทิศทางของการไล่ระดับนั้นจะตั้งฉากกับขอบเสมอ ซึ่งจะถูกปัดให้เป็น 1 ใน 4 มุมที่แสดงถึงแนวตั้ง แนวนอน และแนวทแยงอีก 2 ทิศทาง

3) Non-maximum Suppression

หลังจากหาขนาดและทิศทางของการไล่ระดับได้แล้ว จะทำการลบพิกเซลที่ไม่ต้องการซึ่งไม่ได้เป็นส่วนหนึ่งของขอบ โดยทุก ๆ พิกเซลจะถูกตรวจสอบว่าเป็นค่าสูงสุดของพิกเซลใกล้เคียงในทิศทางการไล่ระดับเดียวกันหรือไม่ ดังรูปที่ 2.7



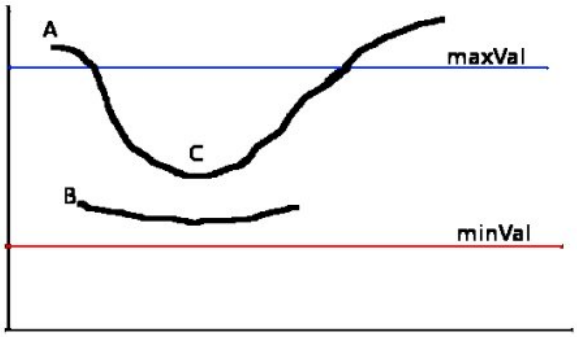
รูปที่ 2.5 ตัวอย่างการทำ Non-maxima Suppression

จากรูป จุด A นั้นอยู่บนขอบ (ในแนวตั้ง) ทิศทางของการไล่ระดับ (Gradient Direction) นั้นตั้งฉากกับขอบ จุด B และ C นั้นอยู่ในทิศทางการไล่ระดับ ดังนั้นจุด A จึงถูกตรวจสอบกับจุด B และ C ว่ามีค่าสูงสุดหรือไม่ หากใช่ก็จะถูกนำไปใช้ในขั้นตอนถัดไป มิฉะนั้นก็จะถูกลบทิ้ง

(ทำให้เป็น 0) ซึ่งขั้นตอนนี้จะให้ผลลัพธ์เป็นภาพแบบ Binary (ภาพที่แต่ละพิกเซลจะมีค่าความเข้มแสงที่เป็นไปได้แค่ 2 ค่า) ที่มีเส้นขอบบาง ๆ

4) Hysteresis Thresholding

ในขั้นตอนนี้จะตัดสินว่าเส้นขอบใดเป็นเส้นขอบจริง ๆ โดยจะใช้ Threshold สองค่าได้แก่ minVal และ maxVal ซึ่งเส้นขอบใดก็ตามที่มีความเข้มสูงกว่าค่า maxVal จะถูกตัดสินว่าแน่ใจว่าเป็นเส้นขอบจริง ส่วนเส้นขอบที่มีค่าต่ำกว่า minVal จะถูกตัดสินว่าแน่ใจว่าไม่เป็นเส้นขอบและถูกลบทิ้งไป ส่วนเส้นขอบที่มีค่าอยู่ระหว่างค่า Threshold ทั้งสอง จะถูกคิดว่าเป็นเส้นขอบก็ต่อเมื่อมีส่วนที่เชื่อมต่ออยู่กับเส้นขอบที่แน่ใจว่าเป็นเส้นขอบจริง มิฉะนั้นก็จะถูกลบทิ้งไป เช่น



รูปที่ 2.6 ตัวอย่างการทำ Hysteresis Thresholding

เส้นขอบ A นั้นมีค่าสูงกว่า maxVal จึงถูกตัดสินว่าแน่ใจว่าเป็นเส้นขอบจริง ส่วนเส้นขอบ C ถึงแม้ว่าจะอยู่ต่ำกว่าค่า maxVal แต่เส้นขอบ C เชื่อมต่อกับเส้นขอบ A จึงถูกตัดสินว่าเป็นเส้นขอบจริง แต่เส้นขอบ B นั้นถึงแม้จะสูงกว่าค่า minVal แต่ก็ไม่ได้เชื่อมต่อกับเส้นขอบที่แน่ใจว่าเป็นเส้นขอบจริงใด ๆ จึงถูกลบทิ้งไป

2.1.4 **Hough Transform**

เป็นเทคนิคที่ใช้ในการหารูปร่างที่ต้องการในภาพ ซึ่งโดยทั่วไปแล้ว Classical Hough transform จะถูกใช้บ่อยสุด เพื่อหาเส้นโค้งปกติ เช่น เส้นตรง วงกลม และ วงรี เป็นต้น

2.1.4.1 Hough Line Transform

Hough Line Transform ถูกใช้เพื่อหาเส้นตรง โดยมักใช้กับภาพที่ผ่านการทำ Edge detection มาแล้ว ซึ่งวิธี Hough Line Transform เพื่อจะกล่าวถึงเส้นตรงในระบบ Polar ซึ่งเขียนเป็นสมการเส้นตรงได้ดังนี้

$$y = \left(-\frac{cos\theta}{sin\theta}\right)x + \left(\frac{r}{sin\theta}\right)$$

จัดรูปได้เป็น $r = xcos\theta + ysin\theta$

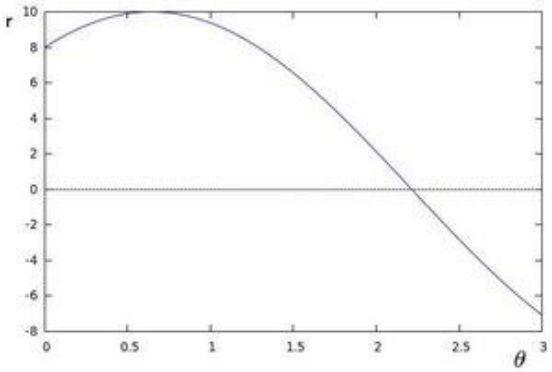
จากนั้นการทำ Hough Line Transform จะมีวิธีการดังนี้

1) โดยปกติแล้วในแต่ละจุด (x_0, y_0) จะสามารถหาเส้นตรงที่ผ่านจุดนั้นได้โดย

$$r\theta = x_0 \cdot cos\theta + y_0 \cdot sin\theta$$

หมายความว่าแต่ละคู่ของ (r_θ, θ) จะแสดงถึงเส้นตรงที่ผ่านจุด (x_0, y_0)

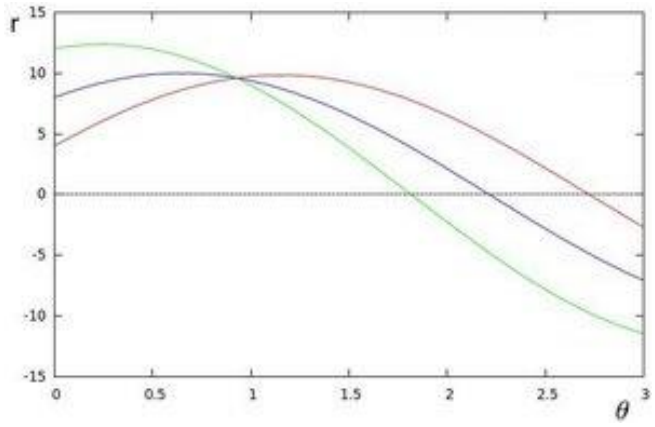
2) ถ้าหากพล็อตกราฟของทุกเส้นที่ผ่านจุด (x_0, y_0) จะได้คลื่นแบบ Sine Wave เช่นให้ $x_0 = 8$ และ $y_0 = 6$ จะพล็อตกราฟได้ดังนี้



รูปที่ 2.7 กราฟของเส้นตรงที่ผ่านจุด (x_0,y_0)

ซึ่งจะพิจารณาแค่จุดที่ค่า $r > 0$ และ $0 < \theta < 2\pi$

3) ทำแบบข้อ 2 กับทุก ๆ จุดในภาพ ถ้าหากเส้นโค้งใดตัดกันแสดงว่าจุดทั้งสองนั้นเป็นจุดในเส้นตรงเดียวกัน เช่น จากตัวอย่างด้านบน พล็อตจุดเพิ่มอีก 2 จุดได้แก่ $x_1 = 4, y_1 = 9$ และ $x_2 = 12, y_2 = 3$ จะได้



รูปที่ 2.8 กราฟของเส้นตรงที่ผ่านจุด (x_0,y_0), (x_1,y_1) และ (x_2,y_2)

กราฟทั้งสามนั้นตัดกันที่จุด $(0.925, 9.6)$ ซึ่งพิกัดนี้คือค่า (θ, r) ของเส้นที่จุด $(x_0, y_0), (x_1, y_1)$ และ (x_2, y_2) อยู่

4) จากขั้นตอนที่ผ่านมานั้นทำให้สามารถหาเส้นตรงได้จากการหาจำนวนของจุดตัดระหว่างเส้นโค้งทั้งหลาย ยิ่งโค้งตัดที่จุดตัดใดมาก หมายความว่าเส้นตรงที่ผ่านจุดตัดนั้นก็ยิ่งมีหลายจุด ดังนั้นสามารถกำหนด Threshold เป็นค่าที่น้อยสุดที่ต้องการเพื่อตรวจหาเส้นตรงได้ หากจำนวนของจุดตัดมีค่ามากกว่าค่า Threshold ก็จะถูกตัดสินให้เป็นเส้นตรงที่มีค่า (θ, r) เป็นค่าของจุดตัดนั้น

2.1.5 **Segmentation**

2.1.5.1 Edge-based Segmentation

คือการทำ Segmentation ที่อาศัยขอบของวัตถุเป็นตัวแบ่งแยกระหว่างวัตถุและพื้นหลัง โดยมีวิธีการทำคือ หาขอบของวัตถุด้วยการทำ Edge Detection จากนั้นจึงนำขอบที่ได้มาเติมช่องว่างภายใน ซึ่งจะได้ผลลัพธ์เป็น Segment ของวัตถุแต่ละชิ้น

2.1.5.2 Region-based Segmentation

คือการทำ Segmentation ที่อาศัยขอบเขตอื่น ๆ ในการแบ่งแยกระหว่างวัตถุ เช่น ใช้ลักษณะพื้นผิวของวัตถุ เป็นต้น

2.2 Machine Learning

ช่วยให้เกิดการพัฒนาปรับปรุงประสิทธิภาพการทำงานของระบบให้ดีขึ้น ในการทำให้เครื่องจักรสามารถเรียนรู้ได้ด้วยตัวเองต้องมีการป้อนข้อมูลตัวอย่าง หรือสภาพแวดล้อม เมื่อเครื่องจักรเรียนรู้ข้อมูลตัวอย่างแล้ว จึงนำความรู้ที่เรียนได้เก็บไว้ในฐานความรู้ด้วยรูปแบบการแทนความรู้

2.2.1 **XGBoost**

XGBoost คือ ไลบรารี่ Gradient Boosting ที่ถูกปรับปรุงเพื่อให้มีประสิทธิภาพและความยืดหยุ่นสูง ซึ่ง XGBoost นั้นใช้อัลกอริทึมในเฟรมเวิร์คของ Gradient Boosting โดย XGBoost นั้นใช้งาน Parallel Tree Boosting (หรือรู้จักในชื่อ GBDT, GBM) ซึ่งสามารถแก้ปัญหาทางวิทยาศาสตร์ได้รวดเร็วและแม่นยำ

2.2.2 **Deep Learning**

Deep Learning เป็นซับเซ็ตของ Machine Learning ซึ่งประกอบไปด้วยอัลกอริทึมที่ได้แรงบันดาลใจมาจากโครงสร้างและการทำงานของเซลล์สมอง ซึ่งถูกเรียกว่า Artificial Neural Networks

2.2.2.1 Convolutional Neural Network (ConvNet/CNN)

CNN เป็นอัลกอริทึม Deep Learning ซึ่งสามารถรับอินพุตเป็นรูปภาพจากนั้นจึงกำหนดความสำคัญ (Learnable Weights and Biases) ให้กับลักษณะหรือวัตถุในภาพ ทำให้สามารถแยกแยะความแตกต่างของรูปแต่ละรูปได้ ซึ่ง CNN นั้นต้องการการทำ Pre-Processing น้อยกว่าเมื่อเทียบกับอัลกอริทึมอื่น เพราะ CNN นั้นมีความสามารถในการเรียนรู้ Filters หรือ Characteristics ได้

2.2.2.2 You Only Look Once (YOLO)

YOLO เป็น Neural Network ที่ใช้อัลกอริทึม Deep Learning สำหรับการทำ Object Detection ซึ่ง YOLO จะทำการแยกประเภทวัตถุในภาพและคำนวณหาว่าวัตถุนั้นอยู่ที่ใดในภาพ โดย YOLO นั้นจะประกอบไปด้วย 3 ส่วนหลัก ๆ ได้แก่

1) Algorithm (Prediction Vector)

ซึ่งจะทำการแบ่งภาพออกเป็นตาราง SxS เพื่อใช้คำนวณหา Bounding Box ที่มีวัตถุอยู่ จากนั้นจึงใช้ข้อมูลนี้ไปคำนวณหาประเภทของวัตถุ

2) Network

ซึ่งมีโครงสร้างเหมือน CNN ทั่วไป โดยประกอบไปด้วย Convolutional, Max-Pooling Layers และ Fully Connected CNN อีก 2 Layers

3) Loss Function

YOLO นั้นจะคำนวณ Bounding Boxes หลายกล่องต่อหนึ่งตารางในภาพ Loss Function จึงถูกใช้งานเพื่อคำนวณหา Loss สำหรับ Bounding Box ที่เป็น True Positive เพื่อปรับปรุงให้การคำนวณหา Bounding Boxes นั้นดีขึ้นสำหรับบางสัดส่วนหรือขนาดของวัตถุ

2.3 Software Tool

2.3.1 **Visual Studio Code**

Visual Studio Code (VS Code) เป็นโปรแกรม Code Editor โดยไมโครซอฟต์ ใช้ในการปรับแต่งแก้ไขโค้ดและถูกพัฒนาออกมาเป็น Free Open source ทำให้ผู้ใช้งานสามารถใช้งานได้แบบไม่มีค่าใช้จ่าย รองรับการใช้งานหลากหลายระบบปฏิบัติการทั้งบนระบบปฏิบัติการ Windows macOS และ Linux สนับสนุนหลายภาษาทั้งภาษา JavaScript TypeScript และ Node.js สามารถเชื่อมต่อกับ Git ได้ รองรับการทำงานข้ามแพลตฟอร์ม การนำมาใช้งานทำได้ง่าย ไม่ซับซ้อน มีเครื่องมือส่วนขยายให้เลือกใช้มากมาย เช่น การเปิดใช้งานภาษาอื่น ๆ (C++ C# Java Python PHP Go) , ธีม, ดีบั๊กเกอร์ เป็นต้น Adobe XD

2.3.2 **Adobe XD**

Adobe XD (XD ย่อมาจาก Experience Design) เป็นโปรแกรมสำหรับสร้างต้นแบบของแอปพลิเคชันหรือเว็บไซต์ให้มีความเสมือนจริง ใช้ประกอบการนำเสนองานก่อนจะไปสู่ขั้นตอนของการพัฒนาแอปพลิเคชันหรือเว็บไซต์จริง โดยการนำเสนองานของตัวโปรแกรมสามารถแสดงผลได้จากทั้งบนหน้าเว็บไซต์หรือสมาร์ทโฟนแท็บเล็ต โดยกระบวนการทำงานของโปรแกรมจะแบ่งออกเป็น 3 ขั้นตอนใหญ่ๆ ได้แก่ Design จะเป็นส่วนที่ใช้ออกแบบแอปพลิเคชันหรือเว็บไซต์ว่าจะมีทั้งหมดกี่หน้าและแต่ละหน้านั้นจะมีหน้าตาเป็นอย่างไร Prototype เป็นส่วนที่จะนำแต่ละหน้าจากส่วนของ Design มาเชื่อมโยงกันและสร้างเป็นต้นแบบขึ้นมา Publish เป็นส่วนของการอัพโหลดขึ้นคลาวด์ของ Adobe เพื่อนำไปใช้งานนำเสนอในลำดับต่อไป

2.3.3 **GitKraken**

GitKraken มีการทำงานเหมือนกับ Git และสามารถใช้งานได้โดยไม่ต้องติดตั้งตัว Git GitKraken เป็นโปรแกรมเบื้องต้นสำหรับการใช้งาน Git มีหน้าที่ในการช่วยดูรายละเอียดกิจกรรมบน Git โดยตัวโปรแกรมมีอินเตอร์เฟซที่ง่ายต่อการใช้งานและสามารถเชื่อมกับ Git Program อื่นได้ GitKraken เป็น GUI (Graphical User Interface) ที่จะช่วยทำให้ไม่ต้องพิมพ์คำสั่งทีละบรรทัด แต่สามารถตรวจสอบและจัดการผ่านตัวโปรแกรมได้

2.4 Runtime Environment

2.4.1 **Node.js**

Node.js คือ JavaScript runtime environment แบบ Open Source และ Cross Platform โดยใช้ V8 JavaScript engine ซึ่งเป็นแกนหลักของ Google Chrome ทำงานนอกเบราว์เซอร์ ทำให้ Node.js นั้นมีประสิทธิภาพสูง

2.5 Software Library

2.5.1 **OpenCV**

OpenCV (Open Source Computer Vision Library) คือซอฟท์แวร์ไลบรารี่แบบ Open source สำหรับ Computer vision และ Machine learning

โดยไลบรารี่นี้มีอัลกอริทึมมากกว่า 2500 อัลกอริทึม ซึ่งอัลกอริทึมเหล่านี้สามารถใช้เพื่อตรวจจับและจำแนกใบหน้า แยกชนิดวัตถุ จำแนกการกระทำของมนุษย์ในวิดีโอ ติดตามการเคลื่อนที่ของกล้อง ติดตามการเคลื่อนที่ของวัตถุ สกัดโมเดล 3 มิติจากวัตถุ รวมภาพหลาย ๆ ภาพเข้าด้วยกันเพื่อสร้างภาพที่มีความละเอียดสูงสำหรับฉากทั้งฉาก หาภาพที่ใกล้เคียงกันใน Database ติดตามการเคลื่อนที่ของดวงตา เป็นต้น

2.5.2 **Scikit-image**

Scikit-image คือ Image Processing ซอฟท์แวร์ไลบรารี่แบบ Open Source สำหรับภาษา Python

โดยอัลกอริทึมในไลบรารี่นี้ถูกออกแบบมาเพื่อใช้กับ SciPy ซึ่งประกอบไปด้วย IO morphology filtering warping color manipulation object detection เป็นต้น

2.5.3 **Scikit-learn**

Scikit-learn คือ Machine Learning ซอฟท์แวร์ไลบรารี่แบบ Open Source สำหรับภาษา Python ซึ่งถูกสร้างขึ้นมาบน NumPy, SciPy และ Matplotlib

2.5.4 **Tensorflow**

Tensorflow คือ ไลบรารี่แบบ Open Source ที่ช่วยพัฒนาและฝึกฝน Machine Learning โมเดล โดย TensorFlow นั้นถูกพัฒนาโดยนักวิจัยและวิศวกรที่ทำงานในทีม Google Brain ซึ่งอยู่ภายใต้องค์กร Google's Machine Intelligence Research เพื่อทำการวิจัยเกี่ยวกับ Machine Learning และ Deep Neural Networks

2.5.5 **Socket.IO**

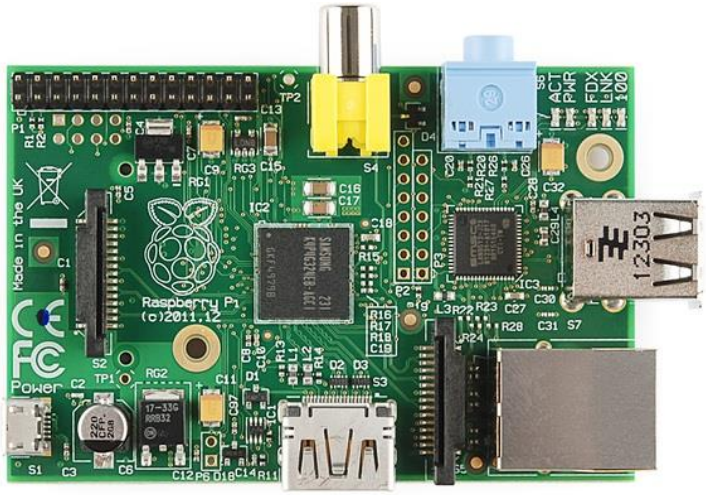
Socket.IO คือไลบรารี่ที่ทำให้สามารถติดต่อสื่อสารระหว่างเบราว์เซอร์กับเซิร์ฟเวอร์ได้แบบ real-time, bidirectional และ event-based โดยประกอบไปด้วย 2 ส่วนคือ

1) เซิร์ฟเวอร์ Node.js

ไลบรารี่ JavaScript Client สำหรับเบราว์เซอร์ และ Node.js

2.6 Raspberry Pi

บอร์ดคอมพิวเตอร์ขนาดเล็ก (Single-Board Computer หรือ SBC) ที่สามารถเชื่อมต่อกับจอมอนิเตอร์ คีย์บอร์ด และเมาส์ได้ สามารถนำมาประยุกต์ใช้ในการทำโครงงานทางด้านอิเล็กทรอนิกส์ การพัฒนาโปรแกรม การเขียนโปรแกรมบนอุปกรณ์ระบบสมองกลฝังตัวหรือเป็นเครื่องคอมพิวเตอร์ตั้งโต๊ะขนาดเล็ก ใช้งานได้เหมือนคอมพิวเตอร์เครื่องหนึ่ง รองรับระบบปฏิบัติการลินุกซ์ (Linux Operating System) ได้หลายระบบ เช่น Raspbian (Debian) Pidora (Fedora) และ Arch Linux โดยบอร์ด Raspberry Pi ถูกออกแบบมาให้มี CPU GPU และ RAM อยู่ภายในชิปเดียวกัน มีจุดเชื่อมต่อ GPIO ให้ผู้ใช้สามารถนำไปใช้ร่วมกับอุปกรณ์อิเล็กทรอนิกส์ได้



รูปที่ 2.9 บอร์ด Raspberry Pi

2.7 Delta Robot Kinematic

เมื่อกล่าวถึงโครงสร้างของหุ่นยนต์ประเภทเดลต้า มีส่วนประกอบคือ

1.)ชิ้นส่วนสามก้านโยงด้านบน (Actuating Arm)

2) ชิ้นส่วนสามก้านโยงด้านล่าง (Passive Arm)

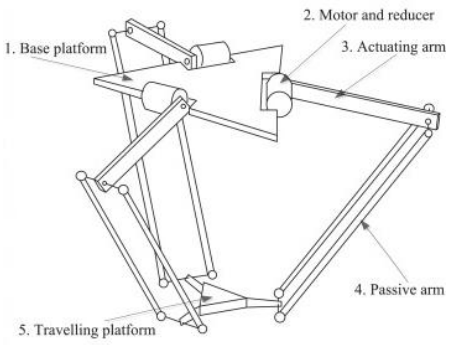
3) แผ่นปลายแขนที่เคลื่อนที่ (Travelling Platform)

จะมีความสมมาตรในแต่ละด้านที่เท่ากัน ซึ่งปลายด้านบนจะถูกเชื่อมต่อกับมอเตอร์ที่วางอยู่บนโครงฐาน ส่วนปลายอีกด้านจะเชื่อมต่ออยู่กับแผ่นปลายแขนที่เคลื่อนที่ โดยมอเตอร์ทั้งสามตัวจะถูกเชื่อมต่อกับแผ่นฐาน

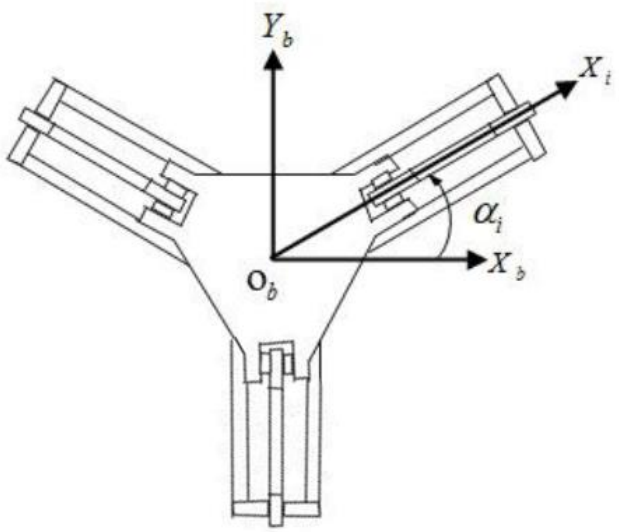
เพื่อยกก้านโยงด้านบนให้เอียงทำมุม θ_{i1} แผ่นปลายแขนที่เคลื่อนที่ถูกกำหนดให้รักษาระนาบให้ขนานกับแผ่นฐาน การเคลื่อนที่ไปพร้อมกันของแขนทั้งสามก้าน จะสร้างการเคลื่อนที่ในแนวเชิงเส้น

ปลายแขนที่เคลื่อนที่จะสามารถติดตั้งแขนจับ (Gripper) เพื่อให้ในการหยิบจับสิ่งของหรือชิ้นส่วนอุปกรณ์ แกนและกรอบอ้างอิงแบบเฉื่อยที่อยู่ตรงกลางของแผ่นฐาน ด้านล่างเรียกว่า (o_b,x_b,y_b,z_b) ส่วนแกนและพิกัดอ้างอิงแบบเฉื่อยที่อยู่ตรงกลางของแผ่นปลายแขนที่เคลื่อนที่เรียกว่า (o_r,x_r,y_r,z_r) และมุม α_i ที่เกี่ยวข้องกับแต่ละก้านโยงคู่ขนานที่ i จะมีค่าเป็น $\pi/6$, $5\pi/6$ และ $3\pi/6$ สำหรับ i= 1,2,3

เมื่อวัดเทียบกับกรอบอ้างอิงแบบเฉื่อยที่จุด o_b โดยที่แต่ละก้านโยงด้านบน จะมีความยาว $L_{1i} = 23$ cm จะถูกติดตั้งกับแผ่นฐานด้านบนที่มีรัศมีเป็น $R_i = R = 10$ cm จากจุดตรงกลาง o_b โดยใช้ข้อแต่แบบหมุน (Revolute Joint) ส่วนความยาวก้านโยงด้านล่างจะมีความยาว $L_{2i} = 24.5$ cm โดยติดตั้งไว้กับแผ่นปลายแขนที่เคลื่อนที่ มีรัศมีเป็น $r = 4$ cm วัดจากจุดตรงกลาง o_r โดยมุม θ_{i2} และ θ_{i3} ของก้านโยงส่วนด้านล่างจะเกิดจากข้อจำกัดในการเคลื่อนที่ของกลไกแบบเดลต้า เพื่อให้สอดคล้องกับการเคลื่อนที่ของสามก้านโยงด้านบนไปพร้อมกัน



รูปที่ 2.10 โครงสร้างของหุ่นยนต์แบบเดลต้า



รูปที่ 2.11 แผ่นฐานของหุ่นยนต์แบบเดลต้าและมุมมองด้านบนของแต่ละก้านโยงคู่ขนานด้านบน

เพื่อจะเปลี่ยนแปลงองศาการหมุนของมอเตอร์ทั้งสามตัว θ_{i1} ที่จะใช้ขับเคลื่อนก้านโยงด้านบนทั้งสาม และตำแหน่งจุดกึ่งกลางของแผ่นปลายแขนที่เคลื่อนที่ (Travelling Plate) จะถูกมาใช้อธิบายโดยทฤษฎีจลนศาสตร์ไปข้างหน้า (Forward Kinematic) และ จลนศาสตร์แบบผกผัน (Inverse Kinematic) โดยทฤษฎีจลนศาสตร์ไปข้างหน้า (Forward Kinematic) สามารถคำนวณหาจุดศูนย์กลางของปลายแขนที่เคลื่อนที่ เมื่อรู้ค่ามุมของก้านโยงด้านบนทั้งสามด้าน θ_{i1} สำหรับ i = 1,2,3 สูตรการนวณหาจุดศูนย์กลางของปลายแขน จะถูกกำหนดในชุดของสมการพีชคณิตกำลังสองที่เชื่อมสามสมการที่จะทำงานสอดคล้องไปพร้อมๆกัน ในทางกลับกันจลนศาสตร์แบบผกผัน (Inverse Kinematic) จะคำนวณหามุมของมอเตอร์ทั้งสาม θ_{i1}จากตำแหน่งของจุดศูนย์กลางของปลายแขนที่ต้องการเคลื่อนที่ ซึ่งจะถูกคำนวณได้จากสามสมการกำลังสองที่เป็นอิสระแก่กัน ดังนั้นในการแก้สมการจะสามารถแก้แต่ละสมการแยกจากกันได้ รวมถึงความเร็วและความเร่งจองแต่ละข้อต่อ จะสามารถหาได้จากสมการทางจลนศาสตร์ ในรูปแบบของเมทริกซ์จาโคเบียน (Jacobian)

2.7.1 **จลนศาสตร์ไปข้างหน้า (Forward Kinematics)**

ในการใช้สมการจลน์ศาสตร์ไปข้างหน้า (Forward Kinematic) ตำแหน่งของจุดกึ่งกลาง o_t ของแผ่นปลายแขนที่เคลื่อนที่จะสามารถถูกคำนวณโดยใช้มุมของก้านโยงทั้งสามก้าน เป็นไปตามข้อจำกัดในการเคลื่อนที่ของโครงสร้างแบบลูปปิดของหุ่นยนต์แบบเดลต้า โดยตำแหน่งของจุดกึ่งกลางของแผ่นปลายแขนที่เคลื่อนที่ (x_t,y_t,z_t) ถูกกำหนดให้อยู่ที่จุดตัดของสามทรงกลมที่รัศมี L_2 ซึ่งแต่ละทรงกลมมีจุดศูนย์กลางที่ปลายก้านโยงด้านบน สามารถเขียนอธิบายได้ดังสมการ

$$x_i = (R + L_1 \cos\theta_{i1} - r)\cos\alpha_i$$

$$y_i = (R + L_1 \cos\theta_{i1} - r)\sin\alpha_i$$

$$z_i = -L_1 \sin\alpha_i \quad \text{สำหรับ } i = 1,2,3$$

สมการของสามทรงกลมที่เชื่อมโยงเป็นสมการกำลังสอง จะต้องถูกแก้สมการไปพร้อมๆกัน สำหรับหาตำแหน่งของจุดกึ่งกลางของแผ่นปลายแขนที่เคลื่อนที่ (x_t,y_t,z_t) ดังสมการ

$$(x_t + x_i)^2 + (y_t + y_i)^2 + (z_t + z_i)^2 = L_2^2$$

จากการแก้สมการดังกล่าวจะมีอยู่ด้วยกันสองคำตอบที่เป็นไปได้สำหรับแต่ละองศา θ_{i1} ซึ่งจะสอดคล้องกับสองท่าทางที่เป็นไปได้ของหุ่นยนต์แบบเดลต้า สำหรับจุดศูนย์กลาง o_t

2.7.2 **จลนศาสตร์แบบผกผัน (Inverse Kinematics)**

สำหรับการกำหนดตำแหน่งของจุดกึ่งกลาง $o_t : P = [x_t,y_t,z_t]$ แต่ละมุม θ_{i1} ที่ไม่ทราบของก้านโยงด้านบนจะถูกคำนวณตามสูตรที่ไม่เป็นแบบเชิงเส้นในสมการ

$$\tan\left(\frac{\theta_{i1}}{2}\right) = \frac{-2z_{pi} \pm \sqrt{4z^2{}_{pi} + 4R_i^2 - S_i^2 + Q_i^2\left(1 - \frac{R_1^2}{L_1^2}\right) - Q_i\left(4R_1 + \frac{2R_1S_i}{L_1}\right)}}{-2R_1 - Q_1\left(\frac{R_1}{L_1} - 1\right) - S_i}$$

เมื่อ

$$-2R_1 = R - r$$

$$Q_i = 2X_{ti}cos_i + 2Y_{ti}sin\alpha_i$$

$$S_i = \frac{\left(-X_{ti}^2 - Y_{ti}^2 - Z_{ti}^2 + L_2^2 - L_1^2 - R_1^2\right)}{L_1}$$

สำหรับ i = 1,2,3 จะมี 2 ผลลัพธ์ที่เป็นไปได้สำหรับมุม θ_{i1} ที่สอดคล้องกับการเคลื่อนที่ขึ้นหรือการเคลื่อนที่ลงของแต่ละก้านโยงด้านบน โดยที่คำตอบของสมการจลน์ศาสตร์แบบผกผันในการสมการ

$$4z_{pi}^2 + 4R_1^2 - S_i^2 + Q_i^2\left(1 - \frac{R_1^2}{L_1^2}\right) - Q_i\left(4R_1 + \frac{2R_1S_i}{L_1}\right) \geq 0$$

จะหาคำตอบได้ก็ต่อเมื่อเงื่อนไขที่กำหนดเป็นจริง โดยจำนวนครั้งของการประมวลผลในการคำนวณหาผลลัพธ์ทั้งสามค่าของ θ_{i1} เมื่อกำหนดตำแหน่ง o_t ที่ต้องการจากสมการจลนศาสตร์แบบผกผัน (Inverse Kinematics) นั้นจะน้อยกว่าจำนวนครั้งของการประมวลผลในการคำนวณหาตำแหน่ง o_t จากสมการจลนศาสตร์แบบไปข้างหน้า (Forward Kinematics) ดังนั้นพื้นที่การทำงานของหุ่นยนต์แบบเดลต้า จะสามารถคำนวณได้อย่างมีประสิทธิภาพโดยใช้สมการจลนศาสตร์แบบผกผัน (Inverse Kinematics)

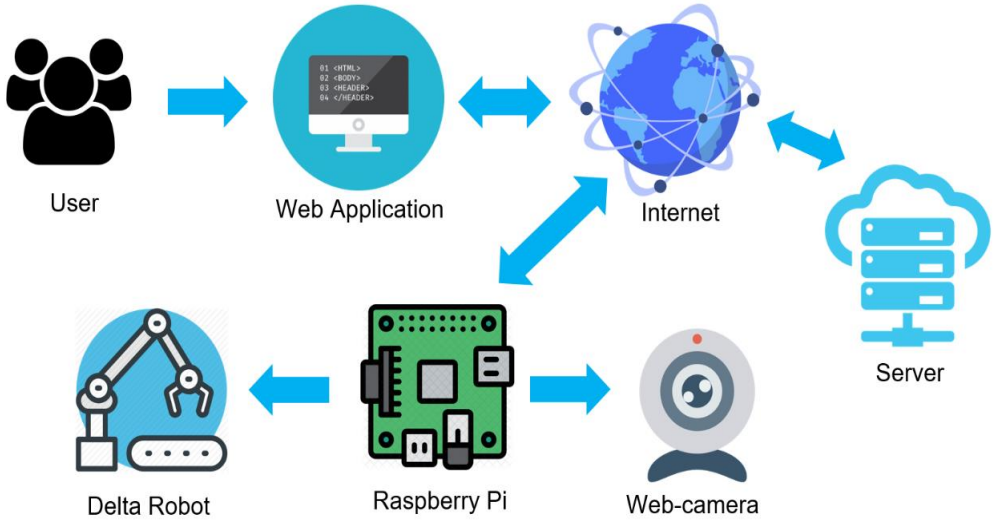
บทที่ 3

การออกแบบและพัฒนาระบบ

การออกแบบและพัฒนาระบบแบ่งเป็น 7 ส่วน คือ ภาพรวมของระบบ (Conceptual Design) แผนภาพยูสเคส (Usecase Diagram) ส่วนต่อประสานกับผู้ใช้ (User Interface) ภาพรวมการจัดการอุปกรณ์ โครงสร้างหลัก ขั้นตอนการเพิ่ม Dataset เพื่อเทรนโมเดล ขั้นตอนการแยกวัตถุ

3.1 ภาพรวมระบบ

ระบบได้รับการออกแบบพัฒนาตามแผนภาพที่แสดงในรูปที่ 3.1 โดยมีรายละเอียด ดังนี้



รูปที่ 3.1 ภาพรวมของระบบแสดงความสัมพันธ์ของการทำงานระหว่างอุปกรณ์และเครื่องมือต่างๆ

แอปพลิเคชันสำหรับระบบคัดแยกและเลือกสิ่งของด้วยการเรียนรู้ของเครื่อง ประกอบไปด้วย 3 ส่วน ได้แก่

3.1.1 **เว็บแอปพลิเคชันสำหรับผู้ใช้งาน (Web Application for User)**

เว็บแอปพลิเคชันสำหรับผู้ใช้งาน (Web Application for User) เป็นตัวกลางที่ควบคุมและจัดการกิจกรรมต่างๆ ที่จะเกิดขึ้นในกระบวนการคัดแยก โดยผู้ใช้งานจะสั่งการ Delta Robot ให้ทำงานผ่านตัวเว็บแอปพลิเคชัน เมื่อผู้ใช้งานสั่งให้มีการ Classification Counting และ Learning ข้อมูลจะถูก

ส่งไปยังเซิร์ฟเวอร์ จากนั้นเซิร์ฟเวอร์จะส่งข้อมูลไปให้ Raspberry Pi เพื่อประมวลผลและสั่งการ Delta Robot

3.1.2 **เว็บแอปพลิเคชันสำหรับผู้ดูแลระบบ (Web Application For Admin)**

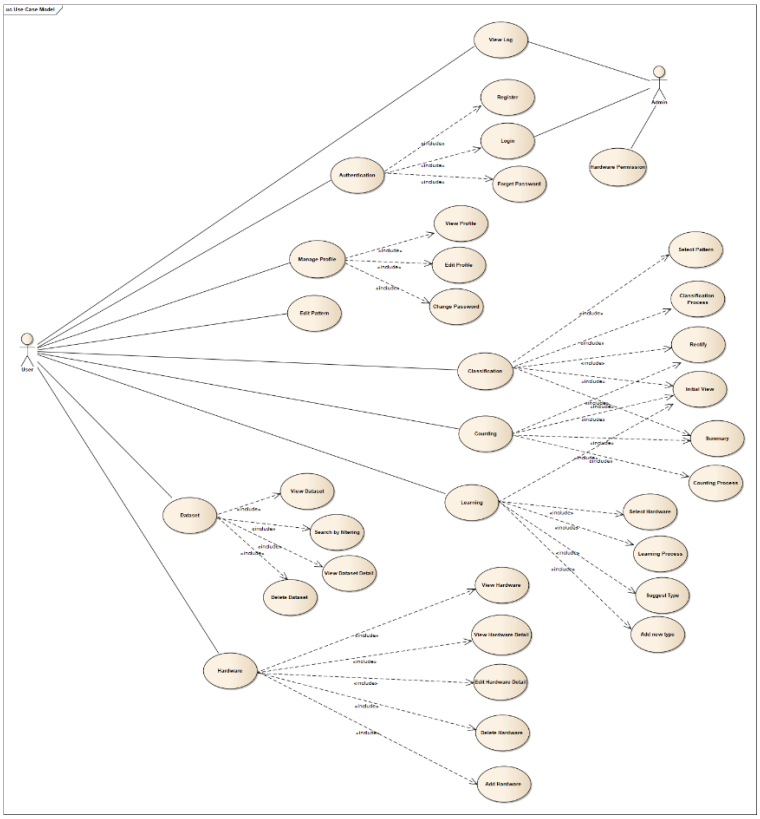
เว็บแอปพลิเคชันสำหรับผู้ดูแลระบบ (Web Application For Admin) เป็นส่วนที่ผู้ดูแลระบบใช้จัดการระบบสำหรับอนุญาติให้ผู้ใช้งานเข้าถึงอุปกรณ์ต่างๆ ในส่วนของการเพิ่มและอนุญาติอุปกรณ์และดูกิจกรรมต่างๆ ที่เกิดขึ้นกับระบบ

3.1.3 **ส่วนของเซิร์ฟเวอร์และฐานข้อมูล**

ส่วนของเซิร์ฟเวอร์และฐานข้อมูลเป็นส่วนที่รวบรวมข้อมูลทั้งเว็บแอปพลิเคชันสำหรับผู้ใช้งานและเว็บแอปพลิเคชันสำหรับผู้ดูแลระบบ

3.2 แผนภาพยูสเคส (Use Case Diagram)

แผนภาพยูสเคสได้รับการออกแบบพัฒนาตามแผนภาพที่แสดงในรูปที่ 3.2 โดยมีรายละเอียดดังนี้



รูปที่ 3.2 แผนภาพแสดงความสัมพันธ์การทำงานของผู้ใช้ระบบและระบบ

จากแผนภาพยูสเคส สามารถแสดงรายละเอียดได้ดังนี้

ตารางที่ 3.1 ตารางแสดงยูสเคสที่ 1 (Register)

Use Case no.	1
Use Case Name	Register
Actors	ผู้ใช้งาน
Pre-Condition	กรอกข้อมูลผู้ใช้งานและกดปุ่ม Register
Post-Condition	ลงทะเบียนสำเร็จ
Brief Description	ส่วนของผู้ใช้งานที่ต้องการสมัครบัญชีผู้ใช้งานระบบและไม่เคยมีบัญชีผู้ใช้งานมาก่อน

ตารางที่ 3.2 ตารางแสดงยูสเคสที่ 2 (Login)

Use Case no.	2
Use Case Name	Login
Actors	ผู้ใช้งานและผู้ดูแลระบบ
Pre-Condition	ทำการ Login เข้าสู่ระบบ
Post-Condition	เข้าสู่หน้าหลัก
Brief Description	ส่วนของผู้ใช้งานและผู้ดูแลระบบที่ต้องการเข้าสู่ระบบ

ตารางที่ 3.3 ตารางแสดงยูสเคสที่ 3 (Forget Password)

Use Case no.	3
Use Case Name	Forget Password
Actors	ผู้ใช้งาน
Pre-Condition	กดปุ่ม Forget Password
Post-Condition	กรอก Email ของผู้ใช้งาน
Brief Description	ส่วนของผู้ใช้งานและผู้ดูแลระบบที่ต้องการลืมรหัสผ่าน

ตารางที่ 3.4 ตารางแสดงยูสเคสที่ 4 (View Profile)

Use Case no.	4
Use Case Name	View Profile
Actors	ผู้ใช้งาน
Pre-Condition	ทำการ Login อยู่ในระบบ
Post-Condition	แสดงข้อมูลส่วนตัวของผู้ใช้งาน
Brief Description	ส่วนของการแสดงข้อมูลส่วนตัวของผู้ใช้งาน

ตารางที่ 3.5 ตารางแสดงยูสเคสที่ 5 (Edit Profile)

Use Case no.	5
Use Case Name	Edit Profile
Actors	ผู้ใช้งาน
Pre-Condition	ดูข้อมูลส่วนตัวของผู้ใช้งาน
Post-Condition	แสดงข้อมูลส่วนตัวที่แก้ไขแล้ว
Brief Description	ส่วนของการแก้ไขรายละเอียดข้อมูลส่วนตัวของผู้ใช้งาน

ตารางที่ 3.6 ตารางแสดงยูสเคสที่ 6 (Change Password)

Use Case no.	6
Use Case Name	Change Password
Actors	ผู้ใช้งาน
Pre-Condition	ทำการ Login อยู่ในระบบ
Post-Condition	-
Brief Description	ส่วนของการเปลี่ยนรหัสผ่านของผู้ใช้งาน

ตารางที่ 3.7 ตารางแสดงยูสเคสที่ 7 (Initial View)

Use Case no.	7
Use Case Name	Initial View
Actors	ผู้ใช้งาน
Pre-Condition	เลือกทำ Classification, Counting, Learning (Select Hardware)
Post-Condition	-
Brief Description	ส่วนของการดูมุมมองสภาพแวดล้อมก่อนเข้ากระบวนการ Classification, Counting, Learning

ตารางที่ 3.8 ตารางแสดงยูสเคสที่ 8 (Select Pattern)

Use Case no.	8
Use Case Name	Select Pattern
Actors	ผู้ใช้งาน
Pre-Condition	Initial View
Post-Condition	แสดง Pattern ที่ต้องการใช้ในขั้นตอนการคัดแยก
Brief Description	ส่วนของการเลือกรูปแบบการคัดแยกจากรูปแบบที่บันทึกไว้หรือจากการสร้างใหม่

ตารางที่ 3.9 ตารางแสดงยูสเคสที่ 9 (Classification Process)

Use Case no.	9
Use Case Name	Classification Process
Actors	ผู้ใช้งาน
Pre-Condition	Select Pattern
Post-Condition	แสดงผลที่ได้จากการคัดแยกเพื่อให้ผู้ใช้งานตรวจสอบความถูกต้อง
Brief Description	ส่วนของการคัดแยกและนับจำนวนของวัตถุ

ตารางที่ 3.10 ตารางแสดงยูสเคสที่ 10 (Rectify)

Use Case no.	10
Use Case Name	Rectify
Actors	ผู้ใช้งาน
Pre-Condition	Classification, Counting
Post-Condition	แสดงสรุปผลที่ได้จากกระบวนการที่ผ่านการตรวจสอบความถูกต้องจากผู้ใช้งาน
Brief Description	ส่วนของการตรวจสอบความถูกต้องของข้อมูลที่ผ่านกระบวนการ

ตารางที่ 3.11 ตารางแสดงยูสเคสที่ 11 (Summary)

Use Case no.	11
Use Case Name	Summary
Actors	ผู้ใช้งาน
Pre-Condition	Rectify
Post-Condition	-
Brief Description	ส่วนของการสรุปผลที่ได้จากกระบวนการ

ตารางที่ 3.12 ตารางแสดงยูสเคสที่ 12 (Edit Pattern)

Use Case no.	12
Use Case Name	Edit Pattern
Actors	ผู้ใช้งาน
Pre-Condition	ทำการ Login อยู่ในระบบ
Post-Condition	แก้ไขข้อมูลรูปแบบการคัดแยกที่บันทึกไว้ได้
Brief Description	ส่วนของการแก้ไขรูปแบบการคัดแยกที่บันทึกไว้

ตารางที่ 3.13 ตารางแสดงยูสเคสที่ 13 (Counting Process)

Use Case no.	13
Use Case Name	Counting Process
Actors	ผู้ใช้งาน
Pre-Condition	Initial View
Post-Condition	แสดงผลที่ได้จากการนับจำนวนเพื่อให้ผู้ใช้งานตรวจสอบความถูกต้อง
Brief Description	ส่วนของการนับจำนวนของวัตถุ

ตารางที่ 3.14 ตารางแสดงยูสเคสที่ 14 (Select Hardware)

Use Case no.	14
Use Case Name	Select Hardware
Actors	ผู้ใช้งาน
Pre-Condition	เลือกทำ Learning
Post-Condition	Initial View
Brief Description	ส่วนของการเลือก Hardware ที่ต้องการใช้งานสำหรับการ Learning

ตารางที่ 3.15 ตารางแสดงยูสเคสที่ 15 (Learning Process)

Use Case no.	15
Use Case Name	Learning Process
Actors	ผู้ใช้งาน
Pre-Condition	Initial View
Post-Condition	Suggest Type, Add New Type
Brief Description	ส่วนของการเรียนรู้วัตถุจากวัตถุที่ป้อนเข้ากระบวนการ

ตารางที่ 3.16 ตารางแสดงยูสเคสที่ 16 (Suggest Type)

Use Case no.	16
Use Case Name	Suggest Type
Actors	ผู้ใช้งาน
Pre-Condition	Learning
Post-Condition	-
Brief Description	ส่วนของการแนะนำวัตถุที่มีลักษณะใกล้เคียงกับวัตถุที่ได้เรียนรู้จากข้อมูลที่มีในระบบ

ตารางที่ 3.17 ตารางแสดงยูสเคสที่ 17 (Add new type)

Use Case no.	17
Use Case Name	Add New Type
Actors	ผู้ใช้งาน
Pre-Condition	Learning
Post-Condition	-
Brief Description	ส่วนของการป้อนข้อมูลใหม่สำหรับวัตถุที่ได้จากการเรียนรู้ในกรณีที่ไม่มีวัตถุที่ใกล้เคียงอยู่ในระบบ

ตารางที่ 3.18 ตารางแสดงยูสเคสที่ 18 (View Dataset)

Use Case no.	18
Use Case Name	View Dataset
Actors	ผู้ใช้งาน
Pre-Condition	ทำการ Login อยู่ในระบบ
Post-Condition	แสดงข้อมูลวัตถุทั้งหมดในระบบ
Brief Description	ส่วนของการแสดงข้อมูลวัตถุทั้งหมดในระบบได้

ตารางที่ 3.19 ตารางแสดงยูสเคสที่ 19 (Search By Filtering)

Use Case no.	19
Use Case Name	Search By Filtering
Actors	ผู้ใช้งาน
Pre-Condition	View Dataset
Post-Condition	แสดงข้อมูลของวัตถุตามรายละเอียดที่ต้องการ
Brief Description	ส่วนของการค้นหาข้อมูลวัตถุตามรายละเอียดที่ต้องการ

ตารางที่ 3.20 ตารางแสดงยูสเคสที่ 20 (View Dataset Detail)

Use Case no.	20
Use Case Name	View Dataset Detail
Actors	ผู้ใช้งาน
Pre-Condition	Select Dataset
Post-Condition	แสดงรายละเอียดของ Dataset ที่เลือก
Brief Description	ส่วนของการแสดงรายละเอียดข้อมูลของวัตถุที่เลือก

ตารางที่ 3.21 ตารางแสดงยูสเคสที่ 21 (Delete Dataset)

Use Case no.	21
Use Case Name	Delete Dataset
Actors	ผู้ใช้งาน
Pre-Condition	Select Dataset
Post-Condition	-
Brief Description	ส่วนของการลบข้อมูลหรือรูปของวัตถุที่เลือกจากระบบ

ตารางที่ 3.22 ตารางแสดงยูสเคสที่ 22 (View Hardware)

Use Case no.	22
Use Case Name	View Hardware
Actors	ผู้ใช้งาน
Pre-Condition	ทำการ Login อยู่ในระบบ
Post-Condition	แสดงข้อมูล Hardware ทั้งหมดของผู้ใช้
Brief Description	ส่วนของการแสดงข้อมูล Hardware ทั้งหมดที่ผู้ใช้งานสามารถใช้งานได้

ตารางที่ 3.23 ตารางแสดงยูสเคสที่ 23 (View Hardware Detail)

Use Case no.	23
Use Case Name	View Hardware Detail
Actors	ผู้ใช้งาน
Pre-Condition	Select Hardware
Post-Condition	แสดงข้อมูล Hardware ที่ผู้ใช้เลือก
Brief Description	ส่วนของการแสดงรายละเอียดข้อมูลของ Hardware ที่ผู้ใช้เลือก

ตารางที่ 3.24 ตารางแสดงยูสเคสที่ 24 (Edit Hardware Detail)

Use Case no.	24
Use Case Name	Edit Hardware Detail
Actors	ผู้ใช้งาน
Pre-Condition	Select Hardware
Post-Condition	แสดงข้อมูล Hardware ที่ผู้ใช้ทำการแก้ไข
Brief Description	ส่วนของการแก้ไขรายละเอียดข้อมูลของ Hardware ที่ผู้ใช้เลือก

ตารางที่ 3.25 ตารางแสดงยูสเคสที่ 25 (Delete Hardware)

Use Case no.	25
Use Case Name	Delete Hardware
Actors	ผู้ใช้งาน
Pre-Condition	Select Hardware
Post-Condition	-
Brief Description	ส่วนของการลบ Hardware ที่ผู้ใช้เลือก

ตารางที่ 3.26 ตารางแสดงยูสเคสที่ 26 (Add Hardware)

Use Case no.	26
Use Case Name	Add Hardware
Actors	ผู้ใช้งาน
Pre-Condition	View Hardware
Post-Condition	แสดงรายละเอียดข้อมูลของ Hardware ที่ผู้ใช้เพิ่มเข้าระบบ
Brief Description	ส่วนของการเพิ่ม Hardware ใหม่

ตารางที่ 3.27 ตารางแสดงยูสเคสที่ 27 (Hardware Permission)

Use Case no.	27
Use Case Name	Hardware Permission
Actors	ผู้ดูแลระบบ
Pre-Condition	ทำการ Login อยู่ในระบบ
Post-Condition	จัดการ Permission ต่าง ๆ สำเร็จ
Brief Description	ส่วนของการอนุญาต Permission ให้กับผู้ใช้งานและจัดการกับ Hardware ในระบบ

ตารางที่ 3.28 ตารางแสดงยูสเคสที่ 28 (View Log)

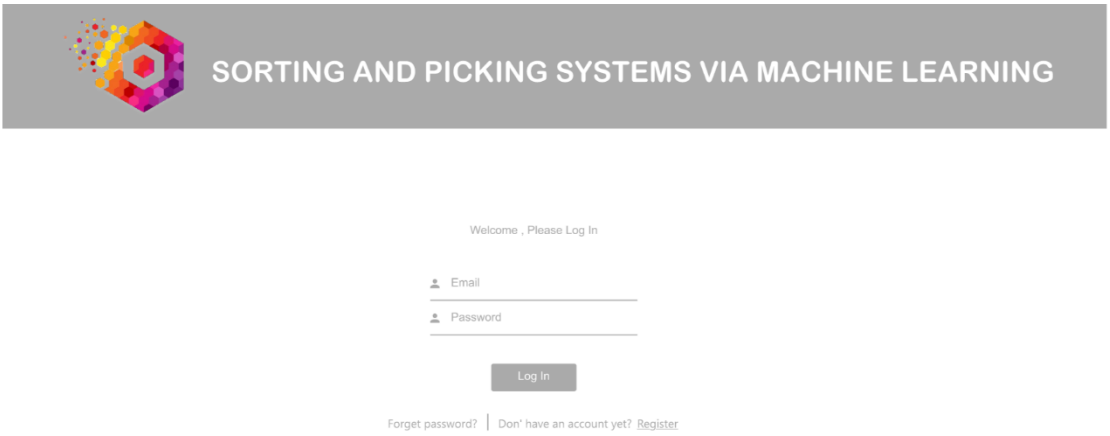
Use Case no.	28
Use Case Name	View Log
Actors	ผู้ใช้งานและผู้ดูแลระบบ
Pre-Condition	ทำการ Login อยู่ในระบบ
Post-Condition	-
Brief Description	ส่วนของการแสดงกิจกรรมต่าง ๆ ที่ผู้ใช้งานทำ

3.3 ส่วนต่อประสานกับผู้ใช้ (User Interface)

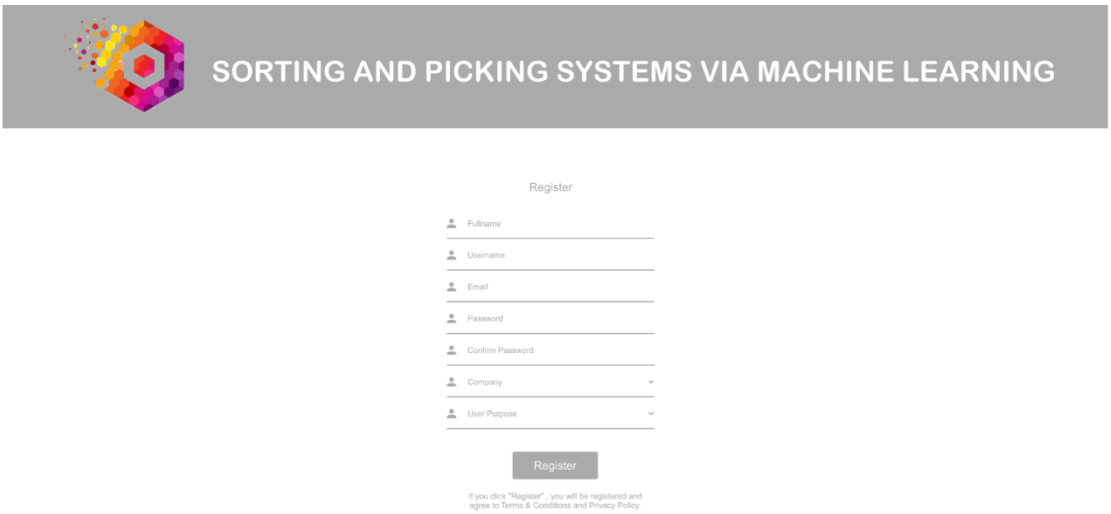
เว็บแอปพลิเคชันของระบบ ประะกอบไปด้วย 2 ส่วน คือ Web Application ส่วนของผู้ใช้งาน (User) และ Web Application ส่วนของผู้ดูแลระบบ (Admin)

3.3.1 **Web Application ส่วนของผู้ใช้งาน (User)**

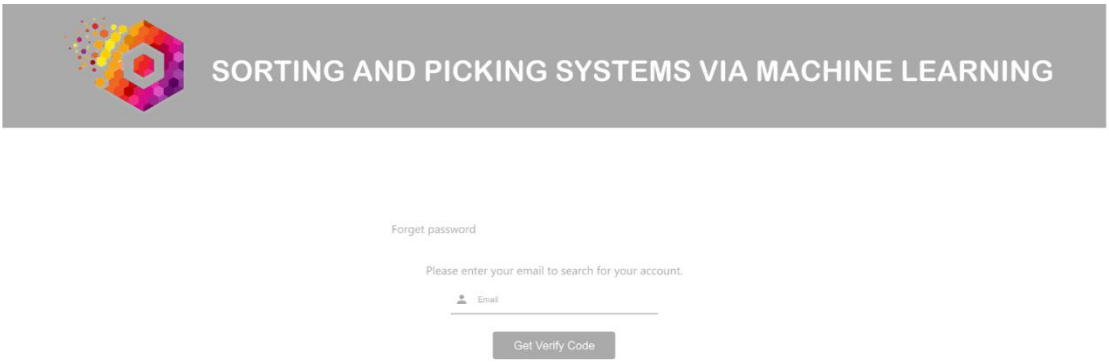
การยืนยันตัวตนของผู้ใช้งาน (Authentication)



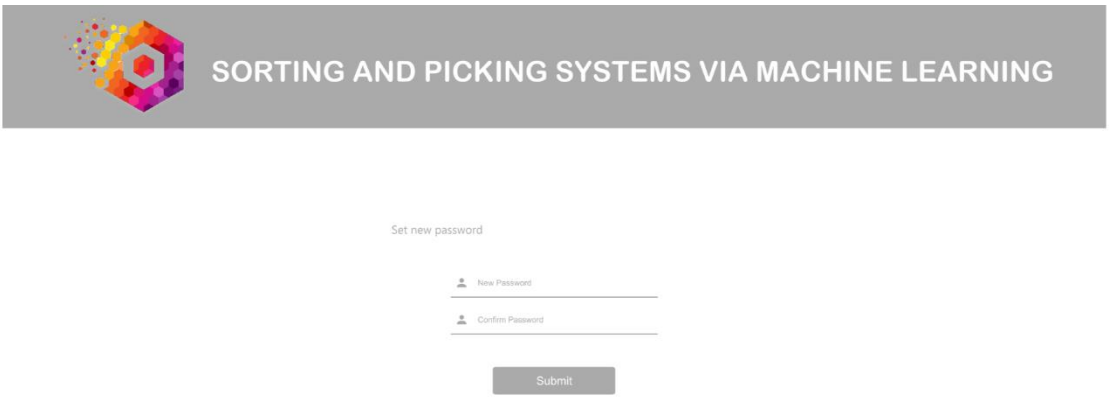
รูปที่ 3.2 หน้าแสดงผลการยืนยันตัวตนเข้าสู่ระบบ



รูปที่ 3.3 หน้าแสดงผลของการสมัครสมาชิก

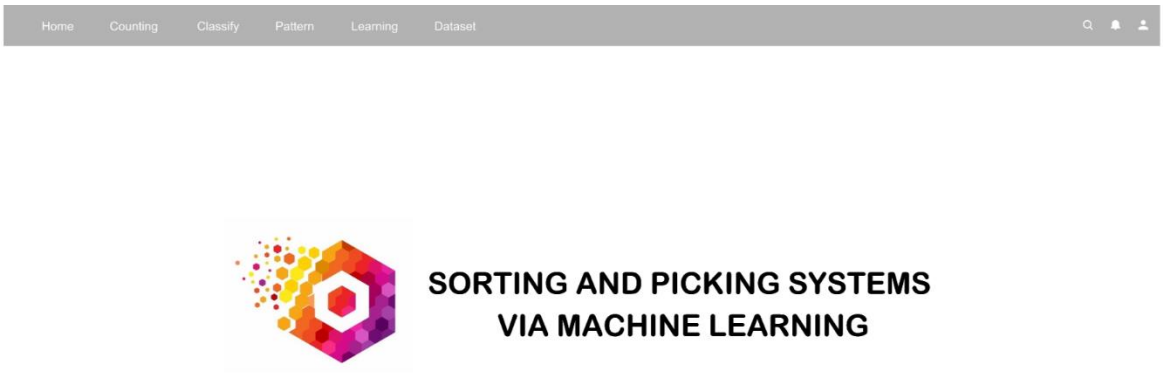


รูปที่ 3.4 หน้าแสดงผลกรณีลืมรหัสผ่าน



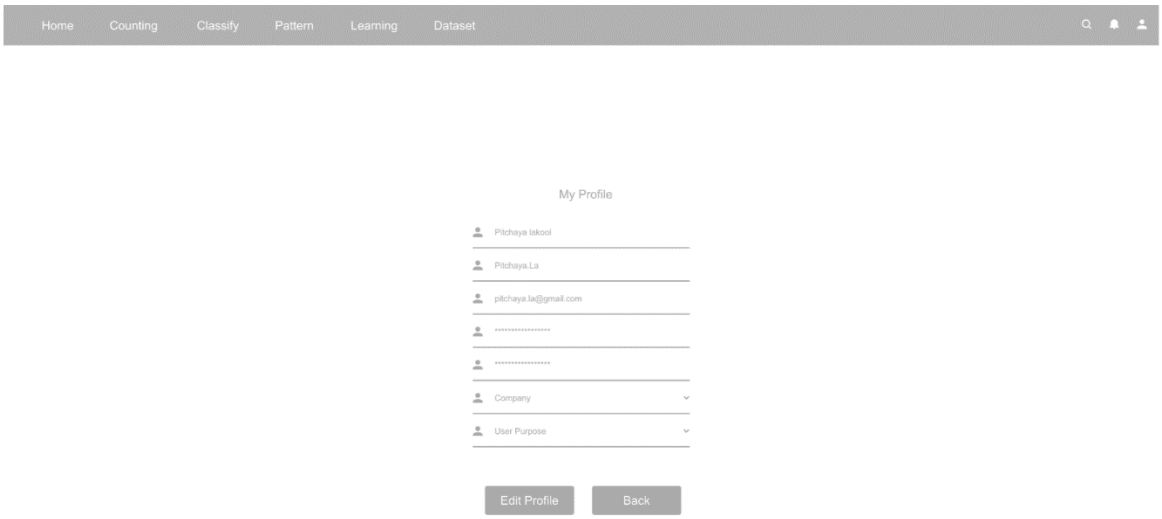
รูปที่ 3.5 หน้าแสดงผลการตั้งค่ารหัสผ่านใหม่กรณีลืมรหัสผ่าน

ส่วนของการยืนยันตัวตนเข้าสู่ระบบ ในกรณีที่ผู้ใช้ได้ลงทะเบียนกับระบบแล้ว ผู้ใช้งานต้องกรอก Email และ Password ที่ลงทะเบียนไว้กับระบบ จากนั้นจึงกดปุ่ม Log In เพื่อทำการเข้าสู่ระบบ และถ้าหากกรอกข้อมูลถูกต้อง ระบบจะแสดงผลเข้าสู่หน้าหลักของระบบ หากไม่ถูกต้องระบบจะแจ้งว่ามีการกรอกข้อมูลที่ไม่ถูกต้อง ในกรณีที่ผู้ใช้งานยังไม่ได้ลงทะเบียนไว้กับระบบ ผู้ใช้งานจะต้องทำการลงทะเบียนกับระบบก่อน จากนั้นจึงทำการยืนยันตัวตนเข้าสู่ระบบได้

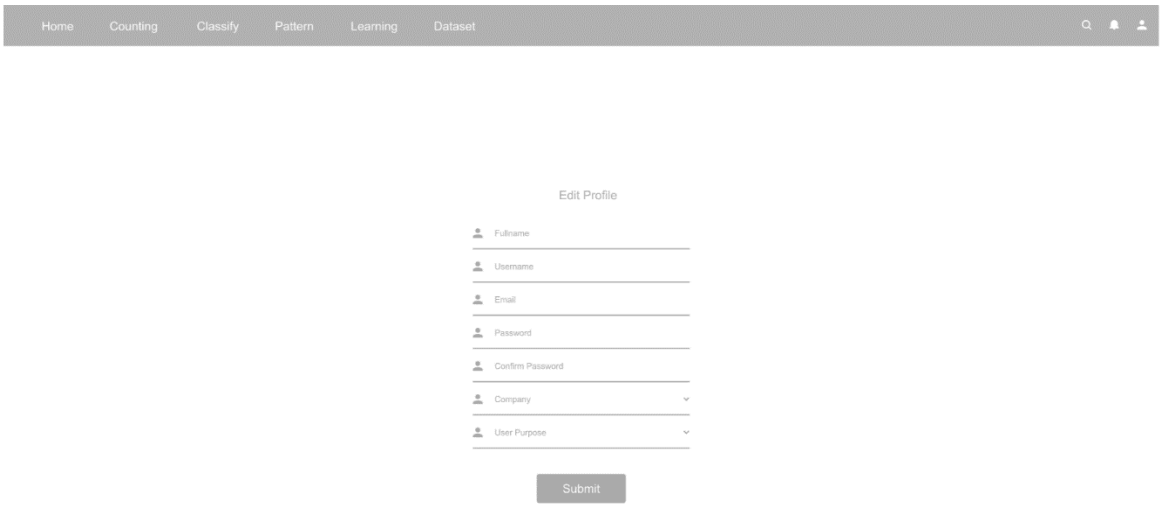


รูปที่ 3.6 หน้าแสดงผลหลัก

หน้าหลักของระบบ (Home) จะแสดงหน้าหลักของระบบ ซึ่งเป็นหน้าแรกที่เจอหลังจากทำการเข้าสู่ระบบ โดยมีเมนูต่างๆ(เรียงลำดับจากซ้ายไปขวา) ประกอบด้วย Home แสดงหน้าหลักของระบบ , Counting ระบบจะเข้าสู่ Counting Process , Classify ระบบจะเข้าสู่ Classify Process , Pattern เป็นเมนูเพื่อแก้ไข Pattern ที่ถูกบันทึกไว้ในระบบ , Learning ระบบจะเข้าสู่ Learning Process , Dataset จะแสดงรายการข้อมูล Dataset ที่อยู่ในระบบ , Search ใช้ค้นหาข้อมูลต่างๆด้วยคีย์เวิร์ดหรือคำค้นหาที่สำคัญ , Notification ใช้แสดงการแจ้งเตือนต่างๆจากระบบ, ManageProfile ช่วยในการจัดการข้อมูลส่วนตัวของผู้ใช้งานและใช้เพื่อออกจากการใช้งานระบบด้วยเมนู Log Out

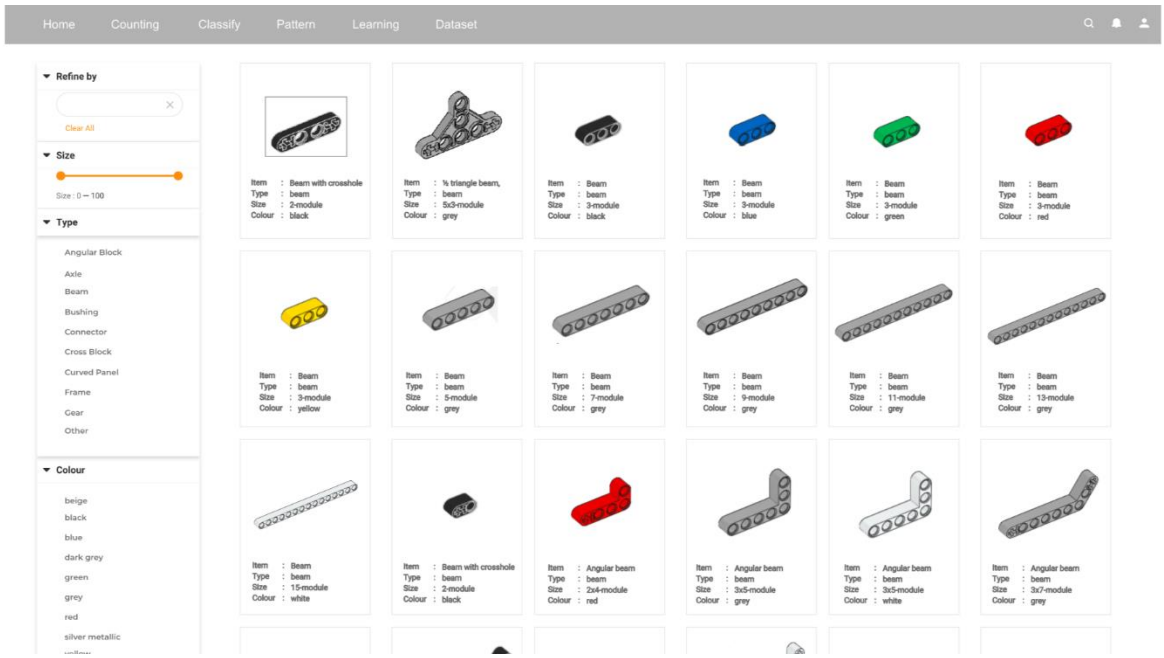


รูปที่ 3.7 หน้าแสดงผลข้อมูลส่วนตัว

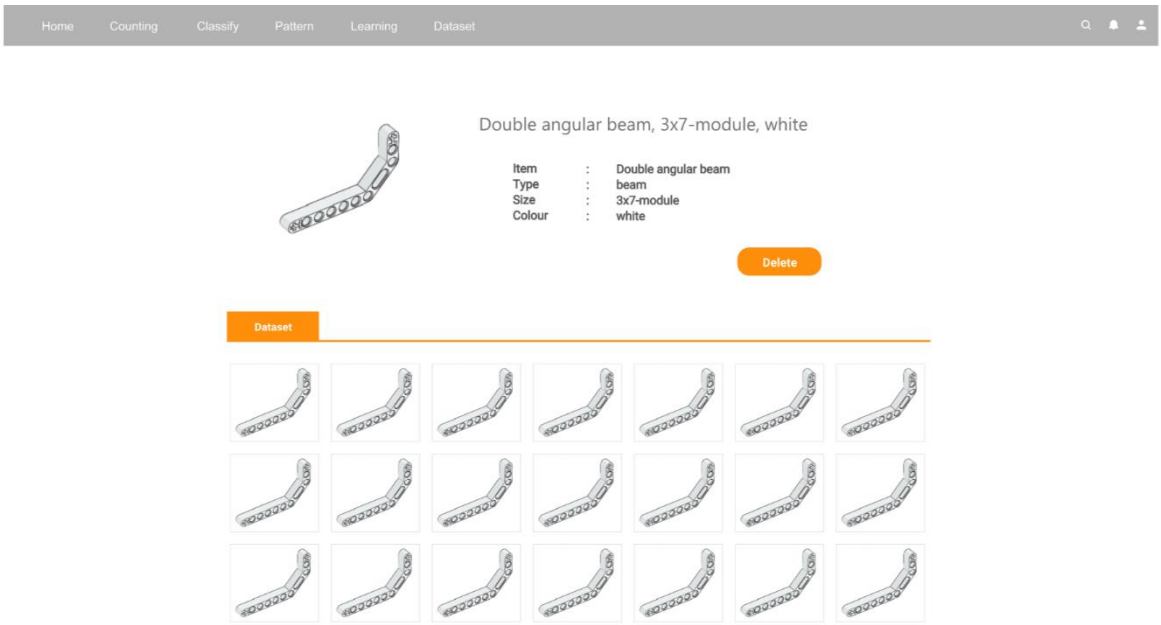


รูปที่ 3.8 หน้าแสดงผลการเปลี่ยนแปลงข้อมูลส่วนตัว

เป็นส่วนของการจัดการข้อมูลส่วนตัวของผู้ใช้งาน (Manage Profile) สามารถแสดงข้อมูลส่วนตัวของผู้ใช้งาน (ViewProfile) ผู้ใช้งานสามารถแก้ไขข้อมูลส่วนตัวได้ โดยการกดที่ปุ่ม Edit Profile

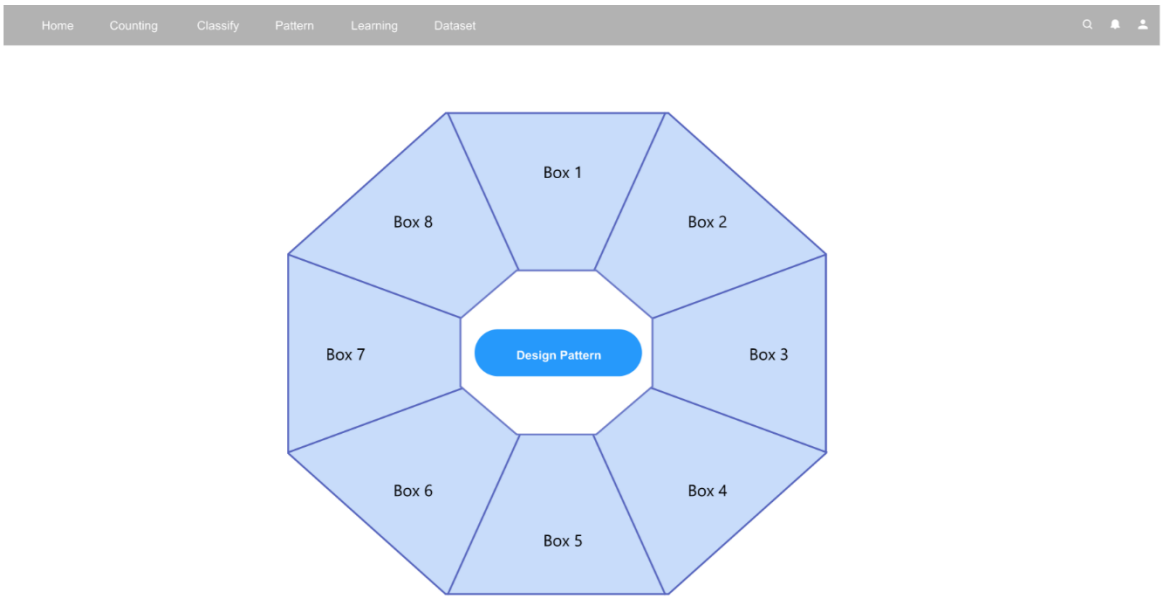


รูปที่ 3.9 หน้าแสดงผลชุดข้อมูล

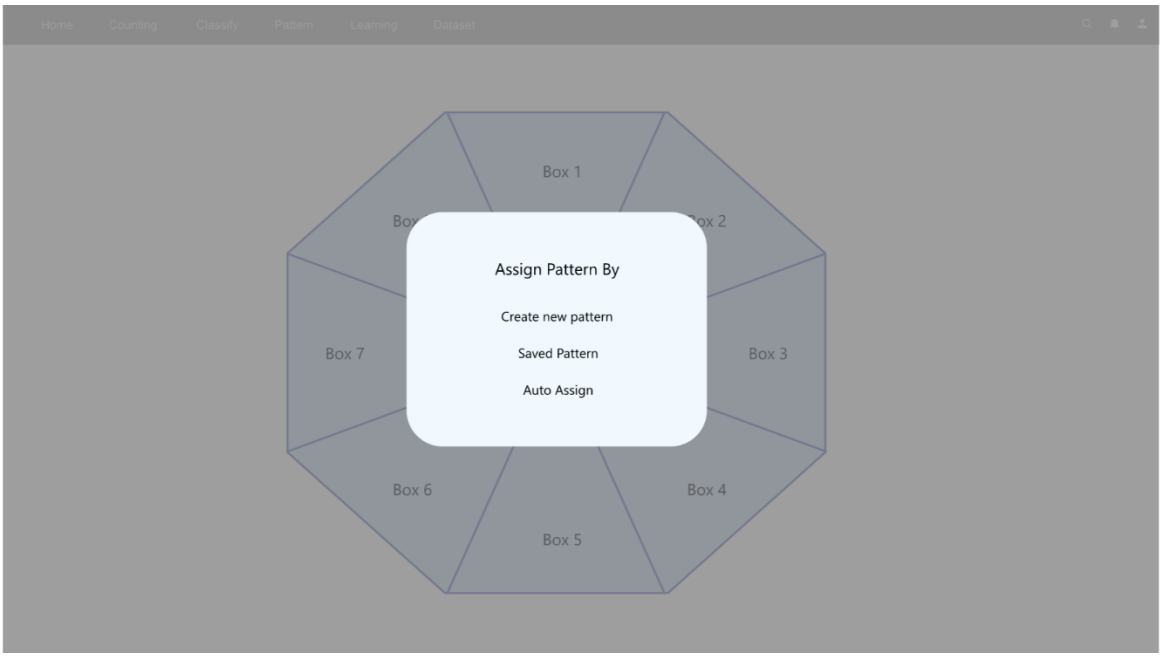


รูปที่ 3.10 หน้าแสดงรายละเอียดข้อมูลแยกชิ้น

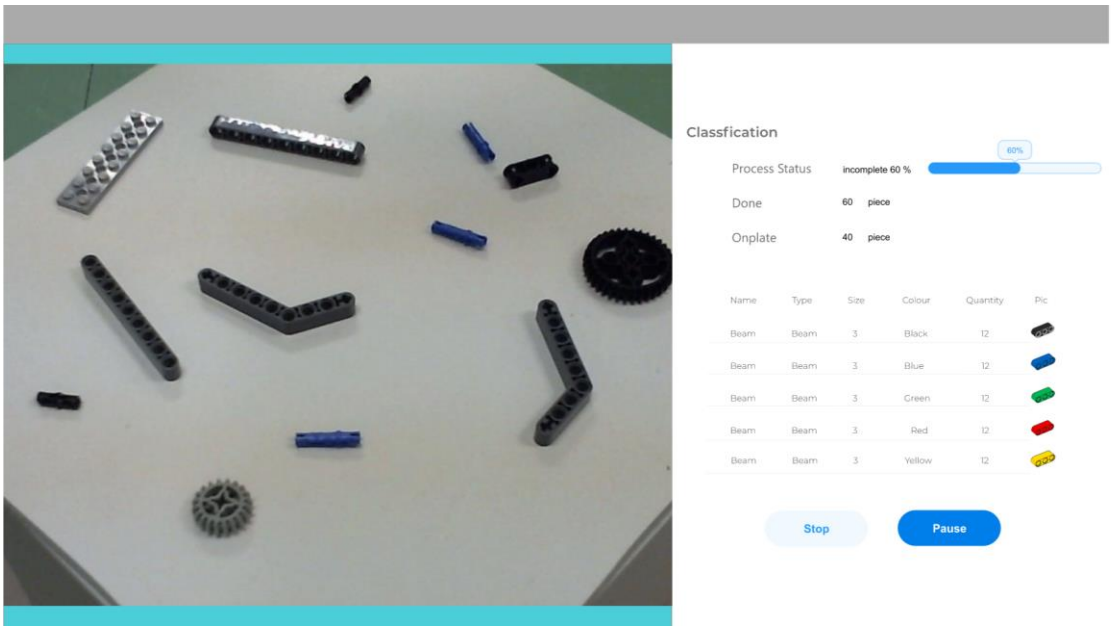
เป็นส่วนแสดงชุดข้อมูล (Dataset) หากไม่มีการเลือกประเภทใดๆ จะแสดงชุดข้อมูลทั้งหมด ถ้าหากมีการเลือกให้แสดงตามประเภทต่างๆ จะแสดงข้อมูลเฉพาะข้อมูลที่มีคุณสมบัติตามที่ผู้ใช้งานเลือก เมื่อผู้ใช้งานเลือกกดที่ข้อมูลใดข้อมูลหนึ่ง จะแสดงหน้ารายละเอียดของข้อมูลนั้น ผู้ใช้จะสามารถดูรายละเอียดของข้อมูล ลบข้อมูลนั้นและลบรูปของข้อมูลนั้นได้จากหน้านี้



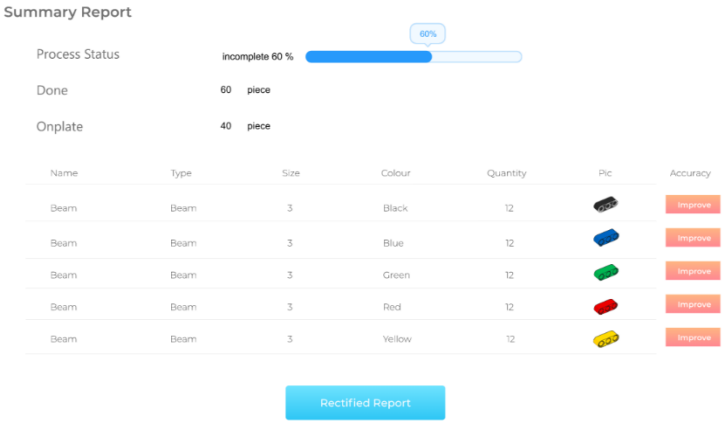
รูปที่ 3.11 หน้าแสดงผลหลักกระบวนการคัดแยก



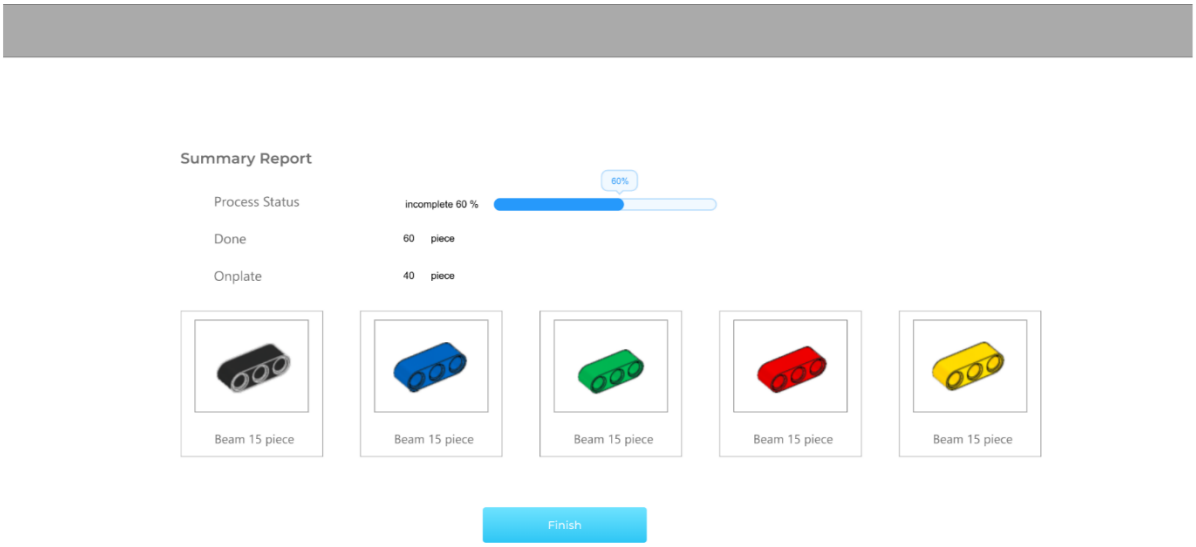
รูปที่ 3.12 หน้าแสดงผลเลือกรูปแบบการคัดแยก



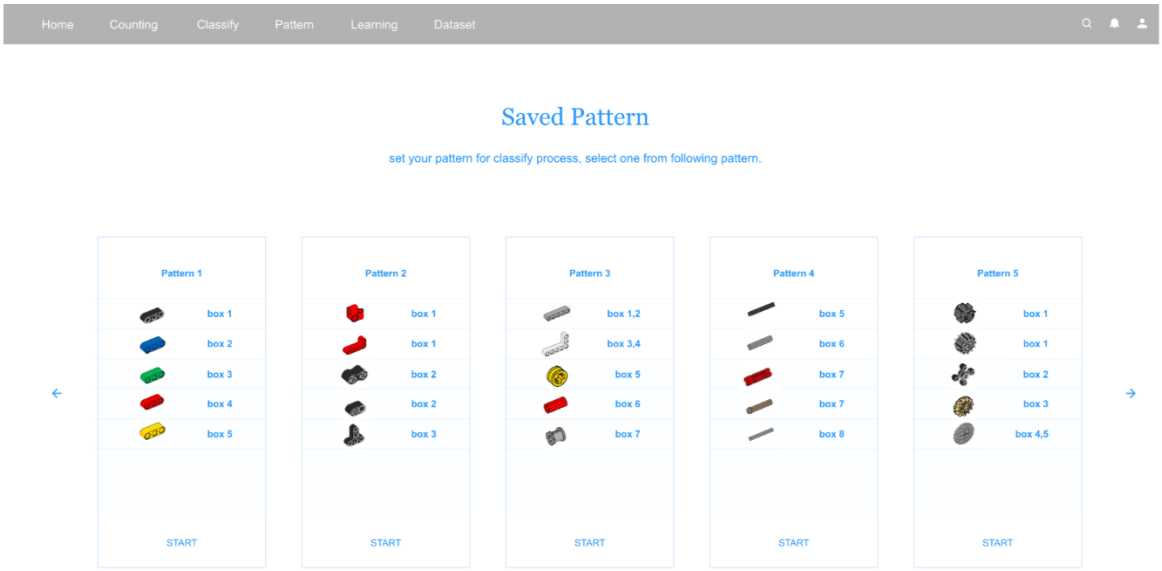
รูปที่ 3.13 หน้าแสดงผลกระบวนการคัดแยก



รูปที่ 3.14 หน้าแสดงผลยืนยันความถูกต้องของข้อมูล

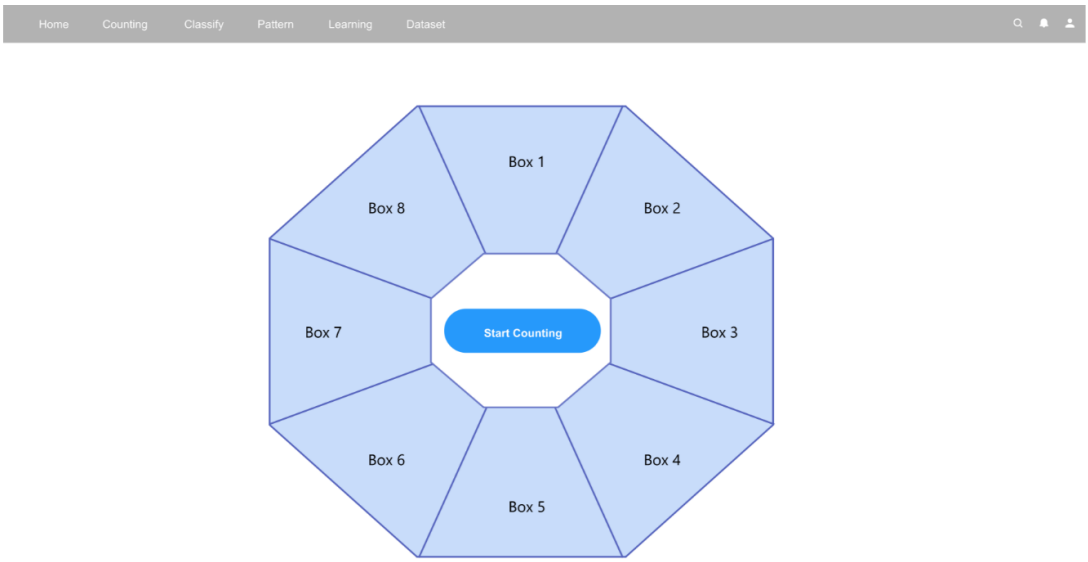


รูปที่ 3.15 หน้าแสดงผลสรุปผลข้อมูลที่ได้จากการคัดแยก

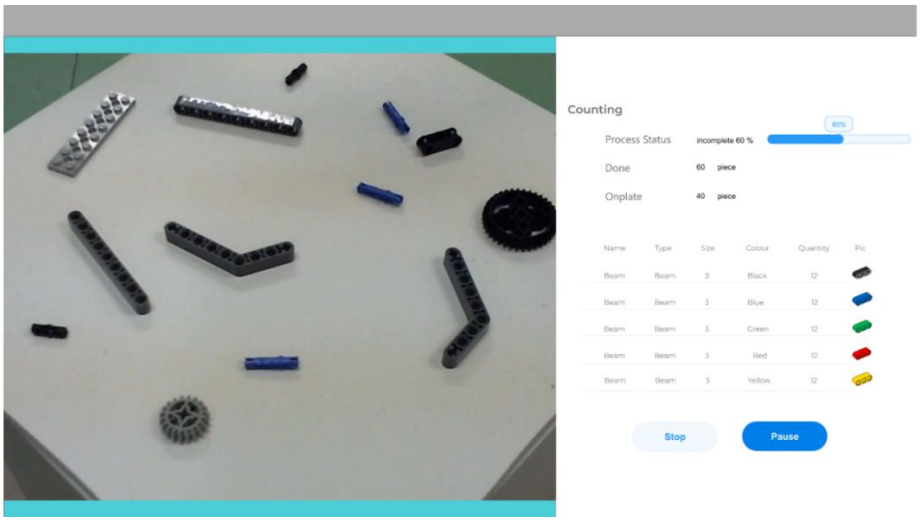


รูปที่ 3.16 หน้าแสดงผลรูปแบบการคัดแยกที่ได้บักทึกไว้

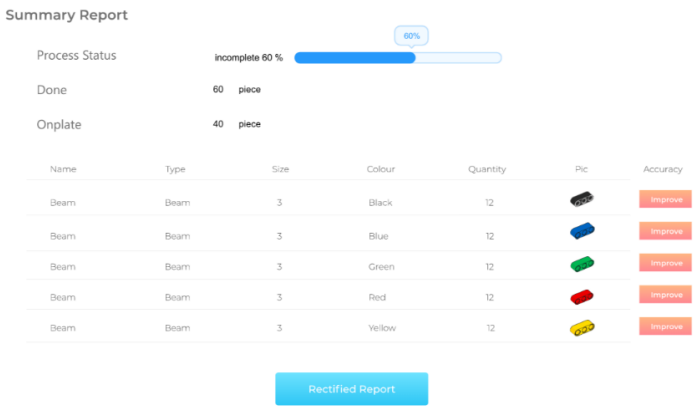
เมนู Classify เป็นเมนูที่ใช้ในการคัดแยกสิ่งของ โดยผู้ใช้ต้องเลือกรูปแบบในการจำแนกจากรูปแบบ 3 รูปแบบ ได้แก่ รูปแบบที่สร้างขึ้นมาใหม่ (Create new pattern) , รูปแบบที่ถูกบันทึกไว้ (Saved pattern), รูปแบบโดยระบบ (Auto assign) เมื่อทำการตั้งค่ารูปแบบและเริ่มกระบวนการแล้วระบบจะเริ่มขั้นตอนการคัดแยกจนเสร็จสิ้นหรือจนกว่าจะมีการกดปุ่ม Stop โดยผู้ใช้งาน ระบบจะแสดงข้อมูลที่ได้จากการคัดแยกเพื่อให้ผู้ใช้ประเมินผลความถูกต้องก่อนจึงออกรายงานการคัดแยกสิ่งของ (Summary Report)



รูปที่ 3.17 หน้าแสดงผลหลักกระบวกการนับ



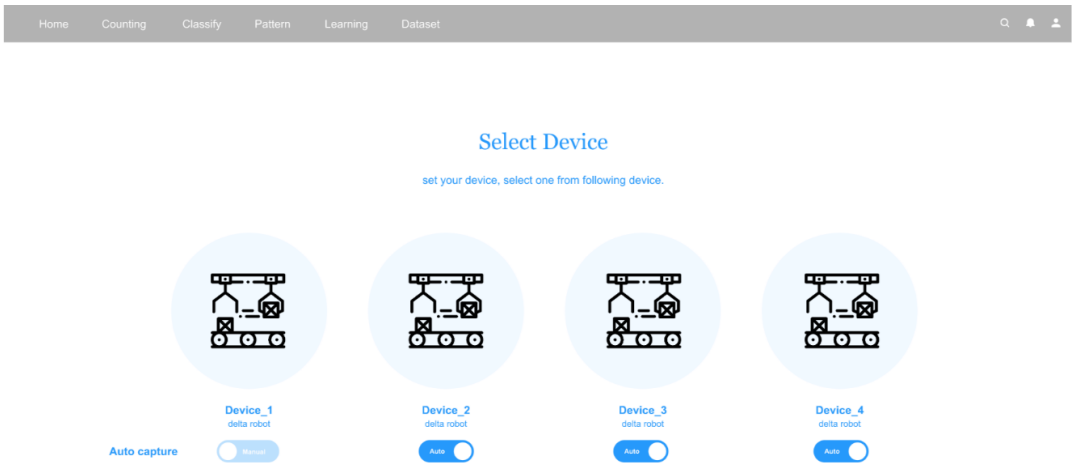
รูปที่ 3.18 หน้าแสดงผลกระบวนการนับ



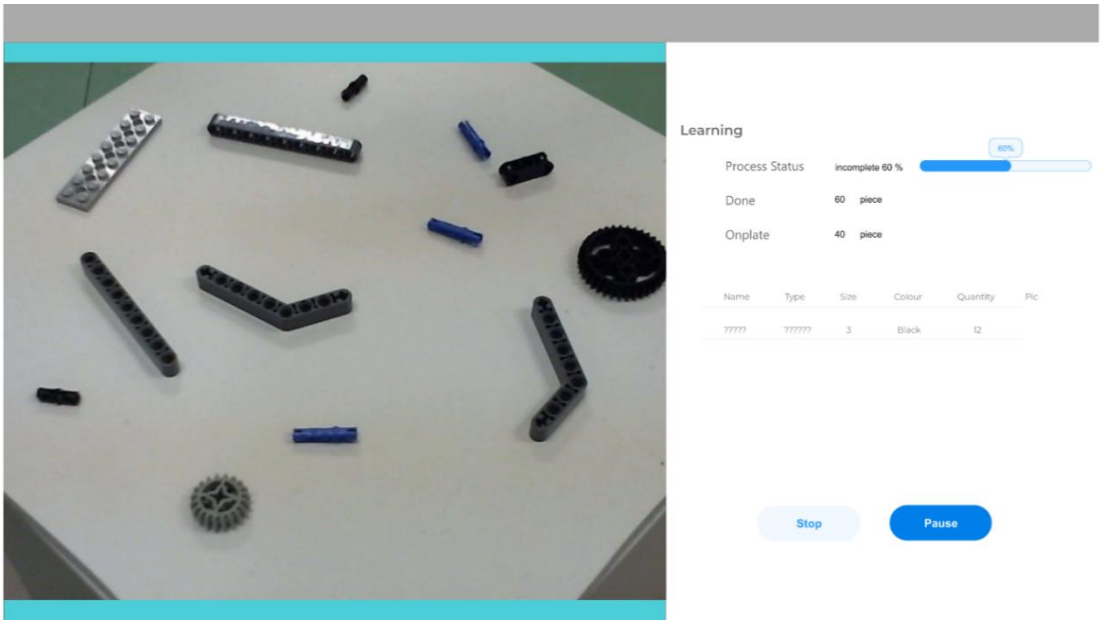
รูปที่ 3.19 หน้าแสดงผลยืนยันความถูกต้องของข้อมูล

รูปที่ 3.20 หน้าแสดงผลสรุปผลข้อมูลที่ได้จากการนับ

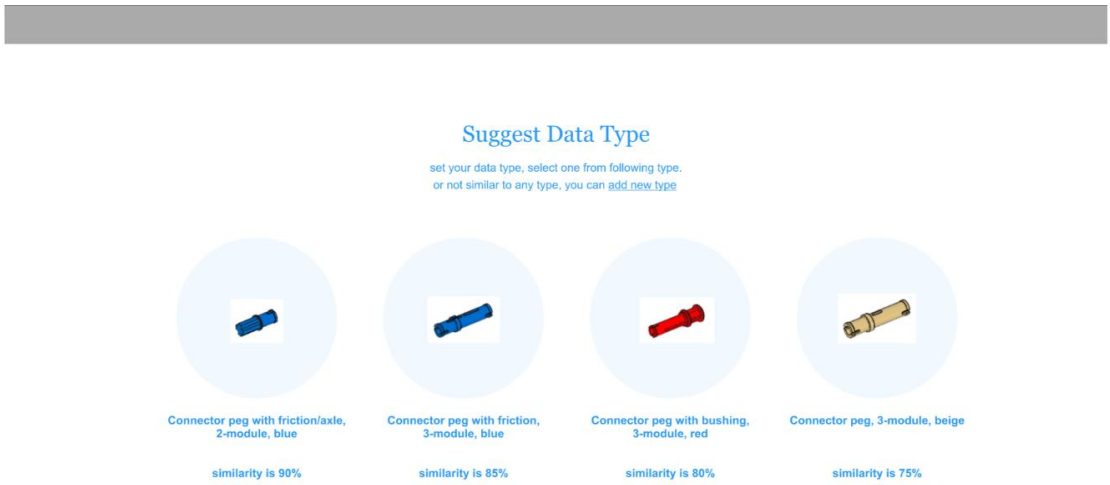
Counting Process เป็นเมนูเพื่อใช้นับจำนวนสิ่งของโดยไม่ต้องตั้งค่าเหมือนกับเมนู Classification ระบบจะทำการตั้งค่าให้เอง เมื่อทำการนับจำนวนเสร็จจะแสดงหน้าตรวจสอบความถูกต้องให้ผู้ใช้ประเมินผลเพื่อใช้ประเมินประสิทธิภาพการนับว่ามีความแม่นยำเพียงใด ต่อจากนั้นจะแสดงรายงานการนับสิ่งของ (Summary Report)



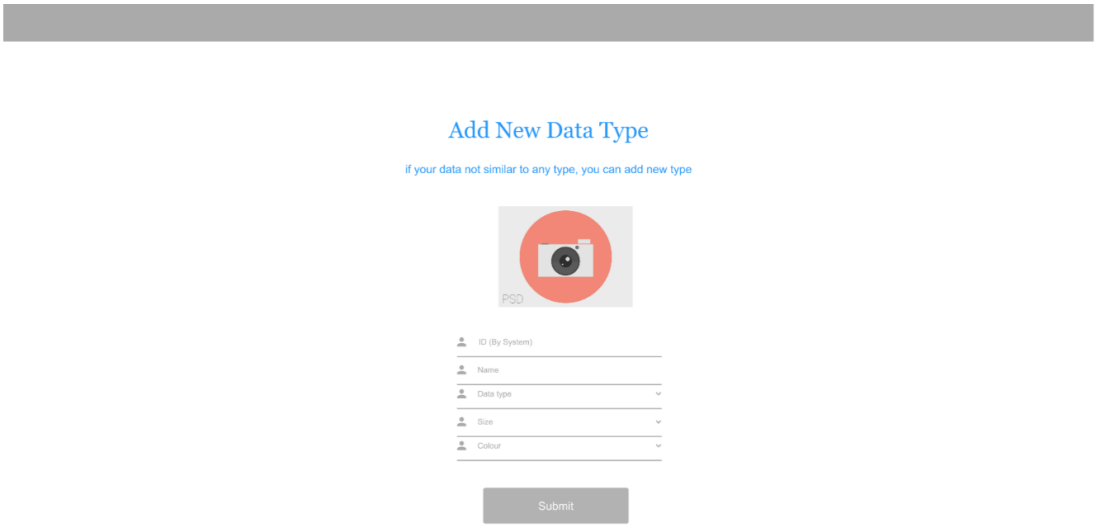
รูปที่ 3.21 หน้าแสดงผลเลือกอุปกรณ์ในการเรียนรู้



รูปที่ 3.22 หน้าแสดงผลกระบวนการเรียนรู้



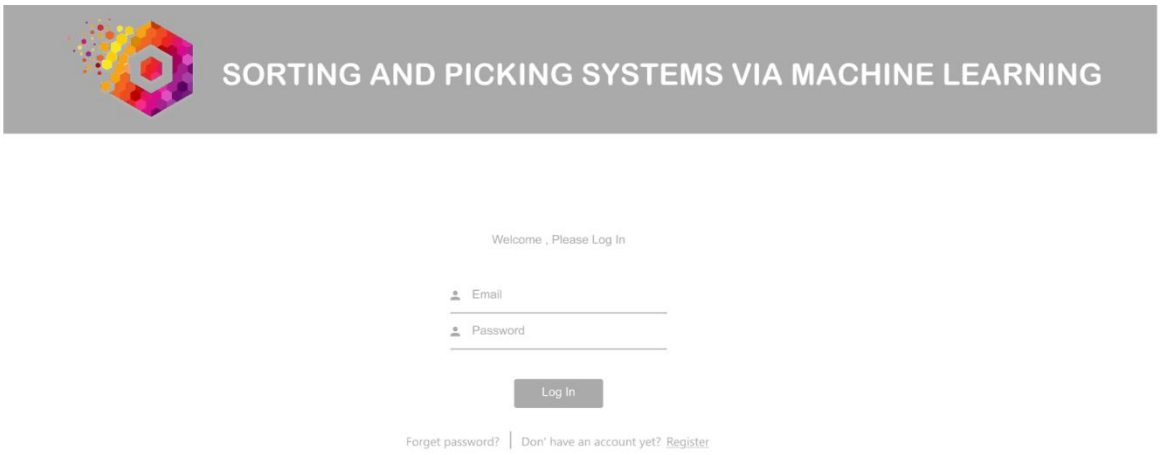
รูปที่ 3.23 หน้าแสดงผลการแนะนำประเภทข้อมูล



รูปที่ 3.24 หน้าแสดงผลเพิ่มข้อมูลประเภทใหม่

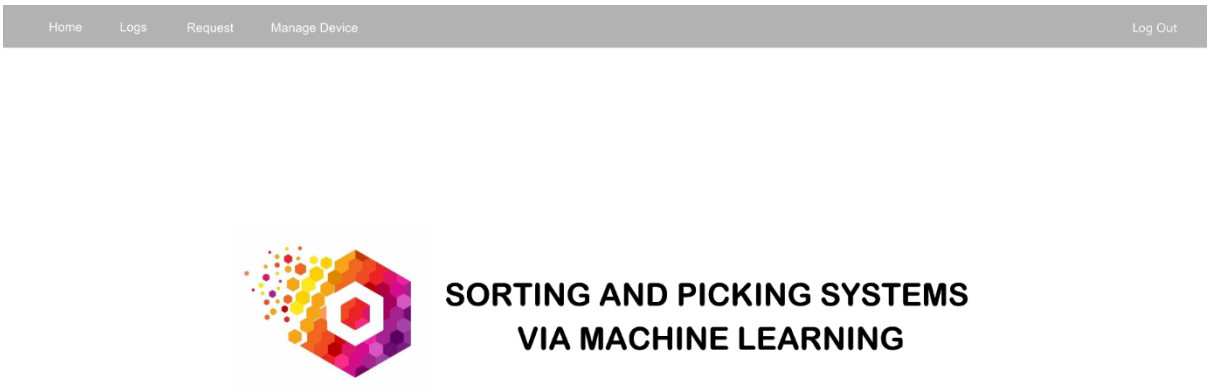
เมนู Learning เป็นเมนูที่ใช้ในการเรียนรู้ของเครื่องเพื่อนำไปใช้ในส่วนของเมนู Classify และ Counting ต่อไป โดยผู้ใช้ต้องเลือกอุปกรณ์ก่อน จากนั้นระบบจะทำการ Learning เมื่อเสร็จสิ้น ระบบจะแนะนำ Data Type ที่มีลักษณะคล้ายคลึงกับสิ่งของที่ได้ทำการ Learning มา หากไม่คล้ายกับ Data Type ชนิดใดเลย ผู้ใช้งานสามารถเพิ่ม Data Type ของข้อมูลใหม่ได้เอง

3.3.2 **Web Application ส่วนของผู้ดูแลระบบ (Admin)**



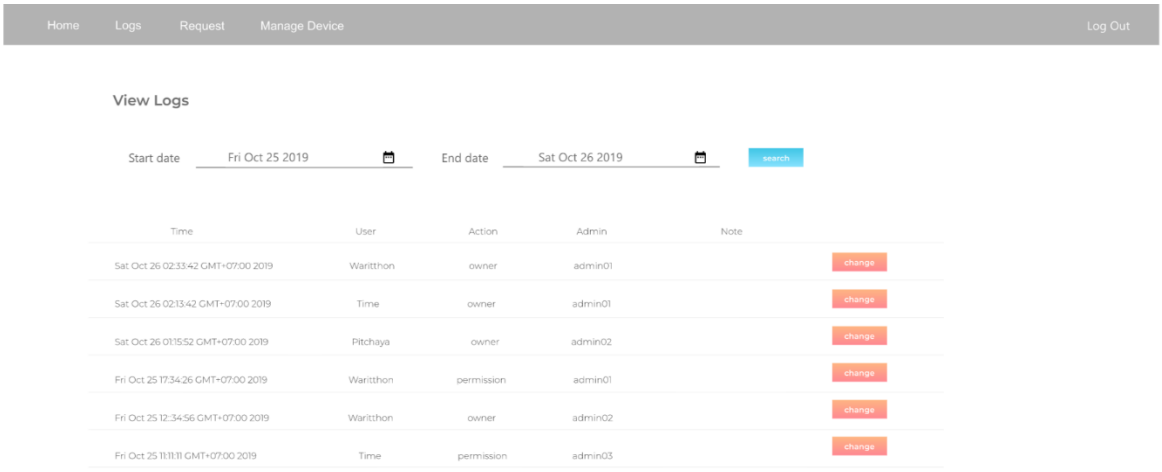
รูปที่ 3.25 หน้าแสดงผลการยืนยันตัวตนเข้าสู่ระบบ

ส่วนของการยืนยันตัวตนเข้าสู่ระบบ ระบบต้องกรอก Email และ Password ก่อน จากนั้นจึงกดปุ่ม Log In เพื่อทำการยืนยันตัวตนเข้าสู่ระบบ



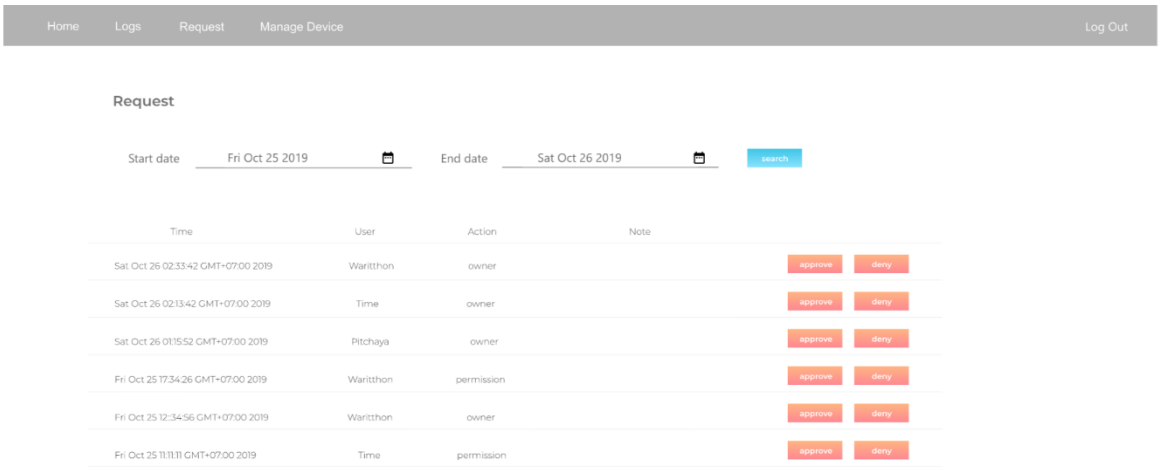
รูปที่ 3.26 หน้าแสดงผลหลัก

หน้าหลักของระบบ (Home) จะแสดงหน้าหลักของระบบ ซึ่งเป็นหน้าแรกที่เจอหลังจากทำการเข้าสู่ระบบ โดยมีเมนูต่างๆ(เรียงลำดับจากซ้ายไปขวา) ประกอบด้วย Home แสดงหน้าหลักของระบบ , Logs จะแสดงรายการ Logs ที่ผู้ดูแลระบบได้ทำอนุมัติหรือเปลี่ยนแปลงการจัดการกับอุปกรณ์ต่างๆ ในระบบ , Request แสดงรายการรออนุมัติคำขอเข้าถึงอุปกรณ์จากผู้ใช้งาน , Manage Profile เป็นเมนูที่ช่วยในการจัดการกับอุปกรณ์ในระบบ, Log Out ใช้เพื่อออกจากการใช้งานระบบด้วยเมนู



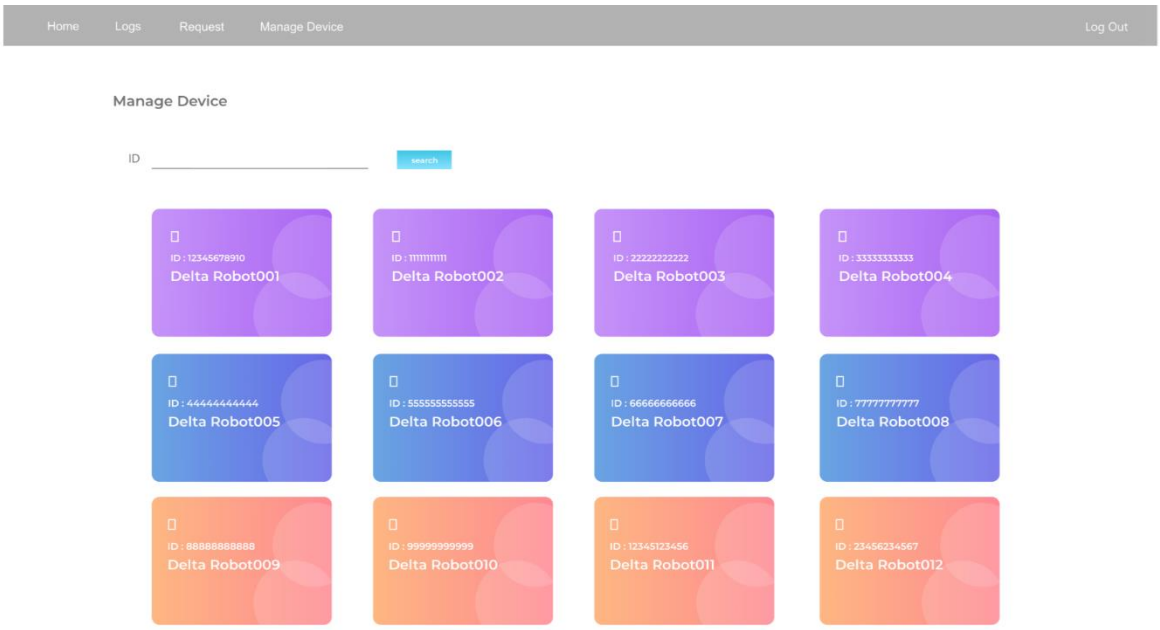
รูปที่ 3.27 หน้าแสดงรายละเอียดกิจกรรม

เป็นเมนูที่จะแสดงรายละเอียดกิจกรรมต่างๆที่เกิดขึ้นกับอุปกรณ์ในระบบ เช่น การอนุมัติคำร้องขอเข้าใช้อุปกรณ์ (Permission) , การยื่นคำขอเป็นเจ้าของอุปกรณ์ (Owner) เป็นต้น ในกรณีที่ผู้ดูแลระบบต้องการเปลี่ยนแปลงแก้ไขสถานะของกิจกรรมที่ได้ทำไปแล้ว สามารถทำได้โดยกดที่ปุ่ม Change



รูปที่ 3.28 หน้าแสดงผลคำร้องขอเข้าถึงอุปกรณ์

เมื่อผู้ใช้งานต้องการเข้าถึงอุปกรณ์ในระบบจะต้องส่งคำร้องขอมายังผู้ดูแลระบบก่อน หากผู้ใช้งานส่งคำร้องขอเรียบร้อยแล้ว ผู้ดูแลระบบจะทำการจัดการคำร้องขอจากผู้ใช้งานด้วยเมนู Request หากผู้ใช้งานที่ร้องขอเข้าถึงอุปกรณ์มีสิทธิ์ที่จะใช้งานอุปกรณ์ตามสถานะที่ได้ร้องขอ ผู้ดูแลระบบจะกดที่ปุ่ม Approve เพื่ออนุมัติคำร้องขอดังกล่าว แต่หากผู้ใช้งานไม่มีสิทธิ์ที่จะเข้าถึงอุปกรณ์ ผู้ดูแลระบบจะทำการปฏิเสธคำร้องขอด้วยการกดที่ปุ่ม Deny



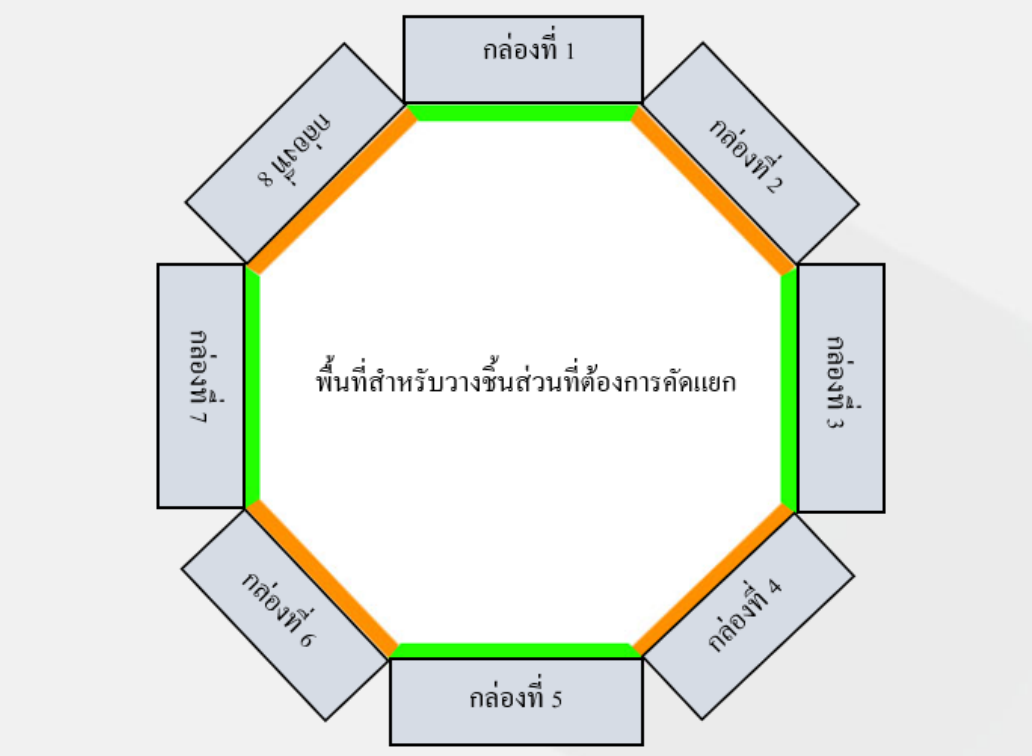
รูปที่ 3.29 หน้าแสดงผลการจัดการอุปกรณ์

ผู้ดูแลระบบจะจัดการอุปกรณ์ต่างๆในระบบได้จากเมนูนี้ โดยเมนูจะให้เลือกค้นหาอุปกรณ์ด้วย ID ของอุปกรณ์ แต่ถ้าหากไม่มีการกรอก ID ของอุปกรณ์ ระบบจะแสดงข้อมูลอุปกรณ์ทั้งหมดที่มีอยู่ในระบบ ผู้ดูแลระบบสามารถดูรายละเอียดของอุปกรณ์ แต่ละชิ้นได้ด้วยการกดเข้าไปที่อุปกรณ์นั้น อีกทั้งผู้ดูแลระบบสามารถเพิ่มอุปกรณ์ใหม่เข้าระบบและลบอุปกรณ์ที่มีอยู่ออกจากระบบได้จากเมนูนี้

3.4 ภาพรวมของฮาร์ดแวร์

3.4.1 ฐานสำหรับวางอุปกรณ์

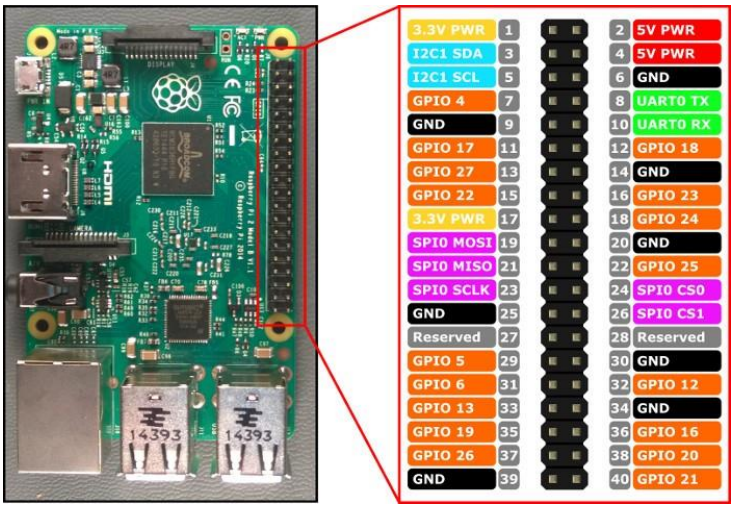
การจัดวางพื้นที่ที่ใช้สำหรับวางสิ่งของ ทำเป็นรูปทรงแปดเหลี่ยม เนื่องจากแขนกล Delta Robot นั้นเคลื่อนที่เป็นแบบวงกลมการทำฐานให้ใกล้เคียงทรงกลม จะทำให้ใช้ประโยชน์ได้มากที่สุด โดยรอบแท่นทรงแปดเหลี่ยมจะเป็นกล่องรูปทรงสี่เหลี่ยมผืนผ้า ที่ไว้สำหรับเก็บอุปกรณ์ที่คัดแยกตามประเภทที่ผู้ใช้งานได้ตั้งค่าไว้



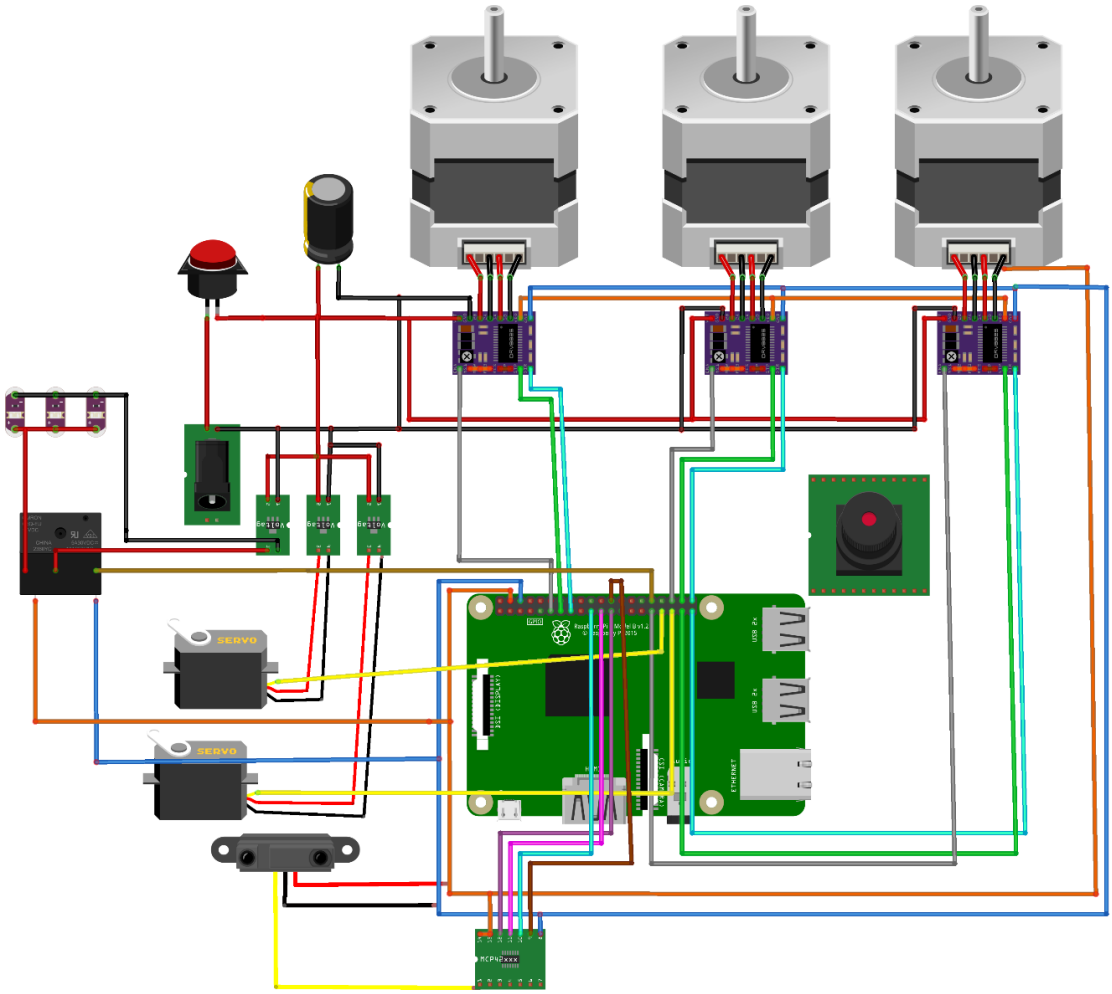
รูปที่ 3.30 แสดงการจัดเรียงอุปกรณ์จากด้านบน

3.4.2 **การต่อวงจร**

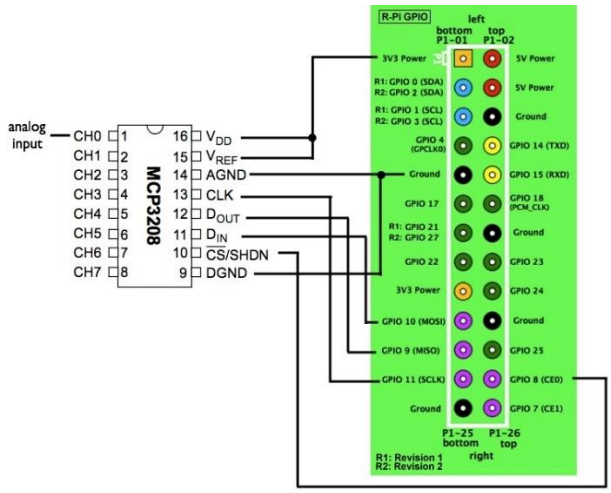
การต่อวงจรที่ใช้สำหรับการควบคุมการทำงานของมอเตอร์นั้น จะต้องคำนึงถึง GPIO PIN ที่เลือกใช้โดยจำเป็นจะต้องใช้สำหรับการควบคุมขา ENA STEPA DIRA ENB STEPB DIRB ENC STEPC และ DIRC สำหรับการควบคุมมอเตอร์ทั้งสามตัว โดยควบคุมการทำงานอิสระจากกัน ไฟที่จ่ายให้กับมอเตอร์มาจาก Power Supply 12VDC โดยระบบไฟทั้งหมดจะผ่านสวิตช์ฉุกเฉิน (Emergency Switch) ต่อเข้ากับ Stepdown เพื่อแบ่งจ่ายให้กับบอร์ด Raspberry Pi และ Servo ทั้งสองตัว และเนื่องจาก Raspberry Pi ไม่มีช่องสัญญาณสำหรับรับค่า Analog โดยตรง จึงต้องใช้ IC รุ่น MCP 3208 โดยแปลงค่าเป็นข้อมูลแบบ 12 บิต หรือ (0 - 4096) เชื่อมต่อด้วยโปรโตตอลแบบ Serial Peripheral Interface (SPI) โดยต่อวงจรดังรูปที่ 3.33



รูปที่ 3.31 แสดง GPIO PIN Rasberry Pi Model B



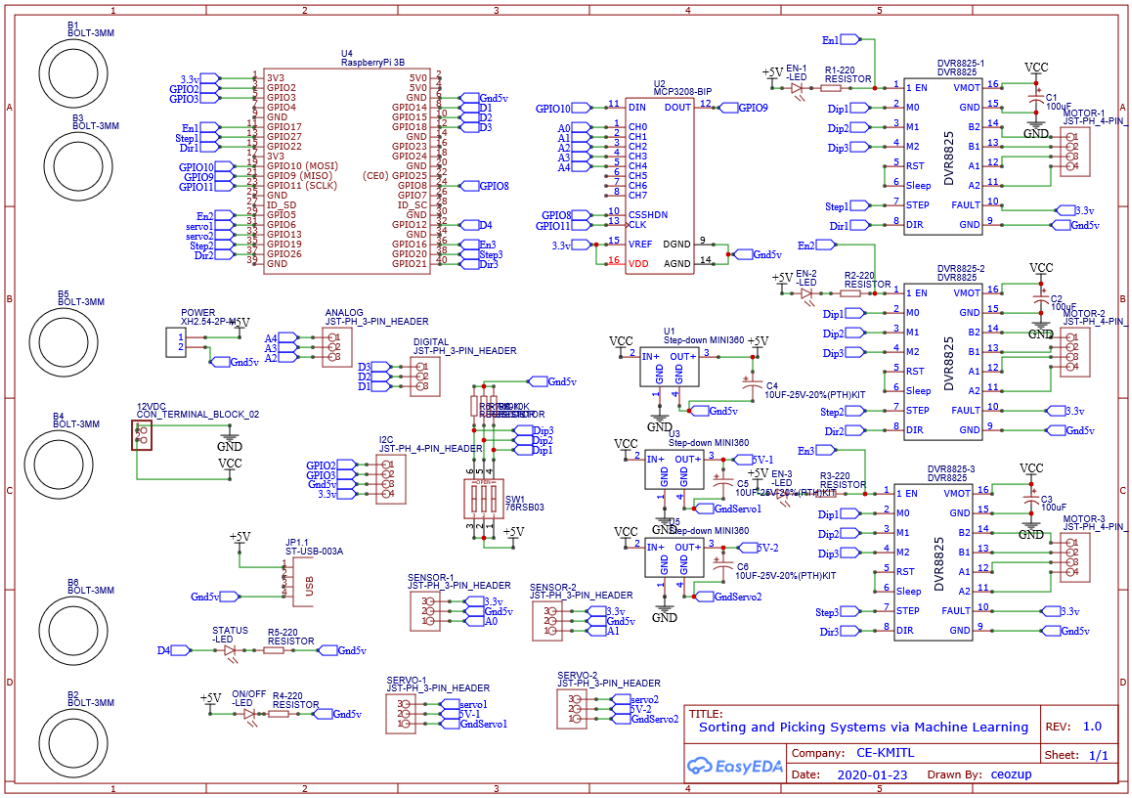
รูปที่ 3.32 แสดงภาพรวมของระบบทั้งหมด



รูปที่ 3.33 แสดงการเชื่อมต่อผ่าน SPI ระหว่าง Raspberry Pi และ MCP3208

3.4.3 การออกแบบวงจรพิมพ์อิเล็กทรอนิกส์

การออกแบบวงจรพิมพ์อิเล็กทรอนิกส์นั้นสร้างเป็นแบบสองด้าน การออกแบบแผนผังวงจรไฟฟ้า (Schematic) โดยใช้ซอฟต์แวร์ EasyEDA



รูปที่ 3.34 แสดงแผนผังวงจรไฟฟ้า (Schematic)

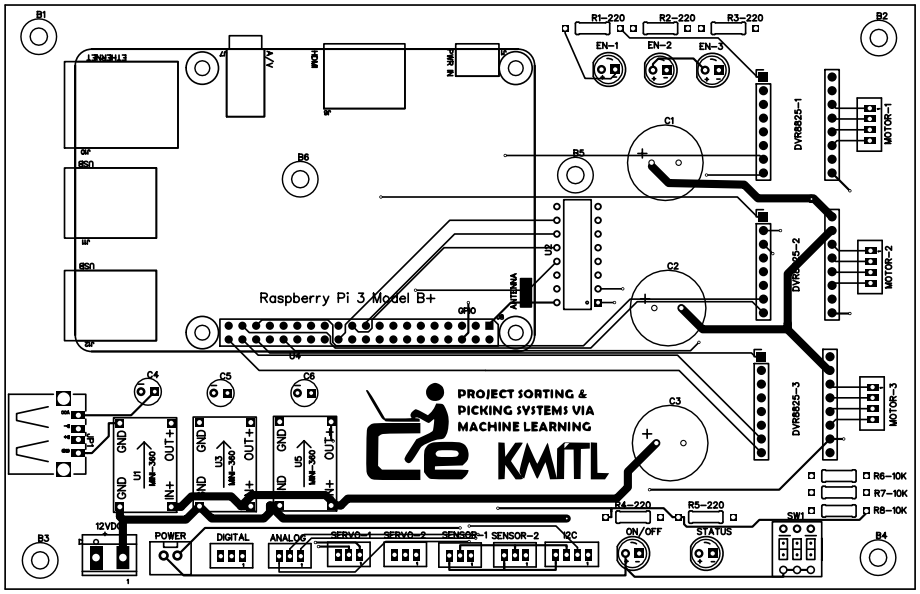
3.4.4 การตั้งข้อกำหนดในการออกแบบวงจรพิมพ์อิเล็กทรอนิกส์

เมื่อได้แผนผังวงจรไฟฟ้าที่ถูกต้องสมบูรณ์แล้วนั้น จะต้องทำการสร้างวงจรพิมพ์อิเล็กทรอนิกส์ หรือ Printed Circuit Board (PCB) เพื่อช่วยให้การทำงานมีประสิทธิภาพดีขึ้นและประสิทธิผลสูงขึ้น โดยการออกแบบนั้นจะต้องคำนวณกระแสไฟฟ้าที่ใช้ภายในวงจร เพื่อกำหนดขนาดและระยะห่างของลายวงจร โดยไฟ 12VDC ที่จ่ายให้กับมอเตอร์เราจะใช้ลายวงจรเส้นใหญ่ (Power) สำหรับไฟ 5VDC เราจะใช้ลายวงจรขนาดกลาง (5V) และสายสัญญาณจะใช้สายไฟขนาดมาตรฐาน (Default) ดังรูปที่ 3.35

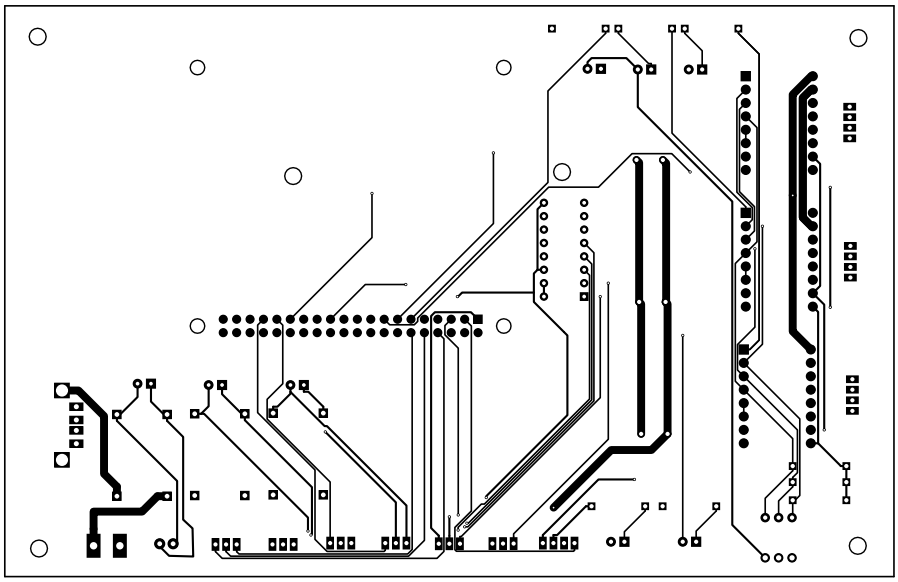
Rule	Track Width	Clearance	Via Diameter	Via Drill Diameter	Track Length
Default	0.25	0.16	0.61	0.32	
5V	0.4	0.25	0.61	0.32	
Power	1.5	0.3	0.61	0.32	

รูปที่ 3.35 แสดงข้อกำหนดในการออกแบบ (Design Rule)

เมื่อกำหนด Design Rule เสร็จสิ้น จะต้องทำการจัดวางอุปกรณ์ให้อยู่ในตำแหน่งที่เหมาะสม เพื่อให้ง่ายต่อการ Routing หลังจากนั้นทำการ Auto Router บางส่วนของวงจรพิมพ์อิเล็กทรอนิกส์ และทำการ Router บางส่วนใหม่ เนื่องจากการสร้างลายวงจรนั้น มีข้อกำหนดหรือข้อควรระวัง เพื่อป้องกันไม่ให้เกิดความเสียหายกับชิ้นงาน



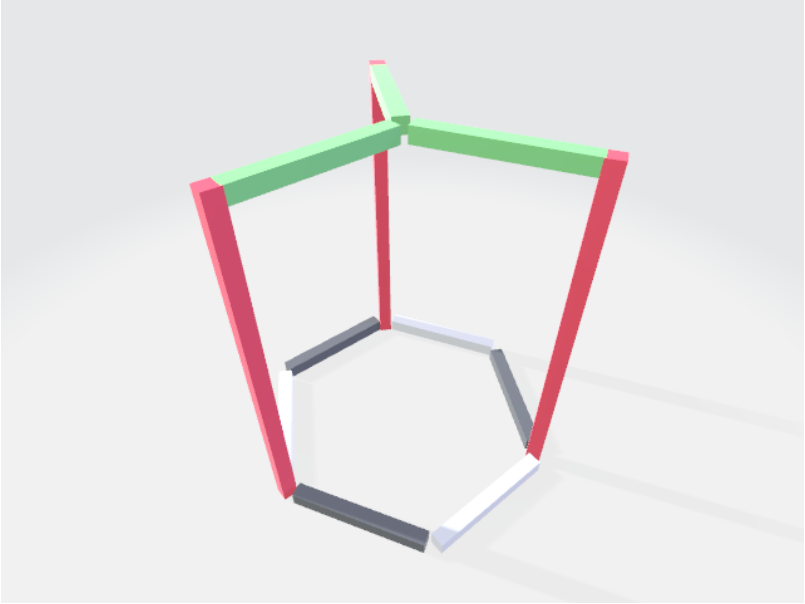
รูปที่ 3.36 แสดงวงจรพิมพ์อิเล็กทรอนิกส์ (ด้านหน้า)



รูปที่ 3.37 แสดงวงจรพิมพ์อิเล็กทรอนิกส์ (ด้านหลัง)

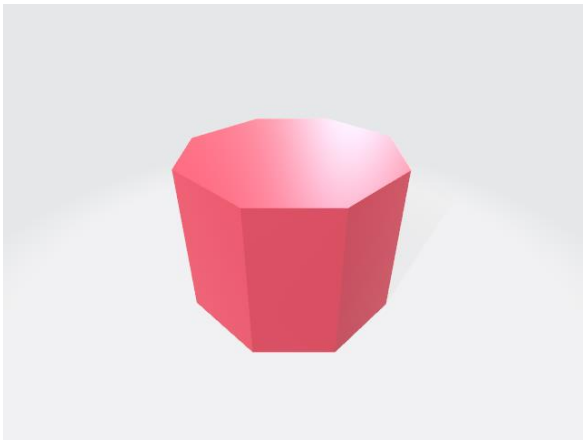
3.5 โครงสร้างหลัก

โครงสร้างหลักทำจากเหล็กกล่อง ประกอบกันโดยใช้ข้อต่อ 90 องศา และ ข้อต่อที่สามารถปรับองศาได้ โดยประกอบกันเป็นรูปหกเหลี่ยม ใช้เสาหลักเพื่อทำคานด้านบนสามเสา ยึดเข้าด้วยกันทำมุม 120 องศาต่อกัน



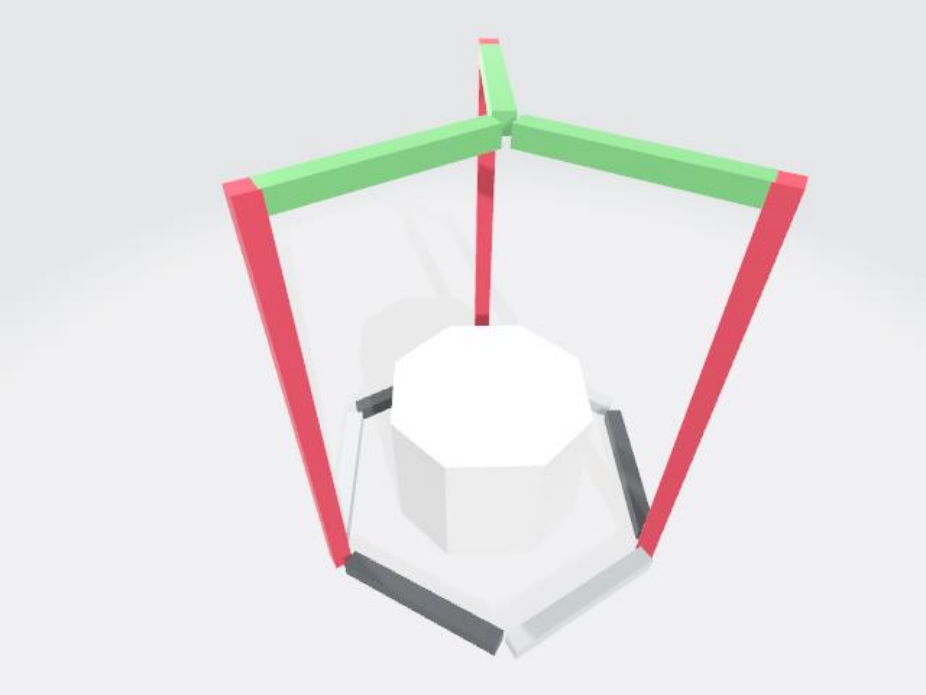
รูปที่ 3.38 โครงสร้างหลักใช้สำหรับวางอุปกรณ์และเป็นจุดติดตั้งมอเตอร์

ออกแบบฐานเป็นรูปทรงแปดเหลี่ยมเหลี่ยมสีขาว เพื่อใช้ในการวางชิ้นส่วนที่ต้องการให้ระบบทำการคัดแยกโดยทำให้มีความสูงจากพื้น เพื่อให้สามารถหยิบชิ้นส่วนได้รัศมีที่กว้างขึ้น



รูปที่ 3.39 ฐานรูปทรงแปดเหลี่ยมที่ใช้สำหรับวางชิ้นส่วนที่ต้องการคัดแยก

นำฐานรูปทรงแปดเหลี่ยมมาประกอบเข้ากับโครงสร้างหลัก เพื่อให้แขนกลสามารถหยิบจับชิ้นส่วนหรืออุปกรณ์ที่ต้องการคัดแยกได้

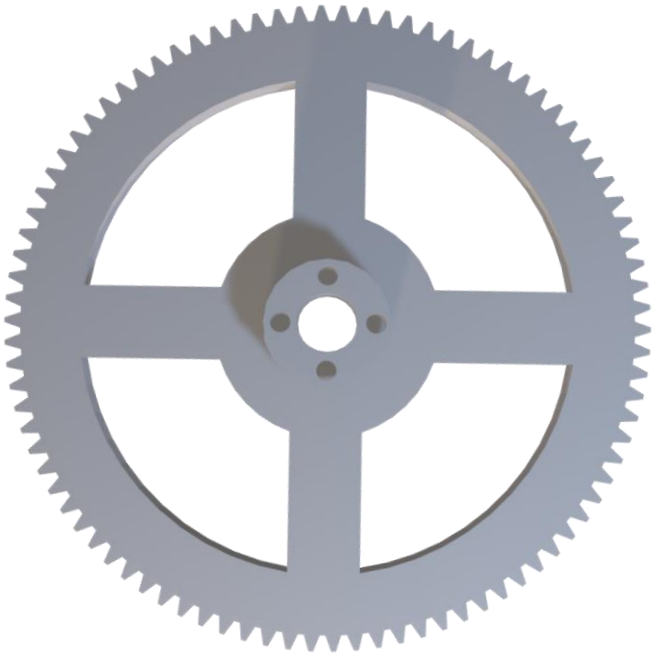


รูปที่ 3.40 แสดงการนำฐานรูปแปดเหลี่ยมมาวางเข้ากับโครงสร้างหลัก

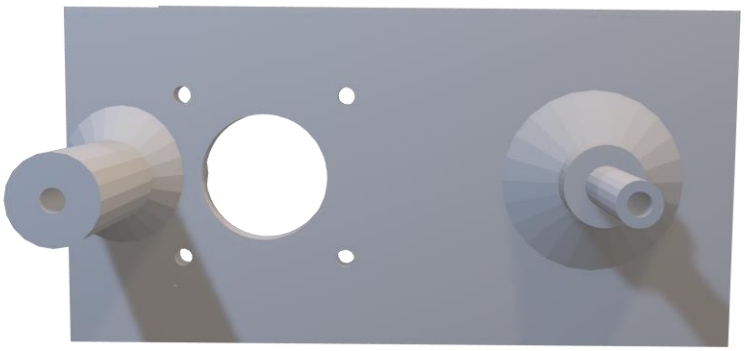
3.6 ออกแบบเฟืองเพื่อช่วยในการเพิ่มกำลังให้กับ Stepper Motor

เนื่องจาก Stepper Motor ที่ใช้นั้นมีกำลังที่ 0.8 นิวตันเมตร ที่ 1.6A ซึ่งเป็นกำลังสูงสุดของมอเตอร์ที่ทำได้ เนื่องจากแขนกลประเภทเดลต้านั้นมีความยาวที่มาก ทำให้เกิดโมเมนต์ของแรงมากตามไปด้วย มอเตอร์ที่ใช้งานจึงไม่สามารถที่จะขับเคลื่อนไปยังตำแหน่งที่กำหนดได้ จึงต้องใช้เฟืองในการทดแรง โดยอัตราส่วนของเฟือง คิดเป็น 1:8 ซึ่งจะทำให้ Stepper Motor มีกำลังมากยิ่งขึ้น โดยออกแบบให้เฟืองขับขนาดเล็กมีเส้นผ่านศูนย์กลาง 14.36 มิลลิเมตร ดังรูปที่ 3.41 และเฟืองตามขนาดใหญ่มีเส้นผ่านศูนย์กลาง 114.96 มิลลิเมตร ดังรูปที่ 3.42 และออกแบบฐานให้ยึดกับ Stepper Motor Bracket และสามารถที่จะทำให้เฟืองทั้งสองทำงานได้พอดี ดังรูปที่ 3.43 และเพิ่มความแข็งแรงด้วยชิ้นส่วนรองรับ ดังรูปที่ 3.44

รูปที่ 3.41 เฟืองขับขนาดเส้นผ่านศูนย์กลาง 14.36 มิลลิเมตร



รูปที่ 3.42 เฟืองตามขนาดเส้นผ่านศูนย์กลาง 114.96 มิลลิเมตร



รูปที่ 3.43 ฐานใช้สำหรับยึดชุดเฟืองกับ Stepper Motor Bracket

รูปที่ 3.44 ชิ้นส่วนช่วยรับแรงที่เกิดขึ้นเมื่อใส่แขนกล

3.7 ขั้นตอนการเพิ่ม Dataset เพื่อเทรนโมเดล

รูปที่ 3.45 ขั้นตอนการเพิ่ม Dataset เพื่อเทรนโมเดล

3.7.1 **ถ่ายภาพ**

ผู้ใช้งานติดต่อ Raspberry Pi ผ่าน Web Socket เพื่อสั่งงานให้ถ่ายรูปผ่านกล้องที่ติดตั้งไว้และบันทึกภาพมาเพื่อประมวลผลต่อ

3.7.2 **ประมวลผลภาพเบื้องต้น (Image Preprocessing)**

ทำ Perspective Transform กับภาพที่บันทึกไว้ จากนั้นจึง Segment วัตถุแต่ละชิ้นเพื่อนำมาหา Bounding Boxes ของวัตถุแต่ละชิ้น

3.7.3 **เทรนโมเดล**

นำ Bounding Boxes ของวัตถุที่ได้มาเทรน Classification Model

3.8 ขั้นตอนการแยกวัตถุ

รูปที่ 3.46 ขั้นตอนการแยกวัตถุ

3.8.1 **ถ่ายภาพ**

ผู้ใช้งานติดต่อ Raspberry Pi ผ่าน API เพื่อสั่งงานให้ถ่ายรูปผ่านกล้องที่ติดตั้งไว้และบันทึกภาพมาเพื่อประมวลผลต่อ

3.8.2 **ประมวลผลภาพเบื้องต้น (Image Preprocessing)**

ทำ Perspective Transform กับภาพที่บันทึกไว้ จากนั้นจึง Segment วัตถุแต่ละชิ้นเพื่อนำมาหา Bounding Boxes ของวัตถุแต่ละชิ้น

3.8.3 **แยกชนิดวัตถุ**

แยกชนิดของแต่ละวัตถุด้วยการนำไป Predict ด้วย Classification Model

3.8.4 **ส่งต่อตำแหน่งของแต่ละวัตถุ**

ส่งต่อตำแหน่งของแต่ละวัตถุไปยังโปรแกรมควบคุมแขนกลที่ใช้ในการหยิบจับเพื่อแยกวัตถุ

บทที่ 4

การทดลองและผลการทดลอง

4.1 ทดลองถ่ายภาพด้วย Raspberry Pi

4.1.1 จุดประสงค์การทดลอง

เพื่อทดสอบว่าสามารถใช้ Raspberry Pi สั่งการให้ Webcam ถ่ายภาพได้จริง

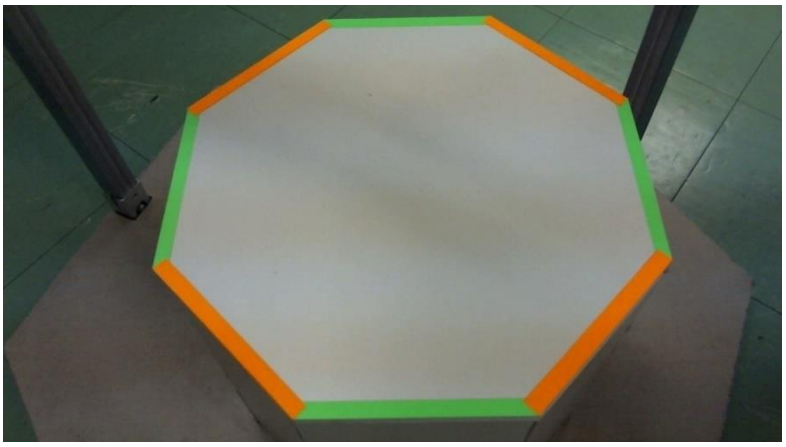
4.1.2 วิธีการทดลอง

นำ Raspberry Pi มาต่อกับ Webcam และทดลองถ่ายภาพด้วยไลบรารี่ OpenCV

```python
import os
import cv2
from classification import settings


def capture_image():
    if not os.path.exists(settings.ASSETS_ROOT):
        os.mkdir(settings.ASSETS_ROOT)
    filename = settings.ASSETS_ROOT + 'Snapshot.jpg'
    cam = cv2.VideoCapture(0)
    ret, image = cam.read()
    if ret:
        cv2.imwrite(filename, image)
    cam.release()
```

รูปที่ 4.1 โปรแกรมที่ใช้สำหรับสั่งให้ Raspberry Pi ถ่ายรูป

4.1.3 **ผลการทดลอง**



รูปที่ 4.2 รูปที่ถูกถ่ายด้วย Webcam

จากภาพเป็นการถ่ายภาพด้วย Webcam และบันทึกรูปภาพเก็บไว้ใน Raspberry Pi

4.1.4 **สรุปผลการทดลอง**

สามารถถ่ายภาพด้วย Raspberry Pi ผ่านการใช้งานไลบรารี่ OpenCV ได้

4.2 ทดลองเปรียบเทียบความแตกต่างของจำนวนพิกเซลวัตถุชิ้นเดียวกัน ก่อนและหลังการทำ Perspective Transform

4.2.1 **จุดประสงค์การทดลอง**

เพื่อทดสอบว่าก่อนการทำ Perspective Transform วัตถุชนิดเดียวกันที่อยู่คนละตำแหน่งกันจะมีจำนวนพิกเซลต่างกันอันเป็นผลมาจากการกล้องไม่ได้ถูกติดตั้งแบบ Bird's-eye View แต่ถูกติดตั้งด้วยมุมเอียง และเมื่อหลังทำ Perspective Transform แล้วจะมีจำนวนพิกเซลที่ใกล้เคียงกันมากกว่าก่อนทำ Perspective Transform

4.2.2 **วิธีการทดลอง**

1. ใช้ฟังก์ชั่น InRange ของ OpenCV กับภาพที่ถ่าย (รูปที่ 4.3) เพื่อทำการ Thresholding หาเส้นขอบฐานสีเขียวและสีส้ม ดังรูปที่ 4.4 และ 4.5

2. นำผลที่ได้จากการ Threshold ไปหาเส้นขอบด้วยการใช้ Canny Edge Detection ดังรูปที่ 4.6

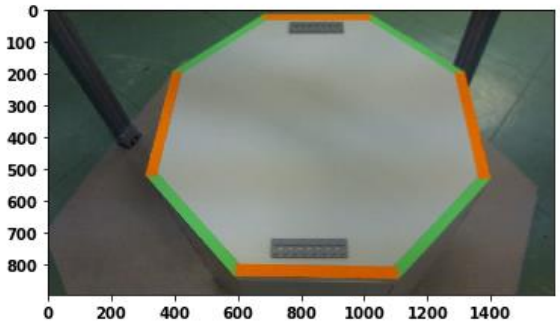
3. นำขอบที่ได้ไปหาเส้นตรงด้วยการใช้ Probabilistic Hough Line Transform ดังรูปที่ 4.7

4. นำเส้นตรงที่ได้มาเปรียบเทียบค่า x ที่มากที่สุดของทุกเส้นว่าเส้นตรงของขอบฐานสีใดมีค่า x น้อยกว่า จึงเลือกใช้เส้นตรงของขอบฐานสีนั้น ดังรูปที่ 4.8

5. นำเส้นตรงที่ได้มาเลือกเพียงแค่เส้นตรงของขอบฐานที่อยู่รอบนอก ดังรูปที่ 4.9

6. นำเส้นที่ได้มาหาจุดตัดของเส้นแต่ละเส้น ดังรูปที่ 4.10

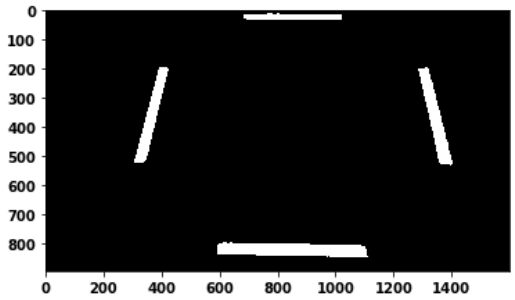
7. ใช้จุดตัดที่ได้มาก่อนหน้า เพื่อหาขนาดของรูปใหม่ในการทำ Perspective Transform ดังรูปที่ 4.21



รูปที่ 4.3 ฐานที่ใช้วางชิ้นส่วนเพื่อทดลองการทำ Perspective Transform

```
img_hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

low_orange = np.array([0, 90, 110])
high_orange = np.array([20, 255, 255])

orange_line = cv2.inRange(img_hsv, low_orange, high_orange)

plt.figure()
plt.imshow(orange_line, cmap='gray')
```

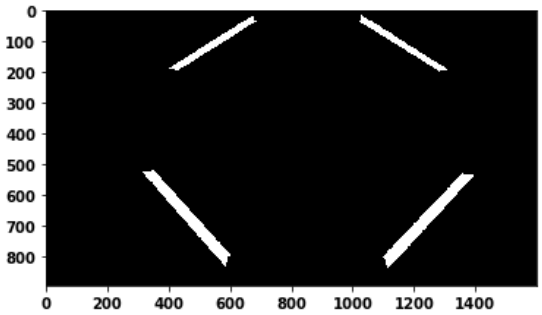
```
<matplotlib.image.AxesImage at 0x1fd0c29bf98>
```



รูปที่ 4.4 Threshold หาขอบสีส้ม

```
img_hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

low_green = np.array([40, 90, 110])
high_green = np.array([80, 255, 255])

green_line = cv2.inRange(img_hsv, low_green, high_green)

plt.figure()
plt.imshow(green_line, cmap='gray')
```

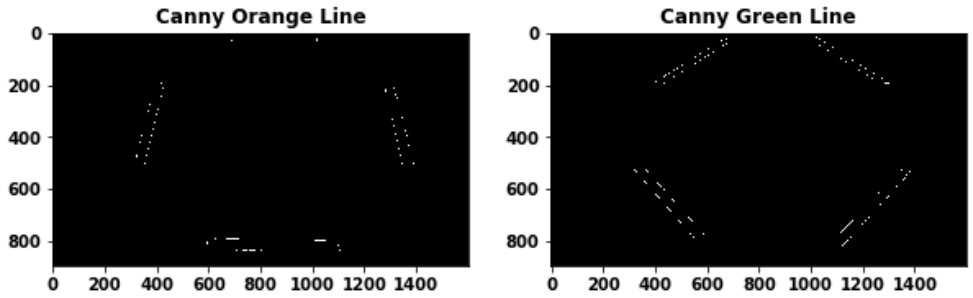
```
<matplotlib.image.AxesImage at 0x1fd0c5e3e80>
```



รูปที่ 4.5 การ Threshold หาขอบสีเขียว

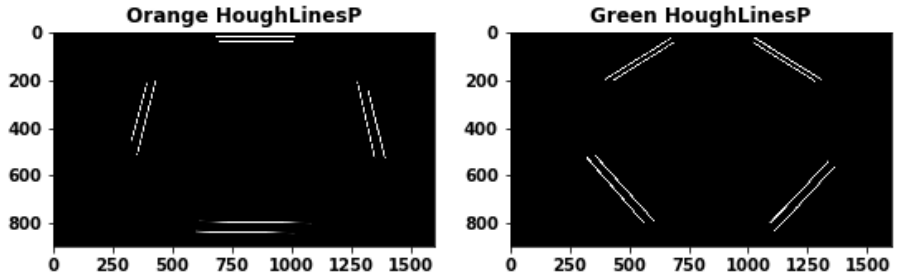
```
orange_edges = cv2.Canny(orange_line,50,150,apertureSize = 3)
green_edges = cv2.Canny(green_line,50,150,apertureSize = 3)

fig = plt.figure(figsize=(10, 8))
fig.add_subplot(1, 2, 1)
plt.title('Canny Orange Line')
plt.imshow(orange_edges, cmap='gray')
fig.add_subplot(1, 2, 2)
plt.title('Canny Green Line')
plt.imshow(green_edges, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x1f32853a940>
```



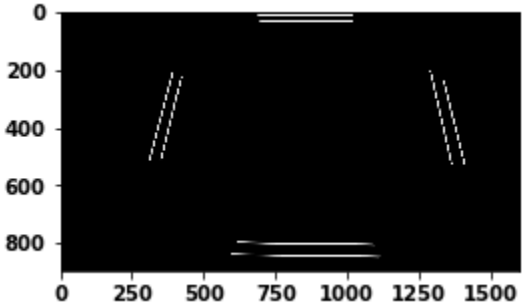
รูปที่ 4.6 การหาขอบโดยใช้ Canny edge detection

```
minLineLength=100

orange_lines = cv2.HoughLinesP(image=orange_edges,rho=1,theta=np.pi/180,
                               threshold=100,lines=np.array([]),
                               minLineLength=minLineLength,maxLineGap=80)
green_lines = cv2.HoughLinesP(image=green_edges,rho=1,theta=np.pi/180,
                              threshold=100,lines=np.array([]),
                              minLineLength=minLineLength,maxLineGap=80)
```



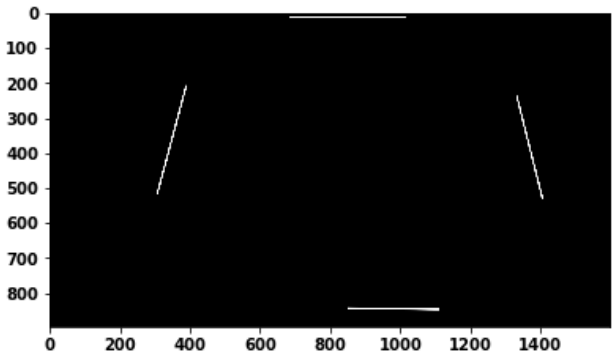
รูปที่ 4.7 เส้นตรงที่ได้จาก Probabilistic Hough Line Transform

```
choosed_lines = green_lines if (np.min(orange_lines[:,0,2]) > np.min(green_lines[:,0,2])) else orange_lines
```



รูปที่ 4.8 เส้นตรงขอบขอบสีที่เลือกใช้

```
left_line = choosed_lines[np.argmin(choosed_lines[:,0,2])][0]
right_line = choosed_lines[np.argmax(choosed_lines[:,0,0])][0]
top_line = choosed_lines[np.argmin(sorted_y[:,1])][0]
bottom_line = choosed_lines[np.argmax(sorted_y[:,0])][0]

usable_lines = {'vertical_line': {
                    'top_line':  [tuple(top_line[:2]), tuple(top_line[2:4])],
                    'bottom_line': [tuple(bottom_line[:2]), tuple(bottom_line[2:4])]
                    },
                'horizontal_line': {
                    'left_line': [tuple(left_line[:2]), tuple(left_line[2:4])],
                    'right_line':  [tuple(right_line[:2]),  tuple(right_line[2:4])]
                    }
                }
```



รูปที่ 4.9 เส้นตรงของขอบนอกของฐาน

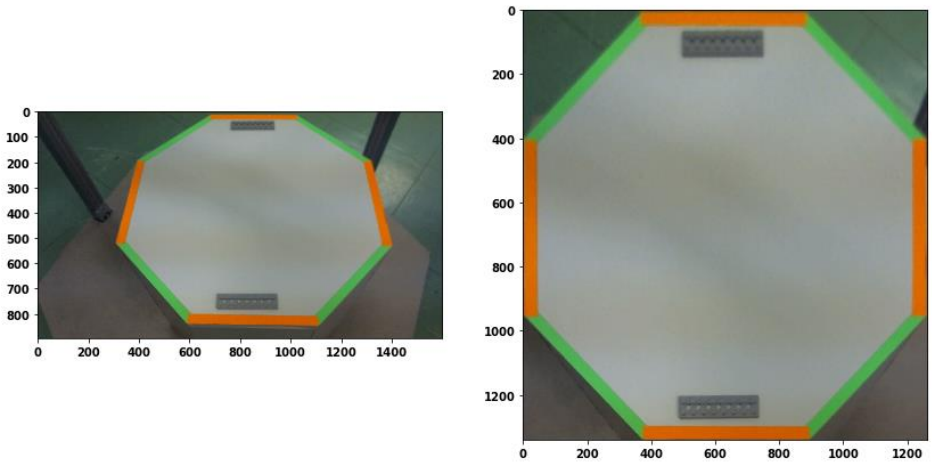
```
def line_intersection(line1, line2):
    xdiff = (line1[0][0] - line1[1][0], line2[0][0] - line2[1][0])
    ydiff = (line1[0][1] - line1[1][1], line2[0][1] - line2[1][1])

    def det(a, b):
        return a[0] * b[1] - a[1] * b[0]

    div = det(xdiff, ydiff)
    if div == 0:
       raise Exception('lines do not intersect')

    d = (det(*line1), det(*line2))
    x = det(d, xdiff) / div
    y = det(d, ydiff) / div
    return x, y
```

```
intersection_pts = []

for vertical_line in usable_lines['vertical_line'].values():
    for horizontal_line in usable_lines['horizontal_line'].values():
        intersection_pts.append(line_intersection(vertical_line, horizontal_line))

print(intersection_pts)
```

```
[(441.08469055374593, 11.0), (1274.5, 11.0), (221.6916661487788, 832.3860791746939), (1484.685576923077, 851.7423076923077)]
```

รูปที่ 4.10 จุดตัดของเส้นตรงที่ได้

```python
def four_point_transform(image, pts):
    # obtain a consistent order of the points and unpack them
    # individually
    rect = order_points(pts)
    (tl, tr, br, bl) = pts

    # compute the width of the new image, which will be the
    # maximum distance between bottom-right and bottom-left
    # x-coordiates or the top-right and top-left x-coordinates
    widthA = np.sqrt(((br[0] - bl[0]) ** 2) + ((br[1] - bl[1]) ** 2))
    widthB = np.sqrt(((tr[0] - tl[0]) ** 2) + ((tr[1] - tl[1]) ** 2))
    maxWidth = max(int(widthA), int(widthB))

    # compute the height of the new image, which will be the
    # maximum distance between the top-right and bottom-right
    # y-coordinates or the top-left and bottom-left y-coordinates
    heightA = np.sqrt(((tr[0] - br[0]) ** 2) + ((tr[1] - br[1]) ** 2))
    heightB = np.sqrt(((tl[0] - bl[0]) ** 2) + ((tl[1] - bl[1]) ** 2))
    maxHeight = max(int(heightA), int(heightB))

    # now that we have the dimensions of the new image, construct
    # the set of destination points to obtain a "birds eye view",
    # (i.e. top-down view) of the image, again specifying points
    # in the top-left, top-right, bottom-right, and bottom-left
    # order
    dst = np.array([
        [0, 0],
        [maxWidth - 1, 0],
        [maxWidth - 1, maxHeight - 1],
        [0, maxHeight - 1]], dtype = "float32")

    # compute the perspective transform matrix and then apply it
    M = cv2.getPerspectiveTransform(rect, dst)
    warped = cv2.warpPerspective(image, M, (maxWidth, maxHeight))

    # return the warped image
    return warped
```



รูปที่ 4.11 รูปก่อน-หลังทำ Perspective Transform

4.2.3 **ผลการทดลอง**

รูปก่อนที่จะทำ Perspective Transform นั้นวัตถุที่อยู่ข้างบนมีขนาด 6,031 พิกเซลและวัตถุด้านล่างมีขนาด 14,380 พิกเซล หลังจากที่ทำ Perspective Transform แล้ว วัตถุบนและล่างมีขนาด 20,804 พิกเซลและ 17,179 พิกเซลตามลำดับ ซึ่งเมื่อเปรียบเทียบกันแล้ว ก่อนทำ Perspective Transform วัตถุชิ้นเดียวกันมีขนาดต่างกัน 8,349 พิกเซล หลังทำมีขนาดต่างกัน 3,625 พิกเซล ประสิทธิภาพดีขึ้นคิดเป็น 56.58 เปอร์เซ็นต์

4.2.4 **สรุปผลการทดลอง**

การทำ Perspective Transform ช่วยให้จำนวนพิกเซลของวัตถุชนิดเดียวกัน ที่อยู่คนละตำแหน่งกัน มีความใกล้เคียงกันมากขึ้น

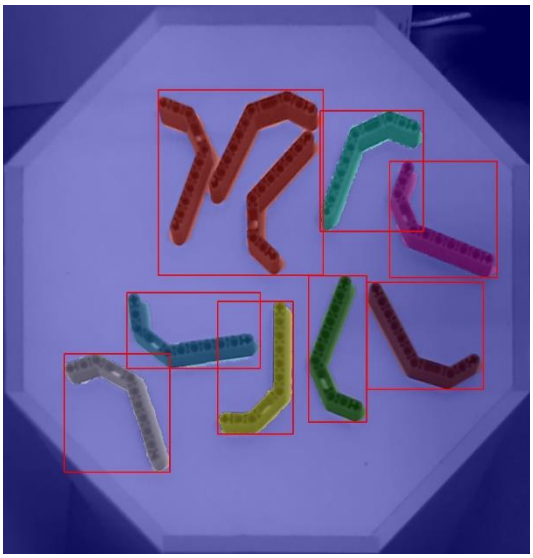
4.3 ทดลองเปรียบเทียบการ Segment วัตถุที่อยู่ติดกันระหว่าง Tiny Yolo และ Yolo

4.3.1 **จุดประสงค์การทดลอง**

เพื่อทดสอบว่า Tiny Yolo หรือ Yolo จะสามารถ Detect วัตถุที่อยู่ติดกันได้

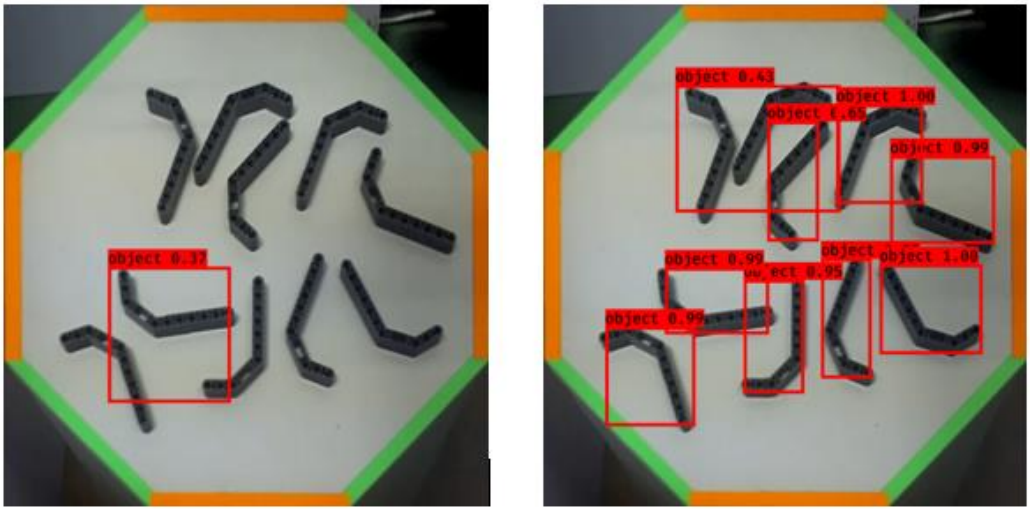
4.3.2 **วิธีการทดลอง**

เทรนโมเดล Tiny Yolo และ Yolo หลังจากนั้นจึงทดลองนำไปใช้ Detect วัตถุจากรูปที่วัตถุนั้นถูกวางไว้ติดกัน เป็นผลให้การ Segment โดยอาศัยขอบของวัตถุนั้นไม่สามารถแยกวัตถุออกจากกันได้ดังรูปที่ 4.12



รูปที่ 4.12 ผลการ Segment ของรูปที่วัตถุติดกัน

4.3.3 **ผลการทดลอง**



รูปที่ 4.13 ผลการ Detect วัตถุโดยใช้ Tiny Yolo และ Yolo ตามลำดับ

จากรูปที่ 4.23 จะเห็นได้ว่าผลลัพธ์ที่ได้จาก Tiny YOLO นั้นผิดพลาดค่อนข้างมากไม่สามารถนำไปใช้งานได้ ส่วน YOLO นั้นให้ผลลัพธ์ที่ค่อนข้างดีกว่า แต่กรอบ Bounding box ของวัตถุนั้นไม่ค่อยตรงอีกทั้งยังมีวัตถุติดกันบางชิ้นที่ถูก Detect ออกมาเป็นชิ้นเดียว ซึ่งอาจจะเป็นผลมาจากการเทรนโมเดลไม่พอ

4.3.4 **สรุปผลการทดลอง**

ไม่สามารถใช้งาน Tiny YOLO และ YOLO เพื่อ Detect วัตถุที่อยู่ติดกันได้ เนื่องจาก Tiny YOLO นั้นให้ผลลัพธ์ที่ไม่ดีพอ ส่วน YOLO นั้นจำเป็นต้องใช้ทรัพยากรในการเทรนมากเกินไป และไม่สามารถรันบน Raspberry Pi ได้

4.4 ทดลองเทรนโมเดล ML Classifiers แต่ละโมเดลเพื่อเปรียบเทียบความแม่นยำ

4.4.1 จุดประสงค์การทดลอง

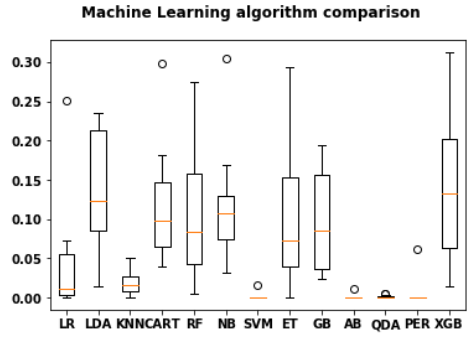
เพื่อทดสอบว่าโมเดลใดที่เหมาะสมกับงานแยกประเภทชิ้นส่วนที่ต้องการ

4.4.2 วิธีการทดลอง

สร้างโมเดล Logistic Regression(LR), Linear Discriminant Analysis(LDA), KNeighbors(KNN) Decision Tree(CART) Random Forest(RF) GaussianNB(NB) Support Vector Machine(SVM) Extra Trees(ET) Gradient Boosting(GB) AdaBoost(AB) Quadratic Discriminant Analysis(QDA) Perceptron(PER) XGBoost(XGB) ขึ้นมาแล้วนำไปเทรนด้วย Dataset ของชิ้นส่วนที่ต้องการแยก จากนั้นจึงนำมาเปรียบเทียบความแม่นยำของแต่ละโมเดลผลการทดลอง

4.4.3 ผลการทดลอง

```
LR: 0.043974 (0.073279)
LDA: 0.135357 (0.076088)
KNN: 0.020526 (0.016528)
CART: 0.117793 (0.074538)
RF: 0.103650 (0.079538)
NB: 0.118606 (0.072868)
SVM: 0.001600 (0.004800)
ET: 0.101247 (0.085346)
GB: 0.095397 (0.062917)
AB: 0.001064 (0.003191)
QDA: 0.001333 (0.002149)
PER: 0.006133 (0.018400)
XGB: 0.137230 (0.091774)
```



รูปที่ 4.14 ผลการเทรนโมเดล ML Classifiers

จากรูปที่ 4.14 ตัวอักษรคือชื่อย่อของแต่ละโมเดล ตามด้วยค่าความแม่นยำ ค่าความเบี่ยงเบนมาตรฐาน(เลขในวงเล็บ) และกราฟ Box & Whisker ซึ่งโมเดล XGBoost นั้นมีค่าความแม่นยำสูงสุด

4.4.4 สรุปผลการทดลอง

เลือกใช้โมเดล XGBoost เนื่องจากมีความแม่นยำสูงสุด

4.5 ทดลองเปรียบเทียบความแม่นยำระหว่าง XGBoost และ Xception (Convnet)

4.5.1 จุดประสงค์การทดลอง

เพื่อทดสอบว่าโมเดลใดมีความแม่นยำสูงกว่าและเหมาะสมกับการใช้งาน

4.5.2 วิธีการทดลอง

เทรนโมเดล XGBoost และ Xception ที่ปรับจูนแล้ว จากนั้นจึงนำมาเปรียบเทียบความแม่นยำ

4.5.3 ผลการทดลอง

```
Model Performance
Average Error: 1.4952 degrees.
Accuracy = 0.83%.
```

รูปที่ 4.15 ผลการทดสอบโมเดล XGBoost

```
Test loss: 0.1914571076631546
Test accuracy: 0.9576944708824158
```

รูปที่ 4.16 ผลการทดสอบโมเดล Xception

4.5.4 สรุปผลการทดลอง

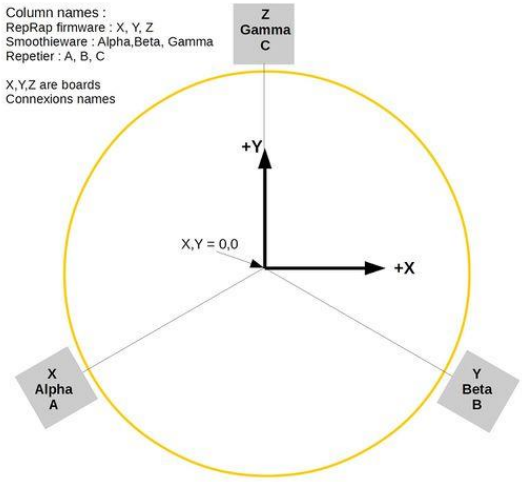
เลือกใช้โมเดล Xception เนื่องจากมีความแม่นยำสูงกว่า

4.6 การทดลองรัศมีของแขนกล โดยการเปลี่ยนความยาวของแขนกลแต่ละส่วน

4.6.1 จุดประสงค์การทดลอง

เพื่อทดสอบความยาวที่เหมาะสมของก้านโยงด้านบนและก้านโยงด้านล่างที่เหมาะสม และสามารถทำงานได้มีประสิทธิภาพมากที่สุด

4.6.2 วิธีการทดลอง



รูปที่ 4.17 ภาพแสดงการเคลื่อนที่ของปลายแขนกล

นำชิ้นส่วน Lego Mindstorms NXT มาต่อจำลองความยาวของก้านโยงด้านบนและก้านโยงด้านล่าง ที่ความยาว ดังนี้

ตารางที่ 4.1 ตารางแสดงขนาดที่ใช้ในการทดลองเพิ่ม-ลดระยะแขนกล

ชิ้นส่วน	รูปแบบที่ 1	รูปแบบที่ 2	รูปแบบที่ 3
Actuating Arm (rf)	23 cm	23 cm	17.5 cm
Passive Arm(re)	24.5 cm	19.5 cm	24.5 cm
Base Radius (f)	10 cm	10 cm	10 cm
End Effector Radius (e)	4 cm	4 cm	4 cm

4.6.3 **ผลการทดลอง**

ตารางที่ 4.2 ตารางแสดงระยะที่ทำได้เมื่อทดลองเปลี่ยนขนาดแขนกลที่ขนาดต่างๆ

ลักษณะแขนกล (คำนวณจาก Model Delta Robot)	ระยะที่แขนกลขึ้นสูงสุด	ระยะที่แขนกลลงต่ำที่สุด	ระยะที่ที่แขนกลเคลื่อนที่ห่างจุดศูนย์กลางที่สุด
Actuating arm (rf)= 23 cm Passive arm(re)= 24.5 cm Base radius (f)= 10cm End Effector radius (e)= 4 cm	25 cm	46 cm	22.5 cm
Actuating arm (rf)= 23 cm Passive arm(re)= 19.5 cm Base radius (f)= 10cm End Effector radius (e)= 4 cm	20.5 cm	42 cm	20 cm
Actuating arm (rf)=17.5 cm Passive arm(re)= 24.5 cm Base radius (f)= 10cm End Effector radius (e)= 4 cm	21 cm	43 cm	19 cm

ลักษณะแขนกล (ใช้สูตร Kinematics Calculations)	ระยะที่แขนกลขึ้นได้ที่สูงสุด	ระยะที่แขนกลลงได้ต่ำที่สุด	ระยะที่ที่แขนกลเคลื่อนที่จากจุดศูนย์กลางไกลที่สุด
Actuating Arm (rf)= 23 cm Passive Arm(re)= 24.5 cm Base Radius (f)= 10cm End Effector Radius (e)= 4 cm	7.8 cm	38.7 cm	40.7 cm
Actuating Arm (rf)= 23 cm Passive Arm(re)= 19.5 cm Base Radius (f)= 10cm End Effector Radius (e)= 4 cm	6.5 cm	22.1 cm	21.8 cm
Actuating Arm (rf)=17.5 cm Passive Arm(re)= 24.5 cm Base Radius (f)= 10cm End Effector Radius (e)= 4 cm	12.9 cm	35.8 cm	16.1 cm

4.6.4 **สรุปผลการทดลอง**

ความยาวที่เหมาะสมของก้านโยงด้านบนและก้านโยงด้านล่างที่เหมาะสม และสามารถทำงานได้มีประสิทธิภาพมากที่สุด คือ

ตารางที่ 4.3 ตารางแสดงสรุปผลการทดลองของขนาดแขนกลที่เหมาะสมที่สุด

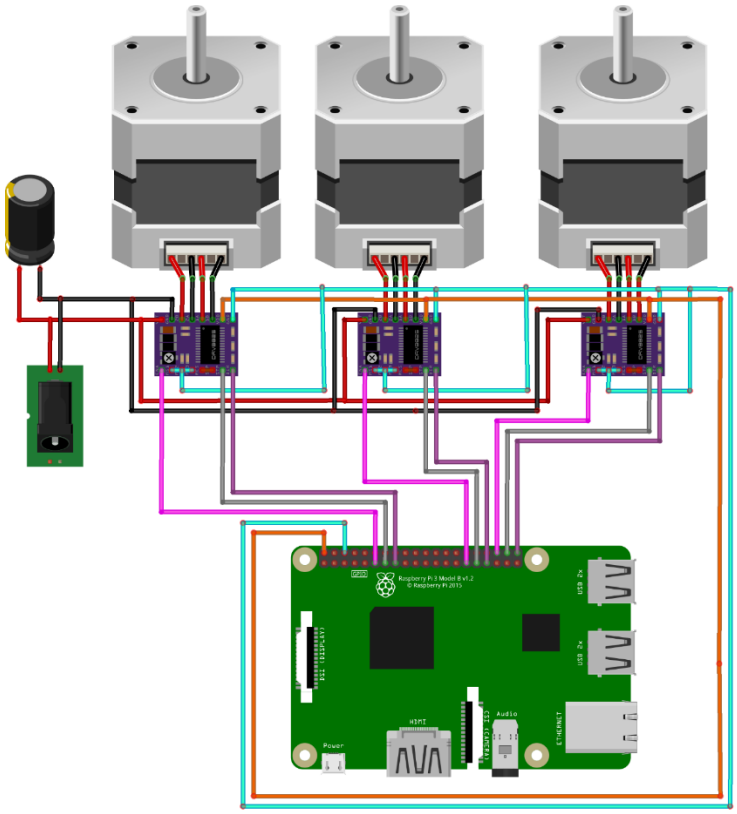
ลักษณะแขนกล (คำนวณจาก Model Delta Robot)	ระยะที่แขนกลขึ้นสูงสุด	ระยะที่แขนกลลงต่ำที่สุด	ระยะที่ที่แขนกลเคลื่อนที่ห่างจุดศูนย์กลางที่สุด
Actuating Arm (rf)= 23 cm Passive Arm(re)= 24.5 cm Base Radius (f)= 10cm End Effector Radius (e)= 4 cm	25 cm	46 cm	22.5 cm

4.7 การทดลองควบคุม Stepper Motor โดยใช้บอร์ด Raspberry Pi

4.7.1 จุดประสงค์การทดลอง

เพื่อทดสอบการควบคุม Driver DRV8825 เพื่อใช้ในการคุมการทำงานของ Stepper Motor ผ่านบอร์ด Raspberry Pi

4.7.2 วิธีการทดลอง



รูปที่ 4.18 วงจรที่ใช้ในการทดสอบการควบคุม Stepper Motor

```python
from time import sleep
import RPi.GPIO as GPIO

EN0 = 11 # Enable Driver GPIO Pin
STEP0 = 13  # Step GPIO Pin
DIR0 = 15   # Direction GPIO Pin

EN1 = 36 # Enable Driver GPIO Pin
STEP1 = 38  # Step GPIO Pin
DIR1 = 40   # Direction GPIO Pin

EN2 = 29 # Enable Driver GPIO Pin
STEP2 = 31  # Step GPIO Pin
DIR2 = 37   # Direction GPIO Pin

CW = 1     # Clockwise Rotation
CCW = 0    # Counterclockwise Rotation
SPR = 200  # Steps per Revolution (360 / 7.5)

GPIO.setmode(GPIO.BOARD)

GPIO.setup(EN0, GPIO.OUT)
GPIO.setup(DIR0, GPIO.OUT)
GPIO.setup(STEP0, GPIO.OUT)

GPIO.setup(EN1, GPIO.OUT)
GPIO.setup(DIR1, GPIO.OUT)
GPIO.setup(STEP1, GPIO.OUT)

GPIO.setup(EN2, GPIO.OUT)
GPIO.setup(DIR2, GPIO.OUT)
GPIO.setup(STEP2, GPIO.OUT)

GPIO.output(EN0, 0)
GPIO.output(EN1, 0)
GPIO.output(EN2, 0)
GPIO.output(DIR0, CW)
GPIO.output(DIR1, CW)
GPIO.output(DIR2, CW)

step_count = SPR
delay = .002

for x in range(step_count):
    GPIO.output(STEP0, GPIO.HIGH)
    GPIO.output(STEP1, GPIO.HIGH)
    GPIO.output(STEP2, GPIO.HIGH)
    sleep(delay)
    GPIO.output(STEP0, GPIO.LOW)
    GPIO.output(STEP1, GPIO.LOW)
    GPIO.output(STEP2, GPIO.LOW)
    sleep(delay)

sleep(.5)
print("end")

GPIO.output(EN0,GPIO.LOW)
GPIO.output(EN1,GPIO.LOW)
GPIO.output(EN2,GPIO.LOW)
```

รูปที่ 4.19 โปรแกรมสำหรับใช้ในการทดสอบควบคุม Stepper Motor

4.7.3 **ผลการทดลอง**

สามารถกำหนด GPIO ที่ต้องการจะใช้ในการควบคุม Stepper Motor ได้และสามารถควบคุมรอบการทำงานของมอเตอร์ และจ่ายไฟให้กับ Driver DRV8825 12VDC ให้กับมอเตอร์แยกได้

4.7.4 **สรุปผลการทดลอง**

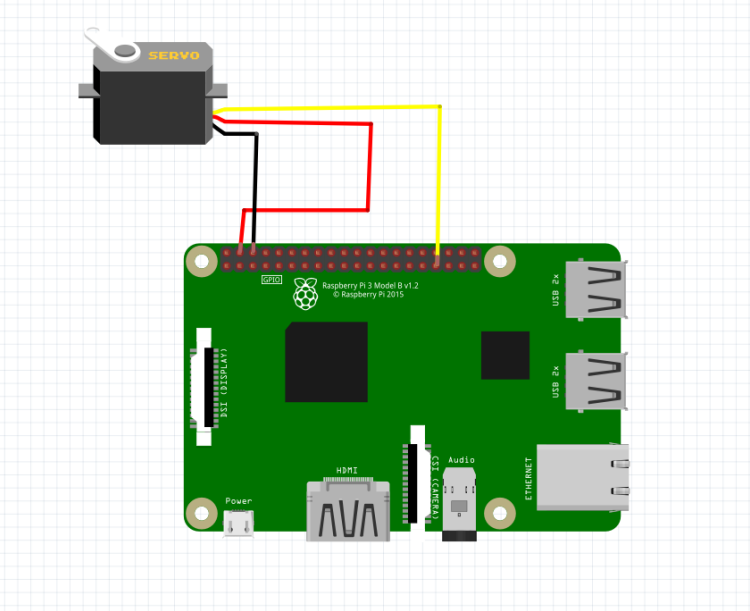
สามารถใช้บอร์ด Raspberry Pi สามารถควบคุมทิศทาง จำนวนรอบ ผ่าน Driver DRV8825 เพื่อใช้ในการควบคุม Stepper Motor ได้ และสามารถควบคุมมอเตอร์ได้อย่างอิสระจากกัน

4.8 การทดลองควบคุม Servo Motor โดยใช้บอร์ด Raspberry Pi

4.8.1 **จุดประสงค์การทดลอง**

เพื่อทดสอบการควบคุม Servo Motor ที่ใช้เพื่อคีบชิ้นส่วน และเปลี่ยนทิศทางแขนคีบ โดยควบคุมผ่านบอร์ด Raspberry Pi

4.8.2 **วิธีการทดลอง**



รูปที่ 4.20 วงจรที่ใช้ในการทดสอบการควบคุม Servo Motor

```python
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)

GPIO.setup(33, GPIO.OUT)

p = GPIO.PWM(33, 50)

p.start(7.5)

try:
    while True:
        p.ChangeDutyCycle(7.5)  # turn towards 90 degree
        time.sleep(1) # sleep 1 second
        p.ChangeDutyCycle(2.5)  # turn towards 0 degree
        time.sleep(1) # sleep 1 second
        p.ChangeDutyCycle(12.5) # turn towards 180 degree
        time.sleep(1) # sleep 1 second
except KeyboardInterrupt:
    p.stop()
    GPIO.cleanup()

```

รูปที่ 4.21 โปรแกรมสำหรับใช้ในการทดสอบควบคุม Servo Motor

4.8.3 ผลการทดลอง

สามารถควบคุมรอบการทำงานของ Servo Motor และสามารถแยกไฟเลี้ยงของ Servo Motor มาใช้ไฟเลี้ยงแยก เพื่อป้องกันไม่ให้คอนโทรลเลอร์เกิดความเสียหายได้

4.8.4 สรุปผลการทดลอง

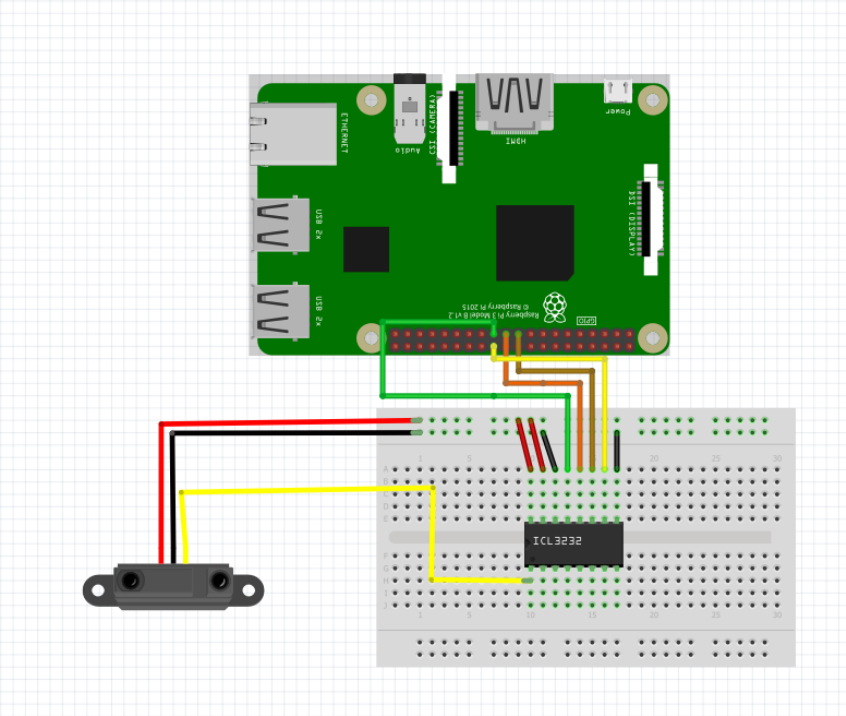
สามารถใช้บอร์ด Raspberry Pi สามารถควบคุมทิศทาง Servo Motor ได้

4.9 การทดลองรับค่า Sensor ผ่าน IC MCP3208 โดยใช้บอร์ด Raspberry Pi

4.9.1 จุดประสงค์การทดลอง

เพื่อทดสอบการรับค่า Analog ของ Sensor โดยใช้ Raspberry Pi เนื่องจากบอร์ดไม่สามารถรับค่า Analog ได้โดยตรงจึงจำเป็นต้องใช้ IC MCP3208 ที่เป็น Analog to Digital Converter เชื่อมต่อผ่าน Serial Peripheral Interface (SPI) ช่วยในการรับค่าสัญญาณจาก Sensor

4.9.2 วิธีการทดลอง



รูปที่ 4.22 วงจรที่ใช้ในการอ่านค่า Analog จาก IC MCP3208

```
1   from mcp3208 import MCP3208
2   import time
3
4   adc = MCP3208()
5
6   while True:
7           for i in range(8):
8                   print('ADC[{}]: {:.2f}'.format(i, adc.read(i)))
9           time.sleep(0.5)
10
```

รูปที่ 4.23 โปรแกรมสำหรับใช้ในการอ่านค่า Infrared Sensor

4.9.3 ผลการทดลอง

สามารถรับค่า Analog จาก Infrared Sensor ได้

4.9.4 สรุปผลการทดลอง

สามารถใช้บอร์ด Raspberry Pi อ่านค่า Infrared Sensor ได้

4.10 ทดลองใช้ฟังก์ชัน Inverse Kinematic เพื่อเปลี่ยนองศาของมอเตอร์ไปยังตำแหน่งที่กำหนดตาม x0,y0,z0

4.10.1 จุดประสงค์การทดลอง

เพื่อใช้ในการคำนวณหาค่า Theta และเปลี่ยนองศาของมอเตอร์ไปยังตำแหน่งที่กำหนดตาม x_0,y_0,z_0 ได้

4.10.2 วิธีการทดลอง

```
# DELTA ROBOT CONFIG
e  =  4.0
f  =  10.0
re = 24.5
rf =  23.0

default_motor = [0,0,0]

box =  [[0,0,0],
        [0,0,0],
        [0,0,0],
        [0,0,0],
        [0,0,0],
        [0,0,0],
        [0,0,0],
        [0,0,0],] # box[0-7] for release you can config position

######################
```

รูปที่ 4.24 แสดงการตั้งค่าขนาดของแขนกล Delta Robot

```
# Trigonometric constants
s      = 165*2
sqrt3  = math.sqrt(3.0)
pi     = 3.141592653
sin120 = sqrt3 / 2.0
cos120 = -0.5
tan60  = sqrt3
sin30  = 0.5
tan30  = 1.0 / sqrt3
######################
```

รูปที่ 4.25 แสดงการคำนวณ Trigonometric Constants

```
def angle_yz(x0, y0, z0, theta=None):
    y1 = -0.5*0.57735*f # f/2 * tg 30
    y0 -= 0.5*0.57735*e # shift center to edge
    # z = a + b*y
    a = (x0*x0 + y0*y0 + z0*z0 + rf*rf - re*re - y1*y1) / (2.0*z0)
    b = (y1-y0) / z0

    # discriminant
    d = -(a + b*y1)*(a + b*y1) + rf*(b*b*rf + rf)
    if d<0:
        return [1,0] # non-existing povar.  return error, theta

    yj = (y1 - a*b - math.sqrt(d)) / (b*b + 1) # choosing outer povar
    zj = a + b*yj
    theta = math.atan(-zj / (y1-yj)) * 180.0 / pi + (180.0 if yj>y1 else 0.0)

    return [0,theta] # return error, theta
```

รูปที่ 4.26 ฟังก์ชันที่ใช้ในการคำนวณมุมแต่ละมุม

```
def inverse(x0, y0, z0):
    theta1 = 0
    theta2 = 0
    theta3 = 0
    status = angle_yz(x0,y0,z0)

    if status[0] == 0:
        theta1 = status[1]
        status = angle_yz(x0*cos120 + y0*sin120,
                          y0*cos120-x0*sin120,
                          z0,
                          theta2)
    if status[0] == 0:
        theta2 = status[1]
        status = angle_yz(x0*cos120 - y0*sin120,
                          y0*cos120 + x0*sin120,
                          z0,
                          theta3)
    theta3 = status[1]

    return [status[0],theta1,theta2,theta3]
```

รูปที่ 4.27 ฟังก์ชันที่ใช้ในการคำนวณค่า Theta1, Theta2, Theta3 ของมอเตอร์แต่ละตัว

4.10.3 ผลการทดลอง

```
print(inverse(10.0,10.5,5))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
[0, -87.31339880855586, 56.36363820087381, -78.62192987634648]
```

รูปที่ 4.28 ผลลัพธ์เมื่อส่งค่าให้กับฟังก์ชัน inverse คำนวณค่าองศาของแต่ละมุม

4.10.4 สรุปผลการทดลอง

ฟังก์ชันสามารถนำค่าที่ x_0,y_0,z_0 ไปคำนวณเพื่อขององศาของแต่ละมุมได้ ถ้าหากเป็นตำแหน่ง x_0,y_0,z_0 ที่ไม่สามารถทำงานได้ตัวฟังก์ชันจะแสดงค่ากลับมาเป็น [1, 0, 0, 0] แสดงว่าเกิดข้อผิดพลาด

บทที่ 5

บทสรุปและข้อเสนอแนะ

5.1 บทสรุป

5.1.1 การทำงานของระบบ

เริ่มจากการที่ผู้ใช้วางวัตถุที่ต้องการคัดแยกลงบนเพลท จากนั้น Raspberry Pi จะทำในส่วน Pre-processing ก่อนด้วยการทำ Perspective Transform กับภาพที่ได้ แล้วแสดง Initial View ให้ผู้ใช้ดูผ่านหน้าเว็บ จากนั้นผู้ใช้ทำการเลือกรูปแบบการคัดแยก ต่อมาเป็นขั้นตอนของการคัดแยก เริ่มด้วยการประมวลผลภาพ จากนั้นจึงทำการควบคุม Delta Robot แล้วจึงส่งข้อมูลกลับไปยังเซิฟเวอร์เพื่อแสดงผลให้กับผู้ใช้ เมื่อจบขั้นตอนของการคัดแยกแล้ว หน้าเว็บจะแสดงข้อมูลที่ได้เพื่อให้ผู้ใช้บันทึกผลความถูกต้อง แล้วจึงแสดงสรุปผลลัพธ์เป็นลำดับสุดท้าย

5.1.2 ความรู้ที่ได้รับ

1) ได้รับความรู้ในการควบคุมการทำงานระหว่าง Hardware และ Software
2) ได้รับความรู้ในการทำ Image Processing และ Machine Learning

5.2 ปัญหาและแนวทางแก้ไข

1) เนื่องจากภาพที่ถ่ายนั้นถ่ายด้วยมุมเอียง จึงไม่สามารถนำมาใช้ได้ทันที ต้องทำ Perspective transform ก่อน
2) ไม่สามารถใช้ YOLO ในการคัดแยกวัตถุได้ เนื่องจากจำเป็นต้องใช้ทรัพยากรจำนวนมากในการเทรนโมเดล ซึ่งไม่เหมาะสมกับโปรเจคที่ทำ
3) โมดูลที่ใช้ในการขับเคลื่อน Stepper Motor มีขนาดเล็ก ควรใช้โมดูลที่ทนกระแสเป็นสองเท่าของกระแสสูงสุดที่ Stepper Motor ทำงาน
4) กระแสไฟที่จ่ายให้กับมอเตอร์ ไม่เสถียร จึงต้องเพิ่มตัวเก็บประจุเพื่อเพิ่มความเสถียร
5) มอเตอร์มีแรงไม่เพียงพอที่จะขับเคลื่อนแขนกล

5.3 แนวทางการพัฒนาต่อ

1) เปลี่ยนข้อต่อของตัวแขนกลให้เป็นแม่เหล็ก เพื่อให้การทำงานอิสระมากยิ่งขึ้น

2) เปลี่ยนโมดูลสำหรับคลับเคลื่อน Stepper Motor รวมถึง Stepper Motor ให้มีขนาดใหญ่ขึ้น

3) ทำให้อุปกรณ์สามารถแยกวัตถุที่วางติดกันได้

บรรณานุกรม

ชาญณรงค์ ชูสุย. 2016. Prototype Of Industrial Delta Robot. [Online].
Available : http://www.repository.rmutt.ac.th/xmlui/bitstream/handle/123456789/2781/RMUTT-151475.pdf.

Ceyhun Kazel. 2016. Why I chose Django over Java Frameworks for my recent project. [Online].
Available : https://hackernoon.com/why-i-chose-django-over-java-frameworks-for-my-recent-project-7ec85cb35756.

บริษัท วีนัส ซัพพลาย จำกัด. 2016. บทความการพัฒนาโปรแกรมบน Raspberry Pi ด้วย Qt. [Online].
Available : https://www.thaieasyelec.com/article-wiki/embedded-electronics-application/raspberry-pi-programming-with-qt-ch1.html.

นิสิต จรูญภาค. 2016. ความหมายของหุ่นยนต์. [Online].
Available : http://krunisit.rwb.ac.th/robot.html.

Bissett. 2017. Interfacing hardware with the Raspberry Pi. [Online].
Available : https://www.rs-online.com/designspark/interfacing-hardware-with-the-raspberry-pi/.

ฝ่ายวิจัยนโยบาย สวทช. 2017. อุตสาหกรรมหุ่นยนต์ของประเทศไทย. [Online].
Available : https://waa.inter.nstda.or.th/prs/pub/20171025-Robot-Whitepaper-final%20-%20Cover%20v2.pdf.

Teerasej Jiraphatchandej. 2017. สิ่งที่คุณจะหลงรักใน Visual Studio Code 1.9. [Online].
Available : https://nextflow.in.th/2017/visual-studio-code-1-9-web-developer-will-love/.

Teerasej Jiraphatchandej. 2017. Adobe Experience Design เครื่องมือสำหรับ UX Designer. [Online].
Available : https://nextflow.in.th/2016/adobe-xd-experience-design-thai/.

Web Admin Visual Studio. 2017. Visual Studio Code Document Overview. [Online].

Available : https://code.visualstudio.com/docs.

Web Admin Mindphp. 2017. รู้จักกับ Visual Studio Code. [Online].

Available : https://mindphp.com/บทความ/microsoft/4829-visual-studio-code.html.

Web Admin Rototron. 2017. Raspberry Pi Stepper Motor Tutorial. [Online].

Available : https://www.rototron.info/raspberry-pi-stepper-motor-tutorial/.

Sarin Kesphanich. 2017. แนะนำ VS Code by Microsoft. [Online].

Available : https://medium.com/hezagon/แนะนำ-vs-code-by-microsoft-611bada6b8a5.

MOROTHAI. 2018. หุ่นยนต์อุตสาหกรรม. [Online].

Available : http://www.moro.co.th/ประเภทของหุ่นยนต์อุตสา.

NISMOD. 2018. มินิรีวิว Adobe Xd โปรแกรมออกแบบ UI/UX. [Online].

Available : https://www.blognone.com/node/102395.

Kaisee Boonsa. 2018. IT Management Tools – GitKraken. [Online].

Available : http://www.glurgeek.com/education/management-tools-gitkraken/.

OpenCV team. 2019. OpenCV about. [Online].

Available : https://opencv.org/about/.

scikit-image development team. 2019. Image processing in Python. [Online].

Available : https://scikit-image.org/.

scikit-learn development team. 2019. scikit-learn Machine Learning in Python. [Online].

Available : https://scikit-learn.org/stable/.

Bradski and Kaehler. 2019. Basic Thresholding Operations. [Online].

Available : https://docs.opencv.org/2.4/doc/tutorials/imgproc/threshold/threshold.html.

OpenCV team. 2019. Image Thresholding. [Online].

Available : https://docs.opencv.org/3.4/d7/d4d/tutorial_py_thresholding.html.

OpenCV team. 2019. Morphological Transformations. [Online].

Available : https://docs.opencv.org/3.4/d9/d61/tutorial_py_morphological_ops.html.

OpenCV team. 2019. Sobel Derivatives. [Online].

Available : https://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/sobel_derivatives/sobel_derivatives.html .

OpenCV team. 2019. Canny Edge Detection. [Online].

Available : https://docs.opencv.org/trunk/da/d22/tutorial_py_canny.html.

OpenCV team. 2019. Hough Line Transform. [Online].

Available : https://docs.opencv.org/3.4/d9/db0/tutorial_hough_lines.html.

Mathworks development team. 2019. Edge Detection. [Online].

Available : https://www.mathworks.com/discovery/edge-detection.html.

Robert Fisher. 2019. Hough Transform. [Online].

Available : http://homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm .

Matt Makai. 2019. WebSockets. [Online].

Available : https://www.fullstackpython.com/websockets.html .

Idan Kober. 2019. YOLO Deep Learning. [Online].

Available : https://missinglink.ai/guides/computer-vision/yolo-deep-learning-dont-think-twice .

XGBoost Developers. 2018. XGBoost Documentation. [Online].

Available : https://xgboost.readthedocs.io/en/latest .

Jason Brownlee. 2019. What is Deep Learning. [Online].

Available : https://machinelearningmastery.com/what-is-deep-learning .

Sumit Saha. 2018. A Comprehensive Guide to Convolutional Neural Networks. [Online].

Available : https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 .

Damien Arrachequesne. 2019. What Socket.IO is. [Online].

Available : https://github.com/socketio/socket.io .

Zeke Sikelianos. 2018. Introduction to Node.js. [Online].

Available : https://nodejs.dev/introduction-to-nodejs .