

การศึกษาเบื้องต้นเกี่ยวกับการตรวจจับฉนวนไฟฟ้าด้วยโครงข่ายประสาทเทียม สำหรับหุ่นยนต์ทำความสะอาดฉนวนผ่าน MQTT โดยประมวลผลบน CPU

ผกามาศ วงศ์สาย^{1*} วรากร เลื่องลือวุฒิ^{1,2} พันศักดิ์ เทียนวิบูลย์² กิตติกร วิริยะศาสตร์¹
อุบล รัชสถาพรวัฒนา¹ ชนาธิป ชื่นมนัส¹ และ พันธุ์เทพ แก้วมงคล¹

วันที่รับ 2 กันยายน 2568 วันที่แก้ไข 9 มีนาคม 2569 วันที่ตอบรับ 17 มีนาคม 2569

บทคัดย่อ

การตรวจจับวัตถุ (Object Detection) เป็นกระบวนการที่ใช้เทคนิคจากโครงข่ายประสาทเทียม (Neural Network) ซึ่งได้รับความนิยมอย่างแพร่หลายในงานวิจัยด้านการมองเห็นของหุ่นยนต์ (Robot Vision) งานวิจัยนี้ได้นำเทคนิคดังกล่าวมาประยุกต์ใช้กับหุ่นยนต์ทำความสะอาดฉนวนไฟฟ้า (Insulator Cleaning Robot) โดยระบบจะรับภาพจากกล้องเพื่อตรวจหาพิกัด Bounding Box ของเป้าหมาย ก่อนส่งข้อมูลไปยังหน่วยควบคุม หุ่นยนต์ผ่านการสื่อสารไร้สายแบบ MQTT โดยการทดสอบดำเนินการบน CPU 6-Core Intel Xeon E5 ความเร็ว 3.5 GHz พร้อม RAM ขนาด 16 GB โดยไม่ใช้ GPU ในการประมวลผล ผลการเปรียบเทียบหลายสถาปัตยกรรม โมเดลพบว่า YOLOv11n ให้ประสิทธิภาพดีที่สุด ด้วยค่า mAP@0.5 = 97.6% และ Precision = 98.66% ความเร็วเฉลี่ยของระบบ (รวมทั้งการประมวลผลและส่งข้อมูล) อยู่ที่ 1.61 FPS แม้จะต่ำกว่างานวิจัยก่อนหน้านี้ซึ่งมีความเร็ว 24 - 51 FPS แต่ระบบที่พัฒนาขึ้นมีจุดเด่นในการส่งพิกัดเป้าหมายล่วงหน้า 0.62 วินาที ทำให้หน่วยควบคุมสามารถเคลื่อนที่ตามพิกัดได้โดยไม่ต้องพึ่งพาภาพแบบเรียลไทม์ในทุกขั้นตอน นอกจากนี้ เวลารวมในการประมวลผลและส่งข้อมูลคิดเป็นเพียง 52% ของเวลาทั้งหมดต่อเฟรม แสดงให้เห็นว่าระบบมีความเหมาะสมต่อการใช้งานจริง แม้จะทำงานด้วย CPU เพียงอย่างเดียวก็ตาม

คำสำคัญ: การตรวจจับวัตถุ, โครงข่ายประสาทเทียม, หุ่นยนต์ทำความสะอาด, ฉนวนไฟฟ้า, การประมวลผลบนซีพียู, MQTT

¹ ส่วนงานวิศวกรรมการสื่อสารข้อมูลทางอิเล็กทรอนิกส์และเครือข่ายคอมพิวเตอร์, สถาบันเทคโนโลยีป้องกันประเทศ

² ภาควิชาวิศวกรรมไฟฟ้า, คณะวิศวกรรมศาสตร์, มหาวิทยาลัยเกษตรศาสตร์

* ผู้แต่ง, อีเมล: pakamaj.w@dti.or.th

Preliminary Study on Neural Network-Based Insulator Detection for an Insulator-Cleaning Robot via MQTT with CPU-Based Processing

Pakamaj Wongsai^{1*} Warakorn Luangluewut^{1,2} Phunsak Thiennviboon² Kittakorn Viriyasatr¹
Ubon Thongsatapornwatana¹ Chanatip Chuenmanus¹ and Pantape Kaewmongkol¹

Received 2 September 2025 Received 9 March 2026 Accepted 17 March 2026

Abstract

Object detection is a process that utilizes techniques from neural networks, which have gained significant popularity in robot vision research. This study applies such techniques to an insulator cleaning robot, where images from a camera are processed to determine the target's bounding box coordinates. The coordinates are then transmitted to the robot's control unit via long-range wireless communication using MQTT. The experiments were conducted on a 6-Core Intel Xeon E5 CPU (3.5 GHz) with 16 GB of RAM, without using a GPU for model inference. Among several tested architectures, YOLOv11n delivered the best performance, achieving an mAP@0.5 of 97.6% and a precision of 98.66%. The overall system speed (including both inference and data transmission) averaged 1.61 FPS. While this is slower than previous research reporting 24-51 FPS, the developed system offers a unique advantage: it can transmit target coordinates 0.62 seconds in advance, allowing the control unit to navigate toward the target without relying on real-time image processing at every step. Furthermore, the combined processing and transmission time accounted for only 52% of the total per-frame time, demonstrating the system's suitability for practical applications—even when running solely on a CPU.

Keywords: Object Detection, Neural Network, Cleaning Robot, Insulator, MQTT Communication

¹ Data Communication Division, Defence Technology Institute

² Department of Electrical Engineering, Faculty of Engineering, Kasetsart University

* Corresponding author, E-mail: pakamaj.w@dti.or.th

1. บทนำ

การพัฒนาหุ่นยนต์ได้รับความนิยมอย่างแพร่หลายในภาคอุตสาหกรรม [1] การทำให้ระบบสามารถทำงานได้โดยอัตโนมัติต้องอาศัยองค์ประกอบหลายด้าน ทั้งระบบควบคุม (Control) [2] และระบบการมองเห็นด้วยคอมพิวเตอร์ (Computer Vision) [3] ในการทดลองครั้งนี้มุ่งเน้นที่ส่วนของ Computer Vision โดยนำหลักการตรวจจับวัตถุ (Object Detection) [4] มาประยุกต์ใช้กับหุ่นยนต์ฉีดน้ำทำความสะอาดฉนวนไฟฟ้า (Insulator Cleaning Robot)

หลักการทำงานคือ การใช้ Object Detection เพื่อระบุตำแหน่งของฉนวนไฟฟ้า (Insulator) จากภาพที่ได้ แล้ววัดความไวของระบบ เพื่อหาวิธีให้สามารถทำงานแบบ Real-Time ภายใต้ข้อจำกัดของฮาร์ดแวร์ที่ใช้ ในงานวิจัยที่ผ่านมา มีการนำ Object Detection มาประยุกต์ใช้ร่วมกับแขนกลและหุ่นยนต์อย่างแพร่หลาย [5], [6] โดยส่วนใหญ่ทดสอบบนระบบ GPU ซึ่งให้ความเร็วสูงถึง 24 - 51 FPS อย่างไรก็ตามงานวิจัยเหล่านั้นมักไม่สามารถควบคุมจากระยะไกลได้เนื่องจากไม่มีระบบส่งข้อมูลไร้สาย อีกทั้งยังมีต้นทุนสูงจากการใช้ GPU

ในงานวิจัยนี้จึงประยุกต์ใช้โมเดล YOLO ซึ่งมีความเร็วและความแม่นยำสูง [7] ร่วมกับ MQTT ซึ่งเป็นโพรโตคอลการส่งข้อความแบบน้ำหนักเบา (Lightweight Messaging Protocol) ที่เหมาะสำหรับการสื่อสารระหว่างอุปกรณ์ในระบบ Internet of Things (IoT) โดยข้อมูลตำแหน่งของฉนวนไฟฟ้าที่ตรวจจับได้จะถูกส่งผ่าน MQTT ไปยังระบบควบคุม เพื่อให้แขนกลสามารถเคลื่อนที่ไปยังตำแหน่งของฉนวนได้อย่างแม่นยำ

2. ทฤษฎีที่เกี่ยวข้อง

ปัจจุบันงานด้าน Object Detection มีโมเดลให้เลือกใช้อย่างหลากหลาย อย่างไรก็ตาม งานวิจัยฉบับนี้เลือกใช้ตระกูล YOLO (You Only Look Once) เนื่องจากมีจุดเด่นด้านความเร็วและเหมาะสมต่อการประยุกต์ใช้กับระบบที่ต้องทำงานแบบ Real-time

นอกจากนี้ งานวิจัยยังศึกษาว่า ระดับการขยายขนาดโมเดล (Model Scaling) ของ YOLO ระดับใด

เหมาะสมที่สุดกับระบบที่ได้ออกแบบไว้ ภายใต้ข้อจำกัดด้าน ปริมาณชุดข้อมูลสำหรับฝึก (Training Dataset Size) และ ต้นทุนการคำนวณ (Computational Cost) เนื่องจากโมเดลแต่ละสเกลให้ประสิทธิภาพแตกต่างกันตามลักษณะและขนาดของข้อมูลฝึก หากเลือกโมเดลที่มีขนาดใหญ่เกินความเหมาะสมเมื่อเทียบกับชุดข้อมูลที่มี อาจเพิ่มความเสี่ยงต่อการเกิดโอเวอร์ฟิตติง (Overfitting) และส่งผลให้ประสิทธิภาพในการใช้งานจริงลดลงได้

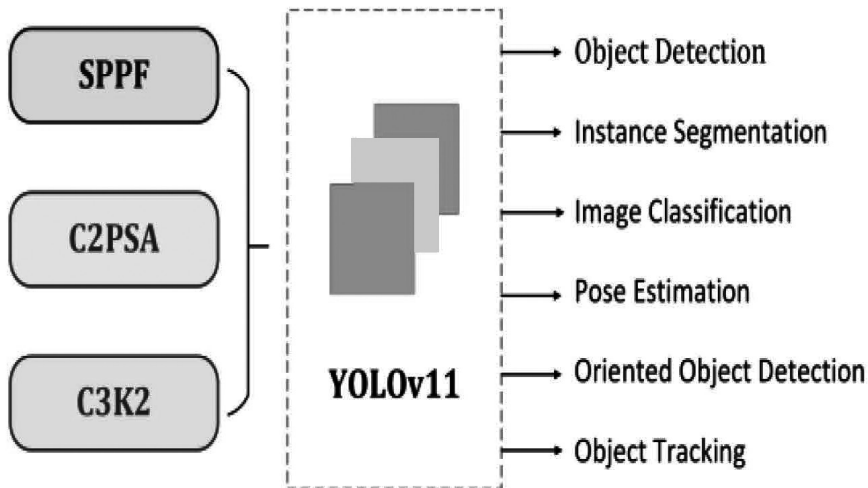
ดังนั้น ในงานนี้จึงทำการเปรียบเทียบโมเดล YOLO หลายสเกลเพื่อคัดเลือกโมเดลที่เหมาะสมที่สุดสำหรับระบบ โดยพิจารณาาร่วมกันทั้งด้าน ความไว และ ค่าหน่วยเวลา (Latency) เพื่อให้ระบบสามารถทำงานได้อย่างมีประสิทธิภาพ และคงค่าดีเลย์ต่ำที่สุด ตามข้อกำหนดของการใช้งานจริง

2.1 การทำ Object Detection ด้วย YOLO

ปัจจุบันการทำ Object Detection มีการพัฒนาโมเดลหลากหลายประเภท เช่น SSD [8], RetinaNet [9] และ YOLO [10] ในงานวิจัยนี้ได้เลือกใช้ YOLO เนื่องจากมีความเร็วและประหยัดทรัพยากรมากกว่าโมเดลอื่น งานวิจัยก่อนหน้านี้มีการพัฒนา YOLO ออกมาหลายรุ่น [11] และถูกประยุกต์ใช้งานในหลายด้าน เช่น Segmentation [12], [13], Classification [14], Pose Estimation [15] และ Object Detection [16] โดยการทดลองในครั้งนี้มุ่งเน้นการใช้งานด้าน Object Detection สำหรับการทดสอบ ได้เลือกใช้ YOLOv11 เพื่อวัดทั้งความแม่นยำและความเร็ว ซึ่ง YOLOv11 [17] มาพร้อมสถาปัตยกรรมที่ปรับปรุงใหม่ โดยในส่วน Backbone ใช้ C3k2 block (Cross-Stage Partial with kernel size 2) แทน C2f แบบเดิม เพื่อลดจำนวนพารามิเตอร์และเพิ่มความเร็ว ด้วยการใช้ Convolution ขนาดเล็กหลายชั้นแทนชั้นใหญ่ เสริมด้วย SPPF (Spatial Pyramid Pooling — Fast) และ C2PSA (Cross-Stage Partial with Spatial Attention) เพื่อเพิ่ม Spatial Attention ให้ระบบสามารถโฟกัสพื้นที่สำคัญ เช่น วัตถุขนาดเล็กหรือวัตถุที่ถูกบัง ส่วน Neck ใช้ C3k2 blocks แทน C2f เพื่อรวมฟีเจอร์หลายระดับได้รวดเร็ว และประหยัดทรัพยากร พร้อมผสม C2PSA เพื่อเพิ่ม

ความสามารถในการตรวจจับในฉากที่ซับซ้อน ส่วน Head ใช้ C3k2 blocks แบบหลายเส้นทาง (Multi-scale Path) เพื่อประมวลผลพีเจอร้อย่างมีประสิทธิภาพ รองรับการทำนายหลายขนาด (Multi-scale inference) สำหรับการตรวจจับวัตถุขนาดเล็ก กลาง และใหญ่ ได้แม่นยำมากขึ้น โดย Head เป็นส่วนที่ทำ Object

Detection โดยตรง ผ่าน Convolution และ Prediction Layers เพื่อทำนายตำแหน่งกรอบวัตถุ (Bounding Box), ค่าความมั่นใจ (Confidence Score) และประเภทของวัตถุ (Class Probability) จากพีเจอรที่ส่งมาจาก Backbone และ Neck ดังแสดงในรูปที่ 1



รูปที่ 1 โครงสร้างของ YOLOv11 [18]

2.2 Model scaling

YOLO (You Only Look Once) [19] เป็นโมเดล Deep Learning ที่ได้รับความนิยมอย่างสูง เหมาะสำหรับการตรวจจับวัตถุแบบเวลาจริง (Real-time) โดยยังคงความแม่นยำในระดับสูง สถาปัตยกรรมของ YOLO แปลงปัญหาการตรวจจับวัตถุให้เป็นปัญหาเชิงถดถอย (Regression Problem) ทำให้สามารถจำแนกประเภท (Classification) และหาตำแหน่ง (Localization) ของวัตถุได้พร้อมกันภายในการประมวลผลเพียงรอบเดียว [19]

ผลลัพธ์ของ YOLO อยู่ในรูปเทนเซอร์ (Tensor Outputs) ซึ่งประกอบด้วยข้อมูลกรอบสี่เหลี่ยม (Bounding Boxes) ที่คาดการณ์ไว้ ได้แก่ ตำแหน่ง ขนาด และค่าความมั่นใจ (Confidence Score) รวมถึงความน่าจะเป็นของคลาส (Conditional Class Probabilities) ของวัตถุที่ตรวจจับได้ เวอร์ชันถัด ๆ มาของ YOLO ได้รับการพัฒนาอย่างต่อเนื่องเพื่อเพิ่ม

ประสิทธิภาพ [17], [18] โดยสามารถจัดการกับฉากที่ซับซ้อนและปรับใช้กับงานตรวจจับที่หลากหลายได้อย่างยืดหยุ่น เฟรมเวิร์ก YOLO จึงถูกพัฒนาต่อขึ้นเพื่อรองรับการปรับแต่งสำหรับงานเฉพาะทาง โดยหนึ่งในเทคนิคสำคัญที่ช่วยเพิ่มประสิทธิภาพคือ การปรับสเกลโมเดล (Model Scaling) [20] ซึ่งเป็นการปรับขนาดสถาปัตยกรรมอย่างเป็นระบบ เพื่อสร้างสมดุลระหว่างความแม่นยำและความเร็ว

การปรับสเกลโมเดลโดยทั่วไปประกอบด้วย

- **การปรับความลึก (Depth scaling):** เพิ่มจำนวนชั้น (Layers) ในโครงข่าย เพื่อให้สามารถเรียนรู้คุณลักษณะที่ซับซ้อนขึ้นผ่าน Convolutional Layers ที่เพิ่มขึ้น [21]
- **การปรับความกว้าง (Width Scaling):** เพิ่มจำนวนหน่วยต่อชั้น เช่น เพิ่มจำนวนตัวกรอง (Convolutional Filters) เพื่อขยายขอบเขตการดึงคุณลักษณะในแต่ละชั้น

ในการทดลองครั้งนี้ ได้ทำการทดสอบการปรับสเกลโมเดล YOLO จำนวน 3 ขนาด ได้แก่ Nano, Small, และ Extra เพื่อครอบคลุมเล็ก-กลาง-ใหญ่สุดภายใต้ข้อจำกัด CPU และดู trade-off latency โดยพบว่าประสิทธิภาพของโมเดลจะเพิ่มขึ้นตามขนาดของโมเดล แต่ความเร็ว (Latency) จะลดลงเมื่อใช้โมเดลที่มีขนาดใหญ่ขึ้น

จากตารางที่ 1 ซึ่งสรุปแนวคิด Model Scaling ของโมเดลที่ผ่านการฝึกบนชุดข้อมูล COCO พบว่าแต่ละโมเดลมี จำนวนพารามิเตอร์ (Parameters) และต้นทุนการคำนวณ (FLOPs) แตกต่างกันอย่างมีนัยสำคัญ อีกทั้งในกระบวนการฝึก โมเดลสามารถประยุกต์ใช้ Data Augmentation ในตัวได้ ส่งผลให้รูปแบบข้อมูลระหว่างการฝึกมีความหลากหลายมากขึ้น

ตารางที่ 1 ประสิทธิภาพของ Model Scaling ในแต่ละตัว [18]

Model	size (pixels)	mAP ^{val} 50-95	Speed CPU ONNX (ms)	Speed T4 TensorRT10 (ms)	params (M)	FLOPs (B)
YOLO11n	640	39.5	56.1 ± 0.8	1.5 ± 0.0	2.6	6.5
YOLO11s	640	47	90.0 ± 1.2	2.5 ± 0.0	9.4	21.5
YOLO11m	640	51.5	183.2 ± 2.0	4.7 ± 0.1	20.1	68
YOLO11l	640	53.4	238.6 ± 1.4	6.2 ± 0.1	25.3	86.9
YOLO11x	640	54.7	462.8 ± 6.7	11.3 ± 0.2	56.9	194.9

โดยภาพรวม แนวโน้มชี้ให้เห็นว่า โมเดลที่มีขนาดใหญ่ขึ้นให้ความแม่นยำสูงขึ้น อย่างไรก็ตามความสามารถในการตรวจจับได้ครบถ้วนหรือความไวอาจลดลงเมื่อขนาดโมเดลเพิ่มขึ้น ซึ่งสะท้อนถึง Trade-off ระหว่างประสิทธิภาพด้านความถูกต้องกับการครอบคลุมการตรวจจับ นอกจากนี้ โมเดลขนาดใหญ่โดยทั่วไปเหมาะกับสถานการณ์ที่มีปริมาณข้อมูลฝึกจำนวนมาก ขณะที่โมเดลขนาดเล็กมักเหมาะกับชุดข้อมูลขนาดเล็กมากกว่า หากเลือกขนาดโมเดลไม่สอดคล้องกับลักษณะและปริมาณข้อมูลอาจนำไปสู่ปัญหาโอเวอร์ฟิตติง (Overfitting) ได้

ดังนั้น งานวิจัยนี้จึงมุ่งประเมินและเปรียบเทียบว่า โมเดลขนาดใดเหมาะสมที่สุดกับข้อจำกัดและเงื่อนไขการใช้งานของระบบที่ได้ออกแบบไว้ เพื่อให้ได้สมดุลระหว่างความแม่นยำ ความไว และต้นทุนการคำนวณในการใช้งานจริง

2.3 การส่งข้อมูลผ่าน MQTT

MQTT (Message Queuing Telemetry Transport) [22] สื่อสารแบบ publish-subscribe

ที่ใช้สถาปัตยกรรมโดยมี Broker ทำหน้าที่เป็นตัวกลางในการรับ-ส่งข้อมูลระหว่าง Publisher และ Subscriber จุดเด่นของ MQTT คือน้ำหนักเบา (Lightweight) ใช้พลังงานต่ำ และเหมาะสำหรับเครือข่ายที่มีแบนด์วิดท์จำกัด ในการทดลองนี้ได้ใช้ MQTT เพื่อส่งข้อมูลผลการตรวจจับวัตถุ (Object Detection) และส่งภาพของฉนวนไฟฟ้า (Insulator) ที่ตรวจจับได้ไปยังระบบควบคุม เพื่อให้หุ่นยนต์สามารถนำพิกัดที่ได้รับไปใช้ในการควบคุมการเคลื่อนที่ได้อย่างแม่นยำ

2.4 Confusion Matrix

Confusion Matrix [23] เป็นเครื่องมือพื้นฐานสำหรับประเมินประสิทธิภาพของ โมเดลจำแนกประเภท (Classification Model) โดยใช้การแสดงผลการทำนายของโมเดลว่าถูกหรือผิดในแต่ละคลาส ซึ่งช่วยให้วิเคราะห์ข้อผิดพลาดและประสิทธิภาพของโมเดลได้อย่างชัดเจน ดังแสดงในรูปที่ 2

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

รูปที่ 2 Confusion Matrix

2.4.1 ความแม่นยำ (Precision) เป็นการวัดความแม่นยำของแบบจำลองจากการพิจารณาแยกทีละผลลัพธ์ โดยคำนวณได้จากสมการที่ (1)

$$\frac{TP_s}{TP_s + FP_s} \quad (1)$$

โดย IoU (Intersection over Union) มักใช้ในงานตรวจจับวัตถุ (Object Detection) หรือการแบ่งส่วนภาพ (Image Segmentation) เพื่อวัดความแม่นยำเชิงพื้นที่ โดยนิยามเป็นอัตราส่วนของพื้นที่ที่ทำนายได้ถูกต้อง (True Positive หรือ Intersection) ต่อพื้นที่รวมทั้งหมด (Union) ของพื้นที่ที่บวกจริง (Actual Positive) กับพื้นที่บวกที่โมเดลทำนายว่าเป็นบวก (Predicted Positive) ดังสมการที่ (2)

$$\frac{TP_s}{TP_s + FP_s + FN_s} \quad (2)$$

2.4.2 Average Precision (AP50) ของแต่ละคลาสดังสมการที่ (3)

$$\int_0^1 P(R) dR \quad (3)$$

คือการคำนวณจริง ใช้วิธีหาพื้นที่ใต้กราฟ Precision-Recall (PR curve) โดยใช้ข้อมูลที่คัดจาก $IoU \geq 0.50$

2.4.3 Mean Average Precision (mAP50)

$$mAP50 = \frac{1}{N} \sum_{n=1}^N AP_{i50} \quad (4)$$

โดย

N = จำนวนคลาสทั้งหมด

AP_{i50} = Average Precision ของคลาสเมื่อ

$IoU \text{ threshold} \geq 0.50$

3. วิธีการดำเนินการ

3.1 การออกแบบระบบหุ่นยนต์ทำความสะอาดฉนวนไฟฟ้า (Insulator Cleaning Robot)

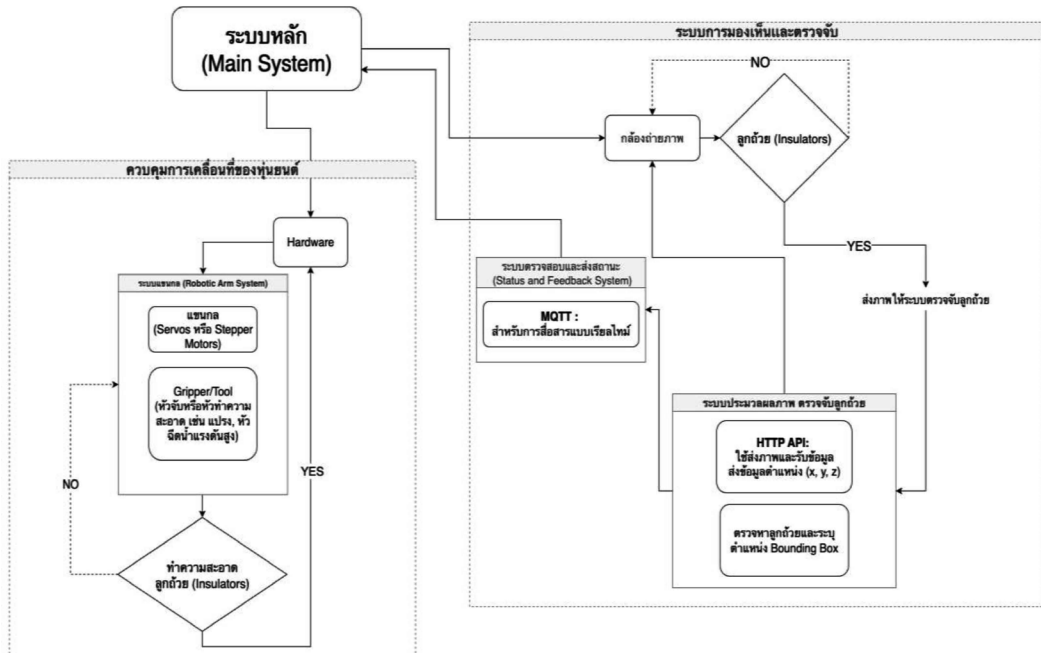
การออกแบบระบบหุ่นยนต์ทำความสะอาดฉนวนไฟฟ้า แบ่งออกเป็น 2 ส่วนหลัก ได้แก่

1. ระบบการมองเห็น (Vision System)

ทำหน้าที่รับภาพจากกล้องและประมวลผลด้วยโมเดล Object Detection เพื่อหาพิกัด bounding box ของฉนวนไฟฟ้า จากนั้นจะส่งข้อมูลพิกัดและภาพที่ตรวจจับได้ผ่านโปรโตคอล MQTT ไปยังระบบหลัก



รูปที่ 3 ตัวอย่างหุ่นยนต์ทำความสะอาด Insulator



รูปที่ 4 โครงสร้างระบบหุ่นยนต์ทำความสะอาด Insulator

2. ระบบควบคุม (Control System) รับข้อมูล พิกัด Bounding Box จากระบบการมองเห็น เพื่อนำไปใช้ควบคุมการเคลื่อนที่ของหุ่นยนต์ให้ตรงกับตำแหน่งของฉนวนไฟฟ้าอย่างแม่นยำ ในการทดลองได้ทำการวัดความเร็ว (Latency) ของการประมวลผลภาพและการส่งข้อมูลจากระบบการมองเห็น เพื่อให้ค่าดังกล่าวสอดคล้องกับความเร็วในการทำงานของระบบควบคุมโครงสร้างโดยรวมของระบบแสดง ดังแสดงในรูปที่ 3 และ รูปที่ 4

3.2 การเลือก Dataset มาทำการทดลอง

ข้อมูลที่ใช้มาทดสอบคือ ข้อมูลจาก Roboflow [24] โดยมีภาพทั้งหมด 1249 ภาพ โดยทำการแบ่งเป็นข้อมูล Train 874 ภาพ Validation 251 ภาพ และข้อมูลที่ Test 124 ภาพโดยข้อมูลชุด Test เราจะนำไปทดสอบของโมเดลทั้งหมด 3 โมเดล และหาความไวที่โมเดลใช้ในการหาพิกัด และความไวในการส่งของข้อมูลโดยใช้ MQTT

3.3 การหาความไวและความแม่นยำของระบบกล้อง และระบบส่งข้อมูล

ในการทดลองด้วย YOLOv11 ได้ทำการปรับ Model Scaling จำนวน 3 ขนาด ได้แก่ Nano, Small

และ Extra เพื่อประเมินความเร็วในการทำงานบน CPU โดยใช้ 6-Core Intel Xeon E5 @ 3.5 GHz พร้อม RAM ขนาด 16 GB และไม่ใช่ GPU ในการประมวลผลโมเดล เนื่องจากข้อจำกัดด้านฮาร์ดแวร์

การวัดประสิทธิภาพประกอบด้วย 2 ส่วน คือ

1. ความเร็วจากการประมวลผลโมเดลในการหาพิกัดของฉนวนไฟฟ้า (Insulator)
2. ความเร็วจากการส่งข้อมูลผ่าน MQTT นอกจากนี้ยังทำการวัดค่า mAP@0.5 เพื่อประเมินความแม่นยำของโมเดลทั้ง 3 ขนาด โดยทดสอบบนชุดข้อมูล Test Set ที่เตรียมไว้

4. ผลการทดลอง

4.1 การทดลองประสิทธิภาพของแต่ละโมเดล (Model)

จากการทดลองเพื่อวัดประสิทธิภาพของ MQTT พบว่าโมเดลแต่ละขนาดให้ค่า mAP@0.5 ใกล้เคียงกัน แต่ YOLOv11n ให้ค่าที่สูงที่สุด เนื่องจากชุดข้อมูลที่เพิ่มความซับซ้อนต่ำ ทำให้ YOLOv11n มีการ overfit น้อยกว่าโมเดลอื่น ๆ นอกจากนี้ ระยะเวลาที่ใช้ในการประมวลผลของโมเดลและการส่งข้อมูลผ่าน MQTT

มีค่าที่ใกล้เคียงกัน โดยรวมคิดเป็น 52% ของเวลาทั้งหมด ต่อเฟรม แสดงว่าระบบสามารถทำงานได้อย่างเหมาะสมบน CPU ซึ่งช่วยประหยัดต้นทุนฮาร์ดแวร์

หากใช้ GPU เพื่อเร่งความเร็วในการประมวลผลระบบอาจทำงานช้าลงเนื่องจากข้อจำกัดของการส่งข้อมูลผ่าน MQTT อย่างไรก็ตาม การนำ MQTT มาใช้ช่วยให้สามารถควบคุมหุ่นยนต์จากระยะไกลระหว่างระบบควบคุมและระบบการมองเห็นได้อย่างมีประสิทธิภาพ ผลการทดลองสรุปดังแสดงในตารางที่ 2

4.2 ประสิทธิภาพ ของ YOLOv11n

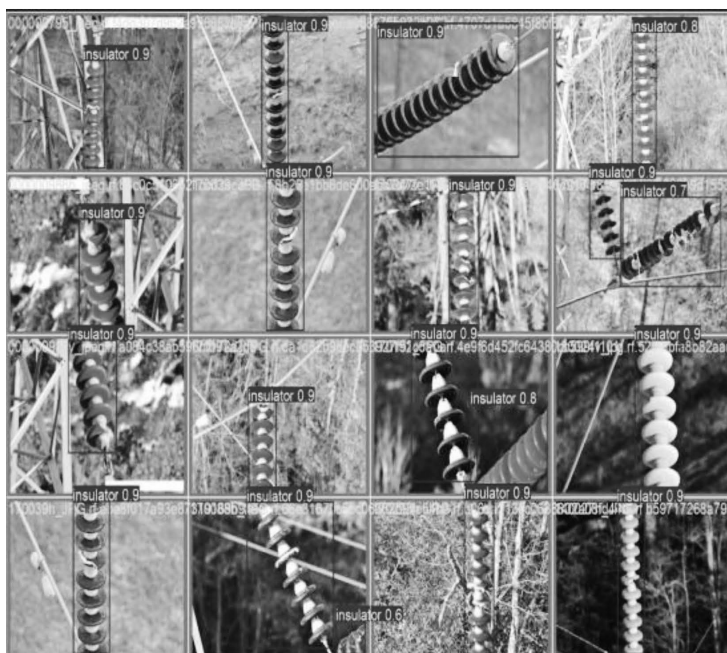
จากการทดลองประสิทธิภาพของแต่ละโมเดลพบว่า YOLOv11n ให้ประสิทธิภาพดีที่สุดทั้งด้านความเร็วและความแม่นยำ ในรูปที่ 5 ซึ่งแสดงการตรวจ

ตารางที่ 2 ประสิทธิภาพของโมเดลแต่ละตัว

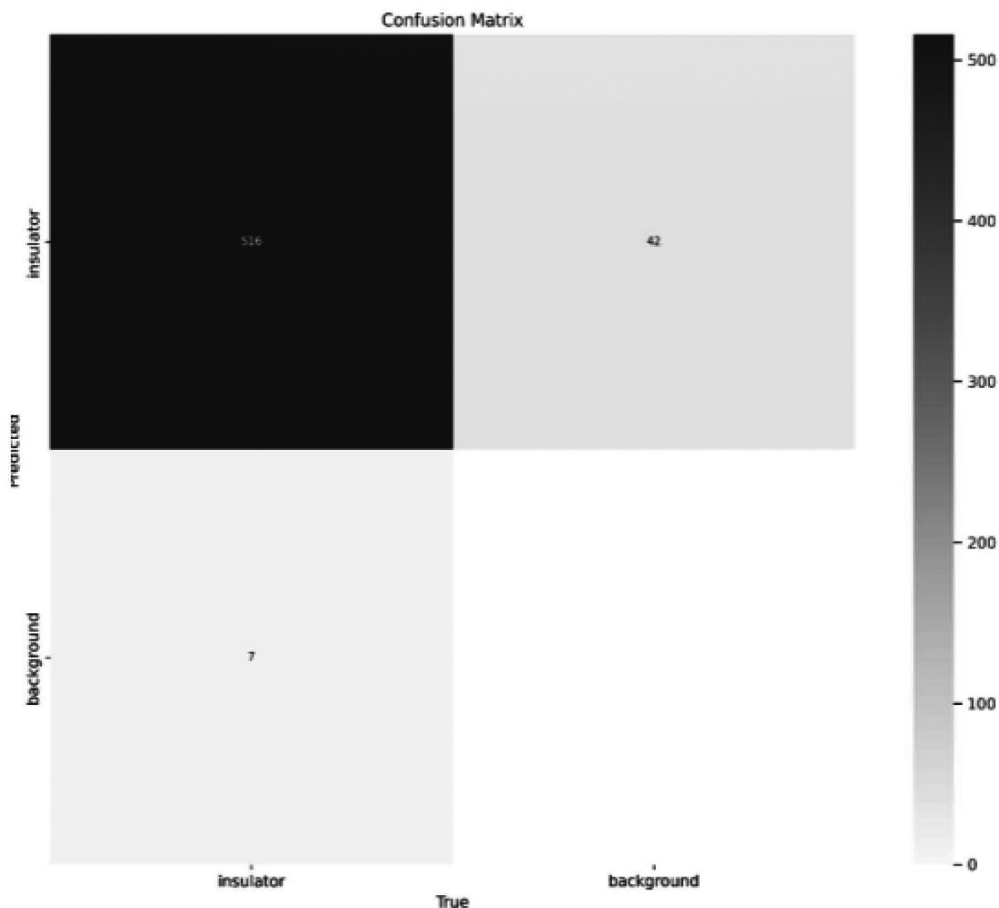
Model	mAP@0.5	Time (Model) [s]	Time (MQTT) [s]	Time Total [s]	FPS Total
YOLOv11n	0.976	0.32	0.30	0.62	1.61
YOLOv11s	0.973	0.44	0.32	0.76	1.32
YOLOv11x	0.974	1.59	0.29	1.88	0.53

จับ Insulator ได้อย่างถูกต้อง และรูปที่ 6 ซึ่งเป็นผลจาก Confusion Matrix โดยมีค่า True Positive จำนวน 516 ตัวอย่าง จากทั้งหมด 565 ตัวอย่าง คิดเป็นความแม่นยำ 98.66% และมีค่า mAP50 สูงถึง 97.6% เมื่อใช้ร่วมกับระบบการมองเห็นมีความไวเท่ากับ 0.62 วินาทีต่อเฟรมซึ่งต่ำกว่า

งานวิจัยก่อนหน้านี้ที่มีความไว 24-51 FPS อย่างไรก็ตาม งานวิจัยก่อนหน้านี้ไม่มีการส่งข้อมูลแบบไร้สายระยะไกลเหมือนระบบที่ออกแบบ โดยระบบจะส่งภาพและพิกัดเป้าหมายไปยังหน่วยควบคุมล่วงหน้า 0.62 วินาที ก่อนที่หน่วยควบคุมจะประมวลผลและขับเคลื่อนตามพิกัด โดยไม่จำเป็นต้องรอข้อมูลภาพแบบเรียลไทม์ในทุกขั้นตอน



รูปที่ 5 ตัวอย่างการตรวจจับ Insulator ด้วย YOLOv11n



รูปที่ 6 Confusion Matrix ของ YOLOv11n

5. สรุปผลการทดลอง

จากการทดสอบโมเดลตรวจจับวัตถุหลายรุ่นพบว่า YOLOv11n ให้ประสิทธิภาพดีที่สุด โดยมีความแม่นยำ (Precision) 98.66%, ค่าความถูกต้องเชิงพื้นที่ (mAP50) 97.6% และจำนวน True Positive 516 ตัวอย่าง จากข้อมูลทั้งหมด 565 ตัวอย่าง ความเร็วการประมวลผลภาพอยู่ที่ 1.61 FPS ซึ่งต่ำกว่างานวิจัยก่อนหน้านี้ที่มีความเร็ว 24-51 FPS อย่างไรก็ตาม ระบบที่พัฒนาขึ้นมีความโดดเด่นตรงที่สามารถส่งข้อมูลภาพและพิกัดเป้าหมายแบบไร้สายระยะไกลได้ และ

ถูกออกแบบให้ส่งพิกัดล่วงหน้า 0.62 วินาที ก่อนการเคลื่อนที่จริง ทำให้หน่วยควบคุมสามารถประมวลผลและขับเคลื่อนตามพิกัดที่ได้รับ โดยไม่ต้องรอข้อมูลภาพแบบเรียลไทม์ในทุกขั้นตอน นอกจากนี้ เวลาที่ใช้ในการประมวลผลของโมเดลและการส่งข้อมูลผ่าน MQTT มีค่าที่ใกล้เคียงกัน โดยรวมคิดเป็นประมาณ 52% ของเวลาทั้งหมดในหนึ่งเฟรม ซึ่งถือว่าเหมาะสมต่อการนำไปใช้งาน เนื่องจากสามารถทำงานได้อย่างมีประสิทธิภาพแม้ใช้เพียง CPU

6. เอกสารอ้างอิง

- [1] J. Arents and M. Greitans, "Smart industrial robot control trends, challenges and opportunities within manufacturing," *Applied Sciences*, vol. 12, no. 2, p. 937, 2022.
- [2] M. Casini, D. Prattichizzo, and A. Vicino, "The automatic control telelab," *IEEE Control Systems Magazine*, vol. 24, no. 3, pp. 36-44, 2004.
- [3] J. Qi et al., "Computer vision-based hand gesture recognition for human-robot interaction: a review," *Complex & Intelligent Systems*, vol. 10, no. 1, pp. 1581-1606, 2024.
- [4] Z. Zou et al., "Object detection in 20 years: A survey," *Proceedings of the IEEE*, vol. 111, no. 3, pp. 257-276, 2023.
- [5] C. Ju et al., "Speed-FairMOT: multi-class multi-object tracking for real-time manipulation," *Signal, Image and Video Processing*, 2025.
- [6] Y. Ye et al., "Dynamic and real-time object detection based on deep learning for home service robots," *Sensors*, vol. 23, no. 23, p. 9482, 2023.
- [7] L. He et al., "Research and application of YOLOv11-based object segmentation," *Buildings*, vol. 14, no. 12, p. 3777, 2024.
- [8] S. Zhai et al., "DF-SSD: An improved SSD object detection algorithm," *IEEE Access*, vol. 8, pp. 24344-24357, 2020.
- [9] Y. Wang et al., "Automatic ship detection based on RetinaNet," *Remote Sensing*, vol. 11, no. 5, p. 531, 2019.
- [10] P. Jiang et al., "A review of YOLO algorithm developments," *Procedia Computer Science*, vol. 199, pp. 1066-1073, 2022.
- [11] M. Sportelli et al., "Evaluation of YOLO object detectors for weed detection," *Applied Sciences*, vol. 13, no. 14, p. 8502, 2023.
- [12] W. Luangluewut et al., "Performance Comparison of LRT and YOLO," *Defence Technology Academic Journal*, 2025.
- [13] C. Kang and S. Kim, "Real-time object detection and segmentation technology," *JMST Advances*, vol. 5, no. 2, pp. 69-76, 2023.
- [14] T. Diwan et al., "Object detection using YOLO," *Multimedia Tools and Applications*, vol. 82, no. 6, pp. 9243-9275, 2023.
- [15] D. Maji et al., "YOLO-pose," *CVPR*, pp. 2637-2646, 2022.
- [16] W. Luangluewut et al., "Domain Adaptation for Drone Detection," *ICAIC*, 2025.
- [17] R. Khanam and M. Hussain, "YOLOv11: An overview," *arXiv*, 2024.
- [18] G. Jocher et al., "Ultralytics YOLO11," *GitHub*, 2024.
- [19] J. Redmon et al., "You Only Look Once," *CVPR*, pp. 779-788, 2016.
- [20] M. Tan et al., "EfficientDet," *CVPR*, pp. 10781-10790, 2020.
- [21] B. Mehlig, *Machine Learning with Neural Networks*. Cambridge University Press, 2021.
- [22] OASIS, "MQTT Version 3.1.1," 2014.
- [23] S. Visa et al., "Confusion Matrix-based Feature Selection," *CEUR Workshop Proc.*, 2011.
- [24] Roboflow, "Insulator Detection Dataset," 2023.