Chulalongkorn University
Chula Digital Collections

Chulalongkorn University Theses and Dissertations (Chula ETD)

2022

การพัฒนาระบบป้องกันการชนสำหรับยานยนต์อัตโนมัติโดยใช้การตรวจจับวัตถุด้วยกล้อง ร่วมกับการสื่อสารระหว่างยานพาหนะอัตโนมัติและโครงสร้างพื้นฐานผ่านเครือข่ายเซลลูลาร์

วริทธิ์ ดิษยศรินทร์
คณะวิศวกรรมศาสตร์

Follow this and additional works at: https://digital.car.chula.ac.th/chulaetd

Part of the Applied Mechanics Commons, and the Engineering Mechanics Commons

Recommended Citation

ดิษยศรินทร์, วริทธิ์, "การพัฒนาระบบป้องกันการชนสำหรับยานยนต์อัตโนมัติโดยใช้การตรวจจับวัตถุด้วยกล้อง ร่วมกับการสื่อสารระหว่างยานพาหนะอัตโนมัติและโครงสร้างพื้นฐานผ่านเครือข่ายเซลลูลาร์" (2022). *Chulalongkorn University Theses and Dissertations (Chula ETD)*. 6443.
https://digital.car.chula.ac.th/chulaetd/6443

This Thesis is brought to you for free and open access by Chula Digital Collections. It has been accepted for inclusion in Chulalongkorn University Theses and Dissertations (Chula ETD) by an authorized administrator of Chula Digital Collections. For more information, please contact ChulaDC@car.chula.ac.th.

การพัฒนาระบบป้องกันการชนสำหรับยานยนต์อัตโนมัติโดยใช้การตรวจจับวัตถุด้วยกล้อง ร่วมกับการสื่อสารระหว่างยานพาหนะอัตโนมัติและโครงสร้างพื้นฐานผ่านเครือข่ายเซลลูลาร์

นายวริทธิ์ ดิษยครินทร์

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาระบบกายภาพที่เชื่อมประสานด้วยเครือข่ายไซเบอร์ ภาควิชาวิศวกรรมเครื่องกล
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2565

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

Development of a Collision Avoidance System for Autonomous Vehicles by using Camera-based Object Detection with cellular C-V2I

Mr. Warit Ditsayakarin

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering in Cyber-Physical System
Department of Mechanical Engineering
FACULTY OF ENGINEERING
Chulalongkorn University
Academic Year 2022

Copyright of Chulalongkorn University

หัวข้อวิทยานิพนธ์	การพัฒนาระบบป้องกันการชนสำหรับยานยนต์อัตโนมัติ โดยใช้การตรวจจับวัตถุด้วยกล้อง ร่วมกับการสื่อสารระหว่างยานพาหนะอัตโนมัติและโครงสร้างพื้นฐานผ่านเครือข่ายเซลลูลาร์
โดย	นายวริทธิ์ ดิษยครินทร์
สาขาวิชา	ระบบกายภาพที่เชื่อมประสานด้วยเครือข่ายไซเบอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	ผู้ช่วยศาสตราจารย์ ดร.นักสิทธ์ นุ่มวงษ์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์

(ศาสตราจารย์ ดร.สุพจน์ เตชวรสินสกุล)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.สัณหพศ จันทรานุวัฒน์)

..... อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(ผู้ช่วยศาสตราจารย์ ดร.นักสิทธ์ นุ่มวงษ์)

..... กรรมการ

(รองศาสตราจารย์ ดร.รัชทิน จันทร์เจริญ)

..... กรรมการภายนอกมหาวิทยาลัย

(ดร.ปาษาณ กุลวานิช)

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

วริทธิ์ ดิษยครินทร์ : การพัฒนาระบบป้องกันการชนสำหรับยานยนต์อัตโนมัติโดยใช้การตรวจจับวัตถุด้วยกล้อง ร่วมกับการสื่อสารระหว่างยานพาหนะอัตโนมัติและโครงสร้างพื้นฐานผ่านเครือข่ายเซลลูลาร์. (Development of a Collision Avoidance System for Autonomous Vehicles by using Camera-based Object Detection with cellular C-V2I) อ.ที่ปรึกษาหลัก : ผศ. ดร.นักสิทธ์ นุ่มวงษ์

จากการที่ศูนย์วิจัยยานยนต์อัจฉริยะได้ทำการทดสอบยานยนต์อัตโนมัติพบว่าจำเป็นต้องขับเคลื่อนเข้าสู่ทางร่วมที่ไม่มีสัญญาณเตือนและซึ่งมีสิ่งกีดขวางบริเวณในการใช้งานเซนเซอร์ ซึ่งยานยนต์อัตโนมัติจำเป็นต้องทำการทดสอบร่วมกับยานยนต์คนขับเคลื่อนบนท้องถนน จึงจำเป็นต้องทำการติดตั้งระบบตรวจจับสำหรับลดความเสี่ยงในการเกิดอุบัติเหตุ ในงานวิจัยนี้จัดทำเพื่อพัฒนาระบบ C-V2I สำหรับกระบวนการตรวจจับยานยนต์คนขับที่เคลื่อนที่ภายในทางหลักและส่งตัวแปรเข้าสู่กระบวนการการตัดสินใจของยานยนต์อัตโนมัติเนื่องจากยานยนต์อัตโนมัติที่เคลื่อนที่เข้าสู่ทางโทไม่สามาถตรวจจับและรับรู้ได้ โดยทดลองในบริเวณหน้าคณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ซึ่งเป็นตัวอย่างของทางร่วมที่ไม่มีสัญญาณเตือนและเป็นทางที่ใช้สำหรับกระบวนการทดลองยานยนต์อัตโนมัติเพื่อที่โดยในระบบนี้จะแบ่งออกเป็น 3 ส่วนคือ ระบบการตรวจจับและติดตามวัตถุ ระบบการสื่อสารระหว่างยานยนต์อัตโนมัติและ ระบบการตัดสินใจของยานยนต์อัตโนมัติ ซึ่งการทำงานของระบบสามารถอธิบายอย่างง่ายโดยเมื่อยานยนต์อัตโนมัติเคลื่อนที่เข้าสู่บริเวณที่กำหนดไว้จะทำการรับค่าจากระบบตรวจจับซึ่งในงานวิจัยนี้ทำการทดลองโดยใช้กล้อง 2 มิติ ซึ่งระบบตรวจจับและติดตามวัตถุด้วยกล้องโดยในระบบนี้จะทำการส่งตัวแปรที่ได้จากระบบตรวจจับและติดตามวัตถุเข้าสู่กระบวนการประมาณหาค่าตำแหน่งของวัตถุที่เคลื่อนที่บนระนาบของถนนโดยค่าที่ ต่อมาค่าตัวแปรที่ได้จากระบบตรวจจับทจะถูกส่งเข้าสู่ระบบของยานยนต์อัติโนมัติผ่านระบบสื่อสารด้วยผ่านระบบ MQTTเพื่อส่งค่าสู่กระบวนการตัดสินใจของยานยนต์อัตโนมัติ การทดลองพบว่ากล้อง สามารถวัดระยะทางและความเร็ว ของยานยนต์ที่ต้องการตรวจจับได้ด้วยระยะ 30 เมตร และได้ทำการนำค่าตัวแปรมาคำนวนหาค่าระยะปลอดภัยของยานยนต์อัตโนมัติจากระบบจำลอง

สาขาวิชา	ระบบกายภาพที่เชื่อมประสานด้วยเครือข่ายไซเบอร์	ลายมือชื่อนิสิต .....
ปีการศึกษา	2565	ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

6470076821 : MAJOR CYBER-PHYSICAL SYSTEM

KEYWORD: c-v2i, 5G, Autonomous Vehicles, Merging Road, Image processing, Computer vision, Object, Object Tracking

Warit Ditsayakarin : Development of a Collision Avoidance System for Autonomous Vehicles by using Camera-based Object Detection with cellular C-V2I. Advisor: Asst. Prof. NUKSIT NOOMWONGS

The research conducted by the Intelligent Vehicle Research Center revealed the necessity of testing autonomous vehicles at intersections without traffic signals and obstacles alongside human-driven vehicles on the main road. To address this, a detection system was developed to minimize the risk of accidents. This research focuses on the development of a C-V2I (Connected Vehicle to Infrastructure) system for detecting human-driven vehicles on the main road and incorporating the detected variables into the decision-making process of autonomous vehicles. The experiments were conducted at Chulalongkorn University's Faculty of Engineering, which represents a signal-less and obstacle-filled intersection used for autonomous vehicle testing. The system consists of three components: object detection and tracking, communication between autonomous vehicles, and decision-making by autonomous vehicles. The system utilizes a 2D camera for object detection and tracking. The detected object variables are used to estimate the position of moving objects on the road surface. These variables are then transmitted to the autonomous vehicle system through the MQTT communication system, providing input for the decision-making process. The experiments demonstrated the camera's capability to measure the distance and speed of detected vehicles within a 30-meter range. These calculated variables were utilized to determine the safe distance of the autonomous vehicle in the simulation system.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

Field of Study:	Cyber-Physical System	Student's Signature .....
Academic Year:	2022	Advisor's Signature .....

กิตติกรรมประกาศ

วิทยานิพนธ์เล่มนี้สามารถสำเร็จลุล่วงได้อย่างสมบูรณ์ด้วยความกรุณาช่วยเหลือจาก ผศ.ดร. นักสิทธ์ นุ่มวงษ์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ซึ่งได้ทำการสนับสนุนและให้คำปรึกษาและสละเวลาให้ข้าพเจ้าในระหว่างการทำวิทยานิพนธ์มาโดยตลอด ข้าพเจ้าต้องขอขอบคุณสมาชิกของ Smart Mobility Research Center ทุกท่านที่ช่วยสนับสนุนและช่วยเหลือในการทดลองงานวิจัยบนท้องถนนจริง ข้าพเจ้าขอขอบคุณ นายกันตวัชร์ ชัยประภา ที่ได้ให้ความช่วยเหลือในด้านการออกแบบและสร้างโปรแกรมสำหรับการทดลอง ข้าพเจ้าขอขอบคุณ นาย ณัชนน กิจประมงศรี และ นางสาว วรดา ขวัญเจริญ ที่ได้ช่วยสละเวลาอันมีค่ามาเพื่อช่วยขับขี่ยานยนต์สำหรับทำการทดลองและช่วยเหลือในกระบวนการเขียนรูปเล่มวิทยานิพนธ์ สุดท้ายนี้ข้าพเจ้าขอขอบคุณครอบครัวของข้าพเจ้าที่ให้การสนับสนุนในการศึกษาต่อในระดับมหาบัณฑิตศึกษา และอยู่เคียงข้างกันตลอดจนงานวิทยานิพนธ์นี้ประสบความสำเร็จได้

วริทธิ์ ดิษยครินทร์

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

สารบัญ

หน้า

..... ค

บทคัดย่อภาษาไทย ..... ค

..... ง

บทคัดย่อภาษาอังกฤษ ..... ง

กิตติกรรมประกาศ ..... จ

สารบัญ ..... ฉ

สารบัญตาราง ..... ฎ

สารบัญรูปภาพ ..... ฐ

บทที่ 1 บทนำ 1

1.1 ที่มาและความสำคัญของปัญหา 1

1.2 วัตถุประสงค์ของการวิจัย 2

1.3 ขอบเขตของการวิจัย 2

1.4 ประโยชน์ที่คาดว่าจะได้รับ 3

1.5 ขั้นตอนการวิจัย 3

1.6 แผนการดำเนินงาน 4

บทที่ 2 แนวคิดทฤษฎีและงานวิจัยที่เกี่ยวข้อง 5

2.1 Detection and Tracking System 5

2.1.1 Object detection theory 5

2.1.1.1 Neural network 5

2.1.1.2 การเรียนรู้เชิงลึก (Deep learning) 6

2.1.1.3 Convolutional neural network (CNN) 7

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2.1.1.5 Computer vision 9

2.1.1.5.1 การตรวจจับวัตถุ (Object Detection) 9

2.1.1.5.2 การจำแนกวัตถุ (Object Classification) 9

2.1.1.5.3 การตัดแยกวัตถุ (Object Segmentation) 10

2.1.1.4 CNN-based Object detection design 10

2.1.1.4.1 Two-stage models 10

2.1.1.4.2 One-state models 11

2.1.1.5การประเมินประสิทธิภาพของการตรวจจับด้วยวัตถุ 12

2.1.1.5.1การประเมินตำแหน่งสามารถวัดได้โดยจากค่าอัตราส่วน Intersection over Union (IoU) 12

2.1.1.5.2การประเมินความสามารถในการจำแนกวัตถุ 13

2.1.2 Object Tracking 15

2.1.2.1 Single Object Tracker 15

2.1.2.2 Multiple Object Tracker (MOT) 16

2.1.3 Position Estimation 16

2.1.3.1 Camera model 16

2.1.4 ArUco Marker 18

2.2 Communication System 20

2.2.1 LTE/4G/5G: Long-Term Evolution (LTE)[5] 20

2.2.2 DSRC (Dedicated short-range communications) [5] 20

2.2.3 C-V2X[5] 20

2.2.4 MQTT (Message Queue Telemetry Transport) 21

2.3 Decision system 21

2.3.1 Time to Collision 21

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

2.3.2 Reaction time (tr) 23

บทที่ 3 แนวคิดการออกแบบระบบป้องกันการชนสำหรับยานยนต์อัตโนมัติ ร่วมกับการสื่อสารระหว่างยานพาหนะและโครงสร้างพื้นฐานผ่านเครือข่าย 5G 24

3.1 เครื่องมือที่ใช้ในการทดลอง 24

3.2 วิธีการทดลอง 28

3.2.1 ระบบตรวจจับและติดตามวัตถุ 29

3.2.1.1 ระบบการตรวจจับวัตถุ (Detection System) 29

3.2.1.1.1 ซอฟแวร์ 30

3.2.1.1.2 ผังการทำงานส่วนระบบการตรวจจับวัตถุ 34

3.2.1.1.5 ลักษณะการทำงานของระบบตรวจจับวัตถุ 34

3.2.1.2 ระบบติดตามวัตถุ (Object Tracking) 35

3.2.1.2.1 Deep SORT Tracking 36

3.2.1.2.2 กระบวนการทำงานของระบบติดตามวัตถุ 36

3.2.1.3 ระบบการประมวลผลหาค่าตำแหน่งและความเร็วของวัตถุ (Position and Velocity Estimation) 38

3.2.1.3.1 สมการการหาระยะของวัตถุจากกล้องสองมิติ 38

3.2.1.3.2 ทดลองสร้างสมการระนาบเพื่อนำไปใช้สร้างระนาบของถนน.....40

3.2.1.3.3 กระบวนการคำนวณความเร็วของวัตถุ 44

3.2.1.3.4 การทดสอบระบบตรวจจับและติดตามวัตถุบนสภาพแวดล้อมจริง 45

การตรวจสอบการวัดระยะของ ArUco 46

การหาระยะเริ่มต้นของ Merging Point เทียบกับกล้อง 47

การทดสอบวัดระยะของวัตถุที่ตรวจจับได้ 48

การวัดค่าความเร็วของวัตถุที่ได้จากกระบวนการตรวจจับวัตถุ ...49

3.2.2 ระบบสื่อสารระหว่างรถและโครงสร้างพื้นฐาน 49

3.2.2.1 ทดสอบการใช้งาน Mosquitto MQTT 49

3.2.2.1.1 กระบวนการสร้าง sever สำหรับการใช้งาน MQTT.....49

3.2.2.1.2 กระบวนการสร้าง Node สำหรับ Publish ข้อความ 50

3.2.2.1.3 กระบวนการสร้าง Node สำหรับกระบวนการรับข้อความ.....50

3.2.2.1 วิธีการทดสอบ 51

3.2.3 ระบบการตัดสินใจของยานยนต์อัตโนมัติ 52

3.2.3.1 ตัวแปรควบคุม และ คุณสมบัติ Turing T2 ที่นำมาประกอบดารคำนวน 52

3.2.3.1.1 ค่าตัวแปรคุณสมบัติของ T2 52

3.2.3.1.2 ค่าตัวแปรที่กำหนดเพื่อเพิ่มความปลอดภัยให้กับระบบการตัดสินใจของรถอัตโนมัติ.....54

3.2.3.1.3 ค่าตัวแปรของตำแหน่ง Conflict Area.....54

3.2.3.2 แผนภาพแสดงระยะทางในการทำงานของระบบระหว่างรถอัตโนมัติ ยานพาหนะที่ตรวจจับได้และ Conflict Area.....55

3.2.3.3 อัลกอริทึมในกระบวนการตัดสินใจของรถอัตโนมัติ 56

3.2.3.3.1 การคำนวนค่า TTC 56

3.2.3.3.2 แผนผังแสดงการทำงานของระบบหลีกเลี่ยงการชน 57

3.2.3.4 กำหนดสถานการณ์ในการทดสอบแบบจำลองรถอัติโนมัติจากกระบวนการตรวจจับวัตถุ 57

3.2.3.5 รูปแบบการใช้งานระบบ Simulation.....58

บทที่ 4 ผลการวิจัย 59

4.1 ระบบการตรวจจับและติดตามวัตถุ 59

4.1.1 ระบบการตรวจจับวัตถุ 59

4.1.2 ระบบการติดตามวัตถุ 60

4.1.3 ระบบการประมาณหาระยะทางของวัตถุ 60

4.1.3.1 กระบวนการหาพิกัดของ ArUco Marker 60

4.1.3.2) กระบวนการสร้างตัวแปรจาก ArUco Marker.....61

4.1.3.2.1 ค่าตัวแปรของสมการระนาบในพื้นที่จริง.....61

4.1.3.2.2 ค่าตัวแปรตำแหน่ง Merging Point ในบริเวณพื้นที่จริง.....62

4.1.3 3) กระบวนการหาตำแหน่งของยานพาหนะที่ได้ทำการตรวจจับโดยเทียบกับตำแหน่งของยานพาหนะบนถนน 62

4.1.3.3.1 ผลการทดสอบหาระยะของวัตถุบริเวณพื้นที่ทดสอบ 62

4.1.3.3.2 ผลการทดสอบหาระยะทางของยานยนต์ในบริเวณพื้นที่ทดลองจริง 64

4.1.5 เวลาทั้งหมดในการทำงานของระบบตรวจจับและติดตามเพื่อคำนวนหาระยะทางและความเร็วของยานยนต์.....65

4.1.4 การคำนวณหาค่าความเร็ว 65

4.2 ระบบการส่งข้อมูลผ่าน MQTT 69

4.2.1 ผลการทำงานของระบบการส่งข้อมูลผ่าน MQTT Broker.....69

4.2.2 ค่าเวลาที่วัดได้จากการส่งข้อมูลผ่าน MQTT Broker ด้วยเครือข่ายเซลลูลาร์ 69

4.3 ระบบ Simulation.....70

4.3.1 ค่าผลลัพธ์จากกระบวนการ Simulation 70

4.3.2 ผลการทดลอง Simulation จากการรับค่าจากระบบตรวจยานยนต์ในขอบเขตการตรวจจับ 20 เมตร 71

บทที่ 5 สรุปผลการทดลอง 75

5.1 ปัญหาที่พบ 76

5.2 ข้อเสนอแนะ 76

บรรณานุกรม 78

ภาคผนวก 81

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

โปรแกรมสำหรับการทดลอง 81
โปรแกรมสำหรับสร้างระนาบของถนนด้วย ArUco Marker ด้วยภาษา Python 82
โปรแกรมในการตรวจจับวัตถุ 84
โปรแกรมในการรับ message MQTT 88
โปรแกรมในกระบวนการ Simulation 88
ประวัติผู้เขียน 94

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

สารบัญตาราง

หน้า

ตารางที่ 1 ความสามารถในการตรวจจับวัตถุจากการสำรวจของคุณ S.S.A.[17] 15

ตารางที่ 2 แสดงคุณสมบัติของกล้อง Logitech BRIO ULTRA HD PRO BUSINESS WEBCAM 24

ตารางที่ 3 คุณสมบัติของ Notebook Acer Nitro 5 AN515-55DM.....25

ตารางที่ 4 คุณสมบัติของ HUAWEI AIS5G CPE 26

ตารางที่ 5 คุณสมบัติของ OPAL T2 27

ตารางที่ 6 ค่าความสามารถของ YOLOv8 .ในทุก Model[28] 32

ตารางที่ 7 ตัวแปรของรถอัตโนมัติ T2 สำหรับใช้ในกระบวนการตัดสินใจของรถอัตโนมัติ 53

ตารางที่ 8 ตารางแสดงตัวแปร Lane Balance และค่า Clearance 54

ตารางที่ 9 ผลความเร็วในการทดลองกระบวนการติดตามวัตถุด้วย YOLOv8 สามโมเดล 59

ตารางที่ 10 ผลระยะทางที่ได้ทำการวัดโดยเครื่องมือวัดเปรียบเทียบกับ ค่าระยะที่ได้จากการตรวจจับ ArUco Marker 61

ตารางที่ 11 ค่าเฉลี่ยระยะทางของยานพาหนะจากกระบวนการประมาณตำแหน่งจากภาพ.....64

ตารางที่ 12 ผลการวัดระยะในสภาพแวดล้อมจริงเปรียบเทียบกับค่าระยะที่ทำการวัดกำหนดไว้....65

ตารางที่ 13 ผลการทดลองเปรียบเทียบค่าความเร็วเฉลี่ยที่วัดค่าได้จากการระบวนการติดตามด้วย GPS และกระบวนตามติดตามวัตถุด้วยภาพ 66

สารบัญรูปภาพ

หน้า

รูปที่ 1 พื้นที่ทดสอบบริเวณหน้าคณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย.....2

รูปที่ 2 โครงสร้างของเพอร์เซปตรอน[6] 5

รูปที่ 3 กราฟเปรียบเทียบระนาบตัดสินใจระหว่างการใช้เพอร์เซปตรอน 1 หน่วยและสองหน่วย....6

รูปที่ 4 ตัวอย่างโครงสร้างของกระบวนการเรียนรู้เชิงลึก[7] 6

รูปที่ 5 ระนาบตัดสินใจเมื่อ Hidden layer เพิ่มขึ้น[8] 7

รูปที่ 6 ตัวอย่างโครงสร้าง Convolutional neural network[9].....7

รูปที่ 7 การเปรียบเทียบระหว่าง Fully-connected layer และ Convolution layer[11] 8

รูปที่ 8 ตัวอย่างกระบวนการคำนวณ Convolve ด้วย filter matrix ภายใน Convolution layer[10].....8

รูปที่ 9 ชนิดและลักษณะของ Activation Function ที่มีการนำมาใช้ในปัจจุบัน[12] 9

รูปที่ 10 ตัวอย่างแสดงข้อมูลที่ได้รับจากกระบวนการ Detection[13] 9

รูปที่ 11 ค่าตัวแปลที่ได้จากกระบวนการ Image Classificationผ.....10

รูปที่ 12 ตัวอย่างของกระบวนการ Object Segmentation.....10

รูปที่ 13 กระบวนการ Region Proposal Network (RPN).....11

รูปที่ 14 กระบวนการทำงานของ Two-state Object Detection[14] 11

รูปที่ 15 กระบวนการทำงานของ One-stage Object detection[14].....12

รูปที่ 16 การใช้กระบวนการ IoU ประเมินผล Bounding Box[15].....13

รูปที่ 17 ตัวอย่างค่า TP FP FN TN.....13

รูปที่ 18 ตัวอย่างพิกัดรูปภาพและวัตถุจริง 16

รูปที่ 19 ArUco Marker 4*4 19

รูปที่ 20 กระบวนการทำงานและการส่งข้อมูลของ MQTT[21].....21

รูปที่ 21 วิธีการคำนวณค่า TTC โดยกำหนดให้รถมีขนาดเป็นจุด[23].....22

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

รูปที่ 22 ตำแหน่งรถและถนนที่ทำมุมกันน้อยกว่า 90 องศา[23].....22

รูปที่ 23 กราฟแสดงความสัมพันธ์ ระหว่างเวลากับระยะทางของยานพาหนะทั้งสองคัน[23].....23

รูปที่ 24 กล้อง Logitech C920E WEBCAM.....24

รูปที่ 25 Notebook Acer Nitro 5 AN515-55DM.....25

รูปที่ 26 HUAWEI AIS5G CPE 26

รูปที่ 27 OPAL T2 27

รูปที่ 28 ทางเอกและทางโทบริเวณหน้าคณะวิศวกรรมศาสตร์ จุฬาลงกรณ์ มหาวิทยาลัย.....28

รูปที่ 29 System Architecture ของระบบป้องกันการชนสำหรับยานยนต์อัตโนมัติโดยใช้การตรวจจับวัตถุด้วยกล้อง ร่วมกับการสื่อสารระหว่างยานพาหนะอัตโนมัติและโครงสร้างพื้นฐานผ่านเครือข่าย 5G.....28

รูปที่ 30 Roadside system architecture 29

รูปที่ 31 กราฟแสดงความสามารถในการตรวจจับวัตถุของแต่ละโมเดลในปัจจุบัน[25] 30

รูปที่ 32 การพัฒนาของกระบวนการตรวจจับวัตถุในปัจจุบัน[25].....31

รูปที่ 33 โครงสร้างของ YOLOv8 ในปัจจุบัน[27].....32

รูปที่ 34 การเปรียบเทียบค่าความแม่นยำของ Yolov5 Yolov7 และ Yolov8 และปริมาณการนำไปใช้ในงานต่างๆในปัจจุบัน[29] 33

รูปที่ 35 แผนผังการทำงานของระบบตรวจจับวัตถุ.....34

รูปที่ 36 การตีกรอบบนรูปภาพเพื่อทำให้สามารถตรวจจับได้ภายในบริเวณที่สนใจตรวจจับ.....35

รูปที่ 37 แผนผังแสดงข้อมูลที่นำเข้าจากระบบตรวจจับเข้าสู่ระบบติดตามวัตถุ 35

รูปที่ 38 แผนผังแสดงวิธีการใช้งานระบบติดตามวัตถุ 37

รูปที่ 39 ตำแหน่งสำหรับหาจุดเพื่อนำเข้าสู่กระบวนการหาค่าตำแหน่ง 37

รูปที่ 40 ตัวอย่างในการตรวจสอบหลักการของสมการของกล้อง 2 มิติ 39

รูปที่ 41 ค่าชนิดของ ArUco Marker ที่นำมาใช้ในการทดลอง 40

รูปที่ 42 แนวความคิดในการสร้างสมการ Plane จาก ArUco Marker จำนวนสามพิกัด.....41

รูปที่ 43 ตัวอย่างการทดสอบในการสร้างสมการระนาบจาก ArUco Marker 41

รูปที่ 44 ตัวอย่างในการเก็ยค่าสมดารระนาบ 42

รูปที่ 45 วิธีการเก็บค่าสัมประสิทธ์ของสมการระนาบ.....42

รูปที่ 46 รูปแบบหลักการในการระบุตำแหน่งของวัตถุอย่างง่าย 42

รูปที่ 47 ทดสอบกระบวนการหาตำแหน่งของขวดในระยะ 74 เซนติเมตร 43

รูปที่ 48 ขั้นตอนในการหาตำแหน่งของวัตถุตัวอย่าง.....43

รูปที่ 49 ผังการทำงานของข้อมูลและโปรแกรมที่ใช้สำหรับกระบวนการหาระยะของวัตถุที่ทำการตรวจจับ 44

รูปที่ 50 ตำแหน่งในการติดตั้งระบบตรสจจะบและติดตามวัตถุ.....45

รูปที่ 51 เครื่องมือในการวัดระยะทางเพื่อนำมาเปรียบเทียบอ้างอิง.....46

รูปที่ 52 ArUco Marker ขนาด 76*76 เซนติเมตรชนิด 6x6_100 ID 1,2,3 46

รูปที่ 53 การทดสอบสร้างระนาบจาก ArUco Marker 3ชนิดบนพื้นที่ทดสอบจริง.....47

รูปที่ 54 การทดสอบสร้างระนาบจาก ArUco Marker 3ชนิดบนพื้นที่ทดสอบจริง.....47

รูปที่ 55ตำแหน่งในการรับค่าพิกัดจากของMerging Point โดยใช้ Aruco Marker.....48

รูปที่ 56 ค่าตำแหน่งของ Merging Point เทียบกัยกล้อง.....48

รูปที่ 57 การทดลองวัดค่าตำแหน่งของยานยนต์ในพื้นที่ทดสอบ.....48

รูปที่ 58 เครื่องมือในการเก็บความเร็วของยานยนต์ที่นำมาใช้สำหรับทดสอบ จับความเร็ว (Ublox ZED -F9P).....49

รูปที่ 59 การสร้าง MQTT Broker Mosquitto.....49

รูปที่ 60 library ที่เลือกใช้ในการส่งค่า MQTT ในรูป JSON 50

รูปที่ 61 การสร้างตัวแปรเพื่อเก็บค่าในรูป Dictionary เพื่อทำการส่งค่าเข้าสู่ MQTT Broker.....50

รูปที่ 62 ค่าฟังก์ชันในการส่งข้อมูลเข้าสู่ MQTT broker ใน topic computer switch.....50

รูปที่ 63 โปรแกรมในการรับค่า message MQTT จาก MQTT Broker.....51

รูปที่ 64 ตำแหน่งของรถอัตโนมัติทำการ Subscribe และ Unsubscribe MQTT Broker.....51

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

รูปที่ 65 บริเวณในการตัดสินใจของรถอัตโนมัติ 52

รูปที่ 66 กราฟแสดงค่าความเร่งในการเบรคจาดระยะ 10 เมตร 53

รูปที่ 67 กราฟค่าความเร่งของรถอัตโนมัติจากหยุดนิ่งจนกระทั่งความเร็วมีค่าเท่ากับ 10 กิโลเมตรต่อชั่วโมง.....53

รูปที่ 68 รูปแสดงตำแหน่งของ Conflict Area และ ค่า LB 54

รูปที่ 69 ตำแหน่งจริงของ Conflict Area และ Merging Point 55

รูปที่ 70 ระยะทางระหว่างรถทั้งสองคันที่ใช้ในกระบวนการตัดสินใจอย่างง่าย.....55

รูปที่ 71แผนผังแสดงกระบวนการตัดสินใจของระบบจำลองการเคลื่อนที่ของยานยนต์อัตโนมัติ ..57

รูปที่ 72รูปแบบข้อมูลที่นำเข้าของระบบจำลองการเคลื่อนที่ผ่านจุด Merge ของ รถอัตโนมัติ Turing T2 58

รูปที่ 73การกำหนดตำแหน่งของรถอัตโนมัติในการรับค่าตัวแปรจากระบบตรวจจับ และติดตามวัตถุ.....58

รูปที่ 74 ตัวอย่างผลลัพธ์มราได้จากกระบวนการตรวจจับวัตถุด้วยโมเดล YOLOv8.....59

รูปที่ 75 ตัวอย่างผลลัพธ์จากกระบวนการติดตามวัตถุด้วย DeepSORT Tracking Algorithm 60

รูปที่ 76 การทดลองวัดค่าพิกัดของ ArUco Marker .ในบริเวณพื้นที่จริง.....60

รูปที่ 77 การติดตั้ง ArUco Marker เพื่อทำการคำนวณสมการระนาบในบริเวณพื้นที่จริง.....61

รูปที่ 78 ตัวแปรสมการระนาบ A B C และ D จากการคำนวนโดยพิกัด ArUco Marker 62

รูปที่ 79 บริเวณในการติดตั้ง ArUco Marker เพื่อทำการวัดระยะMerging Point สำหรับกระบวนการตัดสินใจของรถ.....62

รูปที่ 80 ตัวแปร Merging Point สำหรับการคำนวนระยะทางระหว่างยานยนต์และจุดร่วม 62

รูปที่ 81 ผลการวัดระยะของยานยนต์จากการตรวจจับด้วยภาพโดยสร้างระนาบ จากการติดตั้งแผ่น ArUco Marker ในแนวราบ.....63

รูปที่ 82 ผลการวัดระยะของยานยนต์จากการตรวจจับด้วยภาพโดยสร้างระนาบ จากการติดตั้งแผ่น ArUco Marker ในแนวตั้ง.....63

รูปที่ 83 ผลการทดสอบตรวจจับบริเวณพื้นที่จริง.....64

รูปที่ 84 กราฟแสดงผลการเก็บค่าเวลาที่ใช้ในระบบตรวจจับและติดตามวัตถุ 65

รูปที่ 85 ผลการทดลองทีได้จากการตรวจจับด้วย GPS 66

รูปที่ 86 กราฟแสดงการเปรียบเทียบค่าความเร็วระหว่างกระบวนการวัดด้วย GPS และกระบวนการติดตามวัตถุ ที่ความเร็ว 20 กิโลเมตรต่อชั่วโมงในระยะน้อยกว่า 31.5 เมตร 67

รูปที่ 87 กราฟแสดงการเปรียบเทียบค่าความเร็วระหว่างกระบวนการวัดด้วย GPS และกระบวนการติดตามวัตถุ ที่ความเร็ว 40 กิโลเมตรต่อชั่วโมงในระยะน้อยกว่า 31.5 เมตร 67

รูปที่ 88 กราฟแสดงการเปรียบเทียบค่าความเร็วระหว่างกระบวนการวัดด้วย GPS และกระบวนการติดตามวัตถุ ที่ความเร็ว 40 กิโลเมตรต่อชั่วโมงในระยะน้อยกว่า 31.5 เมตร 68

รูปที่ 89 ผลการทดสอบรส่งค่าตัวแปรจากกระบวนการตรวจจับวัตถุเข้าสู่ MQTT Broker 69

รูปที่ 90 กราฟแสดงผลการเก็บค่าเวลาที่ใช้ในการส่งข้อมูลผ่าน MQTT protocol ด้วยสัญญาณเครือข่ายเซลลูลาร์ 70

รูปที่ 91 กราฟแสดงผลค่าระยะทางเทียบกับเวลาระหว่าง Turing และยานยนต์บนท้องถนน 70

รูปที่ 92 กราฟแสดงค่าคุณสมบัติของยานยนต์อัตโนมัติ Turing T2 71

รูปที่ 93 กราฟแสดงระยะห่างระหว่างรถอัตโนมัติและ Conflict Area ที่น้อยที่สุดที่รับค่าจากระบบตรวจจับที่ความเร็ว 20 km/hr สามารถเคลื่อนรถออกจากConflict Area ได้อย่างปลอดภัย 72

รูปที่ 94 ระยะทางที่สั้นที่สุดในการรับค่าจากระบบตรวจจับยานยนต์ 72

รูปที่ 95 กราฟแสดงค่าความเร่งของยานยนต์อัตโนมัติ Turing T2 เมื่อเริ่มทำการ Emergency Brake 73

รูปที่ 96 กราฟแสดงระยะห่างระหว่างรถอัตโนมัติและ Conflict Area ที่น้อยที่สุดที่รับค่าจากระบบตรวจจับที่ความเร็ว 30 km/hr สามารถเคลื่อนรถออกจากConflict Area ได้อย่างปลอดภัย 73

รูปที่ 97 กราฟแสดงระยะห่างระหว่างรถอัตโนมัติและ Conflict Area ที่น้อยที่สุดที่รับค่าจากระบบตรวจจับที่ความเร็ว 40 km/hr สามารถเคลื่อนรถออกจากConflict Area ได้อย่างปลอดภัย 74

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหา

จากสถิติการรายงานปัญหาอุบัติเหตุที่เกิดจากการเปลี่ยนช่องทางการเดินรถจากฐานข้อมูลในปี ค.ศ.1999 จากระบบ National Automotive Sampling System/General Estimates System (GES) national ในหน่วยงานของ National Highway Traffic Safety Administration (NHTSA) ประเทศอเมริกาพบว่ามีการเกิดอุบัติเหตุถึง 539,000 เหตุการณ์ที่เกิดจากการชนของรถสองคันจากการเปลี่ยนช่องการเดินรถหรือคิดเป็น 9% ของอุบัติเหตุจากการรายงานทั้งหมดของตำรวจ และพบว่าการเกิดการเกิดอุบัติเหตุที่ในทางร่วมถูกจัดอันดับอยู่ใน 7 อันดับสูงสุดจากการเปลี่ยนช่องทางเดินรถ พบว่าประเทศอเมริกามีการเกิดอุบัติเหตุถึง 19,000 ครั้งที่เกิดจากทางร่วม คิดเป็น 3.5 % ของอุบัติเหตุที่เกิดจากการเปลี่ยนช่องทางเดินรถทั้งหมด[1] และล่าสุดในปี ค.ศ.2016 และปี ค.ศ. 2017 [2, 3]จากหน่วยงาน NHTSA ได้ทำการสำรวจการเกิดอุบัติเหตุบนท้องถนนที่เกิดจากรถยนต์และส่งผลถึงชีวิตพบว่าการเกิดอุบัติเหตุบริเวณทางร่วมติดอยู่ในหนึ่งของสาเหตุในการเกิดอุบัติเหตุจากการสำรวจ 43,642 อุบัติเหตุ พบว่า มีเหตุการณ์ที่เกิดขึ้นบริเวณทางร่วมถึง 735 ครั้ง คิดเป็น 1.7 % ของเหตุการณ์ที่ได้มีการสำรวจ โดยพบว่าสาเหตุส่วนใหญ่ของการเกิดอุบัติเหตุจากการเปลี่ยนช่องทางการเดินรถเกิดจากมุมมองของผู้ขับขี่

จากการนำร่องใช้งานรถยนต์อัตโนมัติของ Smart Mobility Research Center มีเส้นทางการเดินรถอัตโนมัติที่ผ่านทางร่วม (Lane Merging) ที่ไม่มีสัญญาณเตือนและจำเป็นต้องขับร่วมกับรถยนต์บนท้องถนนในตำแหน่งเซนเซอร์ของรถอัตโนมัติไม่สามารถตรวจจับได้ เนื่องจากในปัจจุบันมีการนำเทคโนโลยี V2I มาใช้ร่วมกับระบบรถยนต์อัตโนมัติเพิ่มมากขึ้นเพราะสามารถเพิ่มความปลอดภัยในการขับขี่ภายในบริเวณที่มีโอกาสเกิดอุบัติเหตุสูง หรือบริเวณที่เซนเซอร์ของรถอัตโนมัติไม่สามารถทำงานได้อย่างเต็มประสิทธิภาพมี ยกตัวอย่างเช่น สถานการณ์ภายในอุโมงค์ที่ไม่สามารถใช้สัญญาณ GPS ได้ หรือการเคลื่อนที่ผ่านทางร่วมที่มีรถเคลื่อนที่ในตำแหน่งที่เซ็นเซอร์ของรถไม่สามารถตรวจจับได้ [4]และ มีการใช้กล้องเป็นเซนเซอร์หลักในการตรวจจับ[5] ซึ่งกล้องจัดเป็นเซนเซอร์ที่ได้รับความนิยมมากที่สุดในการพัฒนาด้านรถอัตโนมัติเพราะข้อมูลที่ได้รับจากกล้องสามารถ ใช้ในการตรวจจับ และ Tracking ได้ สามารถตรวจจับวัตถุได้ในหลายระยะในระดับเซนติเมตรถึงในระดับ 100 เมตร มีราคาที่ถูก มีการผลิตใช้งานอย่างแพร่หลาย และมีการนำไปประยุกต์ใช้จริงในระบบเกี่ยวกับรถอัตโนมัติ

รายงานวิทยานิพนธ์นี้จึงได้ทำการศึกษาและพัฒนาระบบป้องกันการชน โดยนำการตรวจจับด้วยกล้องมาใช้ร่วมกับการส่งข้อมูลระหว่าง ยานพาหนะ และ สิ่งปลูกสร้างผ่านเครือข่าย

5G โดยมีพื้นฐานมาจากระบบการสื่อสารระหว่างรถอัตโนมัติและโครงสร้างพื้นฐาน(V2I) สำหรับตรวจจับรถยนต์บนท้องถนน ให้สามารถตรวจจับรถยนต์บนท้องถนนในตำแหน่งที่เซนเซอร์ของรถยนต์อัตโนมัติไม่สามารถตรวจจับได้ เพื่อส่งข้อมูลไปยังรถอัตโนมัติสำหรับประกอบการตัดสินใจของรถยนต์อัตโนมัติ เพื่อลดความเสี่ยงต่ออุบัติเหตุของรถยนต์อัตโนมัติ

1.2 วัตถุประสงค์ของการวิจัย

1 พัฒนาระบบป้องกันการชนสำหรับยานยนต์อัตโนมัติโดยใช้การตรวจจับวัตถุด้วยกล้องที่ทำการติดตั้งอยู่ในบริเวณท้องถนนสำหรับใช้งานร่วมกับการสื่อสารระหว่างยานพาหนะอัตโนมัติและโครงสร้างพื้นฐานผ่านเครือข่ายเซลลูลาร์ให้สามารถตรวจจับรถยนต์ที่ขับเคลื่อนบนท้องถนนที่มีความเร็วไม่เกิน 30 กิโลเมตร/ชั่วโมง และส่งข้อมูลไปยังรถอัตโนมัติเพื่อชะลอความเร็วหรือหยุดการเคลื่อนที่แบบอัตโนมัติ

2 พัฒนาระบบที่สามารถควบคุมโดยการชะลอรถอัตโนมัติให้สามารถทำงานได้ทันเวลาสำหรับการขับเคลื่อนรถอัตโนมัติที่ความเร็วต่ำ

1.3 ขอบเขตของการวิจัย

1 โครงงานนี้จำกัดขอบเขตพื้นที่การทดลองบริเวณหน้าคณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัยเนื่องจากเป็นทางร่วมที่เป็นบริเวณทดสอบที่ผู้ทดสอบยานยนต์อัตโนมัติพบว่าเป็นบริเวณที่มีความอันตรายเป็นอย่างมากเนื่องจากมีวัตถุที่สามารถบดบังสัญญาณจากอุปกรณ์เซนเซอร์ซึ่งมีลักษณะดัง รูปที่ 1

รูปที่ 1 พื้นที่ทดสอบบริเวณหน้าคณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

2 กำหนดอัตราเร็วของรถบนท้องถนนไม่เกิน 30 กิโลเมตร/ชั่วโมง ตามพฤติกรรมการขับขี่จริงในจุฬาลงกรณ์มหาวิทยาลัย

3 กำหนดอัตราเร็วของรถอัตโนมัติที่ใช้ในการทดลองมีความเร็วต่ำวิ่งไม่เกิน 20 กิโลเมตร/ชั่วโมง

4 พัฒนา ระบบป้องกันการชน โดยใช้การส่งข้อมูลระหว่าง ยานพาหนะ และ โครงสร้างพื้นฐาน โดยใช้กล้องเป็นเซนเซอร์หลักในการตรวจจับรถยนต์บนท้องถนน

5 พัฒนา ระบบป้องกันการชน โดยใช้การส่งข้อมูลระหว่าง ยานพาหนะ และ โครงสร้างพื้นฐาน ให้สามารถตรวจจับรถยนต์บนท้องถนนได้หลายคัน

6 กำหนดให้การวางแผนการเดินทางของรถอัตโนมัติ เมื่อได้รับการเตือนให้ทำการชะลอความเร็วและเบรกในช่องทางเดินรถเท่านั้น

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1) องค์ความรู้ในการใช้ในการพัฒนาพัฒนา ระบบป้องกันการชน โดยใช้การส่งข้อมูลระหว่าง ยานพาหนะ และ สิ่งปลูกสร้าง ในด้านการประมวลผลภาพ (Image processing) ด้านอินเตอร์เน็ตทุกสรรพสิ่ง (IoT) และด้านการควบคุมรถอัตโนมัติ

2) สามารถพัฒนาพัฒนา ระบบป้องกันการชนต้นแบบ โดยใช้การส่งข้อมูลระหว่างยานพาหนะ และ สิ่งปลูกสร้าง โดยใช้กล้องเป็นเซนเซอร์หลักในการตรวจจับได้

1.5 ขั้นตอนการวิจัย

1 ศึกษางานวิจัยก่อนหน้าที่เกี่ยวข้องกับงานวิจัยนี้ด้านการตรวจจับวัตถุด้วยกล้อง

2 ศึกษางานวิจัยก่อนหน้าที่เกี่ยวข้องกับงานวิจัยนี้เกี่ยวกับระบบการสื่อสารระหว่างรถและโครงสร้างพื้นฐาน

3 ศึกษางานวิจัยก่อนหน้าที่เกี่ยวข้องกับงานวิจัยนี้ด้านการตัดสินใจในการชะลอของรถบริเวณทางร่วม

4 ประยุกต์ใช้แบบรูปในการตรวจจับวัตถุและปรับค่าเพื่อเพิ่มความเร็วในการตรวจจับวัตถุ

5 ออกแบบ สร้างระบบ และทดสอบความสามารถในการตรวจจับ

6 ออกแบบ และสร้างระบบในการประมวลผลข้อมูลที่ได้จากการตรวจจับและส่งสัญญาณเพื่อสื่อสารกับรถอัตโนมัติ

7 ทดสอบการตัดสินใจของรถอัตโนมัติเมื่อรถอัตโนมัติได้รับข้อมูลจากระบบตรวจจับ

8 ทดสอบการใช้ระบบโดยรวมและทดสอบเวลาที่ใช้ในระบบโดยใช้ Carla simulation

9 วิเคราะห์ และสรุปผลการทดลอง

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

1.6 แผนการดำเนินงาน

ขั้นตอน	2565					2566						
	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.	มิ.ย.	ก.ค.
รวบรวมข้อมูลและศึกษางานวิจัยที่เกี่ยวข้อง												
ศึกษาวิธีใช้ ROS, Autoware												
ศึกษาวิธีการใช้ Object detection and Tracking Algorithm												
ศึกษาการทำงานของระบบการสื่อสารระหว่างรถและโครงสร้างพื้นฐาน												
ศึกษาด้านการตัดสินใจในการชะลอของรถบริเวณทางร่วม												
ออกแบบและสร้างระบบตรวจจับและติดตามวัตถุ ระบบการสื่อสารระหว่างรถและโครงสร้างพื้นฐานและระบบการตัดสินใจของรถบริเวณทางร่วม												
ทำการทดสอบระบบแก้ไขและวิเคราะห์ผลของการทดสอบ												
สรุปผลและทำรายงาน												

จุฬาลงกรณ์
CHULALONGK

บทที่ 2
แนวคิดทฤษฎีและงานวิจัยที่เกี่ยวข้อง

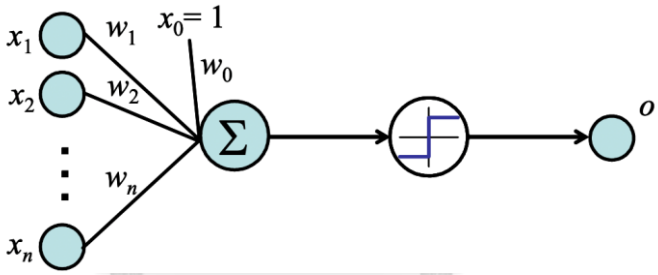
บทนี้จะทบทวนงานวิจัยที่เกี่ยวข้องกับการพัฒนา ระบบป้องกันการชน โดยใช้การส่งข้อมูลระหว่าง ยานพาหนะ และ สิ่งปลูกสร้าง โดยใช้กล้องเป็นเซนเซอร์หลักในการตรวจจับยานพาหนะบนท้องถนน โดยแบ่งงานวิจัยที่เกี่ยวข้องออกเป็น 3 ส่วนหลัก คือ ทฤษฎีและงานวิจัยที่เกี่ยวกับระบบตรวจจับยานพาหนะ งานวิจัยที่เกี่ยวกับการใช้ระบบสื่อสาร งานวิจัยที่เกี่ยวข้องกับการวางแผนการเดินทางของรถ

2.1 Detection and Tracking System

2.1.1 Object detection theory

2.1.1.1 Neural network

ข่ายประสาทเทียม(Artificial Neural Network) เป็นการจำลองการทำงานบางส่วนของเซลล์ประสาทในสมองของมนุษย์ ประกอบด้วยเพอร์เซปตรอน(perceptron)[6] เป็นข่ายประสาทเทียมแบบง่ายโดยจำลองให้มีลักษณะดัง รูปที่ 2



รูปที่ 2 โครงสร้างของเพอร์เซปตรอน[6]

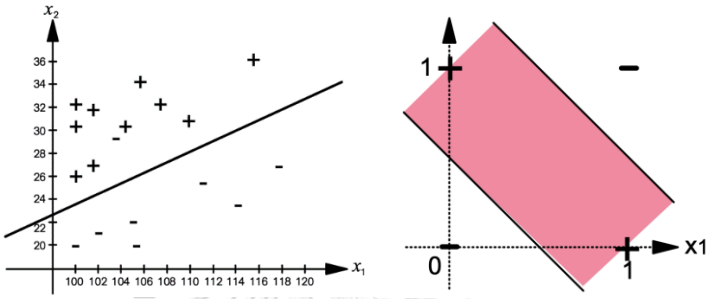
โดยรับค่าอินพุตเป็นเวกเตอร์จำนวนจริงและคำนวณหาผลรวมเชิงเส้นแบบถ่วงน้ำหนักของอินพุต(x1,x2,…,xn) ค่า(w1,w2,…,wn)ในรูปคือค่าน้ำหนักของอินพุต ค่าที่ได้จากการคำนวนจะนำมาคำนวณในฟังก์ชันกระตุ้น (activation function) ดังรูปตัวอย่างคือฟังก์ชันสองขั้ว (Bipolar function) เมื่อนำค่าที่คำนวณจากอินพุตและค่าน้ำหนักเทียบกับค่าขีดแบ่ง ผลเอาต์พุต เป็น 1 และ -1 สามารถแสดงเอาต์พุต (o) ในรูปของฟังก์ชันอินพุต ได้ดังสมการที่ 1

$$o(x_1, x_2, \dots, x_n) = \begin{cases} 1 \; if \; w_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n > 0 \\ -1 \; if \; w_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n < 0 \end{cases} \quad (1)$$

โดยปัญหาในการเรียนรู้ของเพอร์เซปตรอน[6] คือการหาค่าเวกเตอร์น้ำหนัก $(\overrightarrow{w})$ ที่เหมาะสมสำหรับจำแนกข้อมูลที่นำมาสอนเพื่อให้เพอร์เซปตรอนสามารถแสดงค่าเอาต์พุตได้ตรงกับค่าที่สอน โดยจะมีวิธีการปรับค่าน้ำหนักพื้นฐานดังสมการที่ 2

$$\Delta w_i = \alpha(t - o)x_i \quad (2)$$

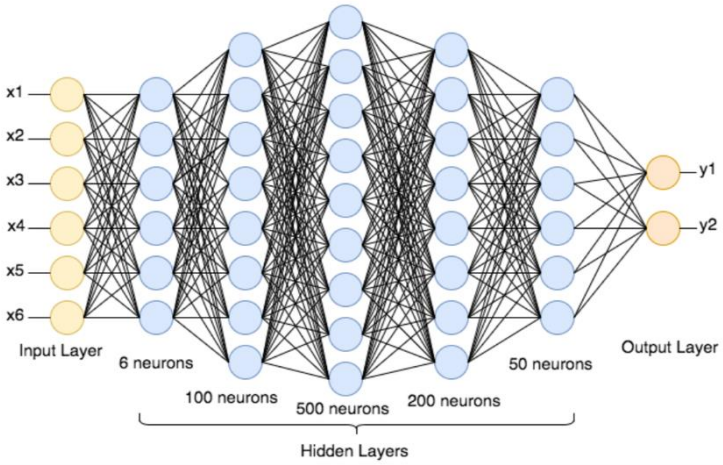
ค่า α คือค่า learning rate ค่า t คือค่าเอาต์พุตที่ต้องการ และค่า o คือค่าเอาต์พุตที่ได้จากเพอร์เซปตรอน โดยเพอร์เซฟตรอนจะสามารถเรียนรู้ฟังก์ชันแยกได้ในรูปแบบเชิงเส้นเท่านั้น เมื่อนำเพอร์เซปตรอนหลายตัวมาเชื่อมต่อกัน จะทำให้เกิดเป็นข่ายงานประสาทหลายชั้น (multilayer neural network) ซึ่งทำให้เกิดผิวตัดสินใจไม่เชิงเส้น (non-linear decision surface) [6] ได้เปรียบเทียบระนาบตัดสินใจได้ดังในรูปที่ 3



รูปที่ 3 กราฟเปรียบเทียบระนาบตัดสินใจระหว่างการใช้เพอร์เซปตรอน 1 หน่วยและสองหน่วย

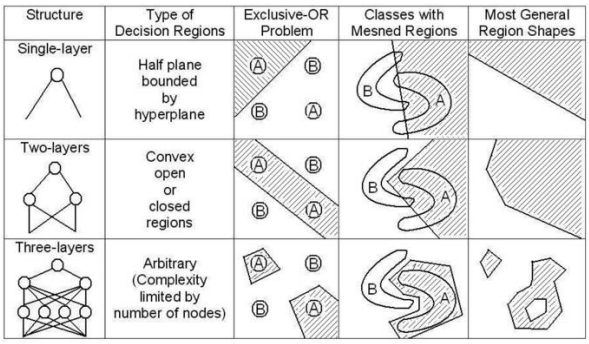
2.1.1.2 การเรียนรู้เชิงลึก (Deep learning)

การเรียนรู้เชิงลึกคือ Multilayer neural network ที่จะประกอบด้วย Hidden layer หลายชั้น ดังรูปที่ 4



รูปที่ 4 ตัวอย่างโครงสร้างของกระบวนการเรียนรู้เชิงลึก[7]

เพื่อเพิ่มความสามารถในการคำนวน ให้สามารถคำนวณให้ซับซ้อนได้มากขึ้น จากรูปที่ 5 แสดงให้เห็นว่า เมื่อเพิ่มจำนวน Hidden layer ระนาบในการตัดสินใจสามารถจำแนกข้อมูลที่มีความซับซ้อนได้มากขึ้น



รูปที่ 5 ระนาบตัดสินใจเมื่อ Hidden layer เพิ่มขึ้น[8]

ปัจจุบันมีการนำกระบวนการ Deep learning มาประยุกต์ใช้ในการแก้ไขปัญหาในหลายด้าน เช่น machine vision ,robotics, Natural language processing

2.1.1.3 Convolutional neural network (CNN)

Convolutional neural network เป็นหนึ่งในประเภทของกระบวนการ Deep learning Neural network ที่มีลักษณะการเชื่อมต่อกันแบบพิเศษทำให้สามารถจำแนกข้อมูลประเภทรูปภาพได้ดีกว่า Neural network ทั่วไป CNN ประกอบด้วย layer หลักคือ Convolve input layer, non-linearity layer, Subsampling and Pooling Layer, fully connected layer และ Input layer รูปที่ 6 แสดงตัวอย่าง CNN ชื่อ LeNet-5

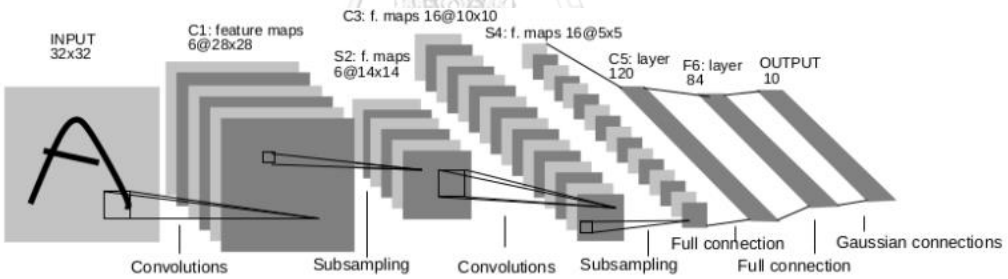
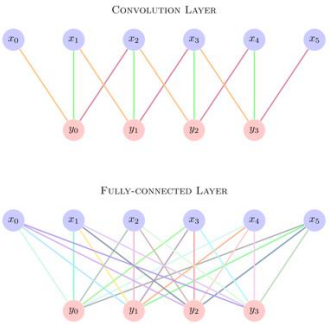


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

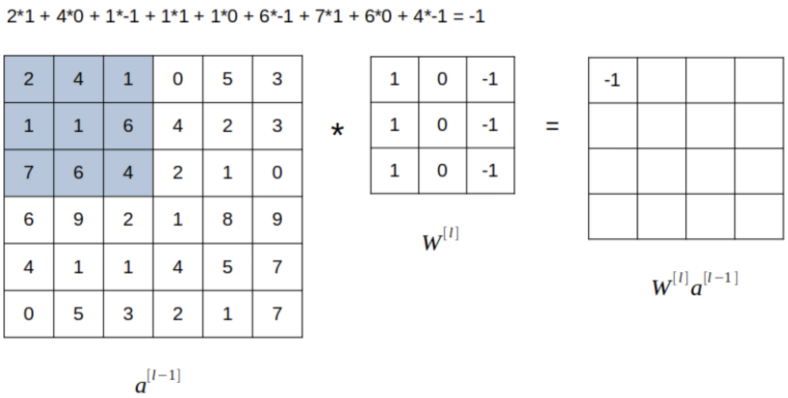
รูปที่ 6 ตัวอย่างโครงสร้าง Convolutional neural network[9]

1) Convolution layer มีลักษณะพิเศษคือแต่ละ node เชื่อมต่อกับ node ใกล้เคียงและมีการปรับค่าน้ำหนักไปในทางเดียวกัน[10] แสดงในรูปที่ 7



รูปที่ 7 การเปรียบเทียบระหว่าง Fully-connected layer และ Convolution layer[11]

หน้าที่ของ convolution layer คือการสกัดลักษณะของภาพ ด้วยการใช้ค่านำหนัก filter matrix ชนิดอาเรย์สองมิติ Convolve (การทำผลคูณจุด Dot Product) กับ Input Matrix ของรูปภาพในทุกตำแหน่งจุดของ Input matrix แสดงดังรูปที่ 8



รูปที่ 8 ตัวอย่างกระบวนการคำนวณ Convolve ด้วย filter matrix ภายใน Convolution layer[10]

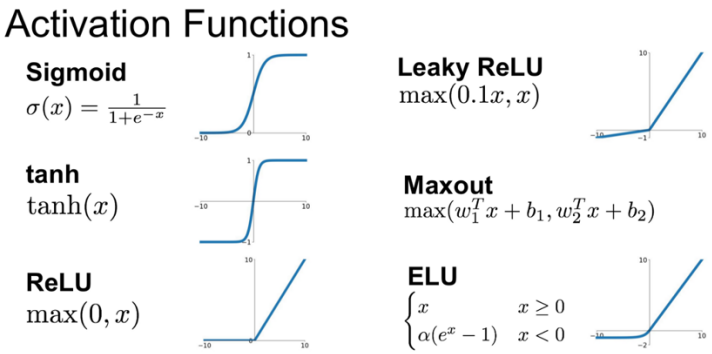
เมื่อทำซ้ำกับ filter matrix หลายชนิดจะทำให้สามารถตรวจจับคุณลักษณะบางส่วนของภาพได้

2)Pooling layer

หน้าที่ของ Pooling layer คือการสกัดเอาส่วนที่สำคัญที่สุดของข้อมูล และเพิ่มประสิทธิภาพการประมวลผลให้รวดเร็วยิ่งขึ้น โดยการสกัดค่าที่มากที่สุดหรือค่าเฉลี่ย ของตารางที่ทาบในInput เก็บค่าใน Matrix Output ดังรูปที่ เมื่อสกัดค่าที่มากที่สุดจะเรียกว่ากระบวนการ Max pooling เมื่อสกัดค่าเฉลี่ยจะเรียกว่ากระบวนการ Average pooling

3)Activation Function

Activation function คือ ฟังก์ชันที่จะรับผลการคำนวณจาก Input layer เพื่อนำมาประมวลผลหาค่า Out put สามารถเป็นได้หลายฟังก์ชันเช่น Sigmoid function, tanh function, ReLU function, Leaky ReLU function, Maxout function, ELU function ดังรูปที่ 9



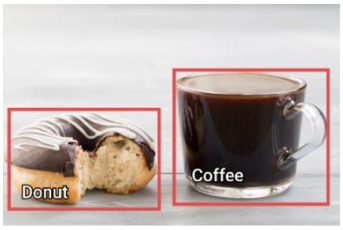
รูปที่ 9 ชนิดและลักษณะของ Activation Function ที่มีการนำมาใช้ในปัจจุบัน[12]

2.1.1.5 Computer vision

คอมพิวเตอร์วิสัย (computer vision) เป็นสาขาวิชาที่สำคัญในหลากหลายด้าน ช่วยให้เครื่องมือสามารถเข้าใจและตีความข้อมูลทางภาพได้ เป็นต้น แบบการทำงานที่สำคัญในคอมพิวเตอร์วิสัยคือการตรวจจับวัตถุ (object detection) การจำแนกวัตถุ (object classification) และการตัดแยกวัตถุ (object segmentation) ในกระบวนการดังกล่าวนี้มีลักษณะใช้งาน วิธีการทำงานและผลลัพธ์ ที่แตกต่างกันดังนี้

2.1.1.5.1 การตรวจจับวัตถุ (Object Detection)

การตรวจจับวัตถุคือกระบวนการที่เป็นการตรวจสอบและระบุวัตถุที่อยู่ในภาพ โดยการระบุตำแหน่งของวัตถุด้วยกรอบสี่เหลี่ยม (bounding box) ซึ่งบ่งบอกถึงพิกัดที่วัตถุอยู่ในภาพ และระบุประเภทหรือชนิดของวัตถุนั้น ๆ เช่น รถยนต์, คน, แมว ฯลฯ วิธีการตรวจจับวัตถุที่ได้รับความนิยมในปัจจุบันเป็นหลักการใช้โครงข่ายประสาทเชิงลึก (Deep Neural Networks) เช่น Faster R-CNN, YOLO (You Only Look Once), และ SSD (Single Shot MultiBox Detector)



รูปที่ 10 ตัวอย่างแสดงข้อมูลที่ได้รับจากกระบวนการ Detection[13]

2.1.1.5.2 การจำแนกวัตถุ (Object Classification)

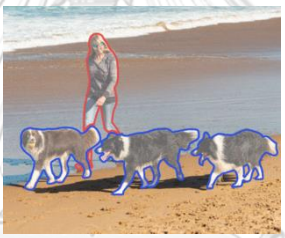
การจำแนกวัตถุเป็นกระบวนการที่เน้นไปที่การระบุประเภทหรือคลาสของวัตถุที่อยู่ในภาพ โดยไม่จำเป็นต้องระบุตำแหน่งของวัตถุเหมือนในการตรวจจับวัตถุ หลักการทำงานของการ

จำแนกวัตถุเน้นไปที่การเรียนรู้รูปแบบของวัตถุในรูปภาพ เช่น การใช้โครงข่ายประสาทเชิงลึกที่ถูกฝึกสอนด้วยชุดข้อมูลที่มีป้ายกำกับ (labeled dataset) เพื่อจำแนกวัตถุต่าง ๆ ตามประเภทที่กำหนด

รูปที่ 11 ค่าตัวแปลที่ได้จากกระบวนการ Image Classificationผ

2.1.1.5.3 การตัดแยกวัตถุ (Object Segmentation)

การตัดแยกวัตถุคือกระบวนการที่เน้นไปที่การระบุขอบเขตของวัตถุแต่ละตัวในภาพด้วยความละเอียด โดยให้ผลลัพธ์เป็นแมสก์ (mask) ที่บ่งบอกถึงพื้นที่ของวัตถุในภาพ ซึ่งช่วยให้สามารถแยกวัตถุที่ต่างกันออกจากภาพได้อย่างชัดเจน การตัดแยกวัตถุใช้เทคนิคที่มีความซับซ้อนกว่าการตรวจจับวัตถุและการจำแนกวัตถุ เช่น Mask R-CNN, U-Net



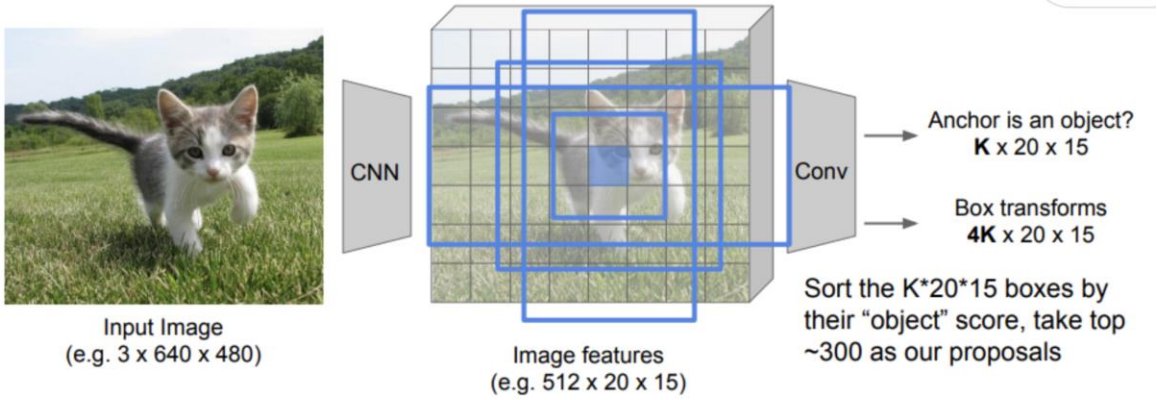
รูปที่ 12 ตัวอย่างของกระบวนการ Object Segmentation

2.1.1.4 CNN-based Object detection design

ลักษณะการตรวจจับวัตถุโดยใช้พื้นฐานของ convolution neural network ในปัจจุบันแบ่งออกเป็น 2 รูปแบบ คือ Two-stage models และ One-stage models

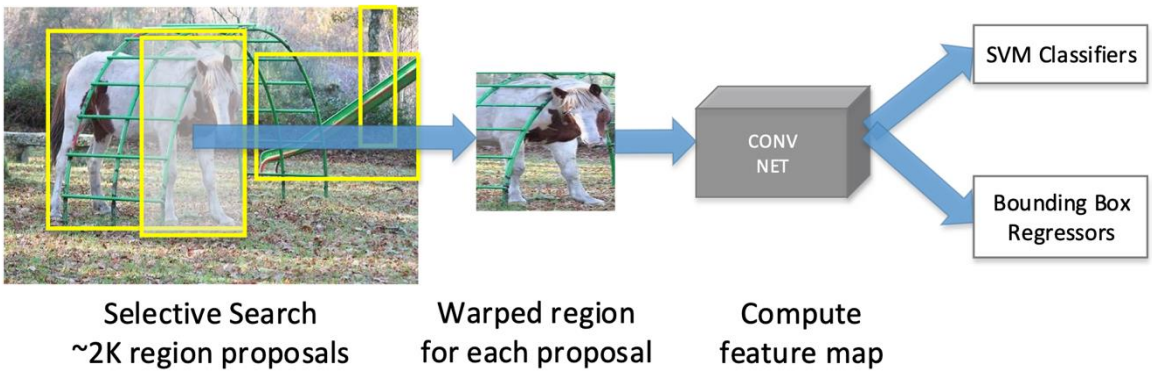
2.1.1.4.1 Two-stage models

มีงานวิจัยส่วนมากในเรื่องเกี่ยวกับการตรวจจับวัตถุมีการใช้ two stage detection ส่วนแรก (first stage) คือ regional proposals extraction เป็น neural network ที่ทำ classification หาขอบเขตที่น่าจะมีวัตถุอยู่และมีการทำ regression เพื่อที่จะได้ขอบเขตที่แม่นยำขึ้น ขอบเขตที่ region proposal network(RPN) นำมาพิจารณา คือกล่องสี่เหลี่ยมหลาย ขนาด ที่ครอบคลุม feature map ในตำแหน่งต่างๆ region proposal network ก็จะทำการหาขอบเขตที่น่าจะเป็น object มากที่สุดมาจำนวนนึง เช่น 300 ขอบเขต เพื่อส่งเป็น region proposal หรือ RoI (Region of Interest) สำหรับขั้นตอนต่อไป ดังตัวอย่างใน รูปที่ 13 บางระบบมีการใช้กระบวน sliding window technique, Deformable Parts Models (DPM) และ OverFeat มาใช้



รูปที่ 13 กระบวนการ Region Proposal Network (RPN)

ขั้นตอนที่สอง(second state) ประกอบด้วยกระบวนการต่างๆคือ RoI pool เป็นการทำ RoI ของ feature map ที่มีขนาดแตกต่างกัน ให้มีขนาดเดียวกัน เพื่อเป็น input ให้กับ neural network ในชั้น classification ที่ทำการประมวลผลเพื่อจำแนก object และ regression ทำการปรับขอบเขตของ RoI ให้แม่นยำขึ้น ดังรูปที่ 14



CHULALONGKORN UNIVERSITY

รูปที่ 14 กระบวนการทำงานของ Two-state Object Detection[14]

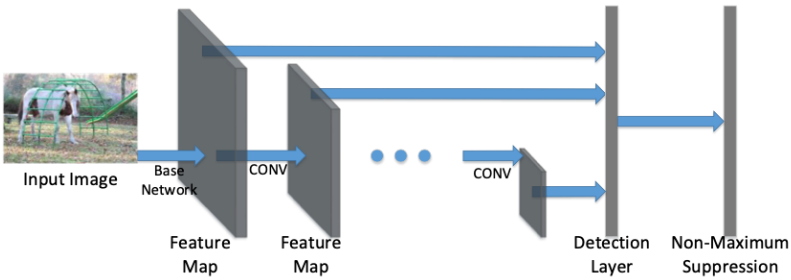
Two-state detection จะมีค่า accuracy ในการตรวจจับที่สูง สามารถตรวจจับวัตถุได้ในหลายๆขนาด แต่เนื่องจากปัจจุบันมีปัจจัยในการตรวจจับที่เพิ่มขึ้นคือ Realtime ซึ่ง Two-state detection จะใช้เวลาประมวลผลภาพที่นานจึงไม่เหมาะสมกับการตรวจจับแบบ Realtime

2.1.1.4.2 One-state models

มีการใช้รูปแบบหลักๆอยู่สองรูปแบบ คือ YOLO (You Only Look Once) กับ SSD (Single Shot Multibox Detector) ซึ่งจะมีความเร็วและปริมาณข้อมูลที่ได้จากการฝึกน้อยกว่าแต่ความแม่นยำในการตรวจจับจะน้อยกว่า Two-state Detector ในปัจจุบัน One-state Detector มีการพัฒนาอย่างต่อเนื่องให้สามารถตรวจจับได้แม่นยำมากขึ้น หลักการทำงานของ One-state Detector คือ การ

แบ่งรูปที่ต้องการตรวจจับเป็นตาราง $N \times N$ โดยแต่ละช่อง (grid cell) จะรับผิดชอบในการตรวจจับวัตถุที่อยู่ในตาราง และให้ Output คือ $N \times N \times S$ โดยค่า S feature map คือค่าของเมทริกซ์ $N \times N$ และค่าของลักษณะในแต่ละช่องของตาราง (grid cell) โดยทั่วไปค่า feature map ของวัตถุที่ต้องการตรวจจับ 1 ตัวจะประกอบด้วย ค่า $S = (5 + C)$ ค่า

โดย 4 ค่าแรกคือ ค่าของกรอบ Bounding box (x coordinate, y coordinate, height, width) ค่าที่ 5 คือค่าของความน่าจะเป็นของ ช่อง (grid cell) ที่จะมีตำแหน่งอยู่บริเวณวัตถุ ค่า C บอกชนิดของวัตถุ[14] ลักษณะของ One state models จะมีความซับซ้อนน้อยกว่า Two stage models แสดงในรูปที่ 15



รูปที่ 15 กระบวนการทำงานของ One-stage Object detection[14]

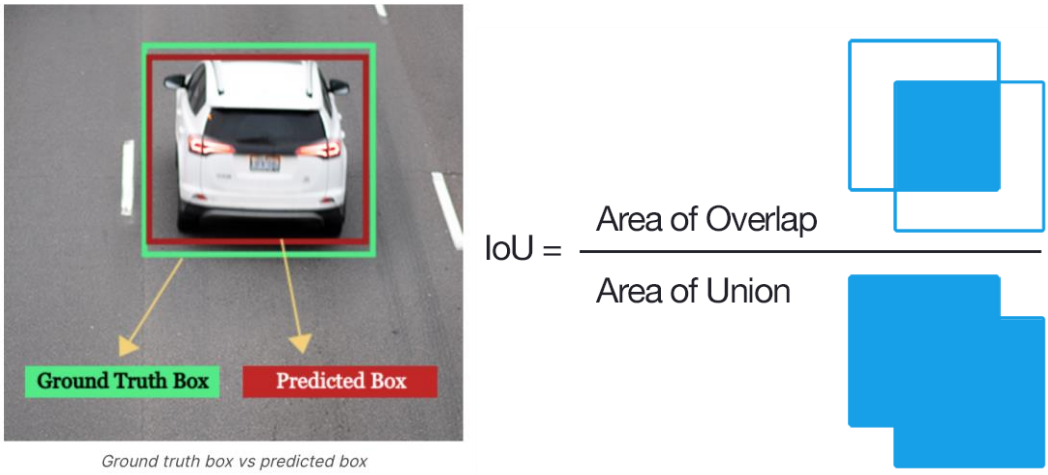
จึงสามารถใช้ระบบประมวลผลที่น้อยกว่าและความเร็วในการประมวลผลมากกว่า ในทางตรงกันข้ามค่าความแม่นยำในการตรวจจับวัตถุจะมีค่าน้อยลง

2.1.1.5การประเมินประสิทธิภาพของการตรวจจับด้วยวัตถุ

การประเมินประสิทธิภาพของการตรวจจับด้วยวัตถุ สามารถแบ่งออกเป็น 2 ส่วนหลักคือ การประเมินตำแหน่งที่สามารถตรวจจับได้และการประเมินความสามารถในการจำแนกวัตถุ

2.1.1.5.1การประเมินตำแหน่งสามารถวัดได้โดยจากค่าอัตราส่วน Intersection over Union (IoU)

Intersection over Union (IoU) เป็นอัตราส่วนของพื้นที่ที่ซ้ำกันระหว่างกรอบสี่เหลี่ยม (bounding box) ที่ระบุโดยระบบ Detection และกรอบสี่เหลี่ยมที่เป็นคำตอบที่ถูกต้อง ค่า IoU สูงสุดเท่ากับ 1 แสดงถึงความเที่ยงตรงสูงสุดของระบบในการระบุตำแหน่งของวัตถุ IoU ต่ำสุดคือค่า 0 ซึ่งหมายความว่ากรอบที่ได้จากการ Detection และกรอบคำตอบ ไม่มีการซ้อนทับกัน



รูปที่ 16 การใช้กระบวนการ IoU ประเมินผล Bounding Box[15]

2.1.1.5.2การประเมินความสามารถในการจำแนกวัตถุ

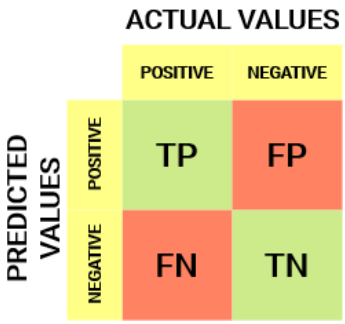
1) Confusion Matrix (เมตริกความสับสน) เป็นเครื่องมือที่ใช้ในการวัดและสรุปผลลัพธ์ของระบบการตรวจจับวัตถุ ช่วยให้สามารถแสดงผลการจำแนกออกเป็นกลุ่มต่าง ๆ ประกอบด้วยตัวแปรต่อไปนี้:

True Positives (TP): จำนวนของตัวอย่างที่ถูกตรวจจับถูกต้องว่าเป็นวัตถุบวก (Positive) โดยระบบการตรวจจับวัตถุก็ระบุว่าเป็นวัตถุบวก

True Negatives (TN): จำนวนของตัวอย่างที่ถูกตรวจจับถูกต้องว่าเป็นวัตถุลบ (Negative) โดยระบบการตรวจจับวัตถุก็ระบุว่าเป็นวัตถุลบ

False Positives (FP): จำนวนของตัวอย่างที่ถูกตรวจจับว่าเป็นวัตถุบวก แต่ในความเป็นจริงแล้วไม่ใช่วัตถุบวก (เทียบเคียงกับ False Alarm)

False Negatives (FN): จำนวนของตัวอย่างที่ถูกตรวจจับว่าเป็นวัตถุลบ แต่ในความเป็นจริงแล้วเป็นวัตถุบวก (เทียบเคียงกับ Miss)



รูปที่ 17 ตัวอย่างค่า TP FP FN TN

Confusion Matrix เป็นตารางที่สร้างขึ้นโดยวางตัวแปรที่กล่าวมาเป็นแกนตาราง โดยแนวตั้งแทนวัตถุที่ระบบการตรวจจับวัตถุทำนายได้ (Predicted) และแนวนอนแทนวัตถุที่จริง (Actual)

2) Precision เกี่ยวกับการตรวจจับวัตถุค่า Precision คือค่าที่อธิบายว่าสามารถหาค่า True Positive ได้ดีหรือไม่โดยวัดจากความสัมพันธ์ระหว่าง True Positive กับค่า Positive Prediction ทั้งหมด

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (3)$$

$$Precision = \frac{True\ Positive}{All\ Observations} \quad (4)$$

3) Recall คือค่าสำหรับการวัดความสามารถในการหาค่า True Positive ได้ดีหรือไม่ในการทำนายทั้งหมด

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (5)$$

$$Recall = \frac{True\ Positive}{All\ Ground\ Truth} \quad (6)$$

4) Average Precision (AP) เป็นตัวชี้วัดที่ใช้ในการประเมินประสิทธิภาพของระบบการตรวจจับวัตถุ (Object Detection) โดยสามารถวัดได้จากการสร้างกราฟระหว่าง ค่า Precision และค่า Recall (PR curve) โดยค่า AP สามารถหาได้จากผลรวมของพื้นที่ใต้กราฟของ PR curve ในทุกขั้นตอนหารด้วยจำนวนขั้นตอนทั้งหมด AP มีค่าระหว่าง 0 ถึง 1 โดยค่ามากยิ่งดี เมื่อ AP เป็น 1 แสดงว่าระบบการตรวจจับวัตถุมีประสิทธิภาพสูงในการระบุวัตถุบวกและความครอบคลุมสูง

$$AP = \int_0^1 p(r)dr \quad (7)$$

5) mean Average Precision (mAP) เป็นค่าเฉลี่ยของ Average Precision ของแต่ละคลาสวัตถุที่ตรวจจับ โดยทำการเฉลี่ยทั้งหมด เป็นเมตริกที่ใช้กันอย่างแพร่หลายในการวัดประสิทธิภาพของระบบการตรวจจับวัตถุ[16]

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

2.1.1.6) State of the art

mean Average Precision ใช้สำหรับ ประเมินค่าความสามารถของ State of the art ที่ใช้ในการตรวจจับ ตารางที่ 1 แสดงความสามารถในการตรวจจับด้วย ชุดข้อมูล MS COCO และ PASCAL VOC 2012 เนื่องจากในงานวิจัยนี้ใช้พัฒนาระบบสำหรับตรวจจับรถบนท้องถนนจึงใช้เกณฑ์ในการเลือกแบบในการตรวจจับรถยนต์คือค่า FPS ร่วมกับค่า mAP

ตารางที่ 1 ความสามารถในการตรวจจับวัตถุจากการสำรวจของคุณ S.S.A.[17]

Table 4
Performance comparison of various object detectors on MS COCO and PASCAL VOC 2012 datasets at similar input image size. Rows colored gray are real-time detectors (>30 FPS).

Model	Year	Backbone	Size	$AP_{[0.5:0.95]}$	$AP_{0.5}$	FPS
R-CNN*	2014	AlexNet	224	-	58.50%	~0.02
SPP-Net*	2015	ZF-5	Variable	-	59.20%	~0.23
Fast R-CNN*	2015	VGG-16	Variable	-	65.70%	~0.43
Faster R-CNN*	2016	VGG-16	600	-	67.00%	5
R-FCN	2016	ResNet-101	600	31.50%	53.20%	~3
FPN	2017	ResNet-101	800	36.20%	59.10%	5
Mask R-CNN	2018	ResNeXt-101-FPN	800	39.80%	62.30%	5
DetectoRS	2020	ResNeXt-101	1333	53.30%	71.60%	~4
YOLO*	2015	(Modified) GoogLeNet	448	-	57.90%	45
SSD	2016	VGG-16	300	23.20%	41.20%	46
YOLOv2	2016	DarkNet-19	352	21.60%	44.00%	81
RetinaNet	2018	ResNet-101-FPN	400	31.90%	49.50%	12
YOLOv3	2018	DarkNet-53	320	28.20%	51.50%	45
CenterNet	2019	Hourglass-104	512	42.10%	61.10%	7.8
EfficientDet-D2	2020	Efficient-B2	768	43.00%	62.30%	41.7
YOLOv4	2020	CSPDarkNet-53	512	43.00%	64.90%	31
DeTR	2020	ResNet-101	-	43.50%	63.80%	20
Swin-L	2021	HTC++	-	57.70%	-	-

Models marked with * are compared on PASCAL VOC 2012, while others on MS COCO.

2.1.2 Object Tracking

Object Tracking คือหนึ่งในการใช้กระบวนการ Deep learning ที่จะทำการโปรแกรมชุดข้อมูลเริ่มต้นที่ได้จากการตรวจจับวัตถุและนำมาระบุชื่อเฉพาะของแต่วัตถุ และสร้างกระบวนการติดตามวัตถุที่ตรวจจับได้ในแต่ละเฟรมของวีดีโอ[18]

การ Tracking มีการนำมาใช้เพื่อประโยชน์หลักคือ การติดตามวัตถุเมื่อวัตถุในช่วงที่ระบบตรวจจับไม่สามารถทำงานได้ การระบุ ID ให้กับวัตถุที่เราสนใจ และ การ Tracker มีความเร็วในการ Track ที่มากทำให้มีการประยุกต์ใช้จริงอย่างแพร่หลาย มีการประยุกต์ใช้การ Tracking ในหลายสถานการณ์ เช่นการตรวจดูบริเวณทางแยกที่มีสัญญาณไฟจราจร การติดตามนักกีฬาหรือลูกฟุตบอลในสนาม และ การติดตามจากกล้องหลายตัวให้สามารถมี ID เดิมได้ Tracker สามารถแบ่งออกเป็นสองประเภทหลักๆคือ Single Object Tracker และ Multiple Object Tracker

2.1.2.1 Single Object Tracker

Single Object Tracker [18]จะสามารถติดตามวัตถุได้ชนิดเดียวถึงแม้ว่าในเฟรมมีวัตถุหลายชนิด การ Tracking วิธีนี้จะมีความเร็วมาก โดยมีหลายหลักการพื้นฐานเช่น CSRT (Channel and Spatial Reliability Tracking) หรือ KCF (Kernelized Correlation Filter) tracker แต่ในปัจจุบันมีการนำกระบวนการ Deep Learning มาใช้ในกระบวนการ Tracking พบว่ามีค่าความแม่นยำ (Accuracy) ในการติดตามมากกว่าวิธีทั่วไปอย่างมาก ยกตัวอย่างเช่น GOTURN: Deep Learning based Object Tracking

2.1.2.2 Multiple Object Tracker (MOT)

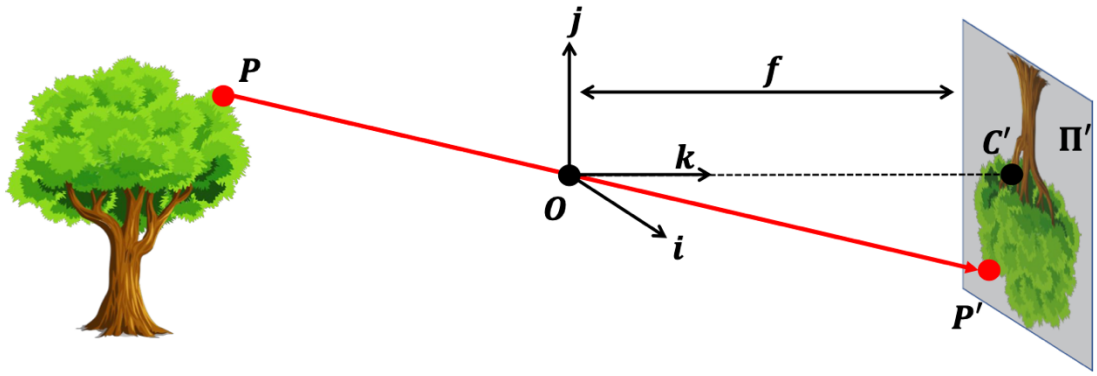
Multiple Object Tracker (MOT)[18]สามารถติดตามวัตถุได้หลายชนิดในเฟรมเนื่องจากมีการ Trained จากวัตถุด้วยข้อมูลมาก ซึ่งทำให้สามารถติดตามวัตถุได้หลายชนิดอย่างแม่นยำได้ในเวลาเดียวกันในขณะที่ยังมีความเร็วที่สูง โดยมีหลายรูปแบบที่ได้มีการสร้างขึ้นมา เช่น DeepSORT, JDEและ Center Track

2.1.3 Position Estimation

Pose Estimation มีความสำคัญมากในด้าน Computer vision โดยสามารถนำไปใช้ประโยชน์ได้หลายๆประเภททั้งในด้าน Robot navigation, augmented reality และงานในด้านอื่นอีกมากมาย.

2.1.3.1 Camera model

Pinhole Camera Model [19]คือสมการอย่างง่ายที่ใช้สำหรับอธิบายสมการทางคณิตศาสตร์ระหว่างพิกัด 3 มิติ กับ ระนาบของกล้อง โดยแสดงการคำนวณอย่างง่าย ดังรูปที่ 18



รูปที่ 18 ตัวอย่างพิกัดรูปภาพและวัตถุจริง

จากรูปที่ 18 กำหนดให้จุด O เป็นจุดศูนย์กลางกล้องมีค่าเท่ากับความยาวโฟกัส (Focal Length:f) เรียกว่า ระนาบของรูปภาพ (Image Plane) กำหนดให้จุดในพื้นที่ 3 มิติ มีค่าพิกัดคือ $p = (x, y, z)^T$ โดยสามารถเปลี่ยนค่าเป็นพิกัดบนระนาบรูปภาพ (Image Plane:Π') ให้อยู่ในพิกัด $p' = (x', y')^T$ กำหนดแกนตำแหน่ง O ได้คือ $(\mathrm{I}, \mathrm{j}, \mathrm{k})$ จากรูป แสดงให้เห็นว่า $[p', c', o]$ และ $[p, c, (0,0,z)]$ เป็นสามเหลี่ยมคล้ายกัน สามารเขียนสมการได้ดังสมการที่

$$p' = [x', y'] = \left[\frac{fx}{z}, \frac{fy}{z}\right]^T \qquad (8)$$

camera matrix model คือเมตริกที่เก็บค่าตัวแปรที่สำคัญในการแปลงพิกัดของวัตถุ p เป็นพิกัดของรูปภาพ p' โดยตัวแปรจะถูกเก็บอยู่ในรูปของเมตริก ตัวแปรแรกคือค่า c_x, c_y คือ ตัวแปรที่ใช้อธิบายความผลต่างของระยะระหว่างพิกัดรูปภาพ และ พิกัดรูปภาพดิจิตอล (Pixel)

$$p' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \frac{fx}{z} + cx \\ \frac{fy}{z} + cy \end{bmatrix} \qquad (9)$$

จากรูปที่ 18 c' มีจุดศูนย์กลางอยู่ที่ตำแหน่ง แกน k ซึ่งพิกัดรูปดิจิตอลมีจุดเริ่มต้นจากพิกัดมุมซ้ายด้านล่างของรูปภาพจึงจำเป็นต้องเลื่อนพิกัดดังสมการที่ 9 ตัวแปรต่อมาคือการกำหนดตัวแปร k, l คือตัวแปรในการแปลงค่าระหว่างเซนติเมตรและพิกเซล เมื่อค่า k=l จะมีความหมายว่าพิกเซลมีลักษณะเป็นรูปสี่เหลี่ยม เมื่อทำการคำนวณตัวแปรลงในสมการจะได้สมการดังสมการที่ 10

$$p' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} k\frac{fx}{z} + cx \\ l\frac{fy}{z} + cy \end{bmatrix} = \begin{bmatrix} \alpha\frac{x}{z} + c_x \\ \beta\frac{x}{z} + c_y \end{bmatrix} \qquad (10)$$

Homogeneous coordinate เนื่องจากการแปลงค่าเมตริก $p \rightarrow p'$ มีขนาดมิติที่ต่างกันจึงจำเป็นต้องมีการนำหลักการ Homogeneous Coordinate เข้ามาประยุกต์ใช้ โดยวิธี Homogeneous Coordinate คือการแปลงค่าพิกัดดังตัวอย่างคือ $p = (x', y')$ เป็นพิกัดใหม่ คือ $p = (x', y', 1)$ หรือสามารถแสดงให้เห็นได้อย่างง่ายคือการเปลี่ยนเวกเตอร์ $(v_1, v_2, \dots, v_n)$ เป็นเวกเตอร์ที่มีมิติเป็น $(v_1, v_2, \dots, v_n, 1)$ สามารถแปลงค่าออกมาได้โดยวิธีการตัวอย่างดังสมการที่ 11

$$p' = \begin{bmatrix} \alpha\frac{x}{z} + c_x \\ \beta\frac{x}{z} + c_y \\ z \end{bmatrix} = \begin{bmatrix} \alpha & 0 & c_x & 0 \\ 0 & \beta & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & c_x & 0 \\ 0 & \beta & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} p \qquad (11)$$

$$p' = \begin{bmatrix} x' \\ y' \\ z \end{bmatrix} = \begin{bmatrix} \alpha & 0 & c_x & 0 \\ 0 & \beta & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & c_x & 0 \\ 0 & \beta & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} p = Mp \qquad (12)$$

จากสมการตัวอย่างแสดงให้เห็นว่าเราสามารถแปลงค่าสมการเมตริกในระนาบรูปภาพ 2 มิติให้กลายเป็นสมการในพิกัด 3 มิติ

$$p' = Mp = \begin{bmatrix} \alpha & 0 & c_x \\ 0 & \beta & c_y \\ 0 & 0 & 1 \end{bmatrix} [I\ 0]p = K[I\ \ 0\ ]p \qquad (13)$$

Complete Camera matrix model

ค่าตัวแปรเพิ่มเติมที่อยู่ใน Camera matrix คือค่า Skewness และ Distortion โดยทั่วไปจะพบว่าค่า skewness จะมีมีค่าคือ 0 แต่เมื่อเกิดเหตุการณ์ skewness สมการ Camera Matrix จะถูกแก้ไขใหม่ดังสมการที่ 14

$$K = \begin{bmatrix} x' \\ y' \\ z \end{bmatrix} = \begin{bmatrix} \alpha & -\alpha cot\theta & c_x \\ 0 & \frac{\beta}{sin\theta} & c_y \\ 0 & 0 & 1 \end{bmatrix} \qquad (14)$$

ส่วนมากการคำนวณสมการ Camera Matrix จะประมาณให้ Distortion มีค่าคือ 0 ทำให้ ในสมการ Camera Matrix ประกอบด้วย 5 DOF คือ Focal length จำนวน 2 ค่า Offset จำนวน 2 ค่า และค่า Skewness 1 ค่า โดยตัวแปลทั้งหมดที่กล่าวข้างต้นมีชื่อเรียกว่า Intrinsic Parameter โดยส่วนใหญ่จะเป็นค่าที่สามารถหาได้จากการข้อมูลของกล้องที่มาจากการผลิต

Extrinsic Parameter เนื่องจากพิกัดของโลกในรูปแบบ 3 มิติ (3D World coordinate system) มีพิกัดที่แตกต่างกันไปในหลายรูปแบบจึงจำเป็นต้องรวมพิกัดการเคลื่อนที่เทียบกับโลกเข้าไปรวมกับพิกัดกล้อง โดยใช้ Rotational matrix R และ Translational vector T โดยกำหนดให้ตำแหน่งของกล่องเทียบกับโลกคือ P_w ทำให้สามารถหาคพิกัดของกล้องได้ ดังสมการที่ 15

$$P = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} p_w \qquad (15)$$

สามารถแทนค่าตัวแปรนี้ลงในสมการ 15 ได้สมการใหม่ออกมาคือสมการ 16

$$p' = K[R\ \ T]p_w = Mp_w \qquad (16)$$

โดยตัวแปร Rและ T ชื่อคือ Extrinsic Matrix เนื่องจากไม่ได้มีค่าขึ้นลงตามคุณสมบัติของกล้อง เนื่องจากตัวแปรข้างต้น สมการกล้องโดยทั่วไปจึงประกอบไปด้วย 11 DOF ประกอบด้วย 5 ส่วนแรกจาก Intrinsic Matrix 3 ส่วนจาก Rotational Matrix และอีกสามส่วนจาก Translational โดยสมการรูปเต็มของ Camera matrix มีลักษณะดังสมการ 17

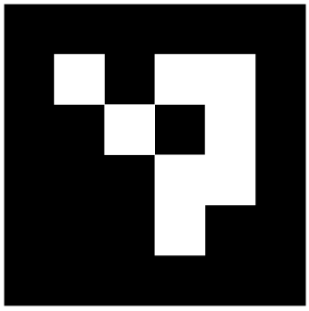
$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad (17)$$

2.1.4 ArUco Marker

ในปัจจุบันวิธีการที่ได้รับความนิยมมากสำหรับกระบวนการ Pose Estimation คือการใช้ binary square fiducial markers ประโยชน์หลักของเครื่องหมายสัญญาณเหล่านี้คือเครื่องหมายเดียวจะมีความสามารถเพียงพอสำหรับการคำนวณตำแหน่งของกล้อง นอกจากนี้การที่ค่าที่อ่านได้จากเครื่องหมายเหล่านี้มีค่าเป็น binary ทำให้ระบบมีความ Robust

ArUco (Augmented Reality University of Cordoba) เป็นคำสัญลักษณ์ที่ใช้แทนไลบรารี (library) ที่ถูกพัฒนาขึ้นโดย Rafael Muñoz และ Sergio Garrido จากมหาวิทยาลัยของ Cordoba ในสเปน ไลบรารีนี้ใช้สำหรับการตรวจจับและระบุเครื่องหมาย ArUco marker ซึ่งเป็นเครื่องหมายสี่เหลี่ยมที่มีเส้นขอบสีดำกว้างและมีรหัสภายในในรูปแบบของมาตริกซ์ไบนารี ไลบรารี ArUco มีการใช้งานกว้างขวางในการประยุกต์ใช้งานทางด้านคอมพิวเตอร์วิสัยทั้งในงานวิจัยและงานพัฒนาแอปพลิเคชันที่เกี่ยวข้องกับ Augmented Reality (AR)

เครื่องหมาย Aruco เป็นเครื่องหมายสี่เหลี่ยมที่ประกอบด้วยขอบสีดำกว้างและ Binary Matrix อยู่ภายในที่ใช้กำหนดตัวระบุ (id) ของเครื่องหมายนั้นขอบสีดำช่วยให้สามารถตรวจจับจากภาพได้ง่ายและเร็วขึ้น การถอดรหัส Binary ของ ArUco ทำให้สามารถระบุ ID หาค่า Error ของการตรวจจับภาพ และเพิ่มความสามารถในการตรวจจับได้ของ โดยขนาดของ ArUco Marker จะกำหนดจาก Internal Matrix ขึ้นกับขนาดของ Marker ยกตัวอย่างเช่น ArUco ขนาด 4x4 จะมีขนาดได้ 16 bits ดังรูปที่



*รูปที่ 19 ArUco Marker 4*4*

ในกระบวนการตรวจจับ ArUco marker บางครั้งเราอาจพบว่าเครื่องหมายมีการหมุนในสภาพแวดล้อม แต่กระบวนการตรวจจับต้องสามารถระบุได้ว่าเครื่องหมายเดิมถูกหมุนอย่างไรเพื่อที่จะระบุมุมของแต่ละมุมได้อย่างชัดเจน การทำนี้สามารถทำได้โดยใช้รหัสภายในของเครื่องหมายที่เป็นรหัสฐานสอง ซึ่งเป็นรหัสที่ถูกเข้ารหัสและนำมาใช้ในกระบวนการตรวจจับโดย ArUco Dictionary คือชุดของเครื่องหมาย ซึ่งข้อมูลที่เก็บคือ list ของ เลขฐานสองที่มาจากเครื่องหมายแต่ละชนิด ซึ่งคุณสมบัติหลักของ ArUco Dictionary คือ ขนาดของ Dictionary และขนาดของ Marker [20]

-ขนาดของ Dictionary คือ จำนวนของ Marker ทั้งหมดที่มีใน Dictionary

-ขนาดของ Marker คือขนาดของ Marker ในรูปแบบจำนวน Bits

2.2 Communication System

ในปัจจุบันกลไกการสื่อสารระหว่างยานพาหนะมีการเพิ่มจำนวนมากขึ้นเช่น DSRC,C-V2X และ 5G [5]เพื่อรับข้อมูลจาก ยานพาหนะคันอื่น โครงสร้างพื้นฐาน บนท้องถนนสามารถอธิบายกลไกในตัวอย่างได้คือ

2.2.1 LTE/4G/5G: Long-Term Evolution (LTE)[5]

LTE/4G/5G: Long-Term Evolution(LTE) เป็นการพัฒนาช่วงจาก 3G ไปสู่ช่วง 4G โดยมีอัตราการดาวน์โหลดสูงสุดอยู่ที่ 300 Mbit/s และอัปโหลดสูงสุดที่ค่า 75 Mbit/s ต่อมาได้มีการเปลี่ยนแปลงไปเป็น Forth-generation (4G) โดยมีความสามารถในการรับข้อมูลถึง 1 Gbit/s ในสถานการณ์แบบคงที่ และสามารถรับข้อมูลได้ถึง 100 Mbit/s สำหรับมือถือ ต่อมาได้มีการพัฒนาเครือข่าย 5G ซึ่งมีความเร็วเฉลี่ยในการดาวน์โหลด เร็วที่สุดถึง 494.7 Mbps ใน Verizon ซึ่งเร็วกว่า 4G ถึง 17.7 เท่า จากรายงานข้างต้นของ Verizon ความหน่วง(latency) พบว่า 5G มีความหน่วงน้อยกว่า 30 millisecond เร็วกว่า 4G 23 millisecond

5Gสามารถใช้ได้ในรูปแบบ low-band, mid-band หรือ high-band millimeter-wave

low band 5G มีความถี่เท่ากับ 4G คือ 600-900 MHz โดยที่ยังมีความเร็วในการดาวน์โหลดมากกว่า 4G อยู่ในช่วง5–250 Mbit/s เสาสัญญาณ low band 5G จะมีระยะสัญญาณใกล้เคียงกับเสาสัญญาณ 4G

mid Band 5G ใช้สัญญาณคลื่น microwave ที่ 1.7-4.7 GHz โดยมีความเร็วอยู่ที่ 100-900 Mbit/s โดยเสาร์สัญญาณมีระยะในการส่งสัญญาณหลายกิโลเมตรมีการนำไปใช้ในหลายๆพื้นที่

High Band 5G มีค่าความถี่ในการใช้อยู่ที่ 24-47 GHz มีความเข้าใกล้ Millimeter Wave ซึ่งทำให้ความเร็วในการ Download อยู่ในระดับ Gbit/s แต่มีข้อเสียคือ ระยะในการส่งสัญญาณจะถูกจำกัดได้มากเนื่องจากมีขนาดใหญ่ทำให้ทะลุผ่าน Material ได้ยาก.

2.2.2 DSRC (Dedicated short-range communications) [5]

DSRC (Dedicated short-range communications) เป็นโปรโตคอลสื่อสารประเภทV2X ที่มีการออกแบบพิเศษสำหรับการเชื่อมต่อระหว่างรถยนต์ DSRC เป็นไปตามมาตรฐาน IEEE 802.11p โดยมีความถี่ในการทำงานคือ 5.9 GHz ข้อความที่ผ่าน DSRC จะมีขนาดเล็กและความถี่ต่ำเนื่องจาก Bandwidth มีขนาดแคบ แต่ DSRC ยังสามารถให้การสื่อสารที่เชื่อถือได้ในความเร็ว 120 ไมล์ต่อชั่วโมง

2.2.3 C-V2X[5]

C-V2X คือการรวมเครือข่าย V2X ดั้งเดิมเข้ากับเครือข่ายเซลลูลาร์ ทำให้ 4G/5G ถูกนำมาใช้ช่วยเหลือในการการขับขี่แบบอัตโนมัติ C-V2X ยังสามารถพัฒนาได้จากการเนื่องจากการพัฒนา

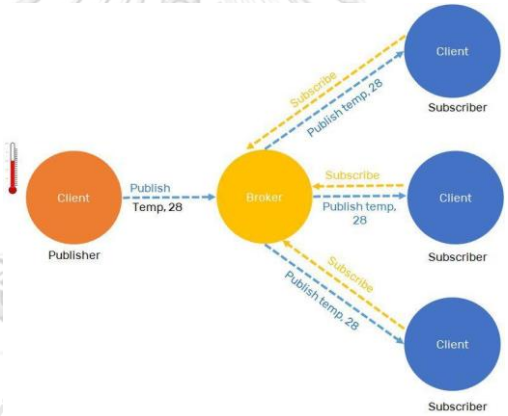
ของ Cellular ได้อีกด้วย C-V2X จะเหมาะกับสถานการณ์ การใช้ V2X ในตำแหน่งที่มีเครือข่าย Cellular อย่ากว้างขวาง

2.2.4 MQTT (Message Queue Telemetry Transport)

MQTT พัฒนาต่อมาจาก TCP/IP เป็น Protocol พื้นฐานที่ใช้ทั่วไปในระบบ IoT ที่สามารถยืนยันได้ว่าข้อมูลไม่หล่นหายในขณะการทำงาน แต่จะมีลักษณะการส่งข้อมูลแบบ one-to-many โดยส่งข้อมูลขนาดเล็ก MQTT ประกอบด้วย Broker (Server), Clients (Publisher/Subscriber) และ Topic

Topic คือ หัวข้อที่ใช้สำหรับส่งหรือรับข้อมูล

Broker คือตัวกลางที่มีหน้าที่ในการรับข้อมูลจาก Client (publisher) ในทุก Topic แล้วทำการจัดส่งข้อมูลไปยัง Clients(subscribe) ที่ทำการ Subscribe Topic ที่ต้องการ ปัจจุบันมี Cloud MQTT broker หรือ global broker ได้ในหลายๆเว็บไซต์ หรือสร้างใน network ของเรา ได้ด้วยดังรูปที่ 20 แสดงโครงสร้างของ MQTT. [21]

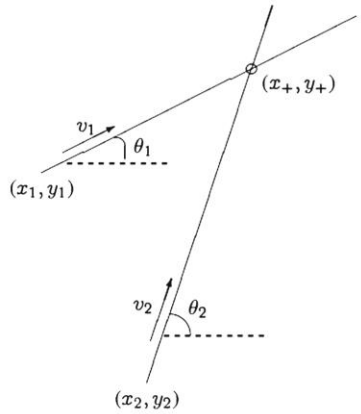


รูปที่ 20 กระบวนการทำงานและการส่งข้อมูลของ MQTT[21]

2.3 Decision system

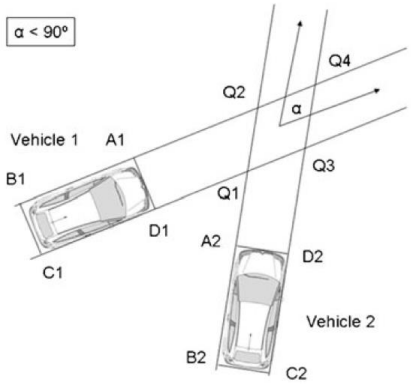
2.3.1 Time to Collision

Time to Collision (TTC) คือ เวลาที่รถจะเกิดการชนกันที่ ความเร็ว ระยะทาง และความเร่งใดๆ ระหว่างรถที่เราสนใจและรถที่ใกล้ที่สุด[22] ค่า TTC เมื่อกำหนดให้การเคลื่อนที่ของรถเป็นจุด สามารถคำนวณได้จาก ค่าตำแหน่งเริ่มต้น ค่าความเร็วและทิศทางของยานพาหนะ การมองรถเป็นจุดดังรูปที่ 21 จำเป็นต้องมีค่า Safety margin (δ) เข้ามาคำนวณเพื่อให้สามารถนำผลการคำนวณมาใช้จริงได้ [23]



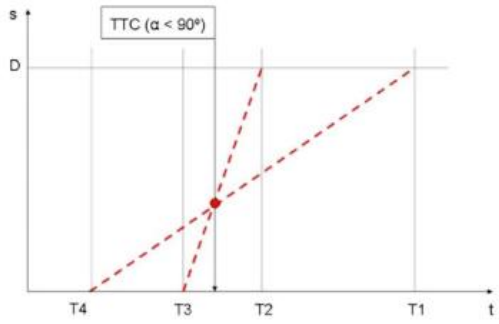
รูปที่ 21 วิธีการคำนวณค่า TTC โดยกำหนดให้รถมีขนาดเป็นจุด[23]

ในปี 2013 [23]ได้เสนอการคำนวณหาค่า TTC ระหว่างยานพาหนะสองคันที่เคลื่อนที่เข้าหากัน โดยกำหนดให้รูปร่างของรถให้มีลักษณะเป็นรูปสี่เหลี่ยมและเคลื่อนที่เข้าหากันด้วยมุม α และกำหนดให้บริเวณที่สามารถเกิดการชนกันได้คือ บริเวณ Q ต่อมาทำการคำนวณค่าเวลาที่รถเข้าสู่ในบริเวณ Q ตามตำแหน่งบริเวณมุมของรถ (A,B,C,D) ดังรูปที่ 22 จากรูปที่ ในงานวิจัยนี้จะวิเคราะห์ค่า TTC จากเหตุการณ์ที่ค่า $\alpha < 90^\circ$



รูปที่ 22 ตำแหน่งรถและถนนที่ทำมุมกันน้อยกว่า 90 องศา[23]

ยกตัวอย่างเหตุการณ์ให้รถทั้งสองคันเคลื่อนที่เข้าสู่ บริเวณ Q ด้วยความเร็วคงที่โดยให้รถคันที่สองมีความเร็วมากกว่ารถคันที่ 1 และอยู่ห่างกันระยะ 10 เมตร เมื่อนำข้อมูลที่ได้มาพิจารณาและทำการคำนวณหาค่าเวลาจากความเร็วของแต่ละตำแหน่งในกรอบของรถที่เข้าสู่บริเวณ Q และนำเวลาที่คำนวณมาสร้างกราฟแสดงความสัมพันธ์ระหว่าง ระยะทางและเวลาในตำแหน่งของรถ ดังรูปที่ 23 จะสามารถคำนวณหาค่า TTC ได้จากจุดตัดของกราฟ ตามสมการที่ 8



รูปที่ 23 กราฟแสดงความสัมพันธ์ ระหว่างเวลากับระยะทางของยานพาหนะทั้งสองคัน[23]

$$TTC = \frac{T_1T_3 - T_2T_4}{T_1 + T_3 - T_2 - T_4} \qquad (18)$$

2.3.2 Reaction time (tr)

Reaction time (tr) คือเวลาที่ใช้ตัดสินใจที่จะขับต่อหรือเบรก และหากหยุดเบรกโดยเวลาในการตอบสนองจะต่างกันไปตามสถานการณ์ มีหลายงานวิจัยได้ทำการสำรวจ มีค่าตั้งแต่ 0.7-3 วินาที ในงานวิจัยของ [24] มีการสำรวจและควบคุมพบว่า มีเวลาการตอบสนองของคนขับต่อการเบรกรถเฉลี่ยคือ 2.3 วินาที โดยในงานวิจัยนี้กล่าวว่า ครอบคลุมผู้ขับขี่ทุกประเภท

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

บทที่ 3

แนวคิดการออกแบบระบบป้องกันการชนสำหรับยานยนต์อัตโนมัติ ร่วมกับการสื่อสารระหว่างยานพาหนะและโครงสร้างพื้นฐานผ่านเครือข่าย 5G

3.1 เครื่องมือที่ใช้ในการทดลอง

อุปกรณ์รับข้อมูลภาพ

Logitech C920E WEBCAM

รูปที่ 24 กล้อง Logitech C920E WEBCAM

ตารางที่ 2 แสดงคุณสมบัติของกล้อง Logitech BRIO ULTRA HD PRO BUSINESS WEBCAM

Height: Width: Depth Cable : Length : Weight	43.3mm:94mm:71mm:1.5m:162g
Mega Pixel	3
Depth Field of View (dFoV)	78 degrees
Video Capture	1920 x 1080 @ 30 fps ¦ 1280 x 720 @ 30 fps
Camera Focal Range	fx: 1394.6 , fy: 1394.60
Camera Offset Coordinate	cx: 995.58 cy:599.32
Camera Distortion Value	k1: 0.1148 k2: -0.219 p1: 0.0012 p2: 0.0085 k3: 0.1127

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

หน่วยประมวลผลข้อมูล

Notebook Acer Nitro 5 AN515-55DM

รูปที่ 25 Notebook Acer Nitro 5 AN515-55DM

ตารางที่ 3 คุณสมบัติของ Notebook Acer Nitro 5 AN515-55DM

Processor	Intel Core i5-7300HQ Processor (6M L3 Cache, up to 3.50 GHz)
Chipset	Intel, NVIDIA
System Memory	4 GB DDR4 2400Mhz
Graphics Engine	NVIDIA GeForce GTX 1050

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

อุปกรณ์สำหรับการเชื่อมต่อ Wi-Fi

รูปที่ 26 HUAWEI AIS5G CPE

HUAWEI AIS5G CPE

ตารางที่ 4 คุณสมบัติของ HUAWEI AIS5G CPE

Dimension	90 mm x 96.6 mm x 178 mm
Weight	600 g
5G/4G	Communication Standard: 3GPP Release 15 Applicable Network: 5G/4G Network Mode: NSA/SA 5G Transmission Rate: 3.6 Gbps/250 Mbps (Theoretical value. The actual rate depends on the operator) 4G Transmission Rate: 1.6 Gbps/150 Mbps (Theoretical value. The actual rate depends on the operator) Antenna Type: Built-in 5G/4G primary and secondary antennas
WIFI	Transmission Standard: Wi-Fi 6, compatible with 802.11ac/n/g/b/a Transmission Rate: DBDC 2976 Mbps, 5 GHz 2402 Mbps (theoretical value), 2.4 GHz 574 Mbps (theoretical value) Frequency Band: 2.4 GHz & 5 GHz Antenna Type: Built-in dual-band Wi-Fi antennas

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

รูปที่ 27 OPAL T2

Turing OPAL T2

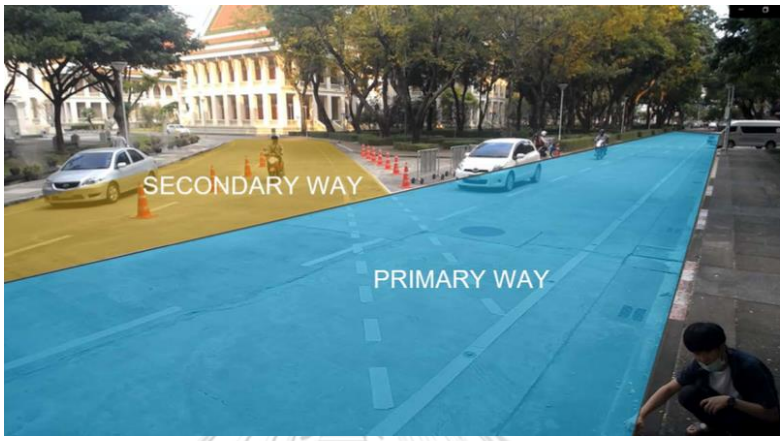
ตารางที่ 5 คุณสมบัติของ OPAL T2

Wheelbase	2894 mm
Passenger	10+2 pax
Curb Weight	1374 kg
Gross Vehicle Weight	2200kg
Maximum Speed	40 km/hr
Tire Size	185/60R15 / 0.30150 m (White car)
	185/65R14 / 0.29805 m (Blue car)

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

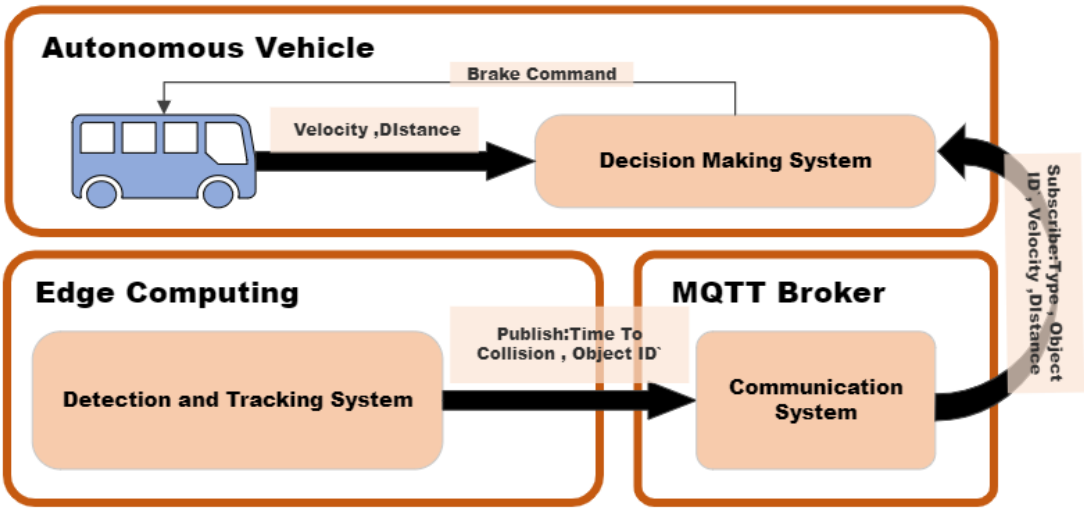
3.2 วิธีการทดลอง

การทดลองระบบป้องกันการชนสำหรับยานยนต์อัตโนมัติโดยใช้การตรวจจับวัตถุด้วยกล้อง ร่วมกับการสื่อสารระหว่างยานพาหนะอัตโนมัติและโครงสร้างพื้นฐานผ่านเครือข่ายเซลลูลาร์จะสนใจทำการทดสอบบริเวณหน้าคณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัยดังแสดงในรูปที่ 28



รูปที่ 28 ทางเอกและทางโทบริเวณหน้าคณะวิศวกรรมศาสตร์ จุฬาลงกรณ์ มหาวิทยาลัย

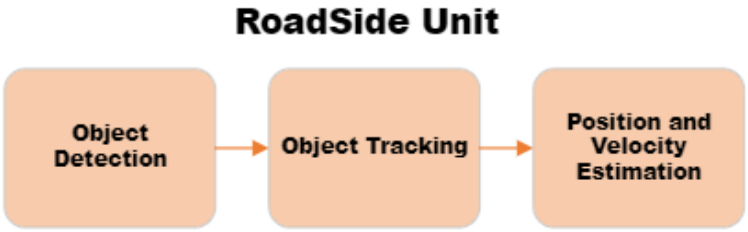
แบ่งระบบออกเป็น 3 ส่วนหลักๆคือระบบตรวจจับและติดตามวัตถุ ระบบการสื่อสารระหว่างรถและโครงสร้างพื้นฐาน และระบบการตัดสินใจในการชะลอรถ ดังรูปที่ 29 แสดงให้เห็นระบบ Decision making และระบบ Object detection and tracking มีการสื่อสารข้อมูลผ่านระบบ Communication system เมื่อรถอัตโนมัติเคลื่อนที่เข้าสู่พื้นที่แจ้งเตือน



รูปที่ 29 System Architecture ของระบบป้องกันการชนสำหรับยานยนต์อัตโนมัติโดยใช้การตรวจจับวัตถุด้วยกล้อง ร่วมกับการสื่อสารระหว่างยานพาหนะอัตโนมัติและโครงสร้างพื้นฐานผ่านเครือข่าย 5G

3.2.1 ระบบตรวจจับและติดตามวัตถุ

ในงานวิจัยนี้ติดตั้งกล้องและหน่วยประมวลผลบริเวณโครงสร้างพื้นฐานเพื่อตรวจจับและติดตามวัตถุ เพื่อหาข้อมูลสำหรับการคำนวณตัวแปรในการประกอบการตัดสินใจของรถอัตโนมัติ ซึ่งในงานวิจัยนี้จะเลือกส่งค่าตำแหน่งและความเร็วของรถอัตโนมัติเพื่อนำไปหาค่า Time to Collision เข้าสู่รถอัตโนมัติ ในการหาตัวแปรสำหรับการการคำนวณ Time to Collision ผู้วิจัยได้แบ่งระบบตรวจจับและติดตามวัตถุออกเป็นดังผังการทำงานดังรูปที่ 30



รูปที่ 30 Roadside system architecture

โดยแผนผังแสดงการทำงานโดยรวมของระบบตรวจจับ เริ่มต้นจากการรรับค่ารูปภาพของวีดีโอในแต่ละเฟรมต่อมานำภาพที่รับมาเข้าสู่กระบวนการประมวลผลภาพ ในงานวิจัยนี้เลือกใช้รูปแบบการตรวจจับ YOLOv4 เนื่องจากเป็นรูปแบบการตรวจจับที่มีลักษณะแบบ One-state model โดยมีข้อดีคือสามารถตรวจจับวัตถุได้แม่นยำและใช้เวลาน้อยกว่า Two-state model ต่อมานำผลที่ได้จากการตรวจจับเข้าสู่กระบวนการ Multi object Tracking เพื่อระบุ ID ตำแหน่งและค่าความเร็วของยานพาหนะที่เราสนใจ เพื่อส่งเข้าสู่ระบบการสื่อสารระหว่างยานพาหนะ และโครงสร้างพื้นฐาน และนำไปใช้ในการตัดสินใจในการเคลื่อนที่ของตัวรถต่อไป

3.2.1.1 ระบบการตรวจจับวัตถุ (Detection System)

เนื่องจากกระบวนการตัดสินใจของแบบจำลองรถอัตโนมัติในบริเวณทางร่วมเลือกใช้ค่า Time to Collision เป็นหลัก ระบบตรวจจับจึงจำเป็นต้องตรวจจับยานพาหนะจำเป็นต้องหาค่าระยะทางและเวลาของรถที่ทำการตรวจจับออกมา กระบวนการนี้จัดทำเพื่อตรวจจับวัตถุจากภาพที่นำเข้าโดยกล้อง ด้วยกระบวนการ Image Object Detection โดยได้ผลลัพธ์คือ ตำแหน่งและชนิดของวัตถุโดยจะได้ เอาต์พุตออกเป็นค่า พิกัด Bounding Box , Number of Class และ ค่า ความมั่นใจ Confidence Level เมื่อเฟรมภาพเปลี่ยนไปกระบวนการ Image Object Detection ยังมีการดำเนินการอยู่แต่ไม่สามารถระบุได้ว่าวัตถุเป็นวัตถุเดิม เพื่อหาความเร็วและระยะทางเพื่อนำไปคิดหาค่า Time to Collision ในการตัดสินใจของระบบจำลองการเคลื่อนที่ของรถอัตโนมัติ กระบวนการ Tracking จึงจะถูกนำมาใช้ในงานวิจัยนี้โดยการ Tracking มีวัตถุประสงค์ในการระบุ Identification ของวัตถุเพื่อให้สามารถรับรู้ได้ว่าวัตถุที่ เปลี่ยนตำแหน่งเป็นวัตถุชนิดเดิม โดยหลักการของกระบวนการ

Tracking จะนำเข้า Bounding boxes ที่คำนวณจากกระบวนการ Object Detection มาคำนวณและทำการคืนค่าเป็น Bounding Box และ ID ของวัตถุ ซึ่งกระบวนการ Image Object Detection และกระบวนการ Tracking มีในปัจจุบันมีวิธีการในการเลือกใช้หลายรูปแบบโดยสามารถเลือกใช้ได้จากข้อจำกัดและวัตถุประสงค์ในการใช้งาน

3.2.1.1.1 ซอฟแวร์

สำหรับกระบวนการ Object detection ในงานวิจัยนี้เลือกใช้ YOLO (You Only Look Once) Detection model เนื่องจากปัจจุบัน model single layer มีการพัฒนาเพิ่มขึ้นอย่างมากเนื่องจากเป็น Model ที่มีความเร็วในการประมวลผลสูงและในปัจจุบันมีการพัฒนาอย่างต่อเนื่องจึงทำให้ Model single layer มีความแม่นยำที่สูงขึ้นอย่างมากดังรูปที่ 31

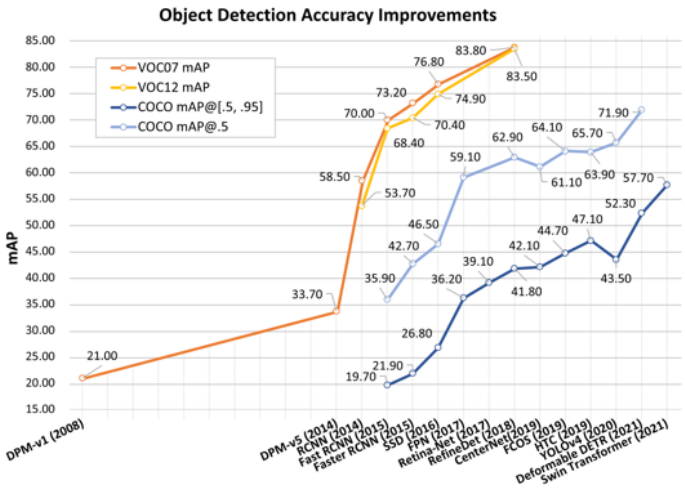


Fig. 3. ***Accuracy improvement of object detection on VOC07, VOC12, and MS-COCO datasets. Detectors in this figure: DPM-v1 [13], DPM-v5 [37], RCNN [16], SPPNet [17], Fast RCNN [18], Faster RCNN [19], SSD [23], FPN [24], Retina-Net [25], RefineDet [38], TridentNet [39] CenterNet [40], FCOS [41], HTC [42], YOLOv4 [22], Deformable DETR [43], and Swin Transformer [44].***

รูปที่ 31 กราฟแสดงความสามารถในการตรวจจับวัตถุของแต่ละโมเดลในปัจจุบัน[25]

จากงานวิจัยสำรวจของ[25]ได้มีการสำรวจว่าโมเดล Single Layer ได้มีการพัฒนามาอย่างต่อเนื่อง ดังรูปที่

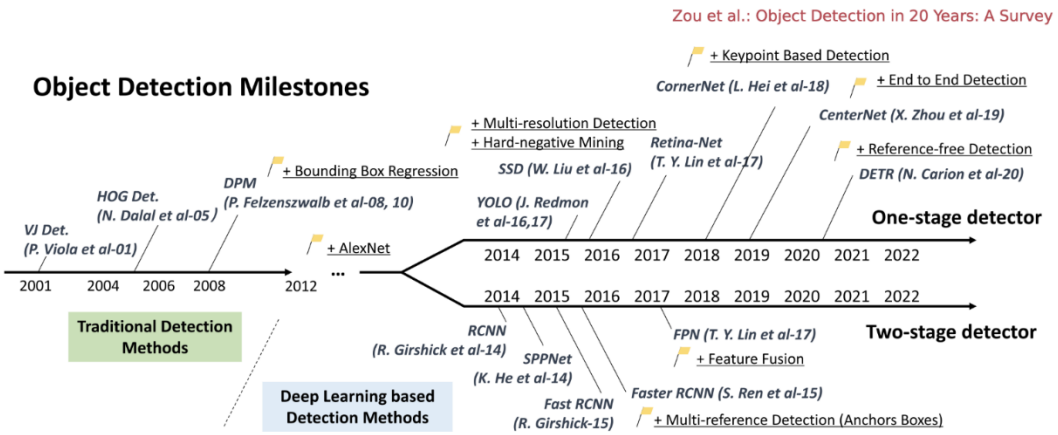
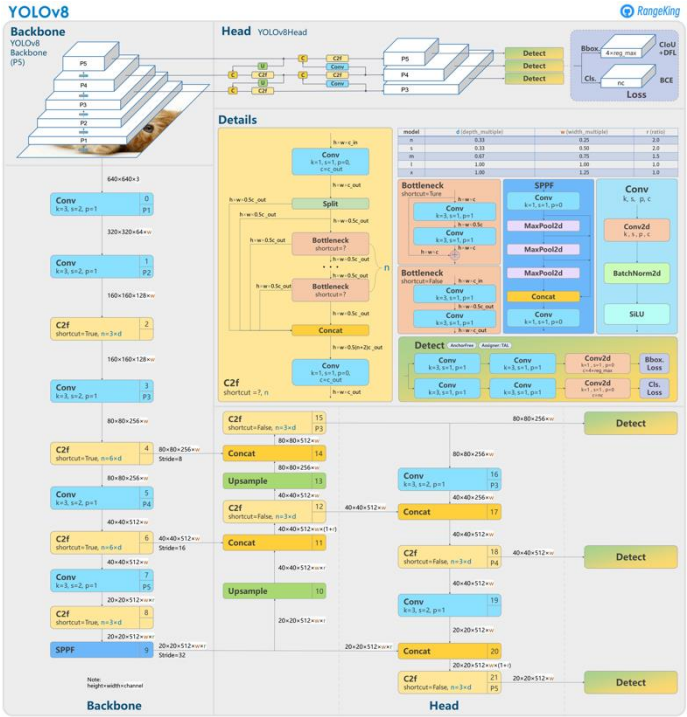


Fig. 2. ***Road map of object detection. Milestone detectors in this figure: VJ Det. [10], [11], HOG Det. [12], DPM [13], [14], [15], RCNN [16], SPPNet [17], Fast RCNN [18], Faster RCNN [19], YOLO [20], [21], [22], SSD [23], FPN [24], Retina-Net [25], CornerNet [26], CenterNet [27], and DETR [28].***

รูปที่ 32 การพัฒนาของกระบวนการตรวจจับวัตถุในปัจจุบัน[25]

YOLO เป็นหนึ่งในโมเดลที่นิยมใช้ในหลายด้านอย่างมากในปัจจุบันเนื่องจาก YOLO model มีการพัฒนากระบวนการอย่างต่อเนื่องจนถึงปัจจุบัน YOLO model ถูกสร้างขึ้นจาก Joseph et al. ในปี 2015[26] เป็น One stage detector แรกที่มีกระบวนการ Deep Learning Yolo ถูกพัฒนาในหลายรูปแบบรวมถึงรูปแบบ Fast Version ทำให้สามารถมีความเร็วในการประมวลผลได้สูงสุดถึง 155 fps จากชุดข้อมูล VOC07 ที่ mAP=52.7 % และสำหรับ Enhanced Version มีความเร็วที่ 45fps ที่ mAP= 63.4% หลักการทำงานของ YOLO จะแบ่ง รูปภาพออกเป็น Region , Predicts Bounding Boxes และค่าความน่าจะเป็น ซึ่งทำให้ YOLO สามารถประมวลผลได้อย่างรวดเร็วแต่ข้อเสียของ YOLO คือ เมื่อเทียบการระบุตำแหน่งกับ Two Stage Detector การระบุตำแหน่งจะมีความคลาดเคลื่อนมากกว่าโดยจะสามารถเห็นได้ชัดเจนใน YOLO model ที่มีขนาดเล็ก ซึ่งในปัจจุบันได้มีการพัฒนาของ YOLO อย่างต่อเนื่องเพื่อทำการแก้ปัญหาเหล่านี้ถึง YOLO version 7 (YOLOv7) ซึ่งได้พัฒนาโดยใช้ขั้นตอนการพัฒนามาจาก Yolov4 ซึ่ง ใน YOLOv7 นั่นสามารถแสดงประสิทธิภาพได้ดีในด้าน Speed และ Accuracy ที่มีความเร็วในการทำงานได้อยู่ในช่วง 5-160 fps โดยในงานวิจัยนี้ได้เลือกใช้ Model YOLO ล่าสุดที่ได้พัฒนาจนถึงปัจจุบัน คือ YOLOv8 [27]โดยได้มีการปรับเปลี่ยน Backbone Network, anchor-free detection head, และค่า Lost function ทำให้ Yolov8 มีประสิทธิภาพมากขึ้นและสามารถ ทำงานได้ใน Hardware ได้อย่างกล้างขวางในการใช้ GPU และ CPU มากขึ้นดังรูปที่ 33



รูปที่ 33 โครงสร้างของ YOLOv8 ในปัจจุบัน[27]

Yolov8 มีการพัฒนาบน Dataset หลักคือ COCO Dataset (Common Objects in Context Dataset) COCO Dataset เป็น Dataset ที่มีมาตรฐานสำหรับการใช้ทดสอบ Benchmark เพื่อที่นำมาประเมินผลของ Object Detection model สามารถเปรียบเทียบผลลัพธ์ที่ได้จากการประมวลผลชุดข้อมูล COCO จากค่า FPS และ mAP โดย YOLOv8 มีการ Train Model จากชุดข้อมูลและได้ทำการเก็บค่าตารางที่ 6 แสดงค่า mAP และ ค่า FPS ของ YOLOv8 ในแต่ละ model โดยจะประกอบด้วย YOLOv8n (nano), YOLOv8s (small), YOLOv8m (medium), YOLOv8l (large), YOLOv8x [28]

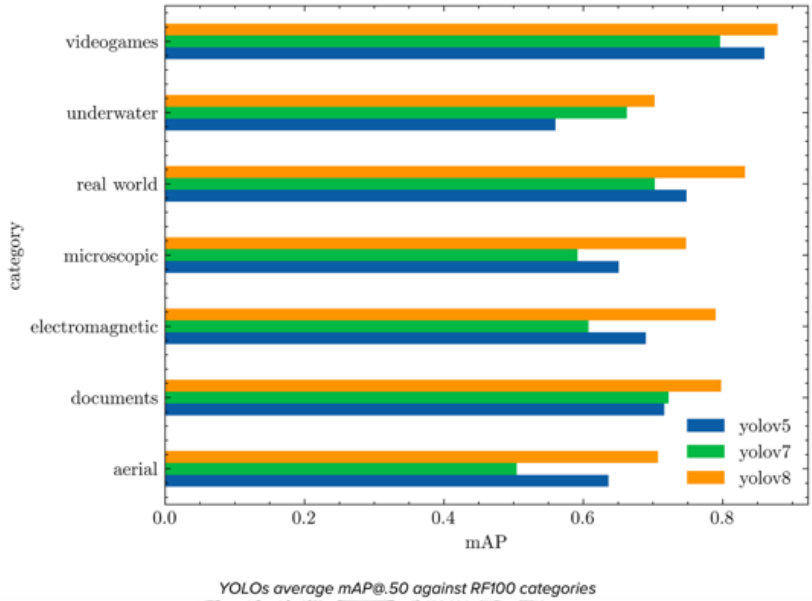
ตารางที่ 6 ค่าความสามารถของ YOLOv8 . ในทุก Model[28]

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

- **mAP^{val}** values are for single-model single-scale on COCO val2017 dataset.
  Reproduce by `yolo val detect data=coco.yaml device=0`
- **Speed** averaged over COCO val images using an Amazon EC2 P4d instance.
  Reproduce by `yolo val detect data=coco128.yaml batch=1 device=0|cpu`

จากข้อมูลการ Train Dataset ของหน่วยงาน[29] Roboflow ได้ทำการเปรียบเทียบค่าของการทดสอบโมเดล YOLOv5, YOLOv7, YOLOv8 กับชุดข้อมูล RF100 ที่ข้อกำหนดเดียวกันและพบว่า

YOLOv8 แสดงผลลัพธ์ได้ดีในด้านการตรวจจับและความเร็วจากกราฟดังรูปที่ 34 ผู้วิจัยจึงเลือกใช้โมเดล YOLOv8 ที่ถูก Train ด้วยชุดข้อมูล COCO เพื่อทำการตรวจจับวัตถุ

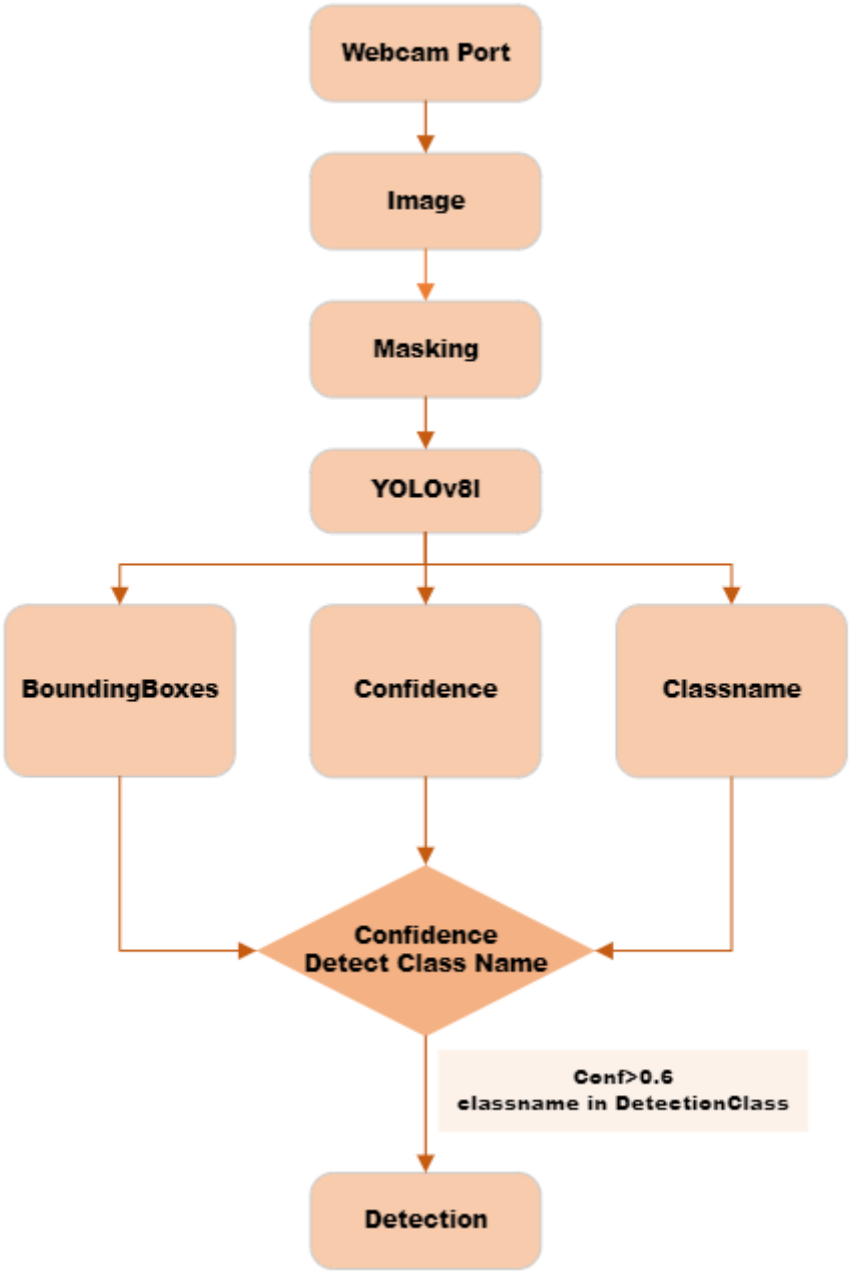


รูปที่ 34 การเปรียบเทียบค่าความแม่นยำของ Yolov5 Yolov7 และ Yolov8 และปริมาณการนำไปใช้ในงานต่างๆ ในปัจจุบัน[29]

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

3.2.1.1.2 ผังการทำงานส่วนระบบการตรวจจับวัตถุ

กระบวนการของการตรวจจับที่ได้จัดทำจะมีลักษณะการทำงานตามแผนผังดังรูปที่ 35



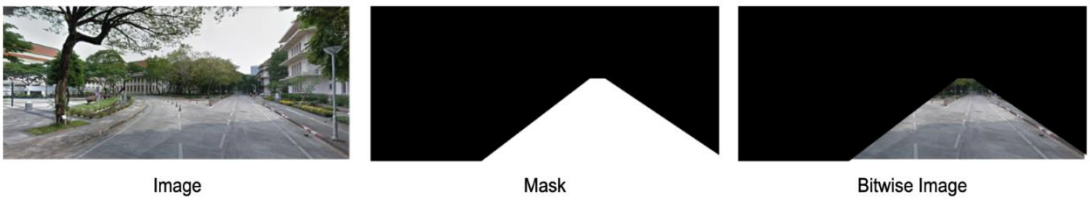
รูปที่ 35 แผนผังการทำงานของระบบตรวจจับวัตตถุ

3.2.1.1.5 ลักษณะการทำงานของระบบตรวจจับวัตถุ

ขั้นตอนที่ 1 นำภาพเข้าสู่กระบวนการตรวจจับวัตถุด้วยไลบรารีของ Open CV ด้วยคำสั่ง cv2.capture โดยในงานวิจัยนี้ ทำการทดลองการตรวจจับด้วยกล้องแบบ Realtime โดยจะ

กำหนดให้อ่านข้อมูลที่ส่งเข้าสู่กล้องจากการกำหนดพอร์ตในการรับข้อมูลตัวอย่างคือ cv2.capture(0) คือการรับข้อมูลจากกล้องที่พอร์ท 0

ขั้นตอนที่ 2 คือการนำภาพเข้าสู่กระบวนการ Masking เพื่อกำหนดขอบเขตของภาพในการตรวจจับวัตถุ มีวิธีอย่างง่ายคือ สร้าง Mask ที่เป็นขอบเขตของรูป ต่อมานำ Mask ทำกระบวนการ Bitwise กับรูปที่อ่านได้จาก Webcam ทำให้สามารถกำหนดขอบเขตได้ดังรูปที่ 36



รูปที่ 36 การตีกรอบบนรูปภาพเพื่อทำให้สามารถตรวจจับได้ภายในบริเวณที่สนใจตรวจจับ

ขั้นตอนที่ 3 คือ การนำรูปที่ได้นำเข้าสู่กระบวนการ Bitwise เข้าสู่กระบวนการ Object ผ่านโมเดล YOLOv8l โดยจากแผนผังดังรูปที่ 33เมื่อทำการตรวจจับค่าผลลัพธ์ ที่ได้รับคือ ค่าพิกัด Bounding Boxes คือ x1,y1,x2,y2 Class คือ ตัวแปรตัวเลขที่ [1…] ซึ่งสามารถนำไปเปรียบเทียบกับชื่อของ Class Confidence คือ ค่าความมั่นใจในการตรวจจับเป็นเลขทศนิยม 0-1

ขั้นตอนที่ 4 ทำการเปรียบเทียบชื่อของ Class กับ ตำแหน่งที่ได้ออกมาจาก YOLOv8 COCO dataset โดยชื่อของ class ที่สามารถตรวจจับได้มีดังต่อไปนี้ ซึ่งในการนำไปใช้งานต่อ ผู้วิจัยกำหนดเงื่อนไขคือให้ส่งข้อมูล Class ["bicycle","car","motorcycle","truck"] และมีค่า Conf ที่มากกว่า 0.6 สำหรับใช้ต่อในกระบวนการ Tracking เท่านั้น

3.2.1.2 ระบบติดตามวัตถุ (Object Tracking)

เนื่องจากจำเป็นต้องคำนวณหาค่าความเร็วของรถที่เคลื่อนที่ผ่านบริเวณที่ต้องการในการนำไปใช้คำนวณต่อเพื่อหาค่า Time to collision ในงานวิจัยนี้จึงจำเป็นต้องระบุค่า ID ของรถให้ได้เพื่อที่จะสามารถมั่นใจได้ว่ายานพาหนะที่ตรวจจับเป็นยานพาหนะคันเดิม ในงานวิจัยนี้ผู้วิจัยเลือกใช้ Deep SORT Tracking Model สำหรับกระบวนการติดตามวัตถุโดยมีกระบวนการดังผังการทำงานในรูปที่ 37

รูปที่ 37 แผนผังแสดงข้อมูลที่นำเข้าจากระบบตรวจจับเข้าสู่ระบบติดตามวัตถุ

3.2.1.2.1 Deep SORT Tracking

Deep SORT คือ อัลกอลิทึมในการติดตามสิ่งของโดยจะระบุ ID ให้วัตถุแต่ละชิ้น โดย Deep SORT [18] เป็นอัลกอริทึมที่ต่อยอดมาจาก SORT (Simple Online Realtime Tracking) โดยการเพิ่ม Deep learning เข้าสู่ SORT และอธิบายลักษณะเพื่อลดการสลับกันในการติดตาม โดย SORT เป็นแนวทางในการติดตามวัตถุโดยใช้วิธีการพื้นฐานเหมือน Kalman filters หรือ Hungarian algorithms ซึ่ง SORT ประกอบด้วยวิธีการ 4 ขั้นตอนคือ

1) กระบวนการ Detection คือกระบวนการตรวจจับวัตถุโดยจะตรวจจับวัตถุในเฟรมที่ต้องการติดตามวัตถุและส่งข้อมูลไปขั้นตอนถัดไป

2) กระบวนการ Estimation คือกระบวนการการส่งเฟรมที่ตรวจจับได้ไปยังเฟรมถัดไป โดยประมาณตำแหน่งของเป้าหมายในเฟรมถัดไปโดยประมาณให้ความเร็วคงที่ และเมื่อการตรวจจับเขาใกล้เป้าหมายที่คาดการณ์ Bounding box จะถูกอัปเดตให้มีค่าเขาใกล้สถานะเป้าหมายและค่าความเร็วจะถูก แก้ให้เหมาะสมด้วย Kalman filter.

3) กระบวนการ Data Association เป็นขั้นตอนการเชื่อมโยงข้อมูลระหว่าง Bounding Box ที่ได้จากการ Estimation และ Detection ทำการคำนวณค่า Cost Matrix โดยใช้วิธีการ Intersection Over Union (IoU) เพื่อหาค่าระยะทางระหว่าง bounding box ที่ได้จากการตรวจจับและขอบเขตที่คาดการณ์ไว้ของเป้าหมายที่มีทั้งหมด นำค่า IoU มาเปรียบเทียบกับค่า Threshold ถ้าค่าน้อยกว่าเกณฑ์จุดนั้นจะถูกเอาออกซึ่งวิธีนี้ช่วนยให้ ID ของวัตถุที่ต้องการ track ไม่สลับกัน.

4) Creation and Deletion of Track Identities เป็นวิธีการในการสร้างและลบ IDs ชื่อเฉพาะตัวจะถูกสร้างขึ้นและสามารถถูกลบได้เมื่อค่า IoU ที่รวมกันแล้วน้อยกว่า IoU min ระบบจะหยุด Track

โดย Deep SORT ใช้เมทริกซ์ที่การเชื่อมโดยที่ดีกว่า SORT ทำให้สามารถบอกลักษณะที่ปรากฏและลักษณะการเคลื่อนไหวของวัตถุได้ทำให้ Deep SORT เป็นอัลกอริทึมที่สามารถ Track ความเร็วการเคลื่อนที่และลักษณะของวัตถุได้

3.2.1.2.2 กระบวนการทำงานของระบบติดตามวัตถุ

ขั้นตอนที่ 1 สร้างตัวแปร Tracker ในรปแบบ Dictionary โดยค่า Tracker.py จะรับตัวแปร Class Name แต่ละชนิดที่ต้องการนำเข้าสู่กระบวนการ Tracking สามารถยกตัวอย่างการสร้างตัวแปร Tracking ได้ดังแผนผังที่ดังรูปที่ 39

รูปที่ 38 แผนผังแสดงวิธีการใช้งานระบบติดตามวัตถุ

โดยผู้วิจัยเลือกใช้กระบวนการในการสร้าง Tracker ของทุก ClassName ที่ต้องการทำการ Tracking เนื่องจาก Deep SORT Tracker ไม่สามารถรับค่า ClassName ได้จึงทำการสร้างตัวแปรเพื่อเก็บค่า ClassName

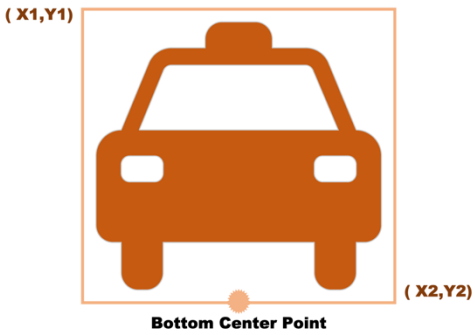
ขั้นตอนที่ 2 ทำการนำเข้าตัวแปรที่ได้จากการ Detection โดยเปรียบเทียบค่าที่ได้จากการ Detection มี ClassName ตรงกับค่า Tracker ตัวเพื่อนำตัวแปรเข้าสู่กระบวนการ Tracking

ขั้นตอนที่ 3 เก็บข้อมูลที่ได้จากกระบวนการ Tracking โดยประกอบด้วย x1,y1,x2,y2 และค่า Track_id ต่อมาสามารถนำค่า ClassName ออกมาได้เนื่องจากกำหนดชื่อ Class ก่อนที่ทำกระบวนการ Tracking

ขั้นตอนที่ 4 นำข้อมูลที่ได้รับจากการ Tracking ไปใช้ในกระบวนการ Visualize ด้วยการวาด Bounding Boxes, Label ClassName และ Tracking ID ลงบนภาพ ต่อมาสร้างจุดศูนย์กลางบริเวณขอบล่างของกล่องดังรูปที่ 39 สำหรับคำนวณหาระยะทางและความเร็วของยานพาหนะ จากสมการ

$$Bottom\ Center = \left(\frac{x_1+w}{2}, y_2\right) \qquad (19)$$

CHULALONGKORN UNIVERSITY



รูปที่ 39 ตำแหน่งสำหรับหาจุดเพื่อนำเข้าสู่กระบวนการหาค่าตำแหน่ง

3.2.1.3 ระบบการประมวลผลหาค่าตำแหน่งและความเร็วของวัตถุ (Position and Velocity Estimation)

การหาระยะของวัตถุที่ ตรวจจับได้มีหลายวิธีโดยทั่วไปมีการ ใช้ Depth Camera โดยทั่วไปจะประกอบด้วย Lidar Depth Camera และ Stereo Depth Camera เนื่องจากในงานวิจัยนี้จำเป็นต้องการวัดระยะของ Object ที่มีระยะมากกว่าขอบเขตของกล้อง Depth Camera ที่มีในท้องตลาด จึงจำเป็นต้องคิดวิธีการคำนวณหาระยะทางของยานพาหนะจากตัวแปรควบคุมของระบบที่สามารถกำหนดได้ คือ กล้องที่ติดตั้งไม่เคลื่อนที่ระหว่างการใช้งาน ยานพาหนะเคลื่อนที่อยู่บริเวณระนาบของถนน และหาระยะจากการใช้กล้องสองมิติ

3.2.1.3.1 สมการการหาระยะของวัตถุจากกล้องสองมิติ

การหาระยะของวัตถุโดยทั่วไปสามารถหาได้จากสมการการคำนวณของกล้องสองมิติ (2D Camera Equation) ดังสมการที่ 20

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad (20)$$

โดยค่า x,y,z คือตำแหน่งของวัตถุที่เราสนใจ u,v คือตำแหน่ง Pixel ของรูป จากข้อกำหนดที่ว่ายานพาหนะต้องวิ่งอยู่บนระนาบของถนน และกล้องอยู่ในบริเวณเดิมระหว่างการทำงาน ทำให้เกิดแนวคิดได้ดังนี้

1) จากการสมการที่กล่าวข้างต้นมีตัวแปรที่รู้คือ $\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$ คือค่าที่ได้จากคุณสมบัติของกล้อง และค่า $\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$ คือค่า Pixel ของรูปภาพที่สามารถคำนวณหาได้จากการ Detection and Tracking Process โดยค่าที่เราต้องการหาคือ ตัวแปร $\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$ ของวัตถุที่ต้องการหาตำแหน่ง

2) เมื่อเราสามารถหาสมการระนาบของถนนเทียบกับกล้องได้และ จุดตำแหน่งของภาพที่ทำการตรวจจับได้อยู่บนระนาบนั้นจะสามารถหาตำแหน่งของวัตถุบนระนาบนั้นเทียบกับกล้องได้ จากแนวคิดสองข้อนี้ผู้วิจัยจึงทำการพิสูจน์สมการว่าสามารถคำนวณตำแหน่งของจุดบนระนาบจากพิกัด Pixel ที่สร้างขึ้นได้ตำแหน่งถูกต้องหรือไม่ ทดสอบการคำนวณด้วย โปรแกรม MATLAB มีขั้นตอนดังนี้

กำหนดค่าตัวแปรพิกัดกล้อง

$$(x_{cam}, y_{cam}, z_{cam}) = (0,0,0) \qquad (21)$$

กำหนดค่า Transformation and Rotational Matrix(Extrinsic Matrix)

$$\begin{bmatrix} 1 & 0 & 0 & x_{cam} \\ 0 & 1 & 0 & y_{cam} \\ 0 & 0 & 1 & z_{cam} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad (22)$$

กำหนดค่า Intrinsic Matric ให้ค่า (fx, fy, cx, cy) = (100,100,0,0)

$$\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 100 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (23)$$

คำนวณหาค่า Camera Matric

$$Camera\ Matric = Intrinsict\ Matrix \times \text{Extrinsic Matrix} = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad (24)$$

สร้างระนาบจากการกำหนดค่า พิกัด 3 จุดคือ

$$p_1 = (2,3,10)\ , p_2 = (9,9,15), p_3 = (4,8,11) \qquad (25)$$

คำนวนหาค่า Plane Equation จาก

$$(p2 - p_1) \times (p_3 - p_1) = (7,6,5) \times (2,5,1) = (-19,3,23) \qquad (26)$$

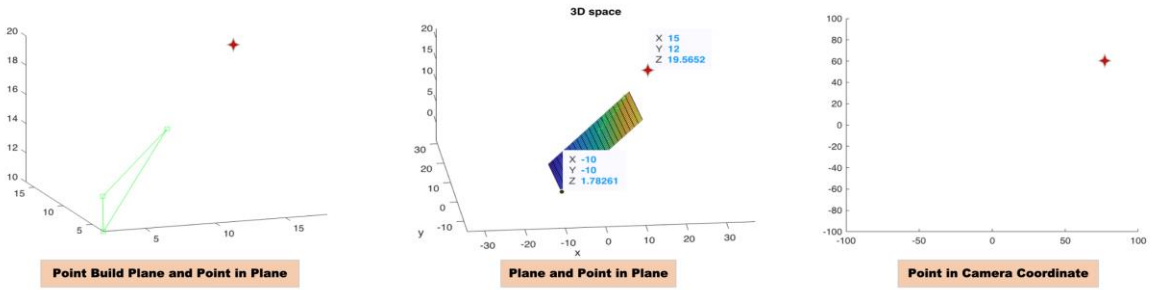
$$Ax + By + Cz + D = -19x + 23y + 23z + D = 0 \qquad (27)$$

เมื่อแทนค่า จุด p_1 ลงในสมการจึงได้ค่าตามสมการที่ 27 ได้ผลลัพธ์เป็น

$$Ax + By + Cz + D = -19x + 23y + 23z - 201 = 0 \qquad (28)$$

กำหนดจุดที่อยู่บนระนาบ มีพิกัด คือ $P_{pip} = (15,12,19.57)$

รูปที่ 40 แสดงพิกัดที่ทดลองคำนวณและแสดงผลด้วยโปรแกรม MATLAB



รูปที่ 40 ตัวอย่างในการตรวจสอบหลักการของสมการของกล้อง 2 มิติ

นำพิกัด P_{pip} มาทำการคำนวณหาค่าพิกัดบนกล้องดังสมการ 29

$$s\begin{bmatrix}u\\v\\1\end{bmatrix}=\begin{bmatrix}100 & 0 & 0 & 0\\0 & 100 & 0 & 0\\0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}15\\12\\19.57\\1\end{bmatrix} \qquad (29)$$

$$\begin{bmatrix}su\\sv\\s\end{bmatrix}=\begin{bmatrix}1500\\1200\\1957\end{bmatrix} \qquad (30)$$

$$s\begin{bmatrix}u\\v\\1\end{bmatrix}=19.57\begin{bmatrix}76.67\\61.3\\1\end{bmatrix} \qquad (31)$$

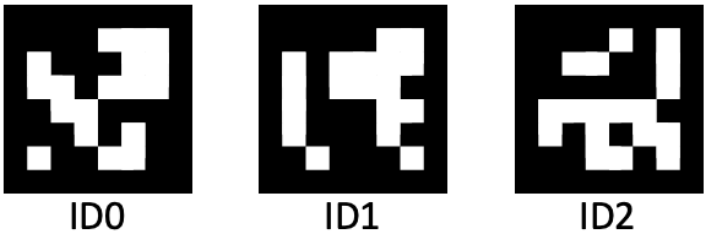
ทำการคำนวณย้อนกลับกำหนดให้เพิ่มสมการระนาบในการคำนวณเพื่อคำนวณหาค่าพิกัดจากสมการโดยรู้ค่า Pixel

$$\begin{bmatrix}s76.67\\s1.33\\s\\0\end{bmatrix}=\begin{bmatrix}100 & 0 & 0 & 0\\0 & 100 & 0 & 0\\0 & 0 & 1 & 0\\-19 & 3 & 23 & -201\end{bmatrix}\begin{bmatrix}x\\y\\z\\1\end{bmatrix} \qquad (32)$$

เมื่อคำนวณสมการ พบว่า ค่า s, x,y,z มีค่าเป็นพิกัดเดิมและสามารถใช้งานได้ในการคำนวน ทดสอบการนำไปใช้ควบคู่กับกระบวนการตรวจจับวัตถุ

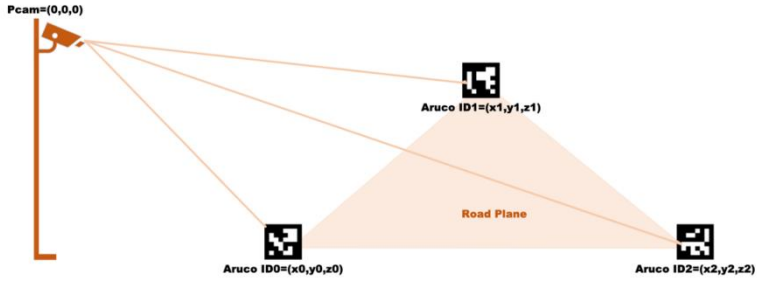
3.2.1.3.2 ทดลองสร้างสมการระนาบเพื่อนำไปใช้สร้างระนาบของถนน

ขั้นตอนที่ 1 กำหนดวิธีในการสร้างสมการระนาบโดยใช้วิธีการหาพิกัดจุด 3 ค่า ในงานวิจัยนี้ใช้การหาพิกัดของจุดจาก ArUco Marker โดยเลือกใช้ ArUco Marker 3 จุด ขนาด 6x6_100 สร้างโดยเว็บไซต์ กำหนดให้โปรแกรมทำการตรวจจับ ArUco ID = 0,1,2 ดังรูปที่ 41 และทำการตรวจจับ ArUco Marker และเอาต์พุตค่าตัวแปร Translational Vector ของ Aruco Marker ของแต่ละ ID



รูปที่ 41 ค่าชนิดของ ArUco Marker ที่นำมาใช้ในการทดลอง

การสร้างจุดบนถนนจะทำการสร้างโดยการนำ ArUco Marker วางบนถนนบริเวณ 3 จุด เพื่อสร้างหาพิกัดและนำมาใช้คำนวณต่อไปดังรูปที่ 42

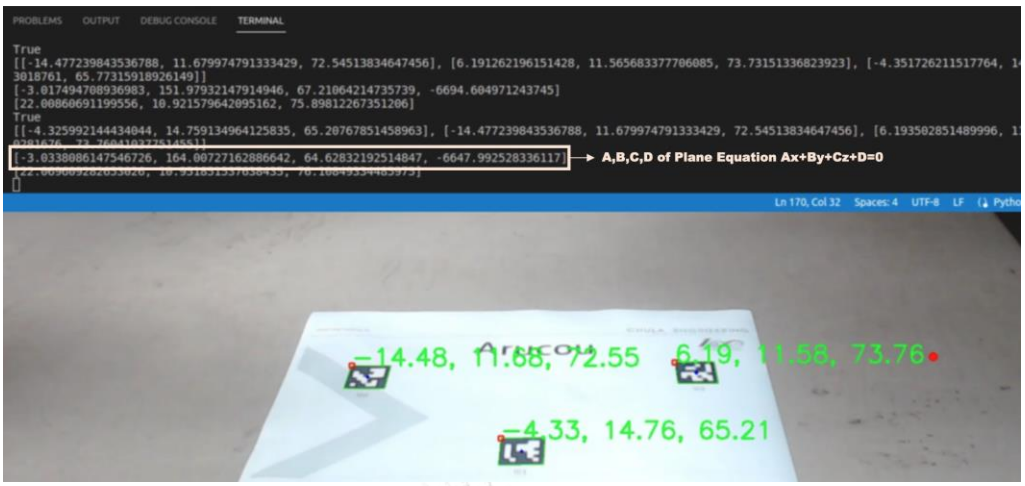


รูปที่ 42 แนวความคิดในการสร้างสมการ Plane จาก ArUco Marker จำนวนสามพิกัด

ทดลองตรวจจับพิกัดของ ArUco Marker โดยทำการสร้างโปรแกรมสำหรับตรวจจับ ArUco สามตำแหน่งขนาด 25.28 เซนติเมตรได้ผลการทดลองดังรูปที่ 43 พิกัดที่ถูกคำนวณโดย ArUco Marker เป็นระยะของ ArUco Marker เทียบกับกล้องในรูปที่ 43 แสดงค่าที่วัดได้ในหน่วยเซนติเมตรเมื่อทำการวัดระหว่างตำแหน่งบริเวณหน้ากล้องถึงบริเวณจุดศูนย์กลางของ Aruco พบว่ามีค่าความคลาดเคลื่อนประมาณ 1 เซนติเมตร ซึ่งอาจเกิดเนื่องจากตัวกล้องมีระยะความหนาอยู่ที่ 24 มิลลิเมตร

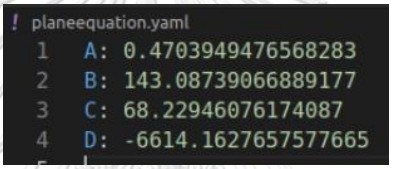
รูปที่ 43 ตัวอย่างการทดสอบในการสร้างสมการระนาบจาก ArUco Marker

ขั้นตอนที่ 2 กำหนดให้โปรแกรมรับค่าจุดพิกัดสามจุดเพื่อเข้าสู่การคำนวนกระบวนการในการสร้างสมการระนาบ Ax+By+Cz+D=0 ดังรูปที่ 44 แสดงค่าตัวแปร A, B, C, D ของสมการระนาบที่เป็นผลลัพธ์จากการคำนวณหาค่าจุดของ Aruco ทั้ง 3 ID จากตัวอย่างเพื่อนำไปใช้ต่อในการหาพิกัดของวัตถุที่ได้ทำการ Tracking



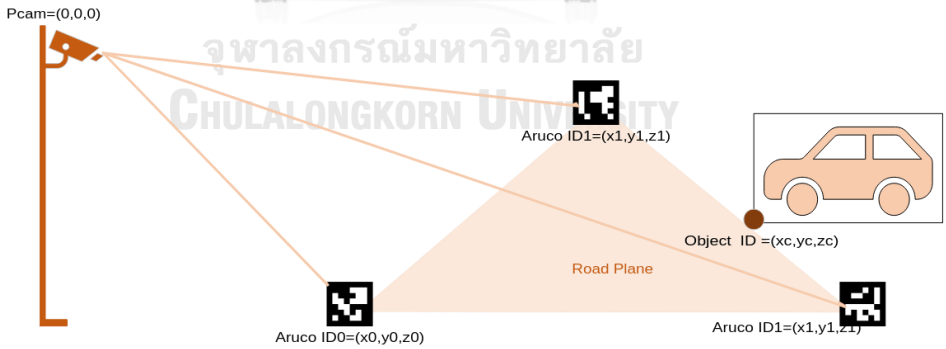
รูปที่ 44 ตัวอย่างในการเก็ยค่าสมดารระนาบ

ขั้นตอนที่ 3 ทำการบันทึกค่าตัวแปร A, B, C, D ลงในไฟล์ในการบันทึกค่าตัวแปร planeequation.yaml ดังรูปที่ 45



รูปที่ 45 วิธีการเก็บค่าสัมประสิทธ์ของสมการระนาบ

จากกระบวนการข้างต้นทำให้สามารถคำนวณหาค่า สมการระนาบของถนนได้ ต่อมาทดลองนำค่าตัวแปรที่ได้จากสมการระนาบและการติดตามวัตถุมาใช้ร่วมกันเพื่อหาระยะทางของวัตถุที่ทำการติดตามโดยขั้นตอนนี้สามารถแสดงค่าอยู่ในรูปอย่างง่ายได้ดังรูปที่ 46



รูปที่ 46 รูปแบบหลักการในการระบุตำแหน่งของวัตถุอย่างง่าย

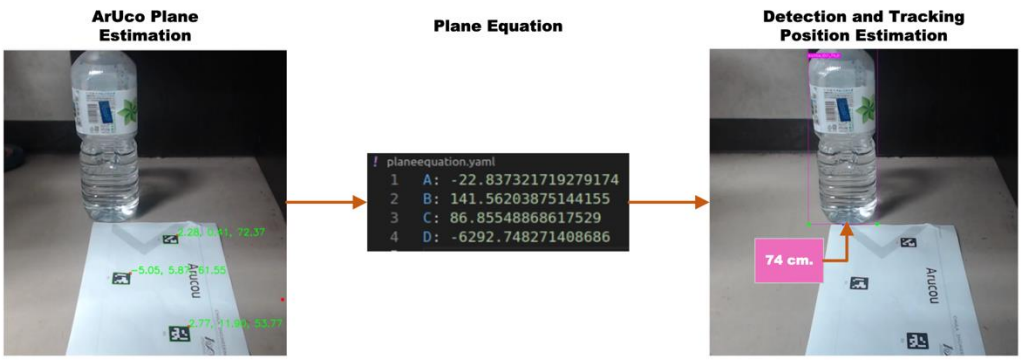
สร้างโปรแกรมสำหรับนำค่าจุด Bottom Center ของ Bounding Box เข้าสู่กระบวนการคำนวนหาตำแหน่ง จากสมการที่ 20

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} s(c_x) \\ s(c_y) \\ s \\ 0 \end{bmatrix} \begin{bmatrix} IntrinsicsMatrix * ExtrinsicMatrix \\ \hline A \quad B \quad C \quad D \end{bmatrix}^{-1} \tag{33}$$

ค่าที่รับจาก Bottom Center คือ $\mathbf{C_x}, c_y$ เมื่อแทนค่าตัวแปรในสมการที่ 20 จะสามารถคำนวนหาค่าพิกัด x,y,z ของวัตถุที่ต้องการได้ทำการทดสอบการหาตำแหน่งของโปรแกรมได้ผลดังรูปที่ 47 จากรูปแสดงตัวอย่างการหาพิกัดของวัตถุที่ตรวจจับและให้แสดงผลลัพธ์คือค่าพิกัด Z ของวัตถุ ซึ่งคือ ระยะระหว่างกล้องและวัตถุ

รูปที่ 47 ทดสอบกระบวนการหาตำแหน่งของขวดในระยะ 74 เซนติเมตร

ขั้นตอนการหาพิกัดของวัตถุสามารถเขียนเป็นขั้นตอนอย่างง่ายได้ดังรูปที่ 48



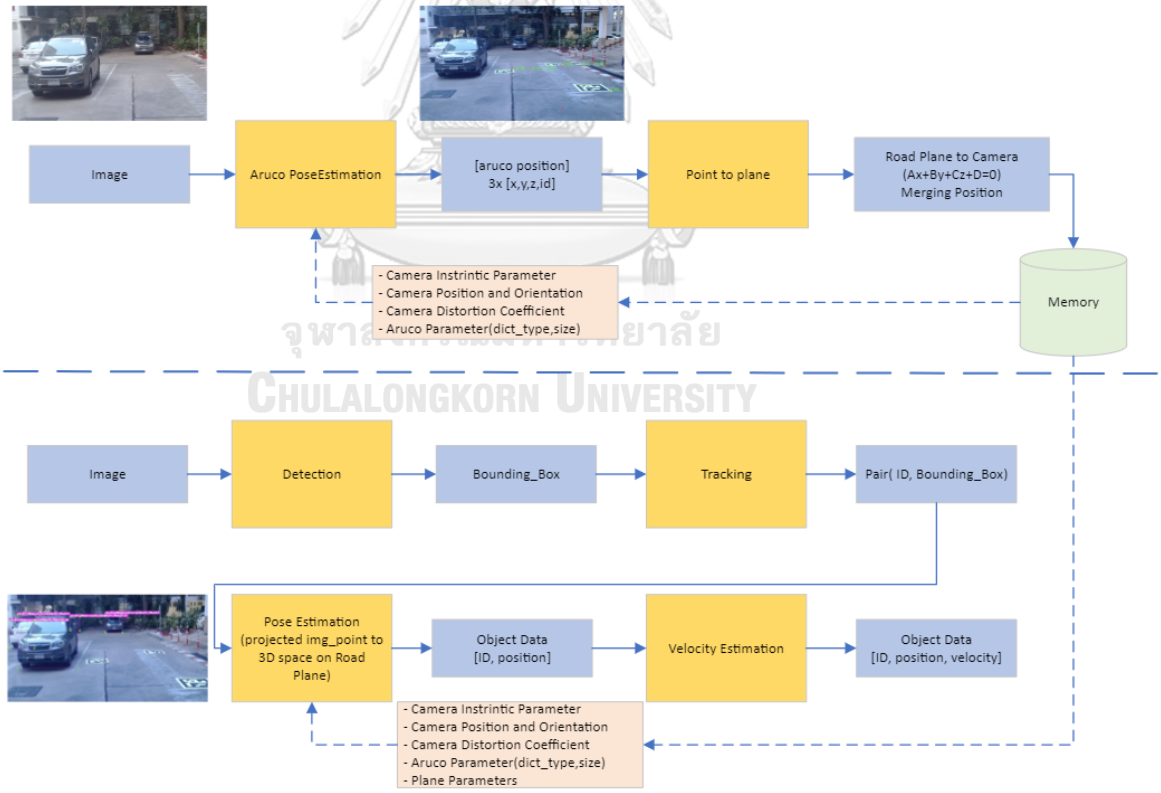
รูปที่ 48 ขั้นตอนในการหาตำแหน่งของวัตถุตัวอย่าง

3.2.1.3.3 กระบวนการคำนวณความเร็วของวัตถุ

ในกระบวนการหาค่าความเร็วของวัตถุสามารถหาได้จำเป็นต้องมีตัวแปรในการเก็บค่าระยะทาง ชนิด และ IDของวัตถุในกระบวนการติดตามวัตถุในเฟรมก่อนหน้าและมาคำนวณเทียบกับเวลาที่เปลี่ยนไป เพื่อให้สามารถคำนวนหาค่าความเร็วของวัตถุที่ทำการติดตามได้

ในขั้นตอนนี้ใช้ไลบรารี่ dequeueในการเก็บค่าของความเร็วในแต่ละกระบวนการโดยกำหนดให้ใช้ค่าระยะทางที่เก็บได้ใน dequeue มีค่า 10 ค่าและเมื่อค่าครบ ค่าให้นำค่าแรกออกจาก Dequeue และนำค่าใหม่เข้ามาเก็บ ต่อมาทำการคำนวนค่าความเร็วของวัตถุจากค่าระยะทางที่เปลี่ยนไปเทียบกับเวลาที่เปลี่ยนไปโดยค่าอัตราเร็วคิดตามสมการที่ 21 และสามารถแสดงผลการทำงานรวมของระบบตรวจจับได้ดังรูปที่ 49

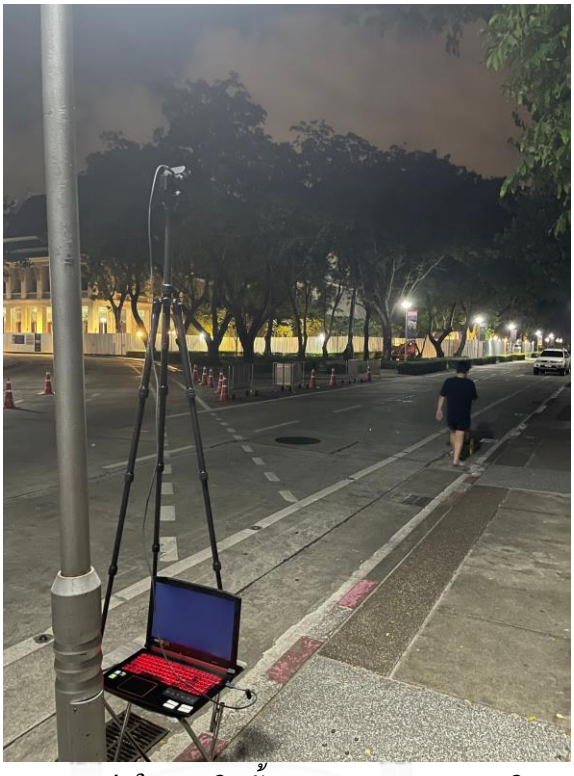
$$|\vec{v}| = \frac{\left|\begin{pmatrix} x_2 - x_1 \\ y_2 - y_1 \\ z_2 - z_1 \end{pmatrix}\right|}{\Delta t} \qquad (34)$$



รูปที่ 49 ผังการทำงานของข้อมูลและโปรแกรมที่ใช้สำหรับกระบวนการหาระยะของวัตถุที่ทำการตรวจจับ

3.2.1.3.4 การทดสอบระบบตรวจจับและติดตามวัตถุบนสภาพแวดล้อมจริง

ในการทดลองบนสภาพแวดล้อมจริงทำการทดสอบ 3 ส่วนคือ การตรวจสอบการวัดพิกัดของ ArUco การวัดผลระยะยานพาหนะที่ทำการตรวจจับได้ การวัดผลความเร็วของยานพาหนะเทียบกับค่า GPS ที่วัดได้จากยานพาหนะทดสอบ โดยทำการตรวจจับบริเวณห่างจากจุดร่วมเป็นระยะ 10 เมตรจากจุดร่วมดังรูปที่ 50



รูปที่ 50 ตำแหน่งในการติดตั้งระบบตรสจจะบและติดตามวัตถุ

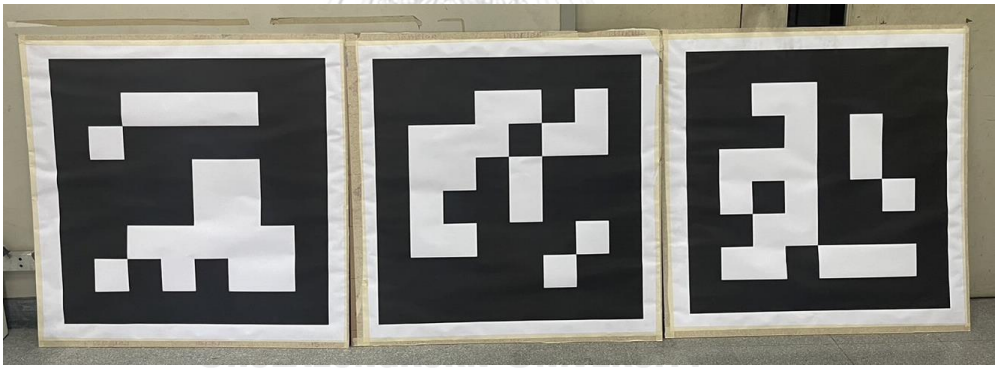
จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

โดยความสูงในการติดตั้งอุปกรณ์จากขาตั้งกล้องอยู่ที่ระยะ 171 เซนติเมตร และความสูงของแท่นวางอยู่ที่ระยะ 66 เซนติเมตร โดยระยะรวมที่วัดได้จากพื้นถนนคือ 237 เซนติเมตร โดยใช้เครื่องมือในการวัดระยะทางคือล้อวัดระยะทาง HUMMER ดังรูปที่ 51

รูปที่ 51 เครื่องมือในการวัดระยะทางเพื่อนำมาเปรียบเทียบอ้างอิง

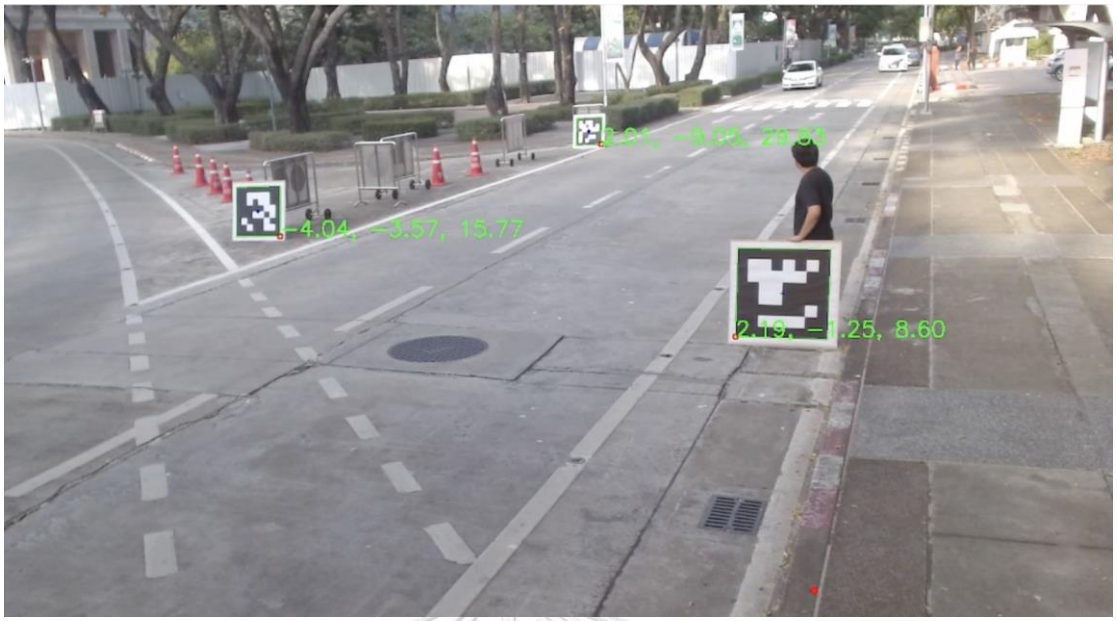
การตรวจสอบการวัดระยะของ ArUco

ในขั้นตอนนี้จะทำการทดลองวัดระยะที่ของ ArUco ที่ทำการตรวจจับได้โดยทำการเทียบกับค่าที่วัดได้จากเครื่องมือวัดและค่าที่สามารวัดได้จากการตรวจจับ ArUco Marker ขนาด 76 * 76 เซนติเมตร ดังรูปที่ 52



*รูปที่ 52 ArUco Marker ขนาด 76*76 เซนติเมตรชนิด 6x6_100 ID 1,2,3*

ในการทดลองนี้ใช้ ArUco ในการวัดระนาบของพื้นที่โดยการวาง ArUco Marker บนถนนในสามตำแหน่งดังรูปที่ 53 โดยวางในตำแหน่งสามตำแหน่ง โดยทำการทดลองเปรียบเทียบค่าระหว่างการวัดโดยใช้ArUco Marker วางในแนวราบและ แนวตั้งดังรูปที่ 53 และรูปที่ 54



รูปที่ 53 การทดสอบสร้างระนาบจาก ArUco Marker 3ชนิดบนพื้นที่ทดสอบจริง

รูปที่ 54 การทดสอบสร้างระนาบจาก ArUco Marker 3ชนิดบนพื้นที่ทดสอบจริง

การหาระยะเริ่มต้นของ Merging Point เทียบกับกล้อง

ในกระบวนการนี้ใช้ Aruco Marker 6x6_100 ขนาด 76*76 เซนติเมตร ดังรูปที่ 55

รูปที่ 55ตำแหน่งในการรับค่าพิกัดจากของMerging Point โดยใช้ Aruco Marker

พิกัดเริ่มต้นของ Merging point แล้วทำการเก็บค่าไว้ในไฟล์ mergingArea .yaml ดังรูปที่ 56

```
1  xc: 6.548259132161859
2  yc: -5.533867414593098
3  zc: 19.58595113903836
```

รูปที่ 56 ค่าตำแหน่งของ Merging Point เทียบกับกล้อง

โดยจะทำการเก็บค่า x_c, y_c, z_c เพื่อนำค่ามาคำนวณหาระยะห่างระหว่าง ยานพาหนะที่ทำการตรวจจับได้และบริเวณ conflict ดังสมการที่ 35

$$Distance = \sqrt{(x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2} \qquad (35)$$

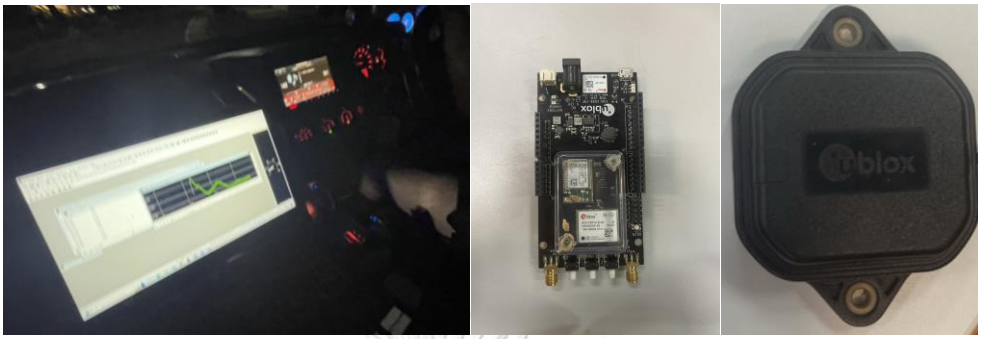
การทดสอบวัดระยะของวัตถุที่ตรวจจับได้

ทำการตรวจจับ ArUco และทดลองวัดค่าระยะของวัตถุที่ตรวจจับได้ดังรูปที่ 57

รูปที่ 57 การทดลองวัดค่าตำแหน่งของยานยนต์ในพื้นที่ทดสอบ

การวัดค่าความเร็วของวัตถุที่ได้จากกระบวนการตรวจจับวัตถุ

ทำการทดสอบหาค่าความเร็วที่ได้จากการวัดเทียบกับค่าความเร็วที่ได้จากการวัดด้วย GPS U-bloxที่ทำการติดตั้งบนยานพาหนะทดสอบโดยมีวิธีการวัดค่าที่ได้จาก GPS ดังรูปที่



รูปที่ 58 เครื่องมือในการเก็บความเร็วของยานยนต์ที่นำมาใช้สำหรับทดสอบ จับความเร็ว (Ublox ZED -F9P)

3.2.2 ระบบสื่อสารระหว่างรถและโครงสร้างพื้นฐาน

3.2.2.1 ทดสอบการใช้งาน Mosquitto MQTT

Eclipse Mosquitto คือ Open source mqtt Broker ที่ใช้ได้กับ MQTT protocol version 5,3.11,3.1 โดยมีลักษณะแบบ Lightweight เหมาะในการใช้ร่วมกับบอร์ดที่ใช้พลังงานต่ำ Mosquitto Protocol มีความสามารถในการใช้ผ่าน command line ได้ เหมาะสำหรับการส่งข้อมูลแบบ Machin to Machine

3.2.2.1.1 กระบวนการสร้าง sever สำหรับการใช้งาน MQTT

Sever Port ของ Mosquitto MQTT ในการทดลองเบื้องต้นผู้วิจัยเลือกใช้พอร์ต 1883 คือพอร์ตชนิด MQTT, unencrypted, unauthenticated โดยส่งข้อมูลในวง LAN ร่วมกัน โดยใช้ HUAWEI AIS5G CPE ในการกระจายสัญญาณ 5G จากเครือข่าย cellular ทดลองเปิด Mosquitto server ผ่าน command line ด้วยคำสั่ง mosquito -v ได้ผลดังรูปที่ 59

```
(base) palmmacbook@PALMs-MacBook-Pro ~ % mosquitto -v
1687338944: mosquitto version 2.0.15 starting
1687338944: Using default config.
1687338944: Starting in local only mode. Connections will only be possible from
clients running on this machine.
1687338944: Create a configuration file which defines a listener to allow remote
 access.
1687338944: For more details see https://mosquitto.org/documentation/authenticat
ion-methods/
1687338944: Opening ipv4 listen socket on port 1883.
1687338944: Opening ipv6 listen socket on port 1883.
1687338944: mosquitto version 2.0.15 running
```

รูปที่ 59 การสร้าง MQTT Broker Mosquitto

3.2.2.1.2 กระบวนการสร้าง Node สำหรับ Publish ข้อความ

ทำการสร้าง Node สำหรับ Publish ข้อมูลที่ได้จากการระบบตรวจจับ และติดตามวัตถุเข้าสู่ MQTT Broker ที่กำหนดไว้โดยการส่งจะส่งข้อมูลโดยใช้ รูปแบบ JSON และเชื่อมต่อเครือข่ายในระยะ LAN ร่วมกันดังตัวอย่างในรูปที่ 60

```
# MQTT
import paho.mqtt.client as mqtt
client = mqtt.Client()
client.connect('192.168.1.247',1883)
import json
```

รูปที่ 60 library ที่เลือกใช้ในการส่งค่า MQTT ในรูป JSON

จากรูปตัวอย่างทำการเชื่อมต่อเข้าที่เลข IP ของเครื่องคือ 192.168.1.247 และเลือกการเชื่อมต่อสัญญาณชนิด 1883 คือสัญญาณชนิด MQTT, unencrypted, unauthenticated เพื่อให้สามารถทดสอบได้อย่างสะดวก ต่อมาทำการสร้างตัวแปรชนิดด Dictionary เพื่อทำการรับค่าข้อมูลที่ต้องการส่งเข้าสู่ระบบการตัดสินใจของรถอัตโนมัติดังตัวอย่างในรูปที่ 61

```
# mqtt
obj_data=dict()
obj_data["id"]=track_id
obj_data["classs Name"]=className
obj_data["velocity"]=int(vel)
obj_data["X"]=f'{pos2[0]:.1f}'
obj_data["Y"]=f'{pos2[1]:.1f}'
obj_data["Z"]=f'{pos2[2]:.1f}'
raw_msg.append(obj_data)
```

รูปที่ 61 การสร้างตัวแปรเพื่อเก็บค่าในรูป Dictionary เพื่อทำการส่งค่าเข้าสู่ MQTT Broker

สร้างฟังก์ชันในการส่งออกค่าข้อความที่ได้ทำการเก็บค่าไว้ โดยเปลี่ยนข้อมูลชนิด Dictionary ให้อยู่ในรูป JSON และส่งค่าออกไปยัง MQTT Broker ดังตัวอย่างในรูปที่ 62 ทำการส่งค่าข้อมูลเข้าสู่ Topic ที่มีชื่อว่า computer/switch

```
def send_mqtt(self,raw_msg):
    encoded_msg = json.dumps(raw_msg)
    client.publish('computer/switch',encoded_msg)
```

รูปที่ 62 ค่าฟังก์ชันในการส่งข้อมูลเข้าสู่ MQTT broker ใน topic computer switch

3.2.2.1.3 กระบวนการสร้าง Node สำหรับกระบวนการรับข้อความ

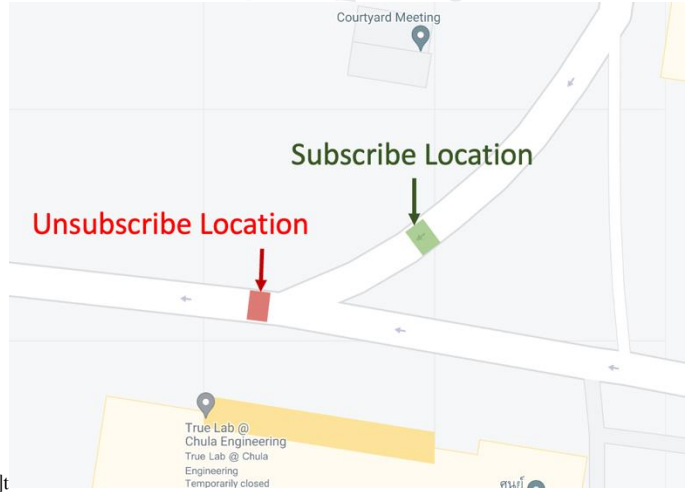
ในกระบวนการ Subscribe ข้อความสามารถทำได้โดยการกำหนดให้เชื่อมต่อกับ Sever ที่เลข IP คือ 192.168.1.247 และกำหนดให้รับค่า Message จาก topic ที่มีชื่อว่า computer/switch และทำการสร้างฟังก์ชัน ในการ Decode ข้อมูลที่อยู่ในรูป JSON ดังตัวอย่างในรูปที่ 63

```python
import paho.mqtt.client as mqtt
import json
(variable) client: Client

client = mqtt.Client()
client.connect('192.168.1.247',1883)


SUBSCRIBE_TOPIC = 'computer/switch'

client.subscribe(SUBSCRIBE_TOPIC)

def on_message_callback(client, userdata, msg):
    topic = msg.topic
    encoded_msg = msg.payload
    decoded_msg = json.loads(encoded_msg)
    print(topic, decoded_msg)

client.on_message = on_message_callback

client.loop_forever()
```

รูปที่ 63 โปรแกรมในการรับค่า message MQTT จาก MQTT Broker

3.2.2.1 วิธีการทดสอบ

ในการทดสอบใช้เครื่อง CPE สำหรับกระจายสัญญาณเครือข่ายเซลลูลาร์ ให้กับระบบในการตัดสินใจของรถอัตโนมัติ และ ระบบตรวจจับและติดตามวัตถุ โดยข้อมูลที่ได้จากการติดตามและตรวจจับจะถูกส่งในรูป MQTT message เป็นค่า ID ชนิดของยานพาหนะ ค่าตำแหน่งและค่าความเร็ว ของยานพาหนะในบริเวณทางเอกที่ตรวจจับได้เข้าสู่ MQTT Broker และตั้งค่าให้เมื่อระบบจำลองรถอัตโนมัติเคลื่อนที่เข้าสู่ตำแหน่งที่กำหนดไว้ ทำการรับค่าที่ถูกส่งมาได้จาก MQTT Broker ดังรูปที่ 64 เพื่อนำประมวลผลสำหรับการชะลอในระบบการตัดสินใจภายในรถอัตโนมัติ



รูปที่ 64 ตำแหน่งของรถอัตโนมัติทำการ Subscribe และ Unsubscribe MQTT Broker

3.2.3 ระบบการตัดสินใจของยานยนต์อัตโนมัติ

ในการสร้างระบบการตัดสินใจจะใช้เกณฑ์ในการกำหนดพื้นฐานทั่วไป (Rule Based) โดยสถานการณ์ที่สนใจเป็นกรณีที่รถอัตโนมัติเคลื่อนที่เข้าสู่ทางร่วมในเส้นทางโทด้วยความเร็วระหว่าง 10-20 กิโลเมตรต่อชั่วโมง และรถบนท้องถนนเคลื่อนที่เข้าสู่ทางร่วมโดยมีข้อกำหนดความเร็วไม่เกิน 30 กิโลเมตรต่อชั่วโมง ในงานวิจัยนี้สร้าง Conflict Area ในบริเวณที่ถนนร่วมกัน และกำหนดตำแหน่งในการจอดของรถอัตโนมัติในกรณีที่จำเป็นต้องจอดรอในบริเวณทางร่วม ดังรูปที่ 65 เพื่อใช้สำหรับประกอบการตัดสินใจในการชะลอของรถอัตโนมัติ

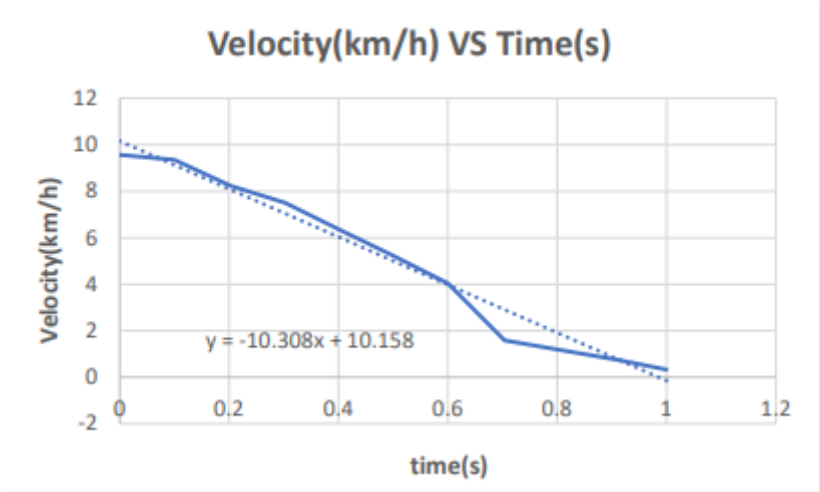
รูปที่ 65 บริเวณในการตัดสินใจของรถอัตโนมัติ

3.2.3.1 ตัวแปรควบคุม และ คุณสมบัติ Turing T2 ที่นำมาประกอบการคำนวน

3.2.3.1.1 ค่าตัวแปรคุณสมบัติของ T2

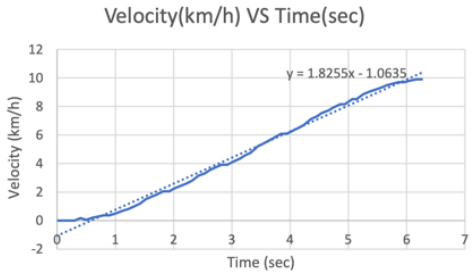
จากงานวิจัยทดลองของ Smart mobility Research Center ได้ทำการทดสอบรถอัตโนมัติเพื่อหาค่าคุณสมบัติต่างๆของรถอัตโนมัติ ในงานวิจัยนี้ได้นำค่าบางค่ามาใช้คำนวนสำหรับการสร้างกระบวนการตัดสินใจของรถอัตโนมัติ

รถอัตโนมัติ Turing นั้นควบคุมการทำงานและระบบสั่งการด้วยระบบ Robot Operating System (ROS) ซึ่ง Robot Operating System (ROS) เป็นระบบที่สร้างขึ้นเพื่อทำให้เกิดความยืดหยุ่นในการเขียนซอฟต์แวร์ควบคุมหุ่นยนต์ ใน ROS จะรวบรวมชุดเครื่องมือและชุดคำสั่งต่างๆที่จำเป็นในการพัฒนาหุ่นยนต์เอาไว้ สิ่งเหล่านี้ช่วยลดความยุ่งยากในการสร้างในการพัฒนาหุ่นยนต์ที่มีความซับซ้อน และทำให้มีประสิทธิภาพในการพัฒนาหุ่นยนต์หลายรูปแบบ Turing T2 เมื่อทำการทดลอง ฐานข้อมูลจะถูกเก็บในรูปไฟล์ของ rosbag ในงานวิจัยนี้ในส่วนระบบตัดสินใจของรถอัตโนมัติผู้วิจัยจะพิจารณาเพียงค่าความเร็วเนื่องจากการเร่งและชะลอความเร็วของรถอัตโนมัติ ซึ่งมีค่าเก็บ อยู่ใน rosbag ส่วนของ message ชื่อว่า msg.brake โดยจะมีค่าอยู่ระหว่าง 0-250 คือค่าความเร็วระหว่างค่าสูงสุดและค่า Full Brake โดยข้อมูลที่คำนวนจากการเก็บค่าตัวแปรในการทดลองจริงของรถและการเทียบค่าจากการทดลองลดความเร็วจริงของรถ ซึ่งได้ค่าความสัมพันธ์ระหว่างเร็วกับเวลาดังกราฟในรูปที่ 66



รูปที่ 66 กราฟแสดงค่าความเร่งในการเบรคจาดระยะ 10 เมตร

เมื่อทำการคำนวณ พบว่าที่ msg.brake เท่ากับ 204 มีค่าความหน่วงอยู่ที่ 2.86 m/s^2 และสามารถหยุดรถอัตโนมัติได้ภายในเวลา 1 วินาที



รูปที่ 67 กราฟค่าความเร่งของรถอัตโนมัติจากหยุดนิ่งจนกระทั่งความเร็วมีค่าเท่ากับ 10 กิโลเมตรต่อชั่วโมง

พบว่ารถอัตโนมัติมีความเร่งในการออกตัวอยู่ที่ 0.51 m/s^2 โดยมีค่า msg.trottle ประมาณ 83 จาก 250 ซึ่งใช้เวลา 6 วินาทีในการออกตัวจาก 0-10 กิโลเมตรต่อชั่วโมง โดยค่าคุณสมบัติของ Turing T2 ที่นำมาใช้ในการคำนวนมีค่าดังแสดงในตารางที่ 7

ตารางที่ 7 ตัวแปรของรถอัตโนมัติ T2 สำหรับใช้ในกระบวนการตัดสินใจของรถอัตโนมัติ

ตัวแปร	ชื่อ	นิยามของตัวแปร	ค่าของตัวแปร
L0	T2 length	ความยาวของรถ T2	4.3 m
V0	T2 velocity	ความเร็วเริ่มต้นของ T2	20 km/hr
A0	T2 accelation	ความเร่งเริ่มต้นของ T2	0 m/s^2

3.2.3.1.2 ค่าตัวแปรที่กำหนดเพื่อเพิ่มความปลอดภัยให้กับระบบการตัดสินใจของรถอัตโนมัติ

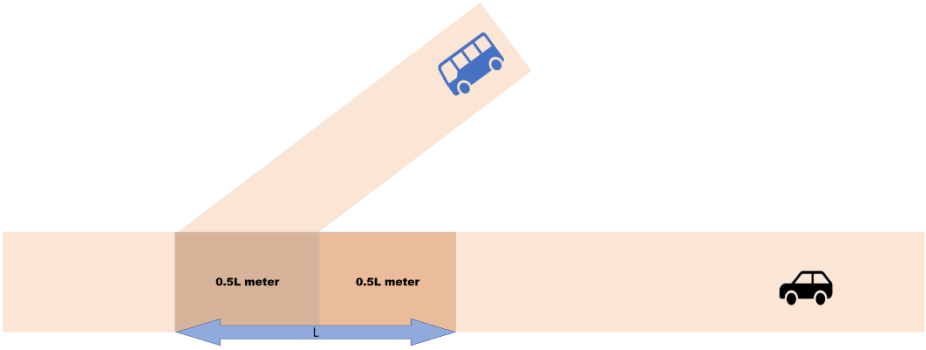
ในระบบจำลองการเคลื่อนที่ของรถอัตโนมัตินั้นได้ทำการนำค่าเพิ่มเติมมาปรกอบการคำนวนดังตารางที่ 8

ตารางที่ 8 ตารางแสดงตัวแปร Lane Balance และค่า Clearance

ตัวแปร	ชื่อ	นิยามของตัวแปร	ค่าของตัวแปร
LB	Lane Balance	ระยะทางจากจุดร่วมถึงตำแหน่งที่ออกจาก Conflict Area	11.5 m
CT	Clearance Time	ระยะเวลาสำหรับการคำนวนระยะปลอดภัยของรถอัตโนมัติ T2	0.5 km/hr

3.2.3.1.3 ค่าตัวแปรของตำแหน่ง Conflict Area

กำหนด Conflict Area จากขอบเขตของรถบนท้องถนนที่วิ่งด้วยความเร็วไม่เกิน 30 กิโลเมตรต่อชั่วโมง หรือประมาณ 8.3 เมตรต่อวินาที กำหนดให้ระยะของ Conflict area ก่อนจุดร่วมและหลังจุดร่วมมีขนาดเท่ากันดังรูปที่ 68



รูปที่ 68 รูปแสดงตำแหน่งของ Conflict Area และ ค่า LB

สร้าง Conflict จากค่า Total Brake Reaction time คือ 2.3 วินาที และค่า Safety factor ที่ 20 % โดยจะสร้าง conflict area ได้ดังสมการที่ 36

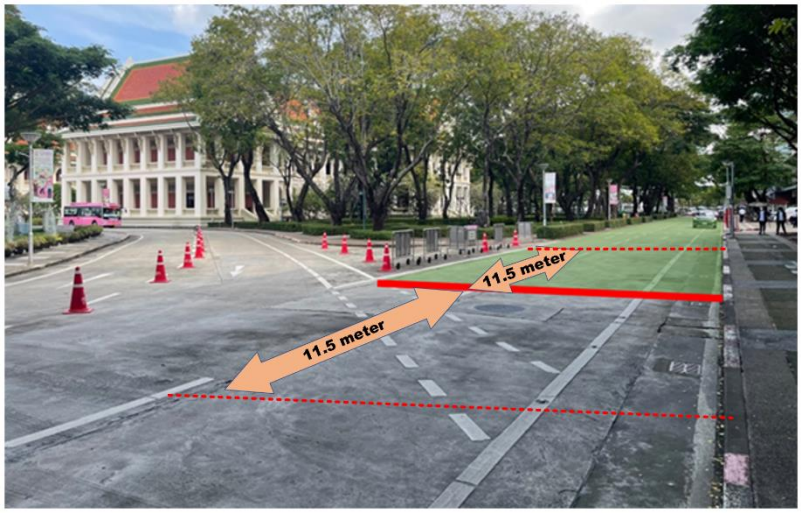
$$Conflict\ Area = Safety\ Factor \times Total\ Brake\ Reaction\ Time\ \times Velocity \quad (36)$$

$$Conflict\ Area = 1.2 \times 2.3 \times Velocity \quad (37)$$

เมื่อนำค่าความเร็วและเวลามาคำนวณหาระยะทางได้ค่าขนาดความยาวของ Conflict Area ที่ความเร็วรถไม่เกิน 30 กิโลเมตรต่อชั่วโมง หรือ 8.33 เมตรต่อวินาที ที่ระยะดังสมการที่

$$Conflict\ Area = 1.2 \times 2.3 \times 8.33 = 23\ meter \quad (38)$$

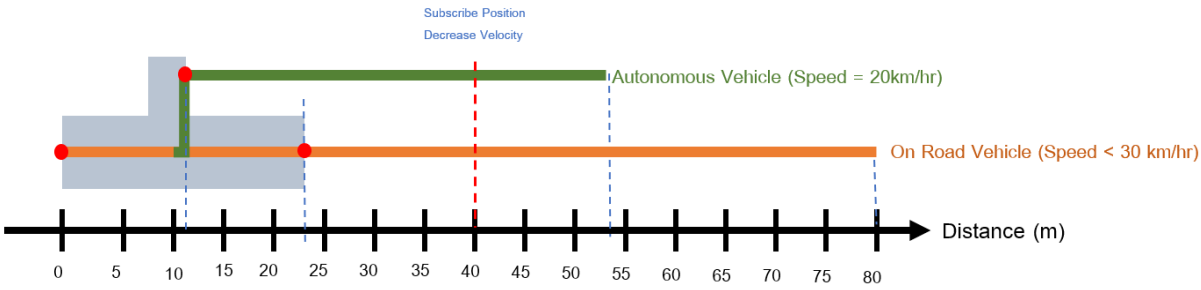
ซึ่งสามารถแบ่งเป็นระยะได้คือ 11.5 เมตรทั้งสองฝั่ง จึงทำให้ LB มีค่าเป็น 11.5 เมตรโดยมีค่าตำแหน่งที่ประมาณค่าดังรูปที่



รูปที่ 69 ตำแหน่งจริงของ Conflict Area และ Merging Point

กำหนดให้รถอัตโนมัติเริ่มชะลอความเร็วลงให้เหลือ 10 กิโลเมตร/ชั่วโมง ด้วยความหน่วง 0.45 m/s^2 จากจุดที่เริ่มรับข้อมูล โดยเริ่มชะลอรถอัตโนมัติในบริเวณก่อนจุดตัดสินใจจอดรถที่ระยะห่าง 40 เมตร จากจุดร่วมของถนน

3.2.3.2 แผนภาพแสดงระยะทางในการทำงานของระบบระหว่างรถอัตโนมัติ ยานพาหนะที่ตรวจจับได้และ Conflict Area



รูปที่ 70 ระยะทางระหว่างรถทั้งสองคันที่ใช้ในกระบวนการตัดสินใจอย่างง่าย

ทำการสร้างแผนภาพอย่างง่ายที่แสดงระยะทางระหว่าง รถอัตโนมัติ ยานพาหนะบนท้องถนน ขนาดของ Conflict Area โดยกำหนดให้ทำการกำหนดตำแหน่งของกล้องอยู่ที่ตำแหน่ง 0 โดยในแผนผังจะแสดงข้อมูลดังนี้

1) ความยาวของ Conflict Area ที่เทียบกับกล้อง

2) ตำแหน่ง Subscribe Position ที่ตำแหน่งระยะ 40 เมตร

3) ความเร็วของรถอัตโนมัติที่ทำการเคลื่อนที่เข้าสู่ตำแหน่ง Subscribe คือ 20 km/hr

4) ความเร็วของยานยนต์บนท้องถนนที่ขับขี่เข้าสู่บริเวณ Conflict Area

3.2.3.3 อัลกอริทึมในกระบวนการตัดสินใจของรถอัตโนมัติ

กำหนดให้รถอัตโนมัติเริ่มชะลอความเร็วลงให้เหลือ 10 กิโลเมตร/ชั่วโมง ด้วยความหน่วง 0.45 m/s^2 จากจุดที่เริ่มรับข้อมูลเพื่อรอรับข้อมูลจากทางเอกที่ได้จากระบบตรวจจับและนำมาคำนวณค่า TTC ใช้ในกระบวนการตัดสินใจของรถอัตโนมัติ

3.2.3.3.1 การคำนวนค่า TTC

ระบบจำลองกการขับขี่ของ Turing T3 จะนำค่าที่รับมาคือค่า ID , Object type ,Velocity,Position ของยานยนต์ที่ๆได้จากระบบตรวจจับมาคำนวณหาค่า Time to Collision (TTC) ออกมา 3 เพื่อประกอบการตัดสินใจของรถโดยจะประกอบด้วยค่า TTC1 ,TTC2 ,TTC3 โดยแต่ละค่ามีการคำนวนดังนี้

TTC1 คือค่าเวลาของตำแหน่งหน้ายานยนต์ที่เคลื่อนที่เข้าสู่ตำแหน่ง Conflict Area

TTC2 คือค่าเวลาของตำแหน่งท้ายยานยนต์เคลื่อนที่เข้าสู่ตำแหน่ง Conflict Area

TTC3 คือค่าเวลาของตำแหน่งท้ายยานยนต์เคลื่อนที่ออกจากตำแหน่ง Conflict Area

ค่า TTC ทั้งสามค่า มีวิธีการคำนวนดังนี้

ของรถอัตโนมัติ T2

$$TTC1 = \frac{Distance\ to\ ConflictArea}{V} \quad (39)$$

$$TTC2 = \frac{Distance\ to\ ConflictArea + T2\ Length}{V} \quad (40)$$

$$TTC3 = \frac{Distance\ to\ ConflictArea + T2\ Length + LB}{V} \quad (41)$$

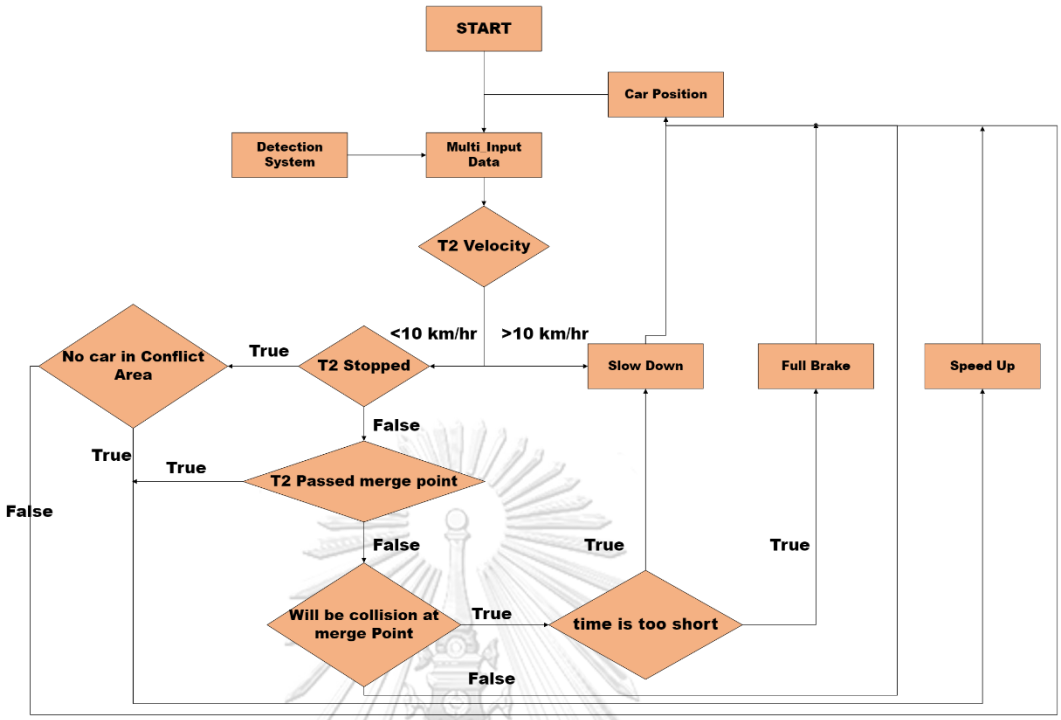
รถยนต์บนท้องถนน

$$TTC1 = \frac{Distance\ to\ ConflictArea}{V} \quad (42)$$

$$TTC2 = \frac{Distance\ to\ ConflictArea + T2\ Length}{V} \quad (43)$$

$$TTC3 = \frac{Distance\ to\ ConflictArea + T2\ Length + Conflict\ Range}{V} \quad (44)$$

3.2.3.3.2 แผนผังแสดงการทำงานของระบบหลีกเลี่ยงการชน



รูปที่ 71แผนผังแสดงกระบวนการตัดสินใจของระบบจำลองการเคลื่อนที่ของยานยนต์อัตโนมัติ

ดังรูปที่ 71 สามารถอธิบายผังการทำงสยของระบบตัดสินใจเป็นขั้นตอนได้ดังนี้

1) ระบบรับค่าตัวแปรของรถอัตโนมัติที่ได้จากระบบตรวจจับคือค่า ID , Type ,Distance,Velocity
2) ตรวจสอบว่าค่าความเร็วของ Turing T2
3) ถ้าค่าความเร็วของรถอัตโนมัติน้อยกว่า 10 km/hr จะทำการตรวจสอบเงื่อนไขว่าอยู่ในเงื่อนไข T2 Stopped หรือไม่ (T2S=2)
4) ถ้าค่าไม่อยู่ในเงื่อนไข T2S=2 ให้ทำการเคลื่อนที่ T2 ออกจาก Merging Point ถ้าทำการเคลื่อนที่ออกแล้วรับค่ามาพบว่ามีการคาดการว่าจะชน รถอัตโนมัติจะทำการเบรค
5) ถ้าอยู่ในเงื่อนไข T2S=2 ให้ทำการตรวจสอบว่ามีรถอยู่ใน Conflict Area หรือไม่ถ้าไม่มีให้ทำการเร่งความเร็วเพื่อออกจากถ้ามีให้ทำการจอดรอเพื่อรับข้อมูลใหม่
6) ถ้าความเร็วของรถอัตโนมัติมากกว่า 10 km/hr. ชะลอต่อ

3.2.3.4 กำหนดสถานการณ์ในการทดสอบแบบจำลองรถอัติโนมัติจากกระบวนการตรวจจับวัตถุ

ในการสร้าง Simulation สำหรับทดสอบการใช้ Algorithm นั้นจะทำการใช้แบบจำลองที่สร้างขึ้นจากไลบรารี่ Matplotlib Python เป็นส่วนหลักในการ Simulation การเคลื่อนที่ของรถอัตโนมัติเนื่องจากการทดสอบด้วยรถอัตโนมัติจริงมีความอันตรายอย่างมาก ในงานวิจัยนี้จึงจะ

สร้าง Simulation method ขึ้นมาทดลองร่วมกับกระบวนการตรวจจับและติดตามวัตถุ โดยกำหนดเหตุการณ์เพื่อทำการจำลองพื้นฐาน 3 เหตุการณ์หลักคือที่ความเร็วคงที่ 20 , 30 , 40 กิโลเมตรต่อชั่วโมง โดยอิงจากการทดสอบของระบบตรวจจับและตืดตามวัตถุ

3.2.3.5 รูปแบบการใช้งานระบบ Simulation

ระบบ Simulation ด้วย Python นั้นถูกออกแบบขึ้นมาโดยมีข้อจำกัดที่ความเร็วของยานพาหนะบนท้องถนนมีความเร็วคงที่ โดยในระบบจำลองของ Turing T2 จะทำการรับค่าที่เหมือนค่าที่ได้รับจากระบบตรวจจับ โดยประกอบด้วยค่า

1) ID ของยานพาหนะบนท้องถนน

2) ชนิดของยานยนต์ โดยประกอบด้วย รถอัตโนมัติ T2 จักรยาน รถยนตร์ และ รถบรรทุก ตามลำดับ โดยระบบจำลองได้ทำการปรับค่าขนาดความยาวของยานยนต์

3) ระยะทางของ ยานยนต์ที่ตรวจจับได้เทียบกัย Merging Point

4) ความเร็วของยานยนต์ที่สามารถทำการตรวจจับได้

ซึ่งใน Simulation ได้ถูกสร้างขึ้นให้สามารถรับค่าทั้งสี่ค่าได้ดังรูปที่ 72

```
(base) palmmacbook@PALMs-MacB
Input Car Data: 1,car,20,40
['1,car,20,40']
Input:2,car,20,40
['2,car,20,40']
```

รูปที่ 72รูปแบบข้อมูลที่นำเข้าของระบบจำลองการเคลื่อนที่ผ่านจุด Merge ของรถอัตโนมัติ Turing T2

โดยในแบบจำลองนั้นมีความสามารถในการรับค่าตัวแปรในตำแหน่งเวลาที่เปลี่ยนไปของรถอัตโนมัติได้โดยสามารถตั้งค่าเวลาได้ดังรูปที่ 73

```
def SIM():# Simulation
    Multi_Input(input("Input Car Data: "))

    for i in range(0,int(TM*RS)+1):
        t.append(i/RS)
        if t[-1] in [2,3,4,5,6]:
            Multi_Input(input("Input:"))
        [A,C] = T2D()
```

รูปที่ 73การกำหนดตำแหน่งของรถอัตโนมัติในการรับค่าตัวแปรจากระบบตรวจจับ และติดตามวัตถุ

บทที่ 4
ผลการวิจัย

ในงานวิจัยนี้แบ่งผลการทดลองเป็น 5 ส่วนหลักคือ การตรวจจับและติดตามวัตถุ การประมาณหาระยะทางของยานพาหนะเพื่อนำมาหาค่าความเร็วของยานพาหนะและค่า TTC ผลการทดสอบการส่งค่าที่ได้จากการตรวจจับเข้าสู่ระบบ MQTT ผลการทดสอบAlgorithm การตัดสินใจของรถในกระบวนการ Simulation และผลการทดสอบระบบโดยรวม

4.1 ระบบการตรวจจับและติดตามวัตถุ

4.1.1 ระบบการตรวจจับวัตถุ

จากการทดสอบระบบตรวจจับวัตถุด้วยการใช้ฮาร์ดแวร์ Acer Nitro 5 และกล้องด้วย Model yolov8 ที่ได้ทำการ Train ด้วย COCO Dataset ได้ผลความเร็วในการตรวจจับรถที่ความเร็วเฉลี่ยและค่าความถูกต้องดังตารางที่ 9

ตารางที่ 9 ผลความเร็วในการทดลองกระบวนการติดตามวัตถุด้วย YOLOv8 สามโมเดล

Model	Speed (ms)	Detection Average Time (ms)	Inference (ms)	Total Time (ms)
Yolov8n	5.4	22.4	5	32.8
Yolov8m	2.1	40 .2	3.4	45.7
Yolov8l	1.4	67.4	1.8	70.6

เมื่อหาความเร็วในการตรวจจับวัตถุในแต่ละโมเดลเทียบกับความถูกต้องแล้วงานวิจัยนี้เลือกใช้ Yolov8l ในการตรวจจับและกำหนดให้ตรวจจับวัตถุแค่ในขอบเขตของกรอบที่ต้องการได้ผลการทำงานตัวอย่างของระบบตรวจจับดังรูปที่ 74

รูปที่ 74 ตัวอย่างผลลัพธ์มราได้จากกระบวนการตรวจจับวัตถุด้วยโมเดล YOLOv8

4.1.2 ระบบการติดตามวัตถุ

ในงานวิจัยนี้เลือกใช้ Bounding Box ที่ได้จากการตรวจจับวัตถุด้วยโมเดล Yolov8l มานำเข้าสู่กระบวนการ ติดตามวัตถุ (Tracking Algorithm) โดยใช้ Model DeepSORT พบว่าสามารถระบุค่า ID ของวัตถุได้ในเฟรมแต่ละเฟรมของวีดีโอและยังสามารถเป็นวัตถุชิ้นเดิมได้ผลดังรูปที่ 75

รูปที่ 75 ตัวอย่างผลลัพธ์จากกระบวนการติดตามวัตถุด้วย DeepSORT Tracking Algorithm

เนื่องจากการตรวจจับโดยใช้วิธีการในรูปขั้นต้นใช้วิธีการคำนวนความเร็วจากการเปลี่ยนแปลงของพิกัด Pixel ของภาพเทียบกับเวลาทำให้ค่าความเร็วที่ทำการคำนวนหาไม่สามารถนำไปใช้จริงได้จึงจำเป็นต้องคำนวนความเร็วจากค่าระยะทางจริงที่วัดได้ดังกระบวนการถัดไป

4.1.3 ระบบการประมาณหาระยะทางของวัตถุ

4.1.3.1 กระบวนการหาพิกัดของ ArUco Marker

เมื่อทำการตรวจจับ ArUco Marker โดยใช้ ArUco Marker ขนาด 76 * 76 เซนติเมตรได้ผลการตรวจจับดังรูปที่ 76

รูปที่ 76 การทดลองวัดค่าพิกัดของ ArUco Marker .ในบริเวณพื้นที่จริง

ผลการทดลองพบว่า ArUco marker สามารถให้ค่าพิกัดที่เปรียบเทียบระหว่าง ArUco marker และ ตัวกล้องได้ โดยผลการทดลองเปรียบเทียบระยะห่างระหว่างการตรวจจับ ArUco Marker และระยะจากการวัดจริงได้แสดงดังตารางที่ 10 เมื่อคำนวนพบว่ามีความคลาดเคลื่อนเฉลี่ยที่ 1.76 เปอร์เซนต์ เทียบจากการวัดจริงของจากระยะระหว่างตัวกล้องและ ArUco marker

ตารางที่ 10 ผลระยะทางที่ได้ทำการวัดโดยเครื่องมือวัดเปรียบเทียบกับค่าระยะที่ได้จากการตรวจจับ ArUco Marker

ระยะเฉลี่ยที่ทำการวัดด้วยเครื่องมือวัดระยะทาง (เมตร)	ระยะเฉลี่ยที่สามารถคำนวณหาได้จากการใช้ ArUco Marker (เมตร)	ค่าความคลาดเคลื่อนระหว่างการวัดทั้งสอค่า (%)
6.1	6.30	3.2
11.4	11.54	1.22
13.4	13.50	0.7
15.1	15.17	0.46
36.7	37.67	2.64
45.8	44.73	2.34

4.1.3.2) กระบวนการสร้างตัวแปรจาก ArUco Marker

4.1.3.2.1 ค่าตัวแปรของสมการระนาบในพื้นที่จริง

ในบริเวณทดสอบจริงผู้วิจัยได้ทำการทดลองวัดระยะของยานยนต์ทดลองโดยวาง ArUco Marker ในบริเวณระยะห่าง 6.7 เมตร 11.5 เมตร และ 13.74 เมตร ดังแสดงในรูปที่ 77

รูปที่ 77 การติดตั้ง ArUco Marker เพื่อทำการคำนวณสมการระนาบในบริเวณพื้นที่จริง

เมื่อทำการคำนวนสมการระนาบแล้วได้ค่าตัวแปรในสมการระนาบ A B C และ D ดังรูปที่ 78

```
! planeequation.yaml
1   A: 0.4703949476568283
2   B: 143.08739066889177
3   C: 68.22946076174087
4   D: -6614.1627657577665
```

รูปที่ 78 ตัวแปรสมการระนาบ A B C และ D จากการคำนวนโดยพิกัด ArUco Marker

4.1.3.2.2 ค่าตัวแปรตำแหน่ง Merging Point ในบริเวณพื้นที่จริง

นำ ArUco Marker ไปติดตั้งในบริเวณพื้นที่จริงดังรูปที่ 79

รูปที่ 79 บริเวณในการติดตั้ง ArUco Marker เพื่อทำการวัดระยะMerging Point สำหรับกระบวนการตัดสินใจของรถ

ผลการตรวจจับ ArUco Marker ได้ค่าตัวแปร X_c,Y_c,Z_C ดังรูปที่ 80

```
1  xc: 6.548259132161859
2  yc: -5.533867414593098
3  zc: 19.58595113903836
```

รูปที่ 80 ตัวแปร Merging Point สำหรับการคำนวนระยะทางระหว่างยานยนต์และจุดร่วม

4.1.3 3) กระบวนการหาตำแหน่งของยานพาหนะที่ได้ทำการตรวจจับ โดยเทียบกับตำแหน่งของยานพาหนะบนถนน

4.1.3.3.1 ผลการทดสอบหาระยะของวัตถุบริเวณพื้นที่ทดสอบ

ผลการทดลองการตรวจจับและคำนวนหาระยะทางของยานยนต์จากสมการระนาบที่สร้างขึ้นด้วยทิศทางการติดตั้งของแผ่น ArUco Marker ที่ต่างกันในบริเวณทดสอบ โดยแสดงตัวอย่างทิศทางการติดตั้งแผ่น ArUco Marker และระยะของวัตถุที่ได้จากการคำนวนดังรูปที่ 81 และรูปที่ 82

รูปที่ 81 ผลการวัดระยะของยานยนต์จากการตรวจจับด้วยภาพโดยสร้างระนาบ จากการติดตั้งแผ่น ArUco Marker ในแนวราบ

รูปที่ 82 ผลการวัดระยะของยานยนต์จากการตรวจจับด้วยภาพโดยสร้างระนาบ จากการติดตั้งแผ่น ArUco Marker ในแนวตั้ง

เมื่อทำการทดสอบการหาระยะจากการตรวจจับวัตถุในบริเวณทดสอบพบว่าสามารถระบุตำแหน่งของวัตถุบนระนาบด้วยกล้อง 2 มิติได้โดยในทิศทางการติดตั้งแผ่น ArUco Marker ในทิศทางที่แตกต่างกันให้ผลลัพธ์ในการคำนวณหาระยะทางของยานยนต์ที่ต่างกันโดยทำการแสดงผลดังตารางที่ 11 โดยจากการทดลองพบว่าการติดตั้งแผ่น ArUco Marker ในทิศทางแนวราบสามารถให้ผลการทดลองที่มีค่าความคลาดเคลื่อนโดยเทียบกับกระบวนการวัด น้อยกว่าการติดตั้งแผ่น ArUco Marker ในแนวตั้งอย่างมีนัยสำคัญ ในการทดลองพื้นที่จริงผู้วิจัยจึงเลือกใช้รูปแบบการติดตั้งแผ่น ArUco Marker ในแนวราบในกระบวนการทดสอบถัดไป

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

ตารางที่ 11 ค่าเฉลี่ยระยะทางของยานพาหนะจากกระบวนการประมาณตำแหน่งจากภาพ

ระยะทางจากการวัดจริง (เมตร)	ระยะทางเฉลี่ยจากการตรวจจับและติดตามวัตถุแบบวางราบ (เมตร)	ระยะทางเฉลี่ยจากการตรวจจับและติดตามวัตถุแบบวางตั้ง (เมตร)	ความคลาดเคลื่อนระหว่างระบบตรวจจับและติดตามวัตถุแบบวางราบเปรียบเทียบกับการวัดระยะห่าง (เปอร์เซนต์)
7.3	7.0	5.5	2.74
12.9	12.5	10.3	3.1
22.5	22.0	17.3	2.22

4.1.3.3.2 ผลการทดสอบหาระยะทางของยานยนต์ในบริเวณพื้นที่ทดลองจริง

ในบริเวณทดสอบจริงเมื่อทำการตรวจจับและติดตามวัตถุได้ผลการทดสอบดังรูปที่ 83

รูปที่ 83 ผลการทดสอบตรวจจับบริเวณพื้นที่จริง

ทำการบันทึกผลทดลองวัดตำแหน่งของการตรวจจับยานยนต์เทียบกับจุดร่วมและนำมาหาค่าเฉลี่ยในของตำแหน่งผลลัพธ์จากการทดสอบในระยะ 10 – 50 เมตรได้ผลการทดลองดังตารางที่ 12 โดยในตารางจะประกอบด้วยค่าระยะที่ได้จากการวัดด้วยเครื่องมือวัด ค่าระยะที่ได้จากการตรวจจับภาพ และค่าความคลาดเคลื่อนที่คำนวนเปรียบเทียบระหว่างระยะทางจากการวัดด้วยเครื่องมือและระยะทางจากการตรวจจับภาพ

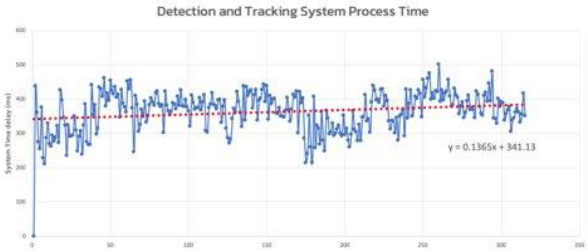
จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

ตารางที่ 12 ผลการวัดระยะในสภาพแวดล้อมจริงเปรียบเทียบกับค่าระยะที่ทำการวัดกำหนดไว้

ระยะที่ทำการวัดโดยเครื่องมือวัด (เมตร)	ระยะทางเฉลี่ยที่ได้จากการตรวจจับด้วยภาพ (เมตร)	ค่าความคลาดเคลื่อนในการตรวจจับระยะทาง (เปอร์เซนต์)
50	66.04	32.08
40	52.18	30.44
30	35.9	19.68
25	24.03	3.89
20	18.20	8.98
10	11.67	16.66

4.1.5 เวลาทั้งหมดในการทำงานของระบบตรวจจับและติดตามเพื่อคำนวนหาระยะทางและความเร็วของยานยนต์

เมื่อทำการวัดค่าเวลาทั้งหมดในการทำงานของกระบวนการตรวจจับและติดตามวัตถุ และทำเก็บข้อมูลเวลามาสร้างกราฟได้ดังรูปที่ 84 จากการคำนวนพบว่าค่าเฉลี่ยของเวลาในการทำงานของระบบคือ 366.03 มิลลิวินาที และค่าสูงสุดที่ได้จากการเก็บค่าคือ 501 มิลลิวินาที

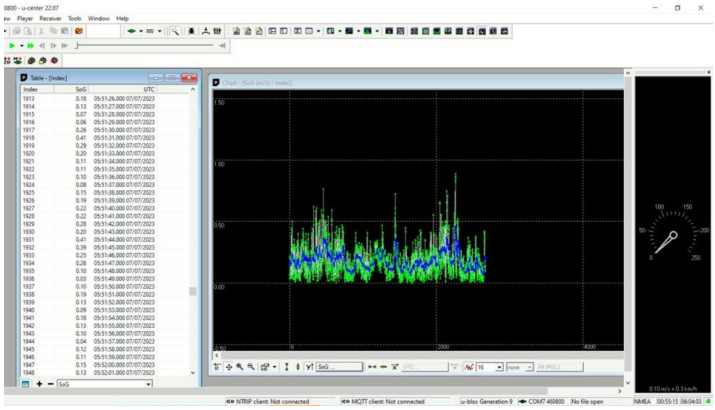


รูปที่ 84 กราฟแสดงผลการเก็บค่าเวลาที่ใช้ในระบบตรวจจับและติดตามวัตถุ

4.1.4 การคำนวณหาค่าความเร็ว

เมื่อทำการทดลองในบริเวณจริงพบว่าที่บริเวณที่ไกลกว่า 30 เมตรจากจุดร่วมหรือบริเวณ 40 เมตรจากกล้องมีค่าความคลาดเคลื่อนที่สูงมากจากหลายปัจจัย ในการทดลองจับค่าความเร็วผู้วิจัยจึงเลือกเก็บค่าความเร็วในบริเวณที่ไม่เกิน 40 เมตรจากกลอง

ทดลองการคำนวณหาความเร็วของยานพาหนะโดยเทียบการวัดค่าจากกระบวนการตรวจจับและติดตามวัตถุและนำมาเปรียบเทียบกับค่าจริงโดยค่าจริงสามารถวัดค่าได้จากค่า GPS U-Blox Zed-F9P ดังรูปที่ 85



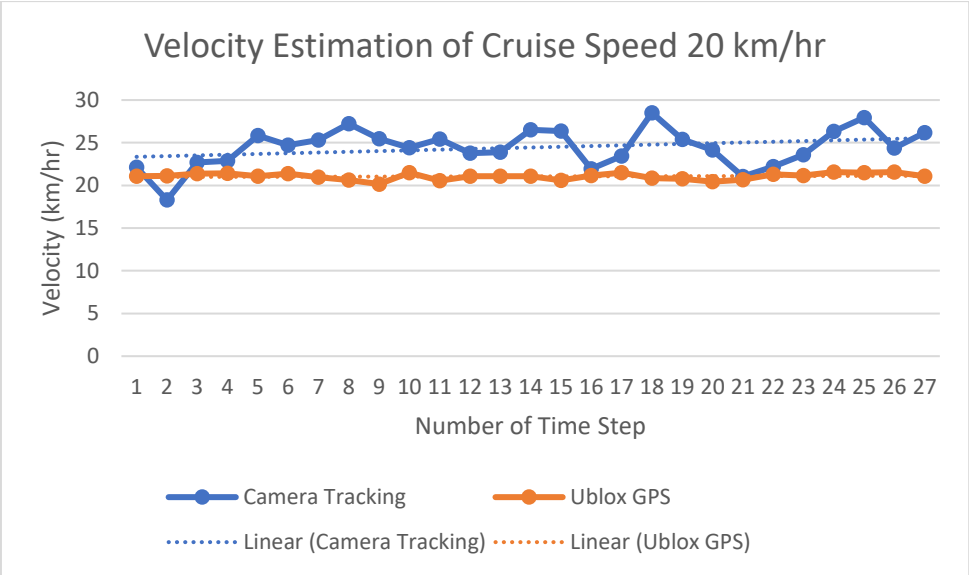
รูปที่ 85 ผลการทดลองที่ได้จากการตรวจจับด้วย GPS

เมื่อทำการขับเคลื่อนด้วยความเร็วคงที่ Cruise Speed และนำค่ามาเปรียบเทียบกับค่าความเร็วจากกระบวนการ Detection and Tracking ในบริเวณระยะห่างจากกล้องน้อยกว่า 41.5 เมตร ในความเร็ว 20 30 และ 40 กิโลเมตรต่อชั่วโมง และนำมาคำนวนหาค่าความคลาดเคลื่อนจากการวัดได้ผลการทดลองดังตารางที่ 13

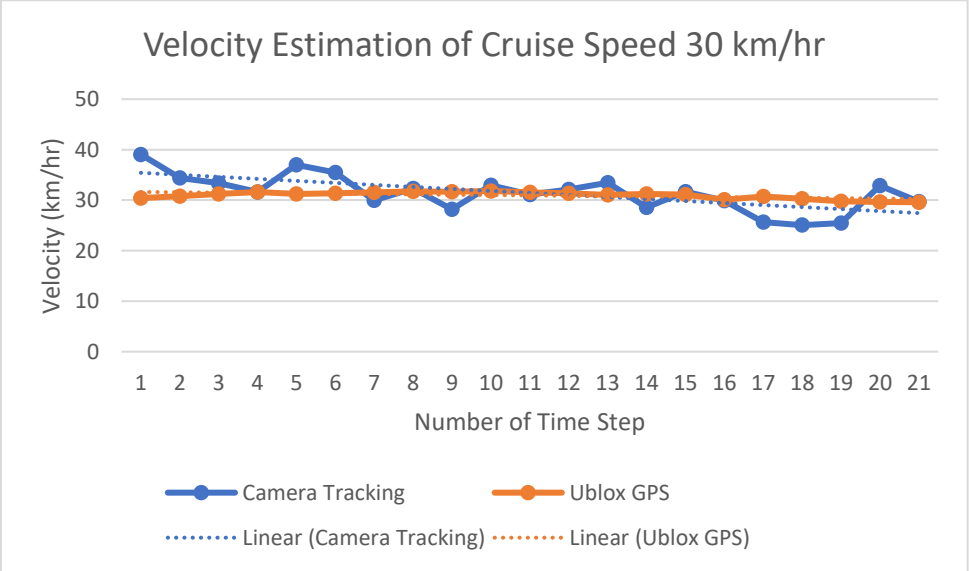
ตารางที่ 13 ผลการทดลองเปรียบเทียบค่าความเร็วเฉลี่ยที่วัดค่าได้จากการระบวนการติดตามด้วย GPS และกระบวนตามติดตามวัตถุด้วยภาพ

ความเร็วจากการตั้ง Cruise Control (km/hr.)	ความเร็วเฉลี่ยจากการวัดค่าด้วย GPS (km/hr.)	ความเร็วเฉลี่ยจากกระบวนการตรวจจับและติดตามวัตถุ (km/hr)	ค่าความคลาดเคลื่อนระหว่างระบบตรวจจับและติดตามวัตถุเปรียบเทียบกับการวัดด้วย GPS (เปอร์เซนต์)
20	19.84	24.44	23.18
30	30.8	31.43	2.05
40	40.4	38.58	4.5

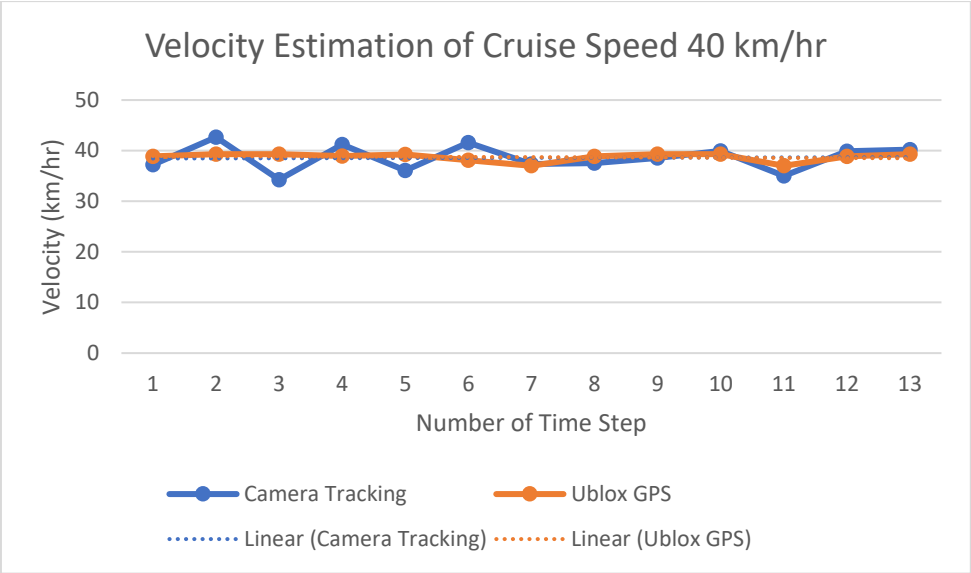
ซึ่งสามารถแสดงผลการการทดลองวัดค่าความเร็วเปรียบเทียบระหว่าค่าความเร็วจาก GPS เปรียบเทียบกับค่าความเร็วจากกระบวนการติดตามวัตถุได้ดังกราฟในรูปที่ 86 รูปที่ 87 และรูปที่ 88



รูปที่ 86 กราฟแสดงการเปรียบเทียบค่าความเร็วระหว่างกระบวนการวัดด้วย GPS และกระบวนการติดตามวัตถุ ที่ความเร็ว 20 กิโลเมตรต่อชั่วโมงในระยะน้อยกว่า 31.5 เมตร



รูปที่ 87 กราฟแสดงการเปรียบเทียบค่าความเร็วระหว่างกระบวนการวัดด้วย GPS และกระบวนการติดตามวัตถุ ที่ความเร็ว 40 กิโลเมตรต่อชั่วโมงในระยะน้อยกว่า 31.5 เมตร



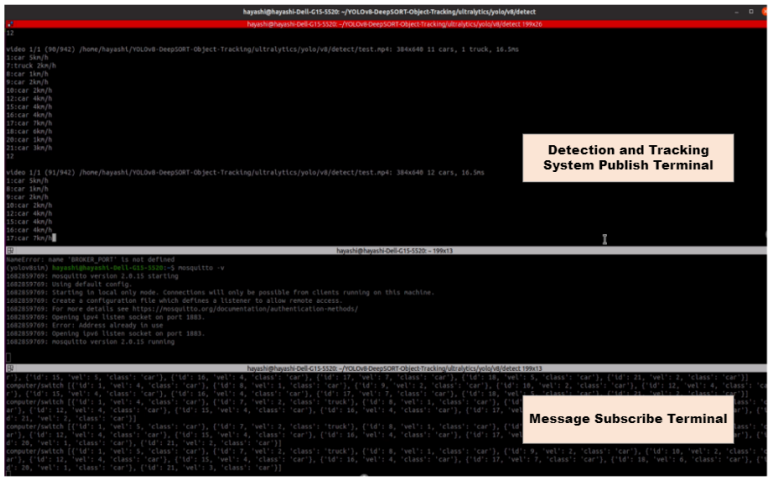
รูปที่ 88 กราฟแสดงการเปรียบเทียบค่าความเร็วระหว่างกระบวนการวัดด้วย GPS และกระบวนการติดตามวัตถุ ที่ความเร็ว 40 กิโลเมตรต่อชั่วโมงในระยะน้อยกว่า 31.5 เมตร

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

4.2 ระบบการส่งข้อมูลผ่าน MQTT

4.2.1 ผลการทำงานของระบบการส่งข้อมูลผ่าน MQTT Broker

จากการทดลองส่งข้อมูลที่ได้จากการตรวจจับเข้าสู่ระบบ MQTT โดยทดลองผ่านการใช้เครื่อง CPE สำหรับกระจายสัญญาณเครือข่ายเซลลูลาร์ และทดสอบในบริเวณเครือข่าย LAN เดียวกันได้ผลการทดสอบดังรูปที่ 89

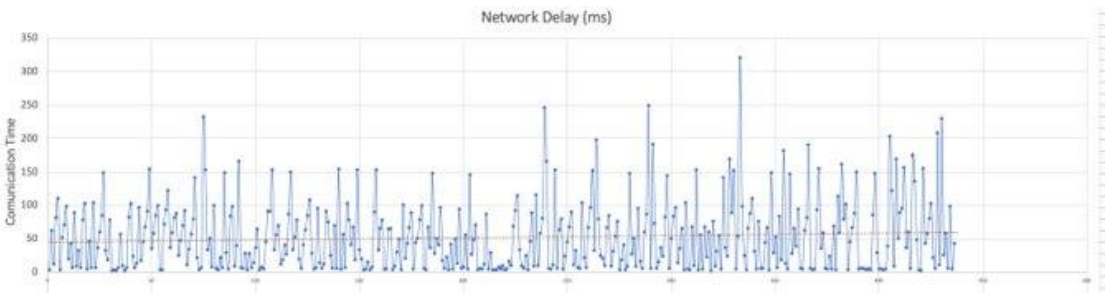


รูปที่ 89 ผลการทดสอบรส่งค่าตัวแปรจากกระบวนการตรวจจับวัตถุเข้าสู่ MQTT Broker

ในรูปแสดงผลการทดลองประกอบด้วย Command ในการสั่งงานระบบตรวจจับโดยในระบบตรวจจับจะกำหนดให้ทำการ Publish ค่าที่ได้จากการตรวจจับและติดตามวัตถุเข้าสู่ MQTT Broker โดยใช้เครือข่ายเซลลูลาร์ที่เชื่อมต่อโดยใช้อุปกรณ์ HUAWEI 5G CPE

4.2.2 ค่าเวลาที่วัดได้จากการส่งข้อมูลผ่าน MQTT Broker ด้วยเครือข่ายเซลลูลาร์

เมื่อได้ทำการทดสอบความเร็วในการรับส่งข้อมูลของระบบ MQTT โดยทำการเก็บค่าเวลาที่ใช้ในการรับและส่งข้อความจากกระบวนการตรวจจับและติดตามวัตถุได้ผลข้อมูลดังกราฟในรูปที่ 90 ซึ่งจากการคำนวนพบว่าระบบสามารถรับส่งข้อความได้ค่าความเร็วเฉลี่ยที่ 52 มิลลิวินาที โดยมีค่าเวลาสูงสุดที่ใช้ในกระบวนการส่งข้อมูลคือ 301 มิลลิวินาที และค่าเวลาต่ำสุดคือ 2 มิลลิวินาที

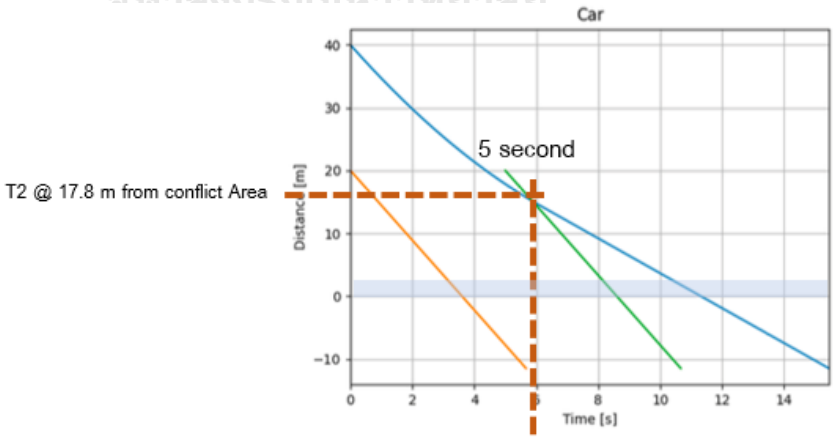


รูปที่ 90 กราฟแสดงผลการเก็บค่าเวลาที่ใช้ในการส่งข้อมูลผ่าน MQTT protocol ด้วยสัญญาณเครือข่ายเซลลูลาร์

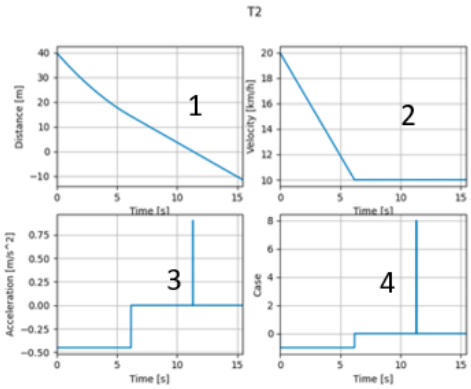
4.3 ระบบ Simulation

4.3.1 ค่าผลลัพธ์จากกระบวนการ Simulation

จากการทดสอบการตัดสินใจของรถอัตโนมัติ ทดลองโดยการกำหนดค่าระยะทางและความเร็วของยานยนต์บนท้องถนนที่ระบบสามารถตรวจจับได้ ทำการทดสอบระบบด้วยค่าความเร็วของยานยนต์บนท้องถนนที่ได้รับจากกระบวนการตรวจจับและติดตามวัตถุ 3 ค่า คือ 20 กิโลเมตรต่อชั่วโมง 30 กิโลเมตร ต่อชั่วโมงและ 40 กิโลเมตรต่อชั่วโมงโดยโปรแกรม Simulation เมื่อรับค่าตัวแปรความเร็วและตำแหน่งของยานยนต์บนท้องถนน ทำการคำนวนและแสดงผลการทดลองดังกราฟตัวอย่างในรูปที่ 91 และรูปที่ 92 โดยจากกราฟในรูปที่ 91 เป็นกราฟเปรียบเทียบระหว่างระตำแหน่งของรถอัตโนมัติ Turing และตำแหน่งยานยนต์บนท้องถนนโดยวัดจากจุดร่วมขอองถนน เทียบกับเวลาตั้งแต่รถอัตโนมัติเริ่มรับค่าจากระบบตรวจจับและติดตามยานยนต์ จนกระทั่งรถอัตโนมัติเคลื่อนที่ออกจาก Conflict Area จากตัวอย่างในกราฟแสดงให้เห็นว่ารถอัตโนมัติ Turing รับค่าจากระบบตรวจจับเมื่อรถอัตโนมัติอยู่ในตำแหน่งที่มีระยะห่างจาก Conflict Area ประมาณ 18 เมตรโดยแถบสีนำเงินบนกราฟแสดงระยะทางที่เกิดจากค่าเวลารวมของระบบตรวจจับและติดตามยานยนต์ และระบบสื่อสารผ่าน MQTT Broker ด้วยเครือข่ายเซลลูลาร์ โดยมีระยะเวลาเฉลี่ยรวมอยู่ที่ 418 มิลลิวินาทีเมื่อทำการคำนวนออกมาเป็นระยะทางในการเคลื่อนที่ด้วยความเร็วต้นคือ 20 กิโลเมตรต่อวินาทีและความหน่วงคือ -0.45 m/s^2 ได้ค่าระยะทางคือ 2.28 เมตร ซึ่งนำมาแสดงในรูปแถบสีในกราฟรูปที่ 91



รูปที่ 91 กราฟแสดงผลค่าระยะทางเทียบกับเวลาระหว่าง Turing และยานยนต์บนท้องถนน



รูปที่ 92 กราฟแสดงค่าคุณสมบัติของยานยนต์อัตโนมัติ Turing T2

จากรูปที่ 92 แสดงกราฟความสัมพันธ์ระหว่างตัวแปรต่างๆของยานยนต์อัตโนมัต Turing เทียบกับค่าเวลาโดยสามารถอธิบายค่าตัวแปรแต่ละกราฟได้คือ

กราฟที่ 1 ความสัมพันธ์ ระหว่างระยะทางกับเวลา

กราฟที่ 2 ความสัมพันธ์ ระหว่างความเร็วกับเวลา

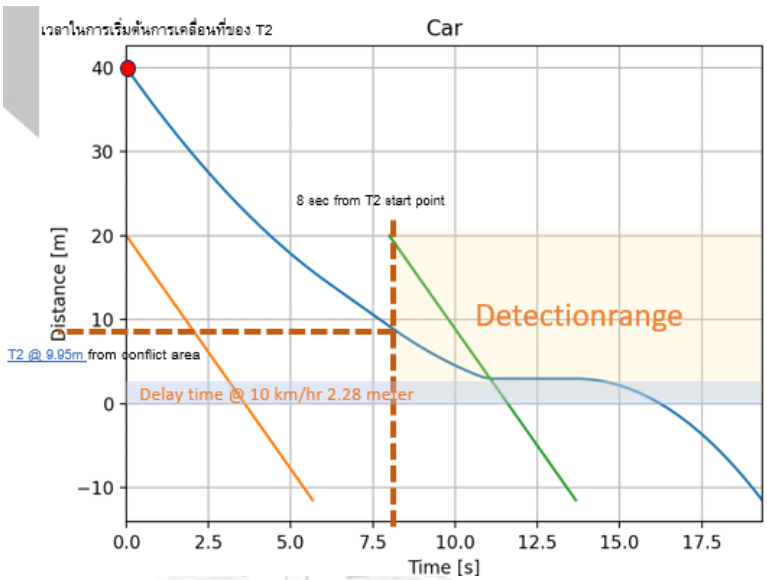
กราฟที่ 3 ความสัมพันธ์ ระหว่างความเร่งกับเวลา

กราฟที่ 4 ความสัมพันธ์ ระหว่างความสถานะของรถอัตโนมัติกับเวลา

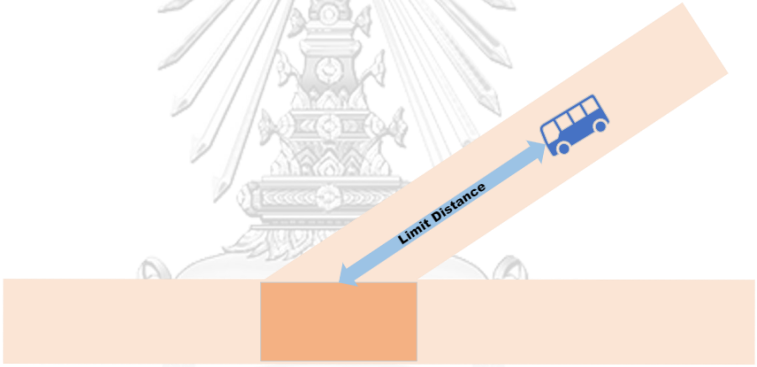
โดยในการ Simulation นี้ทำการหาค่าระยะห่างของยานยนต์อัตโนมัติ Turing จาก Conflict Area ที่น้อยที่สุด ที่สามารถรับค่าจากระบบตรวจจับในค่าความเร็วของยานยนต์บนท้องถนน 20 กิโลเมตรต่อชั่วโมง 30 กิโลเมตรต่อชั่วโมงและ 40 กิโลเมตรต่อชั่วโมง ยังสามารถขับเคลื่อนได้อย่างปลอดภัยโดยเมื่อมีการตรวจจับที่ระยะของรถอัตโนมัติ Turing น้อยกว่าระยะที่หาได้ จำเป็นต้องให้ Safety driver เข้ามาทำการควบคุม

4.3.2 ผลการทดลอง Simulation จากการรับค่าจากระบบตรวจยานยนต์ในขอบเขตการตรวจจับ 20 เมตร

จากสถานการทดสอบจริงซึ่งสามารถวัดค่าตำแหน่งของยานยนต์บนท้องถนนได้ในระยะ 20 เมตรด้วยความเร็ว 20 เมตรต่อวินาที เมื่อนำผลจากการตรวจจับเข้าสู่กระบวนการ Simulation ได้ผลการทดสอบหาระยะสุดห่างของยานยนต์อัตโนมัติ และ Conflict Area สุดท้ายที่จำเป็นต้องรับข้อมูลการตรวจจับดังกราฟในรูปที่ 93 โดยแสดงตำแหน่งที่ต้องการหาระยะดังรูปที่ 94

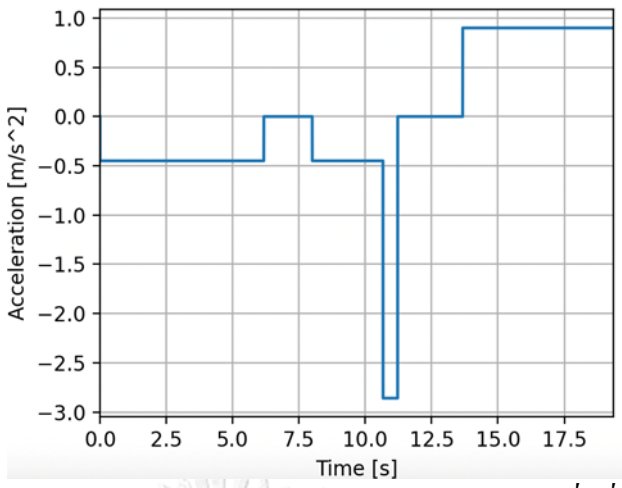


รูปที่ 93 กราฟแสดงระยะห่างระหว่างรถอัตโนมัติและ Conflict Area ที่น้อยที่สุดที่รับค่าจากระบบตรวจจับที่ความเร็ว 20 km/hr สามารถเคลื่อนรถออกจากConflict Area ได้อย่างปลอดภัย



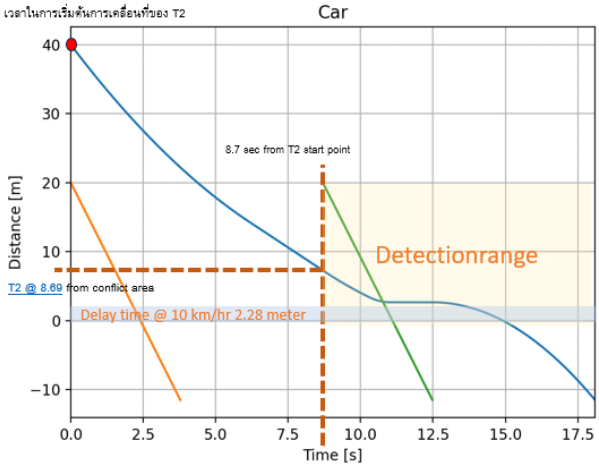
รูปที่ 94 ระยะทางที่สั้นที่สุดในการรับค่าจากระบบตรวจจับยานยนต์

จากราฟในรูปที่ 93 ที่ได้รับจากระบวนการ Simulation พบว่ามีค่า Delay Time คือ 418 มิลลิวินาทีซึ่งคำนวนเป็นระยทางคือ 2.28 เมตรดังแถบสีฟ้าดังกราฟที่ 93 โดยรถอัตโนมัติสามารถชนกับยานยนต์บนท้องถนนเมื่อจุดตัดของกราฟอยู่ในบริเวณตำแหน่งระยะทางเท่ากับ 0 โดยในผลการทดสอบนี้จำคำนึงว่าจุดตัดไม่ควรอยู่ในช่วงบริเวณ Delay time ซึ่งเป็นบริเวณตำแหน่งที่ ไม่สามารถควบคุมตัวแปรได้โดยจากการทดสอบ Simulation พบว่าระยะที่น้อยที่สุดที่ควรตรวจจับยานยนต์บนท้องถนนที่วิ่งด้วยความเร็ว 20 กิโลเมตรต่อชั่วโมงได้คือ 9.95 เมตร โดยในบริเวณนี้จะเป็นจุดเริ่มต้นของการใช้ Emergency brake ดังรูปที่ 95

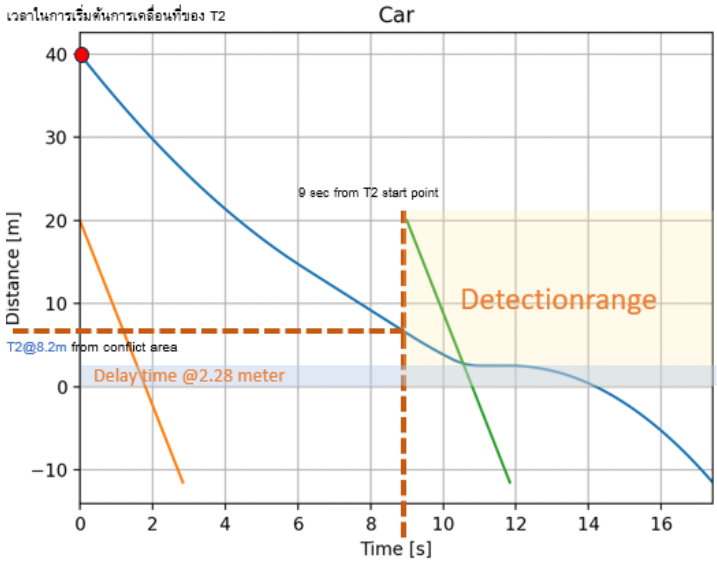


รูปที่ 95 กราฟแสดงค่าความเร่งของยานยนต์อัตโนมัติ Turing T2 เมื่อเริ่มทำการ Emergency Brake

เมื่อทำการทดสอบด้วยวิธีการเดียวกันในการตรวจจับระดับความเร็วของยานยนต์บนท้องถนนที่ต่างกันคือ 30 กิโลเมตรต่อชั่วโมงและ 40 กิโลเมตรต่อชั่วโมง ได้ผลการ Simulation ดังกราฟในรูปที่ 96 และรูปที่ 97 สามารถหาระยะห่างที่สั้นที่สุดของรถอัตโนมัติและบริเวณ Conflict Area ได้ที่ค่าความเร็ว 30 กิโลเมตรต่อชั่วโมง มีระยะคือ 8.69 เมตร และที่ค่าความเร็ว 40 กิโลเมตรต่อชั่วโมง มีระยะคือ 8.2 เมตรซึ่งพบว่าในค่าระยะทางที่กำหนดในแต่ละความเร็วและมีจุดตัดของกราฟอยู่ในบริเวณก่อนตำแหน่งที่ระยะทางเท่ากับ 0 และไม่อยู่ในช่วงระยะ Time Delay ส่งผลคือ 2.28 เมตร



รูปที่ 96 กราฟแสดงระยะห่างระหว่างรถอัตโนมัติและ Conflict Area ที่น้อยที่สุดที่รับค่าจากระบบตรวจจับที่ความเร็ว 30 km/hr สามารถเคลื่อนรถออกจากConflict Area ได้อย่างปลอดภัย



รูปที่ 97 กราฟแสดงระยะห่างระหว่างรถอัตโนมัติและ Conflict Area ที่น้อยที่สุดที่รับค่าจากระบบตรวจจับที่ความเร็ว 40 km/hr สามารถเคลื่อนรถออกจากConflict Area ได้อย่างปลอดภัย

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

บทที่ 5
สรุปผลการทดลอง

ในงานวิจัยนี้ผู้วิจัยได้ทำการพัฒนาระบบป้องกันการชนสำหรับยานยนต์อัตโนมัติโดยใช้การตรวจจับวัตถุด้วยกล้อง ร่วมกับการสื่อสารระหว่างยานพาหนะอัตโนมัติและโครงสร้างพื้นฐานผ่านเครือข่ายเซลลูลาร์โดยได้ทำการพัฒนาในสามส่วนหลักคือ ระบบการตรวจจับและติดตามวัตถุ ระบบการสื่อสารระหว่างระบบตรวจจับและระบบการตัดสินใจของรถอัตโนมัติ

ในส่วนแรกผู้วิจัยได้ทำการพัฒนาระบบตรวจจับด้วยภาพของ ระบบการป้องกันการชน โดยการเปลี่ยนแปลงรูปแบบของโมเดลสำหรับกระบวนการตรวจจับวัตถุด้วยภาพจากโมเดล YOLOv4 เป็น โมเดล YOLOv8 โดยการเปลี่ยนโมเดลนี้ทำให้การตรวจจับวัตถุมีความเร็วและประสิทธิภาพที่ดีขึ้นอย่างมากเมื่อเทียบโดยใช้เครื่องมือในการประมวลผลชนิดเดียวกัน

เนื่องจากในกระบวนการตัดสินใจของยานยนต์อัตโนมัติ จำเป็นต้องใช้ค่าตัวแปร ความเร็วและระยะทางในการคำนวนหาการตัดสินใจของยานยนต์อัตโนมัติในงานวิจัยนี้ผู้วิจัยจึงเลือกนำวิธีการติดตามวัตถุ (Object Tracking) เข้ามาประยุกต์ใช้ร่วมกับกระบวนการตรวจจะจับยานพาหนะ โดยในงานวิจัยนี้ผู้วิจัยเลือกใช้วิธีการ DeepSORT Tracking เข้ามาประยุกต์ใช้กับกระบวนการตรวจจับยานพาหนะ ทำให้ระบบตรวจจับวัตถุของงานวิจัยนี้สามารถระบุค่า Identification Number ของยานพาหนะได้แม้ยานยนต์ที่ทำการตรวจจับจะถูกบดบัง ต่อมาผู้วิจัยได้ทดลองติดตามความเร็วของยานพาหนะที่ได้จากการติดตามวัตถุและพบว่าค่าที่ได้มีการเปลี่ยนแปลงอย่างมากเมื่อเทียบกับค่าความเร็วจริง เพื่อที่จะสามารถหาค่าความเร็วของวัตถุให้แม่นยำมากขึ้นและหาตำแหน่งของวัตถุที่ได้ทำการตรวจจับในสภาพแวดล้อมเดิมได้ผู้วิจัยได้ทำการทดลองการคำนวนสมการการทำงานของกล้องสองมิติเพื่อทำการพิสูจน์ว่าสามารถหาระยะของวัตถุที่สามารถตรวจจับได้เมื่อรู้ว่าวัตถุเคลื่อนที่อยู่ในระนาบที่สนใจ โดยในงานวิจัยนี้ได้นำ ArUco Marker มากำหนดจุดเพื่อสร้างระนาบของถนนที่ยานยนต์กำลังเคลื่อนที่อยู่ทำให้สามารถคำนวนหาค่าระยะจริงของยานพาหนะที่บนท้องถนนได้โดยการใช้กล้องสองมิติต่อมาได้นำไปทดสอบวัดในพื้นที่จริงและได้พบว่าที่กระบวนการที่สร้างขึ้นสามารถวัดค่าตำแหน่งของวัตถุได้ และพบว่าสามารถตรวจจับระยะทางในการตรวจจับวัตถุได้ที่ระยะ 31.5 เมตรเทียบจากกล้องและสามารถตรวจจับความเร็วของยานยนต์ในระยะ 31.5 เมตรได้

ต่อมางานวิจัยนี้ได้ทำการทดสอบส่งค่าข้อมูลที่ได้จากการตรวจจับเข้าสู่ระบบ MQTT Communication ผ่าน mosquito Broker และได้มีการสร้าง Node Mqtt เพื่อมารับค่าข้อมูลจากการทดตรวจจับวัตถุ โดยสามารถรับค่าข้อมูลได้ คือ ความเร็ว ตำแหน่ง ชื่อของวัตถุ และชนิดของวัตถุ

ในงานวิจัยนี้สร้างแบบจำลองสำหรับกระบวนการตัดสินใจบริเวณทางร่วมของรถอัตโนมัติ จากโค้ด Python ด้วยการนำค่าตัวแปรที่จำเป็นต่อการใช้งานสำหรับกระบวนการตัดสินใจมาจากค่าตัวแปรที่ทำหารเก็บค่าจากกระบวนการใช้งานจริงของรถอัตโนมัติTuring OPAL T2 ด้วยข้อกำหนดที่ความเร็วของรถบนท้องถนนคงที่ และได้สร้าง Scenario สำหรับทดสอบกระบวนการตัดสินใจของรถอัตโนมัติจากค่าความเร็ว ตำแหน่งและสถานการที่ได้จากการทดสอบระบบตรวจจับและติดตามวัตถุ โดยจากการทดสอบได้ทำการหาตำแหน่งของรถอัตโนมัติเทียบกับตำแหน่ง Conflict Area ที่น้อยที่สุดในแต่ละย่านความเร็วของยานยนต์บนท้องถนนที่สามารถตัดสินใจชะลอยานยนต์อัตโนมัติได้อย่างปลอดภัยโดยได้คำนึงถึงเวลาในการทำงานของระบบตรวจจับและระบบส่งข้อมูลแล้ว

5.1 ปัญหาที่พบ

เนื่องจากกระบวนการตรวจจับภาพเพื่อให้ผลลัพธ์ในการคำนวนที่ดีจำเป็นต้องมีหน่วยประมวลผลภาพที่ดีด้วย จากการทดสอบด้วยหน่วยประมวลผลภาพที่ทำการใช้งานในปัจจุบันพบว่าหน่วยประมวลผลภาพที่ใช้งานอยู่มีอายุการใช้งานที่นานมากทำให้ไม่สามารถแสดงประสิทธิภาพในการตรวจจับได้ไม่ดีเท่าที่คาดการณ์ไว้

ในกระบวนการทดลองจริงเนื่องจากความละเอียดของกล้องค่อนข้างน้อย ในกระบวนการหาระยะของยานยนต์ในตำแหน่งระยะไกลพบว่ามีค่าความคลาดเคลื่อนสูง เมื่อเทียบกับการทดลองในพื้นที่ขนาดเล็ก

ในกระบวนการตรวจจับ ArUco Marker จากข้อจำกัดของความสูงในการติดตั้งกล้องทำให้ไม่สามารถติดตั้งกล้องได้ในบริเวณความสูงที่มากพอได้ทำให้กระบวนการตรวจจับ ArUco มีข้อจำกัดในการวัดระยะทางเพื่อนำค่าตัวแปรมาประกอบการคำนวนต่อ

จากกระบวนการตรวจจับ ArUco Marker พบว่า เมื่อมีตัวแปรที่เราไม่สามารถควบคุมได้คือแสง ตกกระทบเข้าสู่ แผ่น ArUco Marker ที่มากเกินไปทำให้ArUco Marker สามารถตรวจจับได้ยาก

ในบริเวณพื้นที่ทดสอบมีการขับขี่ยานยนต์เป็นจำนวนมากทำให้มีความอันตรายในการทดสอบเมื่อเทียบกับบริเวณทดลอง

5.2 ข้อเสนอแนะ

ผลงานวิจัยนี้สามารถพัฒนาการตรวจจับและติดตามวัตถุได้โดยการเพิ่มความสามารถของกล้องและหน่วยประมวลผลภาพและความสูงในการติดตั้งกล้องซึ่งในกระบวนการดังกล่าวนี้จะสามารถปรับปรุงระบบให้สามารถประมาณค่าระยะทางได้ในระยะทางที่ไกลขึ้นและมีความแม่นยำที่มากขึ้น

สามารถพัฒนาต่อยอดเพื่อนำข้อมูลที่ได้ไปทดลองร่วมกับการใช้รถจริงเพื่อทำการตัดสินใจจริงเพื่อพัฒนาให้รถอัตโนมัติ Turing T2 เคลื่อนที่ออกจากทางร่วมได้อย่างปลอดภัย

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

บรรณานุกรม

1. Sen, B., J.D. Smith, and W.G. Najm, *Analysis of lane change crashes*. 2003, United States. National Highway Traffic Safety Administration.
2. Administration, N.H.T.S., *Traffic safety facts 2017: A compilation of motor vehicle crash data.* DOT HS, 2019. **812**: p. 806.
3. Protection, O., *Traffic Safety Facts.* 2016.
4. 3M. *What is Vehicle-to-Infrastructure (V2I) communication*. 2022; Available from: https://www.3m.com/3M/en_US/road-safety-us/resources/road-transportation-safety-center-blog/full-story/~/what-is-vehicle-to-infrastructure-v2i-communication-and-why-do-we-need-it/?storyid=021748d7-f48c-4cd8-8948-b7707f231795.
5. Liu, L., et al., *Computing systems for autonomous driving: State of the art and challenges.* IEEE Internet of Things Journal, 2020. **8**(8): p. 6469-6486.
6. กิจศิริกุล, ผ.ด.บ., *Artificial intelligence*, ed. 1.0.2. 2007, จุฬาลงกรณ์มหาวิทยาลัย.
7. Bahi, M. and M. Batouche. *Deep learning for ligand-based virtual screening in drug discovery*. in *2018 3rd international conference on pattern analysis and intelligent systems (PAIS)*. 2018. IEEE.
8. Antkowiak, M., *Artificial Neural Networks vs. Support Vector Machines for Skin Diseases Recognition. Department of Computing Science*. 2006, Oxford, UK.
9. LeCun, Y., et al., *Gradient-based learning applied to document recognition.* Proceedings of the IEEE, 1998. **86**(11): p. 2278-2324.
10. กิตตินราดร, ช. *Convolutional Neural Network*. 2563; Available from: https://guopai.github.io/ml-blog19.html.
11. Indaba, D.L.; Available from: https://colab.research.google.com/github/deep-learning-indaba/indaba-pracs-2019/blob/master/3a_conv_nets.ipynb#scrollTo=NcAHF4g8Xa93.
12. JadonShruti *Introduction to different activation functions for deep learning*. 2018. **16**.
13. Yuheng, S. *Classification, Object Detection and Image Segmentation*. 2023; Available from: https://developer.qualcomm.com/software/qualcomm-neural-processing-sdk/learning-resources/image-segmentation-deeplab-neural-processing-sdk/classification-object-detection-segmentation.

14. Li, K., et al., *Object detection with convolutional neural networks*, in *Deep Learning in Computer Vision*. 2020, CRC Press. p. 41-62.
15. Shah, D., *Mean Average Precision (mAP)*. 2022.
16. Bhandari, A. *Understanding & Interpreting Confusion Matrices for Machine Learning*. 2020; Available from: https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/.
17. Zaidi, S.S.A., et al., *A survey of modern deep learning based object detection models.* Digital Signal Processing, 2022: p. 103514.
18. SamYam. *Understanding Multiple Object Tracking using Deepsort*. 2022; Available from: https://learnopencv.com/understanding-multiple-object-tracking-using-deepsort/.
19. Hata, K., *CS231A Course Notes 1: Camera Models*. 2023.
20. Muñoz, R. *Detection of ArUco Markers*. 2023.
21. Mishra, B. and A. Kertesz, *The use of MQTT in M2M and IoT systems: A survey.* IEEE Access, 2020. **8**: p. 201071-201086.
22. Beuse, N., C. Harper, and K. Shain, *A COMPARISON OF VEHICLE ALERT MODALITIES'TIME-TO-COLLISION WARNINGS TRIGGERED BY THE VEHICLE'S CONTROLLER AREA NETWORK SYSTEM.* 2009.
23. Jiménez, F., J.E. Naranjo, and F. García, *An improved method to calculate the time-to-collision of two vehicles.* International Journal of Intelligent Transportation Systems Research, 2013. **11**(1): p. 34-42.
24. McGehee, D.V., E.N. Mazzae, and G.S. Baldwin. *Driver reaction time in crash avoidance research: Validation of a driving simulator study on a test track.* in *Proceedings of the human factors and ergonomics society annual meeting*. 2000. Sage Publications Sage CA: Los Angeles, CA.
25. Zou?, Z. and K.C. , Zhenwei Shi, Member, IEEE, *Object Detection in 20 Years: A Survey.* 2023.
26. Farhadi, J.R.a.S.D.a.R.G.a.A., *You Only Look Once: Unified, Real-Time Object Detection*. 2016.
27. OpenMMLab *Dive into YOLOv8: How does this state-of-the-art model work?* 2023.
28. Jocher, G., *Ultralytics YOLOv8:The State-of-the-Art YOLO Model.* 2023.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

29. Solawetz, J. *What is YOLOv8? The Ultimate Guide.* 2023.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

ภาคผนวก

โปรแกรมสำหรับการทดลอง

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

โปรแกรมสำหรับสร้างระนาบของถนนด้วย ArUco Marker ด้วยภาษา Python

```
    fx = 1394.6  # focal length in x direction
    fy = 1394.6  # focal length in y direction
    cx = 995.6  # x coordinate of principal point
    cy = 599.32  # y coordinate of principal point
    # # 4kcameralogitect
    # fx = 1027.861096044397  # focal length in x direction
    # fy = 1028.3420769929078  # focal length in y direction
    # cx = 340.54872314317566  # x coordinate of principal point
    # cy = 165.19220659193354  # y coordinate of principal point
    camera_matrix = np.array([[fx, 0, cx], [0, fy, cy], [0, 0, 1]],dtype=np.float32)
    return camera_matrix
def get_distortion_coefficient():
    # # Define the distortion coefficients
    k1 = 0.115  # radial distortion coefficient 1
    k2 = -0.219  # radial distortion coefficient 2
    p1 = 0.0012  # tangential distortion coefficient 1
    p2 = 0.0086  # tangential distortion coefficient 2
    k3 = 0.112  # radial distortion coefficient 3(not always present)
    dist_coeffs = np.array([k1, k2, p1, p2, k3])
    # 4k cam Define the distortion coefficients
    # k1 = 0.318240 # radial distortion coefficient 1
    # k2 = -2.93258  # radial distortion coefficient 2
    # p1 = 0.007015  # tangential distortion coefficient 1
    # p2 = 0.004752  # tangential distortion coefficient 2
    # k3 = 17.84034 # radial distortion coefficient 3(not always present)
    # dist_coeffs = np.array([k1, k2, p1, p2, k3])
    return dist_coeffs
def pose_esitmation2(frame, aruco_dict_type, matrix_coefficients, distortion_coefficients):
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    corners, ids, rej = detector.detectMarkers(gray)
    if ids is None:
        return frame,[]
    point=[]
    for i in range(0, len(ids)):
        # Estimate pose of each marker and return the values rvec and tvec---(different from
those of camera coefficients)
        rvec, tvec, markerPoints = cv2.aruco.estimatePoseSingleMarkers(corners[i], 0.75,
matrix_coefficients,distortion_coefficients)
        # Draw a square around the markers
        cv2.aruco.drawDetectedMarkers(frame, corners)
        # Draw Axis
        origin = (int(corners[i][0][0][0]),int(corners[i][0][0][1]))
        cv2.drawFrameAxes(frame, matrix_coefficients, distortion_coefficients, rvec, tvec, 0.01)
        if ids[i] in [0,1,2,3]:
            writeXYZ(frame,tvec,origin,ids,point)
    return frame,point
def writeXYZ(frame, tvec, origin, ids, point):
    tvec = tvec[0][0]
    txt = f'{tvec[0]:.2f}, {tvec[1]:.2f}, {tvec[2]:.2f}'
    print("--------------------------------")
    point.append([tvec[0], tvec[1], tvec[2]])
    cv2.putText(frame, txt, origin, cv2.FONT_HERSHEY_SIMPLEX, 1, thickness=2, color=(0, 255, 0))
    return tvec, point
```

```
def draw_board(img):
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    corners, ids, rej = detector.detectMarkers(gray)

    if len(corners) > 0:
        cv2.aruco.drawDetectedMarkers(img, corners)
        print("----------------------------------------")
        for i in range(len(ids)):

            origin = (int(corners[i][0][0][0]),int(corners[i][0][0][1]))
            # print(origin)
            cv2.putText(img, str(ids[i][0]), origin, cv2.FONT_HERSHEY_SIMPLEX, 1, (255,0,0))
    return img
def calibrateCamera(camera, charuco_board):
    cam_mat = get_camera_matrix()
    dist_coef = get_distortion_coefficient()
    return cam_mat, dist_coef
def pointToplane(point):
    points_plane = np.array(point).T
    # Find plane
    p1 = points_plane[:, 0]
    p2 = points_plane[:, 1]
    p3 = points_plane[:, 2]
    p1p2 = p2 - p1
    p1p3 = p3 - p1
    norm_vec = np.cross(p1p2, p1p3)
    A = norm_vec[0]
    B = norm_vec[1]
    C = norm_vec[2]
    D = -np.dot(norm_vec, p1)
    equation=f'{A:.2f}, {B:.2f}, {C:.2f},{D:.2f}'
    # print (equation)
    EQ=[A,B,C,D]
    return EQ
def findObjectPoint(Upip,Vpip,eq):
    x_cam = 0
    y_cam = 0
    z_cam = 0
    transform = np.array([[1, 0, 0, x_cam], [0, 1, 0, y_cam], [0, 0, 1, z_cam]])
    cam_mat= get_camera_matrix()
    # print(cam_mat)
    fullcam_mat= np.dot(cam_mat,transform)
    cam_con = np.vstack((fullcam_mat, eq))
    # print(cam_con)
    inv_mat = np.linalg.inv(cam_con)
    # print(inv_mat)
    vec = np.array([Upip, Vpip, 1, 0])
    sol_vec = np.dot(inv_mat, vec)
    s = 1/sol_vec[3]
    x = sol_vec[0]*s
    y = sol_vec[1]*s
    z = sol_vec[2]*s
    position_mat=[x,y,z]
    return position_mat
def main():
    print(cv2.__version__)
    camera = cv2.VideoCapture(2)
    camera.set(3, 1920)
    camera.set(4, 1080)
    cam_mat, dist_coef = calibrateCamera(camera, board)
```

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

```
    while True:
        Upip=1400
        Vpip=800
        ret, img = camera.read()
        print(ret)
        img = cv2.circle(img,(Upip,Vpip),radius=3,color=(0,0,255),thickness=4)
        frame,point=pose_esitmation2(img, dictionary, cam_mat, dist_coef)
        if len(point) == 3:
            print(point)
            eq=pointToplane(point)
            print(eq)
# chanf path-----------------------------------------------------
            write_eq_to_yaml(eq, 'planeequation.yaml')
#-------------------------------------------------------------
            poseEs=findObjectPoint(Upip,Vpip,eq)
            print(poseEs)
        cv2.imshow("test", img)
        cv2.waitKey(10)
if __name__ == '__main__':
    main()
```

โปรแกรมในการตรวจจับวัตถุ

```
import os
from ultralytics import YOLO
import cv2
import cvzone
import math
from tracker import Tracker
from collections import deque
import numpy as np
from loaddata import read_eq_from_yaml, read_conflict_from_yaml

# csvfilesave
import csv
import datetime

# MQTT
# import paho.mqtt.client as mqtt
# client = mqtt.Client()
# client.connect('170.20.10.5',1883)
# import json

# Plane equation load
import yaml
from arucoequationsave import get_camera_matrix,get_distortion_coefficient,findObjectPoint


# velocity calculation
import time

# class TrackObj:
#     def __init__(self, class_id, obj_id, pos,whbbox):
#         self.class_id = class_id
#         self.obj_id = obj_id
#         self.pos = pos
#         self.whbboxes = whbbox


class ObjectTracker:
    def __init__(self):
```

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

```
        self.classNames =
["person","bicycle","car","motorcycle","airplane","bus","train","truck","boat","traffic
light",
       "fire hydrant","stop sign","parking
meter","bench","bird","cat","dog","horse","sheep","cow",
       "elephant","bear","zebra","giraffe","backpack","umbrella","handbag","tie","suitcase",
       "frisbee","skis","snowboard","sports ball","kite","baseball bat",
       "baseball glove","skateboard","surfboard","tennis racket","bottle",
       "wine glass","cup","fork","knife","spoon","bowl","banana","apple",
       "sandwich","orange","broccoli","carrot","hot dog","pizza","donut","cake",
       "chair","couch","potted plant","bed","dining table","toilet",
       "tv","laptop","mouse","remote","keyboard","cell phone","microwave",
       "oven","toaster","sink","refrigerator","book","clock","vase","scissors",
       "teddy bear","hair drier","toothbrush"]
        self.detectClassNames = ["bottle","car","motorcycle","bicycle","bus","truck"]
        self.list_cls_detectClasses = [self.classNames.index(name) for name in
self.detectClassNames]
        self.trackers = {}
        for detectedClass_cls in self.list_cls_detectClasses:
            self.trackers[detectedClass_cls] = Tracker(max_age=20 , max_cosine_distance= 0.4)
        self.model = YOLO('yolov8m.pt')
        self.cap = cv2.VideoCapture(2)
        # self.cap = cv2.VideoCapture("calibcar.mp4")
        self.cap.set(3, 1920)
        self.cap.set(4, 1080)
        self.data_deque = dict()
        #Region selection Image
        self.mask=cv2.imread("mask.png")
        # Choose section for detect
        # imgRegion = cv2.bitwise_and(img,mask)
        # results = model(imgRegion,stream=True)
        # results = model(img,stream=True,verbose=True)
        self.time_now = time.time()
        self.eqFilePath = 'planeequation.yaml'
        self.eq = read_eq_from_yaml(self.eqFilePath)
        # print(self.eq)
        self.conflictposition='conflictstart.yaml'
        self.conflict=read_conflict_from_yaml(self.conflictposition)

    def doTracking(self, img, detection):
        # if len(detection) > 0:
        for slct_cls in self.list_cls_detectClasses:
            slct_dets = []
            for det in detection:
                cls = det[-1]
                if cls == slct_cls:
                    slct_dets.append(det)
            self.trackers[slct_cls].update(img, slct_dets)
            if len(slct_dets) > 0:
                self.trackers[slct_cls].update(img, slct_dets)
        # self.processDataBase()
    def visualizeTracking(self, img, classes_id=None):
        for slct_cls in self.list_cls_detectClasses:
            tracker = self.trackers[slct_cls]
            className = self.classNames[slct_cls]

            if tracker.tracks is None:
                continue

            # mqtt part
            raw_msg=[]
```

```
            for track in tracker.tracks:
                bbox = track.bbox
                x1, y1, x2, y2 = bbox
                x1, y1, x2, y2 = int(x1), int(y1), int(x2), int(y2)
                track_id = track.track_id
                w, h = x2 - x1, y2 - y1
                center = int((x1+w/2)), int(y2)
                # find real world theposition
                # P = findObjectPoint(int(center[0]), int(center[1]), list(self.eq))
                # print(position)
                if classes_id is None or slct_cls in classes_id:
                    obj_id = track_id+slct_cls*100

                    # Store center in data_deque
                    if obj_id not in self.data_deque:
                        self.data_deque[obj_id] = deque(maxlen=10) # Change the maxlen to your
desired value
                   self.data_deque[obj_id].append(center)
                    if len(self.data_deque[obj_id])>=2:
                        pos_img1 = self.data_deque[obj_id][0]
                        pos_img2 = self.data_deque[obj_id][1]
                        pos1 = findObjectPoint(pos_img1[0],pos_img1[1], self.eq)
                        pos2 = findObjectPoint(pos_img2[0],pos_img2[1], self.eq)
                        vel = self.calculate_speed(pos1,pos2)
                        disCon = self.calculateDistanceConflict(pos2)
                        print(self.data_deque)
                        print(f'{className},{vel:.1f},{track_id},{pos2[2]:.1f}')
                        timestamp = datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')
                        # mqtt
                        obj_data=dict()
                        obj_data["id"]=track_id
                        obj_data["classs Name"]=className
                        obj_data["velocity"]=int(vel)
                        obj_data["Distance"]=int(disCon)
                        raw_msg.append(obj_data)
                        # csv SAVE
                        csv_file = 'DetectTrack.csv'
                        with open(csv_file, 'a', newline='') as file:
                            writer = csv.writer(file)
                            writer.writerow([timestamp, className, f'{vel:.2f}', track_id,
f'{disCon:.2f}'])

                        if vel is not None:
                            cvzone.putTextRect(img,
f'{className},ID:{track_id},Distance:{int(disCon)}m,Vel:{int(vel*18/5)}km/hr', (max(0, x1), max(35,
y1)), thickness=2, offset=3, scale=1)
                        else:
                            cvzone.putTextRect(img,
f'{className},ID:{track_id},Distance:{int(disCon)}', (max(0, x1), max(35, y1)), thickness=2,
offset=3, scale=1)
                    else:
                        cvzone.putTextRect(img, f'{className},ID:{track_id}', (max(0, x1),
max(35, y1)), thickness=2, offset=3, scale=0.8)
                    cvzone.cornerRect(img, (x1, y1, w, h), l=5)


                    # Draw path of the object
                    path = np.array(self.data_deque[obj_id])
                    cv2.polylines(img, [path], False, (255, 0, 0), thickness=2)
```

```
                    # Check if object is still in frame
                    if len(self.data_deque[obj_id]) == self.data_deque[obj_id].maxlen:
                        self.data_deque.pop(obj_id)  # Remove data for objects that have left
the frame
            # self.send_mqtt(raw_msg)


    def calculate_speed(self,point1, point2):
        distance = np.linalg.norm(np.array(point2) - np.array(point1))
        T=self.time_now-self.time_past
        speed = distance / T
        return speed

    def calculateDistanceConflict(self,point):
        distanceconflict = np.linalg.norm(np.array(point)- np.array(self.conflict))
        return distanceconflict

    def detectionYolo(self, img):
        detection = []
        results = self.model(img, stream=True, verbose=True)
        for r in results:
            boxes = r.boxes
            for box in boxes:
                x1, y1, x2, y2 = box.xyxy[0]
                x1, y1, x2, y2 = int(x1), int(y1), int(x2), int(y2)
                w, h = x2 - x1, y2 - y1
                conf = math.ceil((box.conf[0] * 100)) / 100
                cls = int(box.cls[0])
                currentClass = self.classNames[cls]
                if cls in self.list_cls_detectClasses and conf > 0.6:
                    detection.append([x1, y1, x2, y2, conf, cls])
        # print(detection)
        return detection


        # Nine,MQTT
    def send_mqtt(self,raw_msg):
        encoded_msg = json.dumps(raw_msg)
        client.publish('computer/switch',encoded_msg)

    # def temp(self,img):
    #     for data in self.data_deque.values():
    #         obj:TrackObj = data[0]
    #         # print(obj.obj_id, )
    #         print(obj.obj_id,self.classNames[obj.class_id],obj.pos)
    #         bbox=obj.whbboxes
    #         cvzone.putTextRect(img,
f'{self.classNames[obj.class_id]},ID:{obj.obj_id},{obj.pos[2]:.2}', (max(0, bbox[0]), max(35,
bbox[1])), thickness=1, offset=3, scale=0.8)
    def capture(self):
        self.time_past = self.time_now
        # print("------",self.time_past)
        success, img = self.cap.read()
        self.time_now = time.time()
        # print("------",self.time_now)
        return success, img

    def start_tracking(self):
        while True:
            success, img = self.capture()
            detection = self.detectionYolo(img)
            # print("----------------------------detection--------------------------------")
```

```
            self.doTracking(img, detection)
            self.visualizeTracking(img)
            # print("----------------------------tracking-------------------------------")
            # self.temp(img)
            cv2.imshow("Image", img)
            cv2.waitKey(10)


# Create an instance of the ObjectTracker class
tracker = ObjectTracker()
# Start tracking
tracker.start_tracking()
```

โปรแกรมในการรับ message MQTT

```
import paho.mqtt.client as mqtt
import json
client = mqtt.Client()
client.connect('192.168.8.6',1883)
SUBSCRIBE_TOPIC = 'computer/switch'
client.subscribe(SUBSCRIBE_TOPIC)
def on_message_callback(client, userdata, msg):
    topic = msg.topic
    encoded_msg = msg.payload
    decoded_msg = json.loads(encoded_msg)
    print(topic, decoded_msg)
client.on_message = on_message_callback
client.loop_forever()
```

โปรแกรมในกระบวนการ Simulation

```
import matplotlib.pyplot as plt

import time

# import paho.mqtt.client as mqtt

# def on_connect(client, userdata, flags, rc):1
#     print("Connected to MQTT broker")
#     client.subscribe("computer/switch")  # ตั้งค่าให้ติดตามการเปลี่ยนแปลงใน MQTT topic "computer/switch"

# def on_message(client, userdata, msg):
#     if msg.topic == "computer/switch":
#         input_data = msg.payload.decode("utf-8")
#         print("Received input data:", input_data)
#         # เรียกใช้ฟังก์ชัน Multi_Input() และประมวลผลต่อไป

# # สร้าง MQTT client
# client = mqtt.Client()

# # กำหนด callback function สำหรับการเชื่อมต่อและรับข้อมูล
# client.on_connect = on_connect
# client.on_message = on_message

# # เชื่อมต่อกับ MQTT broker
# client.connect("192.168.8.10", 1883) # แทน mqtt_broker_address ด้วยที่อยู่ IP/hostname ของ MQTT broker

# # เริ่มการรับข้อมูลจาก MQTT broker
# client.loop_start()
# # รอรับข้อมูลจาก MQTT broker ตลอดเวลา
# try:
#     while True:
#         pass
```

```
# except KeyboardInterrupt:
#     # หยุดการเชื่อมต่อ MQTT client เมื่อมีการกด Ctrl+C
#     client.loop_stop()
#     client.disconnect()


# Input
TD=0.035 #AverageDataDelay[s]
CT = 0.5 # TTC between T2 and other car
LB = 11.5 # Distance after Confict Point
V_cornering = 10 # T2 Max Conering
a = 0.45
ba = 2.86
TM = 100
dt=0.001
RS = 1/dt
t = [0]
Car_Length = [["t2",4.3],["bike",2],["car",5],["truck",7]]# Length of Car for each type
Car_Type = [i[0] for i in Car_Length]
Car_Data = [[40/(20*5/18), (40+Car_Length[0][1])/(20*5/18),(40+LB+Car_Length[0][1])/(20*5/18), 0, 0, [40], [20], [0], [0],
[0]]]
Car_ID = {0}
Car_temp = []
Car_Plot = []
T2S = 0
Decision = True
# Car ID for Multi_Input() (eazy to known its have or not)
# Car Data for Car that went off Confict Area
# Merged Data of Car that went off Confict Area
# T2 State
# T2 Acceleration
# T2 Full Brake Acceleration # Max Time for Simulation #Timeinterval
# Simulation Resolution
# Time Data
Car_Type = [i[0] for i in Car_Length] # Type of car in int format(eazy to index)
# Car_Data_Format = [TTC1[s],TTC2[s],TTC3[s],ID[int],TYPE[int],[S[m]],V[km/h],[Time[s]]]
T2(Acceleration[m/s^2],Time[s],Case[int])
# T2_Data_Format =
[TTC1[s],TTC2[s],TTC3[s],ID[int],TYPE[int],[S[m]],[V[km/h]],[Acceleration[m/s^2]]
,[Time[s]],[Case[int]]]
Car_Data = [[40/(20*5/18), (40+Car_Length[0][1])/(20*5/18),(40+LB+Car_Length[0][1])/(20*5/18), 0, 0, [40], [20], [0], [0],
[0]]]
def Multi_Input(text):
    text = text.replace(" ","").lower().split(";")
    print(text)
    for i in text:
        id, ct, cs, cv = i.split(",")
        if ct not in Car_Type:
            print("Unknown Car Type")
        else:
            id, cs, cv = int(id), float(cs), float(cv)
            if id in Car_ID:
                for j in range(len(Car_Data)):
                    if id == Car_Data[j][3]:
                        Car_temp.append(Car_Data[j])
                        Car_Data.pop(j)
                        Car_ID.discard(id)
                        break
            ttc1 = cs / (cv * 5/18)
```

```
            ct_index = [car[0] for car in Car_Length].index(ct)
            ttc2 = (cs + Car_Length[ct_index][1]) / (cv * 5 / 18)
            ttc3 = (cs + LB + Car_Length[ct_index][1]) / (cv * 5 / 18)
            Car_Data.append([ttc1, ttc2, ttc3, id, ct_index, [cs], cv, [t[-1]]])
            Car_Data.sort()
            Car_ID.add(id)


def TC():# Calulate TTC1/2/3 in Car_Data
    for i in Car_Data:
        if i[3] == 0:
            if T2S == 2:
                i[0] = 2.3*1.2
                i[1] = 2.3*1.2
                i[2] = 2.3*1.2
            else:
                i[0] = i[5][-1]/(i[6][-1]*5/18)
                i[1] = (i[5][-1]+Car_Length[i[4]][1])/(i[6][-1]*5/18)
                i[2] = (i[5][-1]+LB+Car_Length[i[4]][1])/(i[6][-1]*5/18)
        else:
            i[0] = i[5][-1]/(i[6]*5/18)
            i[1] = (i[5][-1]+Car_Length[i[4]][1])/(i[6]*5/18)
            i[2] = (i[5][-1]+LB+Car_Length[i[4]][1])/(i[6]*5/18)
def VL(v):# Set Velocity Limit
    v = max(0,v)
    v = min(20,v)
    return v
def SVC(accel,tpass,nt):# Calulate Position and Velocity in Car_Data
    global T2S
    temp = []
    for i in Car_Data:
        if i[4] == 0:
            if i[6][-1] + (accel*tpass*18/5) < 0:
                tpass2 = i[6][-1]*(5/18)/accel
                i[5].append(i[5][-1] - (i[6][-1]*tpass2*5/18 +(0.5*accel*(tpass2**2))))
            else:
                i[5].append(i[5][-1] - (i[6][-1]*tpass*5/18 + (0.5*accel*(tpass**2))))
            i[6].append(VL(i[6][-1] + (accel*tpass*18/5)))
            i[8].append(nt)
            if i[6][-1] == 0:T2S = 2
        else:# Backup Car Data that left Confict Area
            i[5].append(i[5][-1] - (i[6]*(tpass)*5/18))
            i[7].append(nt)
            if i[5][-1] < -LB:
                temp.append(i)
                Car_temp.append(i)
                Car_ID.discard(i[3])
    if len(temp) != 0:# Remove Car that left Confict Area
        for i in temp:
            Car_Data.remove(i)
    TC()
    Car_Data.sort() #Sort Car_Data by TTC1
# def T2D():# Decision Making System
#     global T2S,ba
#     T2I = [i[4] for i in Car_Data].index(0)
#     print(T2I)
#     #T2 Stopped
#     if Car_Data[[j[4] for j in Car_Data].index(0)][6][-1] > V_cornering:
```

```
#         return[-a,-1]
#     elif T2S == 2:
#             for i in Car_Data:
#                 if i[4] != 0:
#                     if ((i[5][-1] > -LB) and (i[5][-1] <= 0)) or ((i[0] < 2.3*1.2) and (i[0] > 0)):return [0,6]
#             return [2*a,7]
#     #T2 Braking
#     elif T2S == 1:
#         return [-ba,5]
#     else:
# #Accelaration after pass merge point
#         if len(Car_Data) == 1 and Car_Data[[j[4] for j in Car_Data].index(0)][6][-1] < V_cornering -
0.1:
#             return [4*a,8]
#     #Full Brake
#     # T2 hit Car
#         if Car_Data[T2I][0] <= 1.4 and T2I > 0 and Car_Data[T2I][0] <= Car_Data[T2I-1][1] + CT:
#             ba = Car_Data[T2I][5][-1]/1.4
#             T2S = 1
#             return [-ba,3]
#     # Car hit T2
#         elif Car_Data[T2I][0] <= 1.4 and T2I < len(Car_Data)-1 and Car_Data[T2I+1][0] <=
Car_Data[T2I][2] + CT:
#             ba = Car_Data[T2I][5][-1]/1.4
#             T2S = 1
#             return [-ba,4]
#     #Slow Down
#     # T2 hit Car
#         elif T2I > 0 and Car_Data[T2I][0] <= Car_Data[T2I-1][1] + CT:
#             return [-a,1]
# # Car hit T2
#         elif T2I < len(Car_Data)-1 and Car_Data[T2I+1][0] <= Car_Data[T2I][2] + CT:
#             return [-a,2]
#         else:
#             return [0,0]

def T2D():#Dicision Making System
    global T2S,ba
    T2I = [i[4] for i in Car_Data].index(0)
    #T2 Stopped
    if  (Car_Data[T2I][5][-1] > 0) and (Car_Data[T2I][6][-1] > V_cornering):
        return [-a,-1]
    elif Decision == True:
        #T2 Stopped
        if T2S == 2:
            for i in Car_Data:
                if i[4] != 0:
                    if ((i[5][-1] > -LB) and (i[5][-1] <= 0)) or ((i[0] < 2.3*1.2) and (i[0] > 0)):return [0,6]
            return [2*a,7]
        #T2 Full Braking
        elif T2S == 1:
            return [-ba,5]
        else:
        #Accelaration after pass merge point
            if (Car_Data[T2I][5][-1] < 0) and (Car_Data[T2I][6][-1] < V_cornering):
                return [2*a,8]
        #Will Collision
            #T2 hit Car
```

```
            elif T2I > 0 and Car_Data[T2I][0] <= Car_Data[T2I-1][1] + CT:
                c = 1
            #Car hit T2
            elif T2I < len(Car_Data)-1 and Car_Data[T2I+1][0] <= Car_Data[T2I][2] + CT:
                c = 2
            else:
                return [0,0]
        #Will Collision Need Full Braking
            if (c in [1,2]) and (Car_Data[T2I][5][-1] <= 3.4):
                ba = Car_Data[T2I][6][-1]
                T2S = 1
                return [-ba,c+2]
            else:
                return [-a,c]
    else:
        return [0,0]
def SIM():# Simulation
    Multi_Input(input("Input Car Data: "))

    for i in range(0,int(TM*RS)+1):
        t.append(i/RS)

        # if t[-1] in []:
        #     Multi_Input(input("Input:"))
  [A,C] = T2D()
# Collect T2 Data (Acceleration and Case)
        Car_Data[[j[4] for j in Car_Data].index(0)][7].append(A)
        Car_Data[[j[4] for j in Car_Data].index(0)][9].append(C)
        SVC(A,t[-1]-t[-2],i/RS)
# Stop Simulation if T2 left Confict Area
        if Car_Data[[j[4] for j in Car_Data].index(0)][5][-1] < -LB:
            print(t[-1])
            break
def Plot():# Plot T2 Data
    fig1, T2 = plt.subplots(2,2,figsize=(2*5,2*4))
    T2[0,0].plot(Car_Data[[j[4] for j in Car_Data].index(0)][8], Car_Data[[j[4] for j in
Car_Data].index(0)][5])
    T2[0,0].set_xlim(0, t[-1])
    T2[0,0].set_xlabel('Time [s]')
    T2[0,0].set_ylabel('Distance [m]')
    T2[0,0].grid(True)
    T2[0,1].plot(Car_Data[[j[4] for j in Car_Data].index(0)][8], Car_Data[[j[4] for j in
Car_Data].index(0)][6])
    T2[0,1].set_xlim(0, t[-1])
    T2[0,1].set_xlabel('Time [s]')
    T2[0,1].set_ylabel('Velocity [km/h]')
    T2[0,1].grid(True)
    T2[1,0].plot(Car_Data[[j[4] for j in Car_Data].index(0)][8], Car_Data[[j[4] for j in
Car_Data].index(0)][7])
    T2[1,0].set_xlim(0, t[-1])
    T2[1,0].set_xlabel('Time [s]')
    T2[1,0].set_ylabel('Acceleration [m/s^2]')
    T2[1,0].grid(True)
    T2[1,1].plot(Car_Data[[j[4] for j in Car_Data].index(0)][8], Car_Data[[j[4] for j in
Car_Data].index(0)][9])
    T2[1,1].set_xlim(0, t[-1])
    T2[1,1].set_xlabel('Time [s]')
```

```
    T2[1,1].set_ylabel('Case')
    T2[1,1].grid(True)
    fig1.suptitle("T2")
def CTM():# Merge Car Data from Car_Data and Car_Temp into Car_Plot
    CTID = {i[3] for i in Car_temp}
    for id in CTID:
        stemp,ttemp = [],[]
        for i in Car_temp:
            if i[3] == id:
                stemp = stemp + [s for s in i[5]]
                ttemp = ttemp + [s for s in i[7]]
                Car_Plot.append([id,stemp,ttemp])
    return
def MCPlot():# Plot all Car Data
    CTM()
    plt.figure()
    plt.plot(Car_Data[[j[4] for j in Car_Data].index(0)][8], Car_Data[[j[4] for j in
Car_Data].index(0)][5])
    for i in Car_Plot:
        plt.plot(i[2],i[1])
    for j in Car_Data:
        if j[4] != 0:plt.plot(j[7],j[5])
    plt.xlim(0, t[-1])
    plt.xlabel("Time [s]")
    plt.ylabel("Distance [m]")
    plt.title("Car")
    plt.grid()
    plt.show()
    return
SIM()
Plot()
MCPlot()
# Example of Input: 1,car,65,10;2,car,30,10
```

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

ประวัติผู้เขียน

ชื่อ-สกุล	วริทธิ์ ดิษยครินทร์
วัน เดือน ปี เกิด	21 สิงหาคม 2542
สถานที่เกิด	ชลบุรี
วุฒิการศึกษา	จุฬาลงกรณ์มหาวิทยาลัย
ที่อยู่ปัจจุบัน	1230/5 ซอย 12 ถนน ปราจีนอนุสรณ์ ตำบลหน้าเมือง อำเภอเมือง จังหวัดปราจีนบุรี 25000

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY