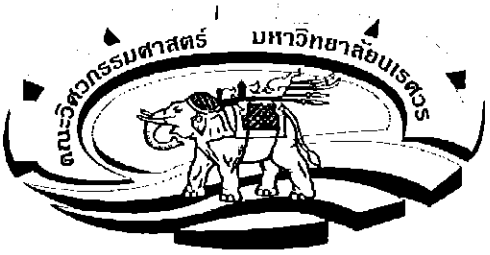
การตรวจจับการขับรถผิดทิศทางการเดินรถด้วยการประมวลผลภาพ

Wrong Traffic Direction Detection using Image Processing

นายธนพัฒน์ อ่อนสี รหัส 54362319

นายนภสินธุ์ กาศสุวรรณ รหัส 54362340

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร
ปีการศึกษา 2557



ใบรับรองปริญญานิพนธ์

หัวข้อโครงงาน	การตรวจจับการขับรถผิดทิศทางการเดินรถด้วยการประมวลผลภาพ			
ผู้ดำเนินโครงงาน	นายธนพัฒน์	อ่อนสี	รหัส	54362319
	นายนภสินธุ์	กาศสุวรรณ	รหัส	54362340
อาจารย์ที่ปรึกษา	อาจารย์รัฐภูมิ	วรานุสาสน์		
อาจารย์ที่ปรึกษาร่วม	ดร.พัฒนาวดี	พัฒนถาบุตร		
สาขาวิชา	วิศวกรรมคอมพิวเตอร์			
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์			
ปีการศึกษา	2557			

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

.....ที่ปรึกษาโครงงาน
(อาจารย์รัฐภูมิ วรานุสาสน์)

.....กรรมการ
(ดร.พัฒนาวดี พัฒนถาบุตร)

.....กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.พนมขวัญ ริยะมงคล)

.....กรรมการ
(รองศาสตราจารย์ ดร.ไพศาล มุณีสว่าง)

ชื่อหัวข้อโครงงาน	การตรวจจับการขับรถผิดทิศทางการเดินรถด้วยการประมวลผลภาพ		
ผู้ดำเนินโครงงาน	นายธนพัฒน์ อ่อนสี	รหัสประจำตัว	54362319
	นายนภสินธุ์ กาศสุวรรณ	รหัสประจำตัว	54362340
ที่ปรึกษาโครงงาน	อาจารย์รัฐภูมิ วรานุสาสน์		
ที่ปรึกษาโครงงานร่วม	ดร.พัฒนาวดี พัฒนถาบุตร		
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2557		

บทคัดย่อ

โครงงานการตรวจจับการขับรถผิดทิศทางการเดินรถด้วยการประมวลผลภาพนี้ จัดทำขึ้นเพื่อตรวจจับผู้ที่กระทำผิดโดยการขับรถผิดทิศทางการเดินรถแบบอัตโนมัติ ซึ่งพิจารณาจากการประมวลผลภาพ โดยจะกำหนดพื้นที่สำหรับใช้ในการตรวจสอบว่ารถที่เข้ามาในพื้นที่ดังกล่าวมีการเคลื่อนที่ไปในทิศทางที่ถูกต้อง ซึ่งการทำงานจะทำโดยการจับวัตถุที่เคลื่อนที่ โดยใช้วิธีการแยกวัตถุที่เคลื่อนที่ออกจากพื้นหลัง ต่อมาจะทำการตีกรอบรอบพื้นที่ของวัตถุที่เคลื่อนที่ และทำการติดตามว่าวัตถุเคลื่อนที่ผ่านเส้นที่ใช้ตรวจสอบใดบ้างในพื้นที่ดังกล่าว สำหรับโปรแกรมนี้สามารถทำงานได้โดยไม่ต้องใช้คนควบคุมซึ่งทำให้เกิดประโยชน์คือ ลดการใช้แรงงานเจ้าหน้าที่ในการตั้งด่านจับผู้กระทำผิด จึงจัดเป็นการอำนวยความสะดวกในการทำงานให้แก่เจ้าหน้าที่ผู้บังคับกฎจราจรได้ดี ซึ่งจากผลการทดลองโปรแกรมที่พัฒนาขึ้นมานี้ให้ความถูกต้องสูงสุดร้อยละ 85 %

Project Title	Wrong Traffic Direction Detection using Image Processing			
Name	Mr.Tanapad	Onsri	ID.	54362319
	Mr.Napasin	Katsuwan	ID.	54362340
Project Advisor	Mr.Rattapoom	Waranusast		
Co-Project Advisor	Dr.Pattanawadee	Pattanathaburt		
Major	Computer Engineering			
Department	Electrical and Computer Engineering			
Academic year	2014			

Abstract

This project was designed to automatically detect wrong-way driving by using images processing. There are designated detecting zones for the direction of the passing vehicles. The system automatically detect and isolate a moving object from background. Then the moving object is marked and tracked automatically. After that, the system check if the object passed the line used for detection in the area. This system requires no manual operator and is completely automatic. This helps reducing strain on Police officers who usually have to operate the checkpoints. After the trail tests, the accuracy of this system is up to 85 %.

กิตติกรรมประกาศ

โครงงานวิศวกรรมคอมพิวเตอร์นี้สำเร็จลุล่วงไปได้ด้วยดี เนื่องจากได้รับความกรุณาจากอาจารย์ที่ปรึกษาทั้งสองท่าน ได้แก่ อาจารย์ รัฐภูมิ วรานุสาสน์ และ ดร. พัฒนาวดี พัฒนถาบุตร ที่คอยแนะนำแนวทาง ให้คำปรึกษา และสละเวลาคอยช่วยเหลือ ในการแก้ปัญหาต่างๆ จนโครงงานนี้สำเร็จลุล่วงไปได้ด้วยดี

ขอขอบพระคุณคณะกรรมการโครงงานทั้งสองท่าน ได้แก่ ผู้ช่วยศาสตร์จารย์ ดร. พนมขวัญ ริยะมงคล และรองศาสตราจารย์ ดร. ไพศาล มุณีสว่าง ที่เป็นผู้ชี้แนะปัญหาของโครงงาน และสิ่งที่ควรปรับปรุงของโครงงาน และช่วยตรวจสอบความถูกต้องของปริญญานิพนธ์ คณะผู้จัดทำจึงขอขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณคณาจารย์คณะวิศวกรรมศาสตร์ ที่ได้คอยสั่งสอนและให้ความรู้มาใช้ในการทำโครงงานแก่คณะผู้จัดทำโครงงาน จนสำเร็จลุล่วงไปได้ด้วยดี

และสุดท้ายขอขอบพระคุณบิดา มารดา และครอบครัว ที่คอยสนับสนุนในทุกด้านและยังเป็นแรงผลักดันในดานการเรียนและการจัดทำโครงงานนี้ได้สำเร็จ

นายธนพัฒน์ อ่อนสี
นายนภสินธุ์ กาศสุวรรณ

สารบัญ

หน้า

สารบัญตาราง

หน้า

สารบัญรูป

หน้า

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

ปัจจุบันมีการใช้พาหนะบนท้องถนนเป็นจำนวนมากไม่ว่าจะเป็นรถยนต์ รถจักรยานยนต์ เนื่องจากความสะดวกจากการใช้งานในการเดินทาง การใช้ถนนเป็นจำนวนมากก็ทำให้เกิดการเร่งรีบประมาท หรือ ความมักง่ายในการใช้รถใช้ถนนจึงเกิดอุบัติเหตุบนท้องถนนอยู่บ่อยครั้ง ทำให้ต้องคำนึงถึงความปลอดภัยของชีวิตในการใช้ยานพาหนะเหล่านี้บนท้องถนนเป็นอย่างมาก

เมื่อต้นปี 2557 สถาบันวิจัยด้านการคมนาคม มหาวิทยาลัยมิชิแกน สหรัฐอเมริกา ร่วมกับข้อมูลจากองค์การอนามัยโลก ได้มีการเปิดเผยอันดับจากการเก็บสถิติของประเทศที่มีคนเสียชีวิตจากอุบัติเหตุบนท้องถนน มากที่สุดในโลก พบว่าประเทศนามิเบียครองแชมป์ เป็นอันดับหนึ่ง ส่วนประเทศไทย ครองอันดับสอง สะท้อนปัญหาอุบัติเหตุบนท้องถนนที่ควรแก้ไขโดย ระบุว่า สถิติการเสียชีวิตจากอุบัติเหตุบนท้องถนนทั่วโลก มีค่าเฉลี่ยอยู่ที่ 18 คน ต่อประชากร 100,000 คนต่อปี แต่สำหรับไทย นามิเบีย และอิหร่าน ซึ่งเป็น 3 อันดับต้น ๆ ที่มีคนเสียชีวิตจากอุบัติเหตุบนท้องถนนมากที่สุดในโลกนั้นมีสถิติมากกว่าค่าเฉลี่ยเกิน 2 เท่า นั่นคือ 45 คน, 44 คน และ 38 คนต่อประชากร 100,000 คนต่อปี [1] ตามลำดับ สาเหตุของการเกิดอุบัติเหตุเหล่านี้อันเนื่องมาจากการทำผิดกฎจราจร เช่น ขับรถเร็วเกินอัตราที่กำหนด ฝ่าไฟแดง แซงในที่ขับขัน ขับรถย้อนศร ขับรถขณะมึนเมาสุรา จึงต้องหาวิธีการลดต้นเหตุของการเกิดอุบัติเหตุที่เกิดขึ้น วิธีหนึ่งคือการบังคับใช้กฎหมาย ใช้แรงงานเจ้าหน้าที่ ในที่นี้ได้สนใจในการลดปัญหาการขับรถผิดทิศทาง โดยใช้เทคโนโลยีเข้ามาช่วยในการทำงาน

ดังนั้นผู้จัดทำจึงได้แนวคิดในการพัฒนาเทคโนโลยี ที่ใช้ซอฟแวร์คอมพิวเตอร์ทำงานร่วมกับกล้องวีดีโอในการตรวจจับการขับรถผิดทิศทางการเดินรถ และนำเทคโนโลยีนี้มาใช้จริงบนท้องถนนที่มีผู้กระทำผิดกฎจราจร ที่เป็นสาเหตุของอุบัติเหตุที่เกิดขึ้น โดยการทำงานโดยใช้การประมวลผลภาพ (image processing)

1.2 วัตถุประสงค์ของโครงงาน

1.2.1 เพื่อศึกษาเทคโนโลยีการประมวลผลภาพในการตรวจจับผู้กระทำผิดกฎหมาย

1.2.2 เพื่อพัฒนาระบบตรวจจับการขับรถผิดทิศทางการเดินรถด้วยการประมวลผลภาพ

1.3 ขอบเขตของโครงงาน

1.3.1 กล้องวีดีโอต้องมีความละเอียดขั้นต่ำ 500x300 และจะต้องไม่โฟกัสแบบอัตโนมัติ

1.3.2 สภาพแวดล้อมของการทำงานของการตรวจจับภาพ มีแสงแดดเพียงพอ ไม่มีหมอก หรือฝนตกและต้องเป็นเวลากลางวัน

1.3.3 จุดที่ตรวจจับต้องติดตั้งกล้องอยู่ในมุมสูง

1.3.4 ยานพาหนะที่ตรวจจับเป็นรถยนต์และรถจักรยานยนต์

1.3.5 จะต้องไม่มียานพาหนะที่จอดนิ่งหรือวัตถุชนิดอื่นเข้ามาขวางภายในกรอบที่ใช้ในการตรวจสอบ

1.4 ขั้นตอนการดำเนินงานของโครงงาน

1.4.1 ศึกษาเกี่ยวกับใช้งาน OpenCV Library

1.4.2 ศึกษาวิธีการตรวจจับภาพ โดย OpenCV Library

1.4.3 ค้นคว้าข้อมูลอัตราการเกิดสถิติจากการขับรถผิดกฎจราจร

1.4.4 ศึกษาเกี่ยวกับกฎหมายจราจร (การกระทำผิดกฎจราจรโดยการขับรถย้อนศร)

1.4.5 ออกแบบโปรแกรมและเขียนโปรแกรมการจับรถที่ขับย้อนศร

1.4.6 ทดสอบการทำงาน

1.4.7 แก้ไขข้อผิดพลาด

1.4.8 สรุปผล

1.5 แผนการดำเนินงานของโครงงาน

ตารางที่ 1.1 แผนการดำเนินโครงงาน

กิจกรรม	ปี 2557					ปี 2558				
	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค	เม.ย.	พ.ค.
1. ศึกษาเกี่ยวกับการใช้งาน OpenCV Library										
2. ศึกษาวิธีการตรวจจับภาพโดยใช้ OpenCV Library										
3. ค้นคว้าข้อมูลอัตราการเกิดอุบัติเหตุจากการขับรถผิดกฎจราจร										
4. ศึกษาเกี่ยวกับกฎหมายจราจร (การกระทำผิดกฎจราจรโดยการขับรถย้อนศร)										
5. ออกแบบโปรแกรมและเขียนโปรแกรมการจับรถที่ขับย้อนศร										
6. ทดสอบการทำงาน										
7. แก้ไขข้อผิดพลาด										
8. สรุปผล										

1.6 ผลที่คาดว่าจะได้รับ

1.6.1 ได้ความรู้เกี่ยวกับการประมวลผลภาพ

1.6.2 ได้ระบบตรวจจับการขับรถผิดทิศทางการเดินรถ

1.7 งบประมาณของโครงงาน

1.7.1 ค่าดำเนินโครงงาน		500	บาท
1.7.2 ค่าอุปกรณ์ทดสอบการทำงาน		1,000	บาท
1.7.3 ค่าถ่ายเอกสารและเข้ารูปเล่ม		500	บาท
	รวมเป็นเงินทั้งสิ้น	2,000	บาท

หมายเหตุ ขออนุมัติถัวเฉลี่ยทุกรายการ

บทที่ 2
หลักการและทฤษฎีที่เกี่ยวข้อง

โครงงานนี้ได้อาศัยเทคโนโลยีการประมวลผลภาพเป็นสิ่งสำคัญในการทดลองโครงงาน โดยในบทนี้จะเป็นการกล่าวถึงทฤษฎีต่างๆที่นำมาใช้ในการพัฒนาระบบการทำงานของโครงงานนี้ขึ้น โดยมีรายละเอียดของทฤษฎีต่างๆ ดังนี้

2.1 หลักการประมวลผลภาพ (Image Processing) [2]

การประมวลผลภาพ หมายถึง การนำรูปภาพต้นฉบับ (Input) มาคำนวณด้วยเทคนิคและอัลกอริธึมต่างๆ เพื่อให้ได้รูปภาพผลลัพท์ (Output) ดังแสดงในรูปที่ 2.1 ที่อยู่ในรูปแบบดิจิตอล (ภาพดิจิทัล) ตามที่เราต้องการ ซึ่งลักษณะของภาพจะทีทั้งภาพนิ่งและภาพเคลื่อนไหว (video) โดยภาพเคลื่อนไหวจะเป็นชุดภาพนิ่ง หรือ เฟรม ซึ่งจะเป็นภาพหลายๆภาพต่อกัน

รูปที่ 2.1 หลักการของการประมวลผลภาพเบื้องต้น

ขั้นตอนต่างๆ ที่สำคัญในการทำการประมวลผลภาพจะยกตัวอย่าง คือ การทำให้ภาพคมชัดมากขึ้น การกำจัดสัญญาณรบกวนออกจากภาพ การแบ่งส่วนของวัตถุที่เราสนใจออกมาจากภาพ เพื่อนำภาพวัตถุที่ได้ไปวิเคราะห์หาข้อมูลเชิงปริมาณ และทิศทางการเคลื่อนที่ของวัตถุในภาพ จากนั้นเราจึงนำข้อมูลเชิงปริมาณเหล่านี้ไปวิเคราะห์และสร้างเป็นระบบ เพื่อใช้ประโยชน์ในงานด้านต่างๆ ตัวอย่างเช่น การตรวจจับการขับรถผิดทิศทางการเดินรถด้วยการประมวลผลภาพ จะเห็นได้ว่าระบบนี้จำเป็นต้องมีการประมวลผลภาพจำนวนมาก และเป็นกระบวนการที่ต้องทำซ้ำๆ ในรูปแบบเดิมเป็นส่วนใหญ่ ซึ่งในงานลักษณะเหล่านี้ หากมนุษย์วิเคราะห์เอง จะต้องใช้เวลามากและใช้แรงงานสูง ซึ่งมนุษย์วิเคราะห์ภาพเองอาจเกิดอาการล้าส่งผลให้เกิดความผิดพลาดได้ดังนั้น คอมพิวเตอร์จึงมีบทบาทสำคัญในการทำหน้าที่เหล่านี้แทนมนุษย์ เพราะคอมพิวเตอร์นั้นมีความสามารถในการคำนวณและประมวลผลข้อมูลจำนวนมหาศาลได้ในเวลาอันสั้น จึงมีประโยชน์อย่างมากในการทำงาน

2.2 ไลบรารีโอเพนซีวี [3]

OpenCV ย่อมาจาก Open source Computer Vision เป็นไลบรารีที่พัฒนาขึ้นจากภาษา C/C++ ใช้งานเพื่อพัฒนาโปรแกรมที่เกี่ยวข้องกับ Image Processing และ Computer Vision โดยจะสามารถใช้งานได้ในระบบปฏิบัติการ Window และระบบปฏิบัติการ Linux การใช้งาน OpenCV จะนำมาใช้งานร่วมกับ Visual Studio C++ โดยอันดับแรกเราจะต้องตั้งค่า Visual Studio C++ เพื่อระบุตำแหน่งของ Library ของ OpenCV ตำแหน่งของไฟล์ที่ต้องใช้ในโปรแกรม และตำแหน่งของ Source File ให้ตัวโปรแกรมทราบและดึงมาใช้ได้ โดย OpenCV Library จะมีสัญลักษณ์ดังรูป 2.2

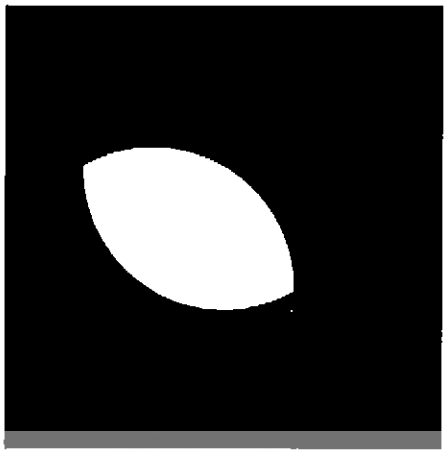
รูปที่ 2.2 สัญลักษณ์ของไลบรารีโอเพนซีวี

2.3 ระบบสี (Color Model)

เป็นการอ้างถึงสีต่างๆ สำหรับคอมพิวเตอร์เพราะจะมีลักษณะหลายประการที่เป็นลักษณะเฉพาะระบบสีของคอมพิวเตอร์ จะเกี่ยวข้องกับการแสดงผลแสงที่แสดงบนจอคอมพิวเตอร์ ถ้าไม่มีการแสดงผลสีใดเลย บนจอภาพจะแสดงเป็น "สีดำ" แต่ถ้าหากทุกสีแสดงพร้อมกันจะแสดงเป็น "สีขาว" และสีอื่นๆ จะเกิดจากการแสดงสีหลายๆ สีแต่มีค่าที่แตกต่างกัน

2.3.1 ระบบสี RGB [4]

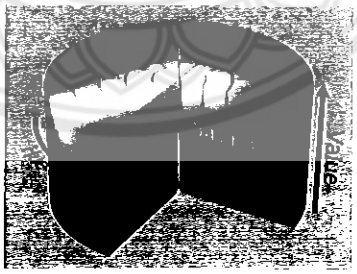
เป็นสีที่เกิดจากการรวมกันของแม่สี 3 สี ดังรูปที่ 2.3 คือ สีแดง (Red) สีเขียว (Green) และสีน้ำเงิน (Blue) แบบจำลองนี้มีที่มาจากหลักการทำงานของจอภาพโทรทัศน์และจอภาพคอมพิวเตอร์ที่ประกอบด้วยหลอดสี 3 หลอดสีดังกล่าว วิธีการกำหนดสี จะใช้ความเข้มของแม่สีทั้ง 3 มาผสมกัน ซึ่งในสีหนึ่งๆ จะมีค่า 256 (ตั้งแต่ค่า 0-255) ดังนั้นเมื่อนำทั้งหมดมาผสมกันก็จะได้สีประมาณ 16,777,216 ล้านสี (256x256x256)



รูปที่ 2.3 ระบบสี RGB

2.3.2 ระบบสี HSV [5]

การพิจารณาสีโดยใช้ค่า Hue Saturation และ Value โดยที่ค่าของ Hue คือค่าสีของสีหลัก (สีแดง สีเขียว สีน้ำเงิน) จะมีค่าตั้งแต่ 0 – 255 โดยถ้า Hue มีค่าเป็น 0 ก็จะแทนเป็นสีแดงและเมื่อเพิ่มค่า เรื่อยๆ Hue ก็จะเปลี่ยนไปตามสเปกตรัมของสี จนถึง 256 ค่าของ Hue จะกลับมาเป็นสีแดงอีกครั้ง และ Saturation หรือค่าความบริสุทธิ์ของสี ถ้าหากค่า Saturation มีค่าเท่ากับ 0 สีที่ได้จะไม่มี Hue จะกลายเป็นสีขาวล้วน และถ้าค่า Saturation เป็น 255 จะหมายความว่าสีนั้นไม่มีสีขาวผสมอยู่เลย และสุดท้ายคือค่า Value หรือค่าความสว่างของสี สามารถวัดได้จากค่าความเข้มของความสว่างของแต่ละสี จะมีตัวอย่างดังรูปที่ 2.4



รูปที่ 2.4 ระบบสี HSV

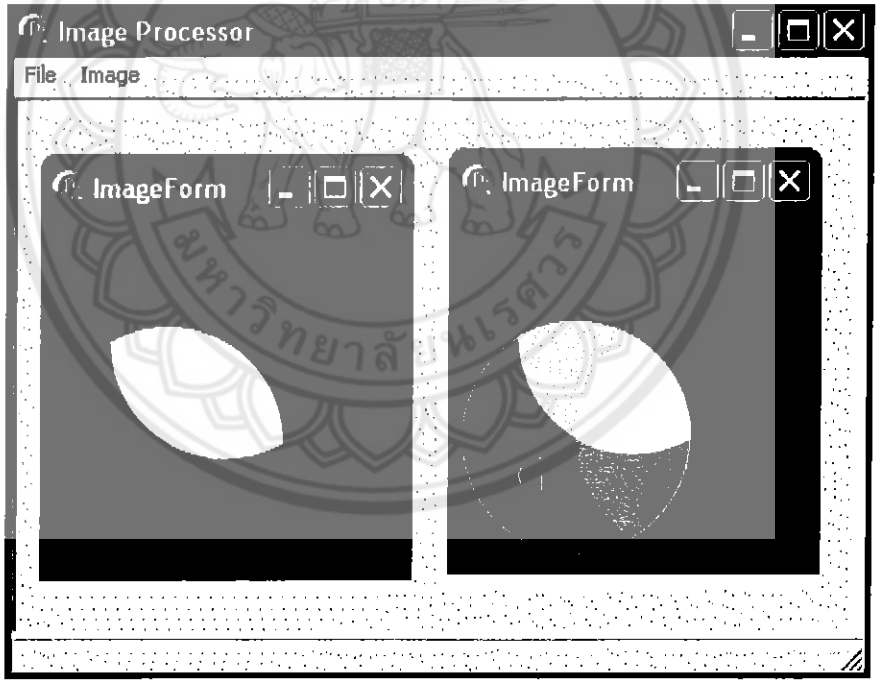
2.4 ภาพระดับเทา (Grayscale Image) [6]

เป็นภาพระดับสีเทา จะมีความเข้มของสีเทา คือ 0-255 (8bit) รูปภาพระดับสีเทาเกิดจากการแปลงภาพสี RGB มาเป็นภาพระดับสีเทา (Grayscale) โดยจะหาค่าได้ดังสมการ (1) และจะได้ผลลัพธ์เป็นภาพระดับเทาดังรูปที่ 2.5

$$Gray = 0.299xR + 0.587xG + 0.114xB \quad (1)$$

โดย

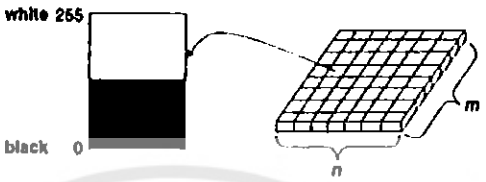
Gray = ค่าความเข้มสีเทา
R = ค่าความเข้มสีแดง
G = ค่าความเข้มสีเขียว
B = ค่าความเข้มสีน้ำเงิน



รูปที่ 2.5 ตัวอย่าง ภาพระดับเทา

2.5 ภาพขาวดำ (Binary Image) [7]

เป็นภาพที่มี 2 สถานะนั่นคือ 0 และ 1 หมายความว่า ใน 1 พิเซลจะเป็นได้เพียงค่าใดค่าหนึ่ง ไม่เป็น 0 ก็จะเป็น 1 โดยที่ 0 นั่นคือพิเซลแสดงสีดำ และ 1 คือพิเซลแสดงสีขาว โดยจะมีตัวอย่างผลลัพธ์ดังรูปที่ 2.6



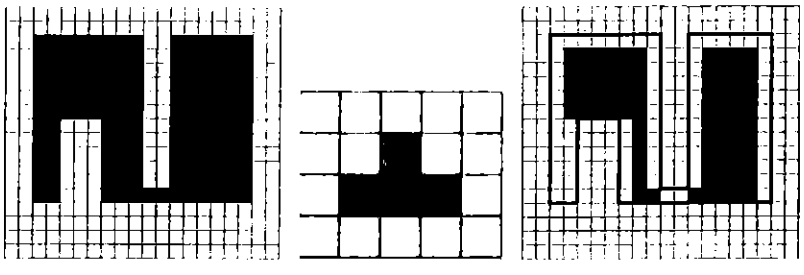
รูปที่ 2.6 ภาพขาวดำ

2.6 การกำจัดสัญญาณรบกวน (Morphological Image Processing) [8]

เป็นการตัดต่อหรือแต่งเติมส่วนขอบของภาพ โครงสร้างของภาพ โดยจะอ้างอิงทฤษฎีเซ็ต โดยเซ็ตใน Morphological จะแทนรูปร่างหรือรูปทรงของวัตถุในภาพ สามารถนำมาใช้ในการกำจัดสัญญาณรบกวน ขยายพื้นที่วัตถุ และกำจัดส่วนเกินของวัตถุออกไปได้

2.6.1 การย่อขนาดพื้นที่ (Erosion)

คือ การลดขนาดพิกเซลของภาพ โดยการใช้ค่า Structuring Element เทียบไปบนแต่ละค่าของพิกเซลภาพ โดยจะทำการเทียบจากตำแหน่งบนซ้ายไปยังตำแหน่งล่างขวา ซึ่งจะทำให้เกิดการเปลี่ยนค่าของพิกเซลที่มีค่า 1 ให้มีค่าเป็น 0 เมื่อค่าของพิกเซลใดๆ บนพิเซลหนึ่งบน Structuring Element มีค่าตรงกับค่าของพิกเซลภาพ และจะมีค่าคงเดิม เมื่อทุกค่าของ Structuring Element มีค่าตรงกับทุกค่าของพิกเซลภาพ ดังรูปที่ 2.7



(ก) (ข) (ค)

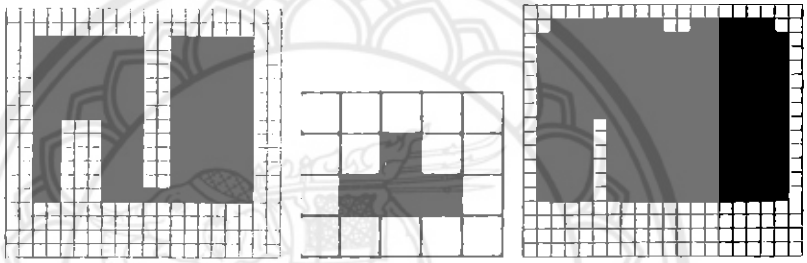
รูปที่ 2.7 แสดงรูปการย่อขนาดพื้นที่

(ก) แสดงรูปภาพเริ่มต้น (Original Image)

(ข) แสดงรูปภาพย่อย (Structuring Image)
(ค) แสดงผลลัพธ์ของการย่อขนาดพื้นที่ (Erosion)

2.6.2 การขยายขนาดพื้นที่ (Dilation)

คือ การขยายขนาดพิกเซลของภาพ โดยการใช้ค่า Structuring Element เทียบไปบนแต่ละค่าของพิกเซลภาพ โดยจะทำการเทียบจากตำแหน่งบนซ้ายไปยังตำแหน่งล่างขวา ซึ่งจะทำให้เกิดการเปลี่ยนค่าของพิกเซลที่มีค่า 0 ให้มีค่าเป็น 1 เมื่อค่าของพิกเซลใดๆ บนพิเซลหนึ่งบน Structuring Element มีค่าตรงกับค่าของพิกเซลภาพ และจะมีค่าคงเดิม เมื่อทุกค่าของ Structuring Element มีค่าตรงกับทุกค่าของพิกเซลภาพ ดังรูปที่ 2.8

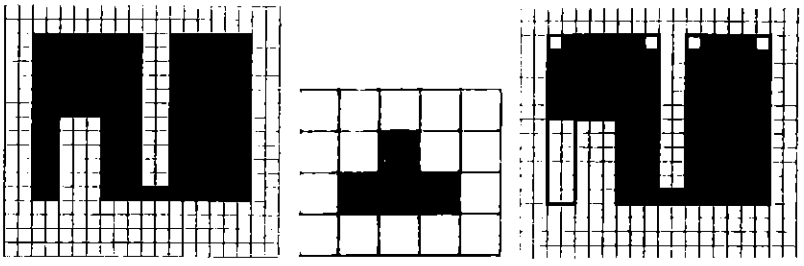


(ก) (ข) (ค)

รูปที่ 2.8 แสดงรูปการขยายขนาดพื้นที่
(ก) แสดงรูปภาพเริ่มต้น (Original Image)
(ข) แสดงรูปภาพย่อย (Structuring Image)
(ค) แสดงผลลัพธ์ของการขยายขนาดพื้นที่ (Dilation)

2.6.3 การเปิดพื้นที่ว่างภายในภาพ (Opening)

คือ การกำจัดรายละเอียดขนาดเล็กของภาพ จะทำให้พิกเซลของภาพถูกเปิดกว้างมากขึ้น วิธีการของ Opening คือการทำ Erosion ก่อน แล้วจึงทำการ Dilation ต่อ ดังรูปที่ 2.9



(ก) (ข) (ค)

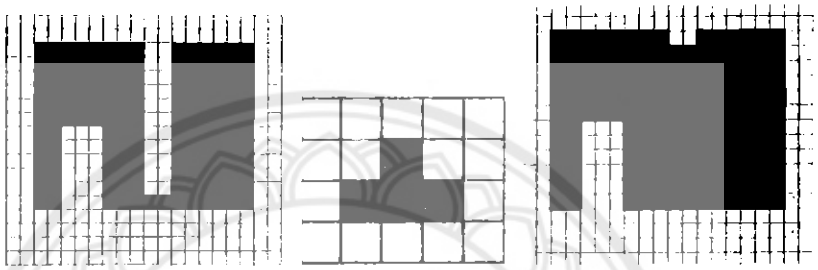
รูปที่ 2.9 แสดงรูปการเปิดพื้นที่ว่างภายในภาพ
(ก) แสดงรูปภาพเริ่มต้น (Original Image)

(ข) แสดงรูปภาพย่อย (Structuring Image)

(ค) แสดงผลลัพธ์ของการเปิดพื้นที่ว่างภายในภาพ (Opening)

2.6.4 การปิดพื้นที่ว่างภายในภาพ (Closing)

คือ การทำให้ภาพมีการเชื่อมต่อกันมากขึ้น โดยการทำ Closing จะมีวิธีการทำ Dilation ก่อน แล้วจึงทำการ Erosion ต่อ ดังรูปที่ 2.10



(ก) (ข) (ค)

รูปที่ 2.10 แสดงรูปการปิดพื้นที่ว่างภายในภาพ

(ก) แสดงรูปภาพเริ่มต้น (Original Image)

(ข) แสดงรูปภาพย่อย (Structuring Image)

(ค) แสดงผลลัพธ์ของการปิดพื้นที่ว่างภายในภาพ (Closing)

2.7 การแปลงภาพสีเป็นภาพขาวดำ (Threshold) [9]

การแปลงภาพระดับสีเทา ให้กลายเป็นภาพขาวดำ โดยจะให้จุดในภาพมีขนาด 1 บิต นั่นคือ 0 (สีดำ) และ 1 (สีขาว) และกำหนดค่า Threshold เป็นค่าที่ใช้เปรียบเทียบ หากมีความเข้มมากกว่าก็จะปรับให้หลายเป็นสีดำ แต่ถ้าหากสว่างกว่าก็จะปรับให้กลายเป็นสีขาว โดยสมมุติค่า threshold มาเป็นค่า 128 ถ้าค่าพิกเซลตรงไหนมีค่ามากกว่า 128 จะถูกปรับให้เป็น 255 แต่ถ้าพิกเซลตรงไหนน้อยกว่า จะถูกปรับให้เป็น 0 โดยจะมีสมการอ้างอิงตามสมการ (2) และ (3) สุดท้ายแล้วจะได้ผลลัพธ์เป็นภาพขาวดำ ดังรูปที่ 2.11

$$g(x,y) = 0 \quad \text{if } f(x,y) < \text{threshold value} \tag{2}$$

$$g(x,y) = 255 \quad \text{if } f(x,y) \geq \text{threshold value} \tag{3}$$

เมื่อ f(x,y) คือ ตำแหน่งพิกเซลของภาพต้นฉบับ

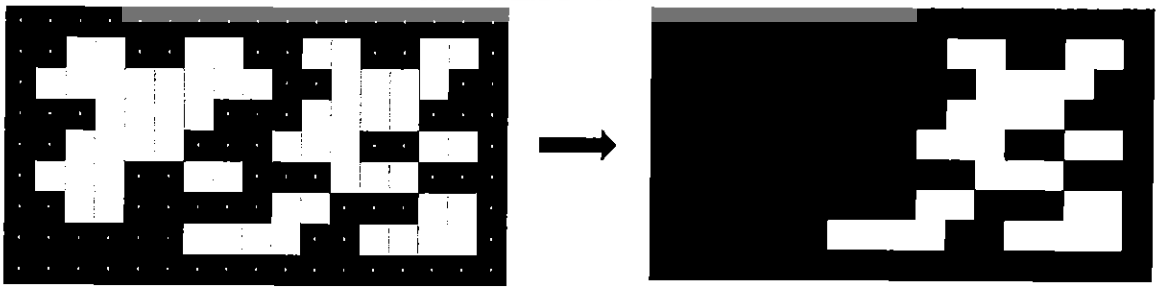
g(x.y) คือ ตำแหน่งพิกเซลของภาพผลลัพธ์

รูปที่ 2.11 แสดงการแปลงภาพให้เป็นขาวดำ

การทำภาพขาวดำโดยใช้เทคนิคเทรชโฮลเพื่อให้ได้ผลลัพธ์ที่เหมาะสม สิ่งที่สำคัญที่สุดคือ ค่าเทรชโฮล เนื่องจากถ้าเลือกค่าเทรชโฮลที่ไม่เหมาะสม ภาพที่ได้อาจจะมืดเกินไป หรือสว่างมากเกินไป หรืออาจมีสิ่งรบกวนเกิดขึ้น ส่งผลทำให้ภาพที่ได้ไม่เป็นไปตามที่ต้องการ ดังนั้นการแก้ปัญหาของการสร้างภาพขาวดำ คือวิธีการกำหนดค่าเทรชโฮลที่เหมาะสมสำหรับแต่ละภาพที่จะนำมาทำการสร้างภาพขาวดำ

2.8 การกำหนดหมายเลขให้ส่วนที่เชื่อมกัน (Connected Component Labeling) [10]

ใช้ในการวิเคราะห์ส่วนประกอบหรือบริเวณที่มีการเชื่อมติดกันและใช้ในการแยกบริเวณต่างๆ ออกจากกันโดยวิธีการกำหนดหมายเลขให้ส่วนที่เชื่อมกัน (Connected Component Labeling) เป็นวิธีที่ใช้ในคอมพิวเตอร์วิทัศน์ (Computer Vision) เพื่อตรวจจับบริเวณที่เชื่อมติดกันของภาพขาวดำหรือภาพสี แต่โดยทั่วไปใช้ในภาพขาวดำที่ผ่านการประมวลผลมาแล้ว เมื่อนำมาหาส่วนที่เชื่อมกัน จะได้พื้นที่แบ่งบริเวณต่างๆ ออกจากกัน ดังรูปที่ 2.12



รูปที่ 2.12 การกำหนดหมายเลขให้ส่วนที่เชื่อมกัน

2.9 พื้นฐานของการติดตาม (Basic of Tracking)

การทำงานกับไฟล์ภาพวิดีโอ จะมีความแตกต่างจากไฟล์ภาพนิ่ง โดยที่เราจะเลือกจับภาพเฉพาะวัตถุใดวัตถุหนึ่งหรือจับภาพตามวัตถุเคลื่อนที่ที่เราต้องการให้อยู่ในกรอบการติดตามภาพนั้น เราจำเป็นจะต้องใช้ความเข้าใจเรื่อง "การเคลื่อนที่ของวัตถุ"

การจำแนกลักษณะ มีความสำคัญในการค้นหาวัตถุที่เราสนใจจากภาพหนึ่งเฟรม ในแต่ละเฟรมที่ต่อเนื่องกันของไฟล์วิดีโอ เทคนิคในเรื่องโมเมนต์หรือฮิสโตแกรมของสี จะช่วยให้สามารถจำแนกวัตถุที่ต้องการได้ สิ่งที่จะทำการติดตามถ้าเรายังจำแนกลักษณะไม่ได้จะเป็นปัญหาเกี่ยวข้องและสัมพันธ์กัน การติดตามที่จำแนกลักษณะไม่ได้จะใช้ก็ต่อเมื่อต้องการจะพิจารณาการเคลื่อนที่ของวัตถุ ว่าวัตถุใดเคลื่อนที่ได้น่าสนใจต่อการติดตาม เทคนิคการติดตามโดยที่ไม่มีการจำแนกลักษณะมักจะเกี่ยวข้องกับการติดตามวัตถุที่เมื่อสังเกตด้วยตาแล้วจะเป็นวัตถุที่ดูมีนัยสำคัญมากกว่าวัตถุอื่น

2.9.1 การแยกวัตถุที่มีการเคลื่อนที่ออกจากพื้นหลัง (Background Subtraction) [11]

เป็นการลบพื้นหลัง ด้วยการนำภาพสองเฟรม ณ บริเวณเดียวในเวลาที่ต่างกัน มาทำการลบกัน หากมีการเคลื่อนที่เข้ามาของวัตถุในภาพเฟรมที่สอง เมื่อนำภาพทั้งสองเฟรมมาลบกัน พื้นหลังจะลบหักล้างกันระหว่างภาพสองเฟรม และมีส่วนผลต่างระหว่างภาพทั้งสองเฟรม ซึ่งผลต่างนี่เองคือการเคลื่อนไหวของวัตถุที่เข้ามายังภาพเฟรมที่สอง เป็นไปดังสมการ (4)

$$O = I - B \tag{4}$$

เมื่อ O คือ Object
I คือ ภาพของเฟรมหนึ่งๆ
B คือ ภาพพื้นหลัง

ในการหาความแตกต่างของภาพ จะกำหนดให้ พิกเซลของภาพสีเทาของไบนารี แสดงพื้นที่รูปภาพกับการเคลื่อนไหว กล่าวคือ พื้นที่ซึ่งมีความแตกต่างอย่างมากระหว่างระดับสีเทาในรูปภาพเป็นลำดับ และได้สมการ (5)

$$\begin{aligned} d(i,j) &= 0 \text{ if } | f1(i,j) - f2(i,j) | e \\ &= 1 \text{ otherwise} \end{aligned} \tag{5}$$

โดย e คือตัวเลขค่าน้อย การตรวจจับภาพเคลื่อนไหวโดยใช้วิธีการหาความแตกต่างของภาพอาศัยการเคลื่อนที่ของวัตถุใดๆที่แตกต่างจากพื้นหลังสามารถแยกวัตถุได้ และเป็นสองรูปภาพตามลำดับที่แยกกันโดยช่วงเวลาของรูปภาพตามความแตกต่างระหว่างรูปภาพ

2.9.1.1 การแยกวัตถุที่เคลื่อนที่ด้วยวิธี Mixture of Gaussians (MOG) [12]

เทคนิคนี้เป็นการสร้างโมเดลของภาพพื้นหลัง ที่สามารถปรับค่าพื้นหลังให้เป็นปัจจุบันได้ตลอดเวลาเพื่อให้เหมือนกับพื้นหลังของทุกๆ เฟรมปัจจุบัน ที่มีวัตถุเคลื่อนที่เกิดขึ้นภายในเฟรมนั้นๆ เนื่องจากสภาพแวดล้อมจริงจะมีความไม่แน่นอนของสภาพแวดล้อมเกิดขึ้น เช่น สภาพภูมิอากาศ ความเข้มแสง เงาจากวัตถุต่างๆ ซึ่งอาจจะส่งผลต่อค่าของจุดสี ณ ตำแหน่งเดิมของในแต่ละเฟรมภาพที่เปลี่ยนแปลงไปตามสภาพแวดล้อมได้โดยแต่ละจุดสีของเฟรมภาพจะถูกจำแนกออกเป็นจุดสีของภาพพื้นหลัง หรือจุดสีของวัตถุด้วยวิธีเกาส์เซียน (Gaussians Distribution) ที่มีมากกว่า 1 ดิสทริบิวชัน เพื่อเป็นประโยชน์ในการวิเคราะห์หาพื้นหลังของภาพที่มีความซับซ้อนมากยิ่งขึ้น

การสร้างโมเดลการหาพื้นหลังของภาพที่มีหลายๆค่านั้น เพื่อแก้ปัญหาเรื่องของจุดสีที่อยู่บนภาพพื้นหลังที่มีหลายค่า ทำให้การแบ่งประเภทของจุดสีมีความใกล้เคียงตามสภาพแวดล้อมจริงมากยิ่งขึ้น ซึ่งความน่าจะเป็นของการพิจารณาค่าของจุดสี พิจารณาได้จากสมการที่ (6) และในสมการที่ (7) เป็นค่าของ Gaussian Probability Density Function

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t}\, \eta(x_t - \mu_{i,t}\sum_{i,t}) \qquad (6)$$

โดยที่ K คือ จำนวน Gaussians Distribution

$\omega_{i,t}$ คือ ค่าน้ำหนักของ Gaussians ตัวที่ i ณ เวลา t

$\mu_{i,t}$ คือ ค่าเฉลี่ยของ Gaussian ตัวที่ i ณ เวลา t

$\sum_{i,t}$ คือ ค่าความแปรปรวนร่วมของ Gaussian ตัวที่ i ณ เวลา t

η คือ Gaussian Probability Density Function

x_t คือ จุดสี ณ เวลา t

$$\text{ซึ่ง } \eta(X_t, \mu, \sum) = \frac{1}{(2\pi)^{\frac{n}{2}}|\sum|^{\frac{1}{2}}} e^{\frac{-1}{2}(X_t - \mu_t)^T \sum^{-1}(X_t - \mu_t)} \qquad (7)$$

โดยที่ $\sum_{k,t} = \sigma_k^2 I$

$$\omega_{k,t} = (1-\alpha)\omega_{k-1} + \alpha(M_{k,t})$$

$$\mu_t = (1-\rho)\mu_{t-1} + \rho X_t$$

$$\sigma_t^2 = (1-\rho)\sigma_{t-1}^2 + \rho(X_t - \mu_t)^T(X_t - \mu_t)$$

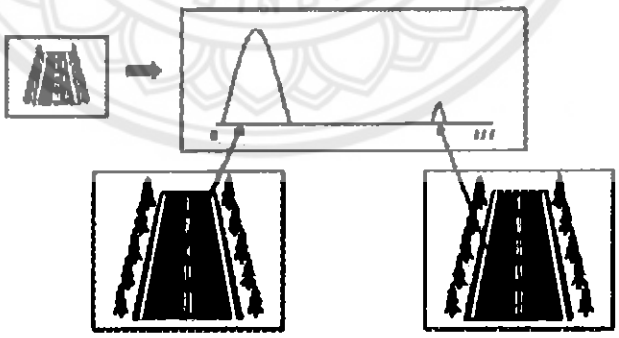
โดยที่ $\rho = \alpha\eta(X_t|\mu_k, \sigma_k)$

เกณฑ์ในการพิจารณาแยกประเภทจุดสีว่าเป็นจุดสีของภาพพื้นหลังหรือจุดสีของวัตถุที่เคลื่อนที่หาได้จากเมื่อพิจารณาจุดสีค่าแอมพลิจูดสูงสุดของทุกๆ ดิสทริบิวชันจะมีค่าเป็น ω_i และมีค่าส่วนเบี่ยงเบนมาตรฐานเป็น σ_i โดยที่

$$\sum_{i=1}^{B} \omega_i > T \tag{8}$$

เมื่อ T คือ ค่าเทรสโฮลด์

ถ้าแอมพลิจูดของดิสทริบิวชันมีค่ามากกว่าเทรสโฮลด์แสดงว่าดิสทริบิวชันนั้นจะเป็นส่วนของพื้นหลังและจากนั้นจะทำการปรับค่าพารามิเตอร์ $(\omega_{i,t}, \mu_{i,t}, \sigma_{i,t})$ ให้เป็นปัจจุบัน แต่ถ้านอกเหนือจากนี้ดิสทริบิวชันนั้นก็คือส่วนของวัตถุหรือรถที่เคลื่อนที่ในเฟรมภาพ ดังรูปที่ 2.13 และผลที่ได้จากการแยกวัตถุเคลื่อนที่ออกจากภาพพื้นหลังเป็นดังรูปที่ 2.14



รูปที่ 2.13 ภาพ Gaussian Distribution

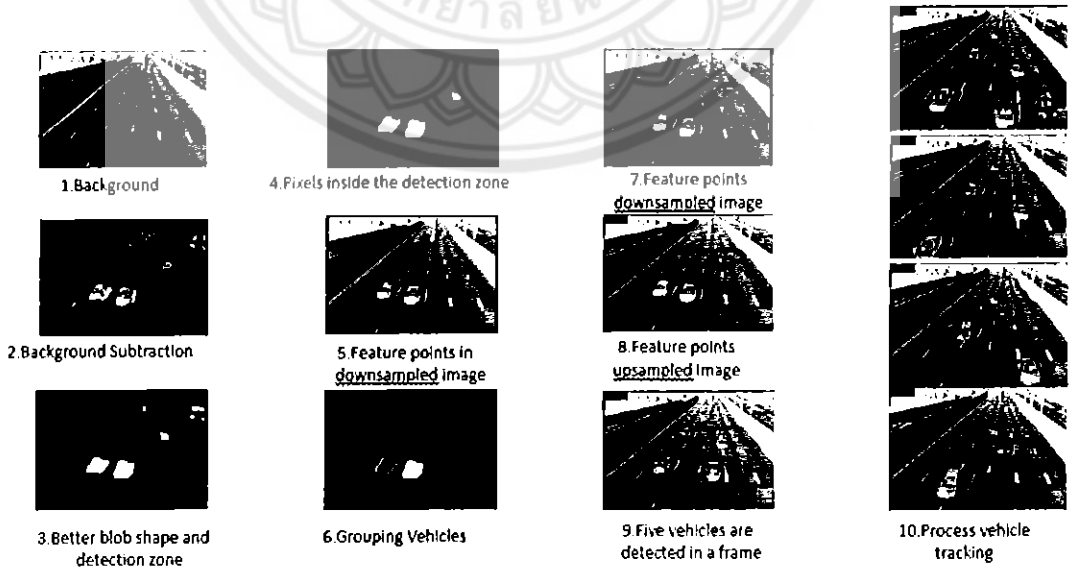
(ก) (ข) (ค)

รูปที่ 2.14 ภาพแสดงการแยกวัตถุเคลื่อนที่ออกจากภาพพื้นหลัง
(ก) ภาพต้นฉบับ
(ข) ภาพพื้นหลัง
(ค) ภาพวัตถุที่เคลื่อนที่โดยใช้เกาส์เซียน

2.10 งานวิจัยที่เกี่ยวข้อง

2.10.1 Vision based Vehicle Tracking using a high angle camera [13]

ผลงานของ Raúl Ignacio Ramos García และ Dule Shu เป็นการกล่าวถึงพื้นฐานการติดตามยานพาหนะโดยใช้กล้องมุมสูง โดยอันดับแรกจะมีการนำภาพถนนที่ได้จากกล้องมุมสูงมาใช้เป็นภาพต้นฉบับและในขั้นตอนต่อมาก็ทำการแยกวัตถุที่เคลื่อนที่ออกจากพื้นหลังโดยวิธีการ Background Subtraction จากนั้นจะทำการกำหนดพื้นที่ที่สนใจที่จะใช้ในการตรวจจับและทำการแบ่งกลุ่มของยานพาหนะ จากนั้นจะได้ผลลัพท์คือยานพาหนะที่ถูกติดตาม จากการเคลื่อนที่ผ่านพื้นที่ทำการตรวจจับยานพาหนะ โดยขั้นตอนดังกล่าว จะเป็นไปตามขั้นตอนดังรูปที่ 2.15



รูปที่ 2.15 แสดงขั้นตอนการติดตามยานพาหนะด้วยวิธี Background Subtraction

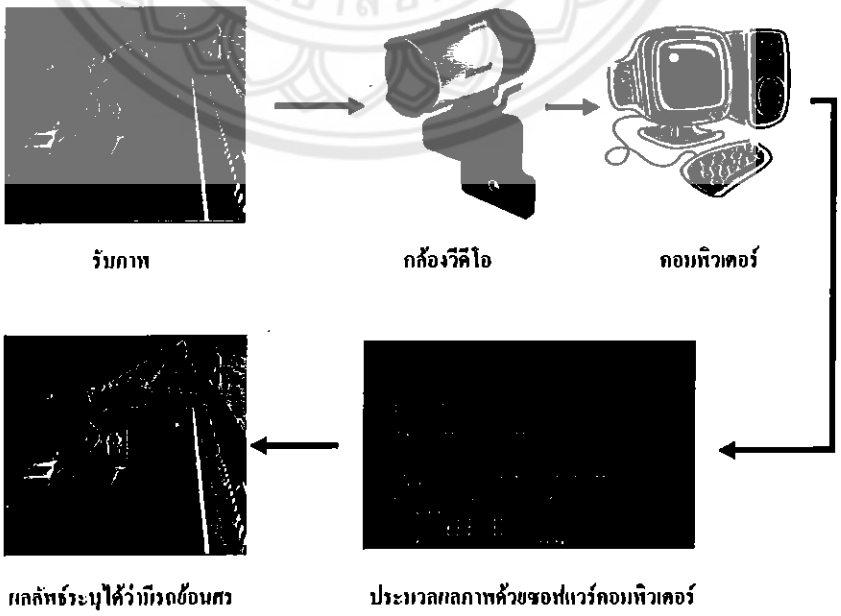
บทที่ 3
ขั้นตอนการดำเนินงาน

ในการสร้างระบบการตรวจจับการขับรถผิดทิศทางการเดินรถ (ย้อนศร) นั้นจะต้องออกแบบการทำงานและทำการศึกษาค้นคว้าข้อมูลและทฤษฎีที่เกี่ยวข้อง ทำให้สามารถเข้าใจหลักการที่จะนำมาทำระบบ ในบทนี้จะกล่าวถึงขั้นตอนละวิธีการดำเนินงานในการออกแบบโครงสร้างการทำงานของระบบนี้ทั้งหมด ซึ่งการออกแบบโดยรวมนั้นจะเป็นการทำงานโดยรับภาพจากกล้องวิดีโอ หรือภาพจากไฟล์วิดีโอมาประมวลผลในการตีความหมายของภาพและตรวจจับรถที่ขับย้อนศร โดยมีรายละเอียดการออกแบบทั้งหมดดังนี้

3.1 ออกแบบการทำงานของระบบ

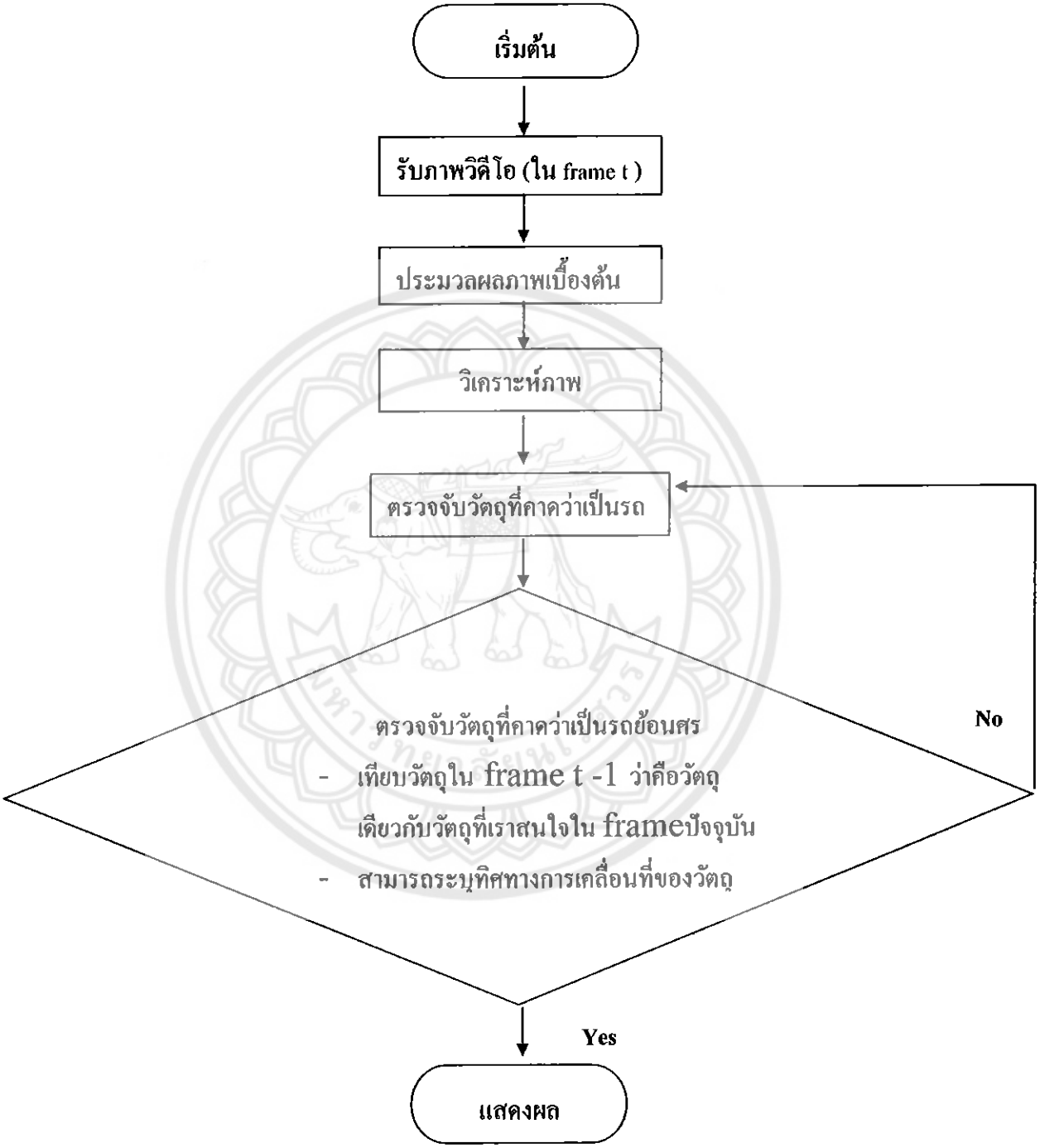
ทำการศึกษาข้อมูลว่าระบบการตรวจจับการขับรถผิดทิศทางการเดินรถ (ย้อนศร) นั้นจะต้องใช้อุปกรณ์อะไรบ้างที่จำเป็นต่อการนำมาใช้พัฒนาระบบนี้ โดยขั้นต้นเราจำเป็นต้องใช้กล้องวิดีโอหรือกล้องเว็บแคมที่มีความละเอียดสูงในการตรวจจับภาพบนท้องถนน เพื่อจะนำภาพวิดีโอมาประมวลผล แล้วทดสอบการการวางมุมกล้องไว้ที่ตำแหน่งต่างๆเลือกมุมที่เหมาะสมที่สุดที่จะนำมาใช้ และศึกษาการเขียนโปรแกรมที่ใช้ในการประมวลผลภาพ โดยมีวิธีการออกแบบดังนี้

3.1.1 การออกแบบระบบโดยสังเขป



รูปที่ 3.1 ภาพการทำงานโดยสังเขป

3.1.2 ออกแบบโครงสร้างการทำงานของโปรแกรม



รูปที่ 3.2 แสดงโครงสร้างการทำงานของโปรแกรม

จากรูปที่ 3.2 เริ่มต้นจากการรับภาพวิดีโอมาเป็นเฟรมเพื่อนำแต่ละเฟรมที่ได้มาประมวลผลต่อ ในการประมวลผลภาพเบื้องต้นจะเป็นการปรับภาพให้มีขนาดที่เท่ากันจากนั้นลดสัญญาณรบกวนกับภาพต้นฉบับและปรับภาพให้เป็นภาพระดับเทา หลังจากประมวลผลภาพเบื้องต้นแล้วจะเป็นการวิเคราะห์ภาพโดยการแยกวัตถุที่เคลื่อนที่ออกจากพื้นหลัง ก่อนนำไปปรับปรุงภาพให้ดีขึ้นเพื่อให้ได้วัตถุที่เคลื่อนที่นำมาตรวจจับและตรวจสอบตามเงื่อนไขว่าเป็นการเคลื่อนที่ไปในทิศทางที่ถูกต้องหรือผิดทิศทางการเดินรถหรือไม่ ก่อนจะแสดงผลออกมา โดยจะอธิบายรายละเอียดแต่ละขั้นตอนในหัวข้อต่อไป

3.2 ขั้นตอนการประมวลผลภาพ

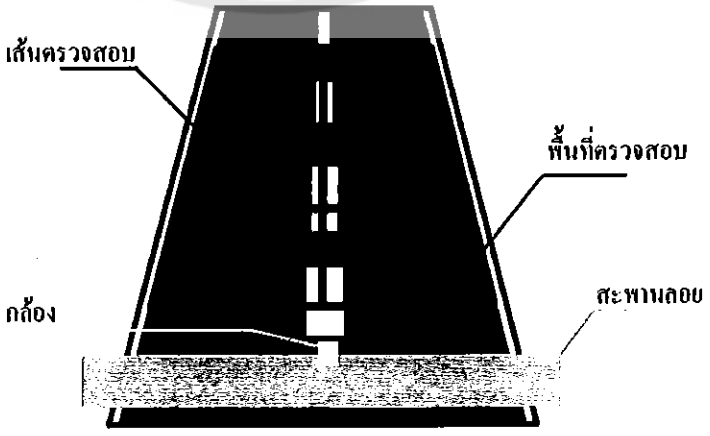
จากหัวข้อ 3.1 เป็นขั้นตอนการออกแบบการทำงานของระบบและการประมวลผลภาพ ซึ่งใช้งานผ่านโปรแกรมที่ผู้จัดทำได้ทำการออกแบบเขียน Code และพัฒนาโปรแกรมด้วยภาษา C/C++ โดยใช้ Visual Studio 2012 และประมวลผลภาพจาก OpenCV Library (Open Source Computer Vision) โดยขั้นตอนการทำงานมีดังนี้

3.2.1 การประมวลผลภาพวิดีโอ

การประมวลผลภาพวิดีโอในที่นี้เป็นการรับภาพวิดีโอไปประมวลผล สามารถทำได้ 2 วิธีคือ การรับภาพวิดีโอจากกล้องโดยตรง และการรับไฟล์วิดีโอที่ได้ถ่ายไว้แล้ว

3.2.1.1 รับภาพจากกล้องวิดีโอโดยตรง

วิธีนี้จะเป็นการทำงานประมวลผลแบบเวลาจริง (Real Time) นั่นคือรับภาพวิดีโอจากกล้องวิดีโอไปแล้วระบบจะประมวลผลในทันที โดยจะต้องเชื่อมต่อกล้องเข้ากับระบบคอมพิวเตอร์และติดต้องกล้องให้อยู่ในมุมสูงเป็นมุมที่เหมาะสมสำหรับการทำงานของระบบนี้ โดยมีตำแหน่งการติดตั้งกล้องดังรูปที่ 3.3



รูปที่ 3.3 รูปแสดงตำแหน่งการติดตั้งกล้องบนสะพานลอย

จากรูปที่ 3.3 จะเห็นว่าเราจะติดตั้งกล้องอยู่บนสะพานลอยแล้วให้หน้ากล้องออกทางถนนจากนั้นปรับตำแหน่งของกล้องและมุมมองกล้องให้อยู่ตรงกลางถนนพอดี

3.2.1.2 รับภาพจากไฟล์วิดีโอ

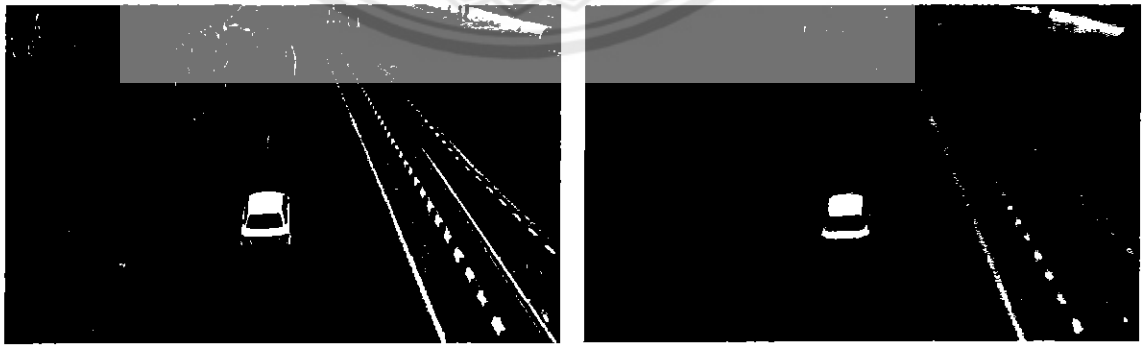
วิธีนี้จะเป็นการทำงานประมวลผลจากไฟล์วิดีโอที่ได้ทำการบันทึกเอาไว้ก่อนแล้ว และจะต้องมีมุมมองกล้องแบบเดียวกันดังรูปที่ 3.3 โดยการเปิดไฟล์วิดีโอที่อยู่บนเครื่องคอมพิวเตอร์ที่มีระบบการตรวจจับการขับรถผิดทิศทางการเดินรถ เพื่อนำภาพวิดีโอไปประมวลผล จะได้ผลลัพธ์ออกมาเช่นเดียวกับการรับภาพจากกล้องวิดีโอโดยตรง

3.2.2 การปรับขนาดภาพวิดีโอ

กดปรับขนาดภาพวิดีโอโดยการนำภาพวิดีโอต้นฉบับ คือ ภาพจากวิดีโอโดยตรงและการรับภาพจากไฟล์วิดีโอ ซึ่งสองวิธีนี้มีขนาดของภาพที่ไม่เท่ากัน โดยจะทำภาพให้เล็กลงเพื่อนำมาพิจารณาคำนวณต่อไป ซึ่งขนาดภาพวิดีโอยิ่งเล็กยิ่งทำให้โปรแกรมหรือคอมพิวเตอร์นั่นสามารถทำงานได้เร็วกว่าไฟล์ภาพวิดีโอขนาดใหญ่ จึงจำเป็นจะต้องมีการปรับขนาดของภาพต้นฉบับ ที่เข้ามาทั้งสองวิธี ให้เป็นขนาดเดียวกัน (ขนาดที่ผู้จัดทำกำหนดคือ 500x300 pixel)

3.2.3 การลดสัญญาณรบกวน

เมื่อเราทำการปรับขนาดภาพแล้ว เราจำเป็นจะต้องทำการลดสัญญาณรบกวน เพื่อที่จะเน้นความสนใจไปในวัตถุที่เคลื่อนที่ ที่เราต้องการจะตรวจสอบ ซึ่งทำให้สัญญาณรบกวนเล็กน้อยที่เกิดขึ้นหายไป โดยใช้การเบลอแบบปกติด้วยค่าเฉลี่ยแบบ 5x5 ได้ผลลัพธ์ดังรูปที่ 3.4



(ก) (ข)

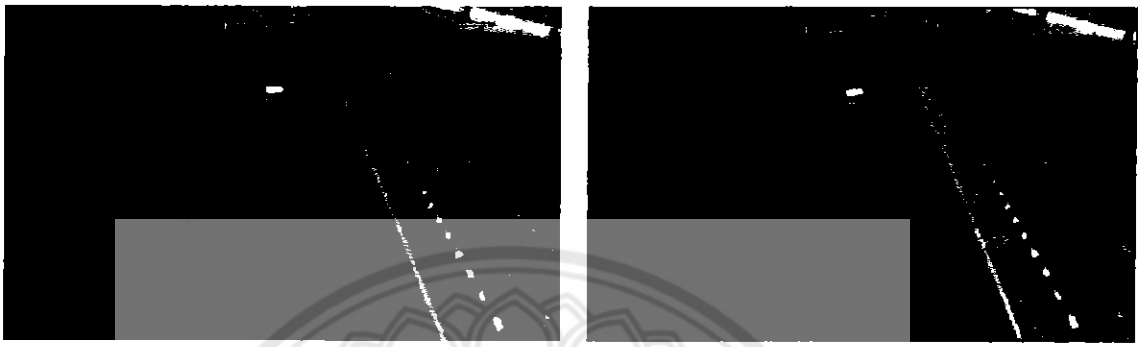
รูปที่ 3.4 แสดงผลการลดสัญญาณรบกวนของภาพ

(ก) รูปภาพต้นฉบับ

(ข) รูปภาพหลังจากนำภาพต้นฉบับไปลดสัญญาณรบกวน

3.2.4 การทำภาพระดับเทา

เมื่อเราทำการลดลดสัญญาณรับกวนของภาพแล้ว ขั้นตอนต่อไปเราจะนำภาพที่ได้มาเปลี่ยนให้เป็นภาพระดับเทาเพื่อที่เราจะนำไปใช้ในการประมวลผลภาพต่างๆ ในขั้นตอนต่อไป ดังรูปที่ 3.5

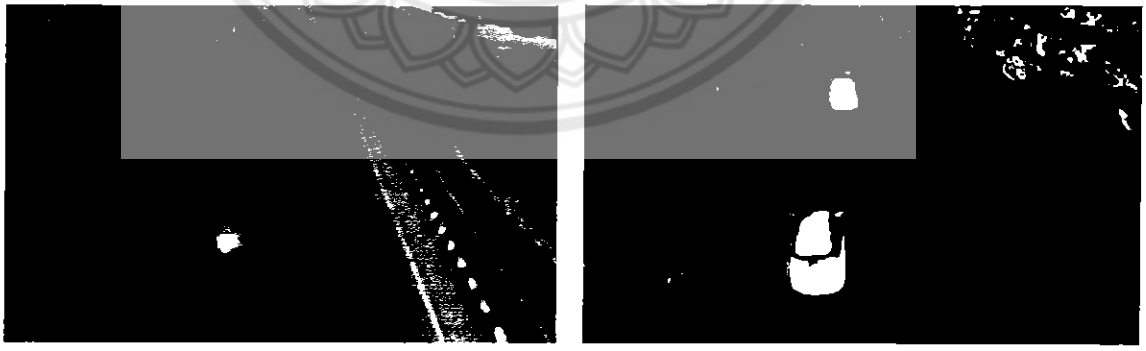


(ก) (ข)

รูปที่ 3.5 แสดงการแปลงภาพสีเป็นภาพระดับเทา
(ก) รูปภาพจากการลดสัญญาณรบกวนก่อนแปลงเป็นภาพระดับเทา
(ข) รูปภาพหลังจากการแปลงภาพระดับเทา

3.2.5 การแยกวัตถุออกจากพื้นหลัง

ขั้นตอนต่อมา เมื่อเราได้ภาพระดับสีเทาแล้ว เราจะนำมาทำการแยกวัตถุที่เคลื่อนที่ออกจากพื้นหลัง โดยใช้หลักการ Background Subtraction ซึ่งจะได้ผลลัพธ์ดังรูปที่ 3.6



(ก) (ข)

รูปที่ 3.6 แสดงการแยกวัตถุที่เคลื่อนที่ออกจากพื้นหลัง
(ก) รูปภาพระดับเทาก่อนนำไปแยกวัตถุออกจากพื้นหลัง
(ข) รูปภาพหลังการแยกวัตถุออกจากพื้นหลัง

จากรูปที่ 3.6 เป็นการแยกวัตถุออกจากพื้นหลังโดยหลักการ Background Subtraction ใช้แบบ Mixture of Gaussians (MOG) โดยการนำภาพสองเฟรมที่บริเวณเดียวกันในเวลาที่ต่างกันมาหาความแตกต่างของเฟรม foreground กับเฟรม background แล้วนำมาหักล้างกัน หากมีวัตถุที่เคลื่อนที่ในเฟรมแรกและเฟรมที่สองเมื่อนำภาพมาลบกันพื้นหลังจะหักล้างกันและมีผลต่างระหว่างเฟรมภาพทั้งสองเฟรม นั่นคือมีการเคลื่อนที่ของวัตถุมายังเฟรมที่สอง จะได้ออกมาดังรูป (ข)

3.2.6 การปรับภาพให้เป็นไบนารี

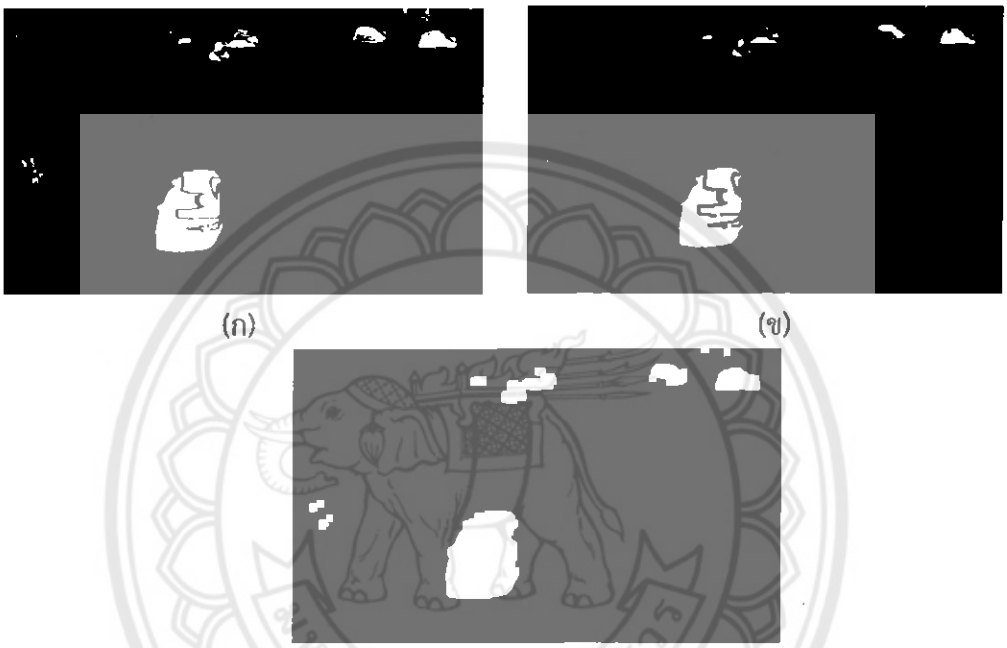
การปรับภาพให้เป็นไบนารี คือการนำผลลัพธ์ที่ได้จากการทำ Background Subtraction มาปรับเป็นภาพแบบไบนารี คือจะมีแค่สองค่า ขาวและดำ โดยกำหนดค่า Threshold อยู่ที่ 100 นั่นคือ พิกเซลที่มีค่าระดับเทาที่อยู่ต่ำกว่า 100 ให้เป็นภาพสีดำและ พิกเซลที่มีค่าระดับเทาที่อยู่มากกว่า 100 ให้เป็นสีขาวทั้งหมด ดังรูปที่ 3.7

(ก) (ข)

รูปที่ 3.7 แสดงการปรับภาพให้เป็นไบนารี
(ก) รูปภาพการแยกวัตถุออกจากพื้นหลังก่อนแปลงภาพเป็นไบนารี
(ข) รูปภาพหลังการแปลงเป็นภาพไบนารี

3.2.7 การปรับรูปร่างและโครงสร้างของภาพ

ในขั้นตอนนี้เราจะนำภาพไบนารี มาทำการปรับรูปร่างและโครงสร้างของภาพโดยใช้วิธีการที่เรียกว่า Morphological เพื่อปรับเปลี่ยนรูปร่างและโครงสร้างของภาพ เพื่อเพิ่มรายละเอียดให้กับวัตถุที่สนใจและลดสัญญาณรบกวนลง ดังรูปที่ 3.8



(ก) (ข)

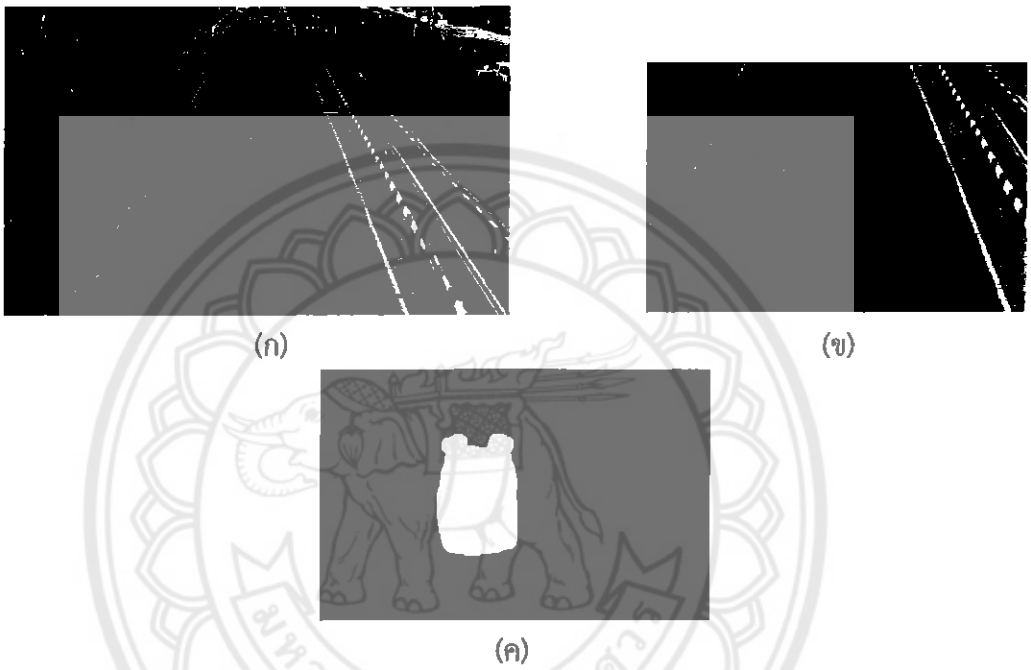
(ค)

รูปที่ 3.8 แสดงการปรับเปลี่ยนรูปร่างและโครงสร้างของภาพ
(ก) รูปภาพไบนารีก่อนการปรับโครงสร้างภาพ
(ข) รูปภาพหลังทำการย่อ (erosion)
(ค) รูปภาพหลังทำการขยาย (dilation)

จากรูปที่ 3.8 ในขั้นตอนแรกนำภาพไบนารีดังรูป (ก) มาทำการย่อ (erosion) ด้วยขนาด 3x3 เพื่อกำจัดสัญญาณรบกวนที่มีขนาดเล็กๆให้หายไปได้ผลลัพธ์ดังรูป (ข) จากนั้นนำมาขยาย (dilation) ด้วย ขนาด 8 x 8 ทำให้ได้ผลลัพธ์ดังรูป (ค)

3.2.8 การกำหนดพื้นที่ที่ใช้ตรวจจับรถ

เราจะทำการระบุตำแหน่งหรือกำหนดพื้นที่ที่ใช้ในการตรวจจับ เพื่อที่เราจะนำภาพเฉพาะในพื้นที่มาใช้ในการคำนวณโดยจะไม่สนใจพื้นที่นอกกรอบ ทำโดยการ Clone image โดยมีจุด เริ่มต้นอยู่ที่มุมซ้ายบน (x , y) = (100,100) และจุดสิ้นสุดอยู่ที่มุมขวาล่าง (x , y) = (320 , 200) โดยประมาณดังรูปที่ 3.9



(ก) (ข) (ค)

รูปที่ 3.9 แสดงการกำหนดพื้นที่ที่สนใจในการตรวจจับ
(ก) รูปภาพการตีเส้นสีฟ้าบนพื้นที่ตรวจสอบจากภาพต้นฉบับ
(ข) รูปภาพต้นฉบับหลังจากการกำหนดพื้นที่ตรวจสอบ
(ค) รูปภาพไบนารีหลังจากการกำหนดพื้นที่ตรวจสอบ

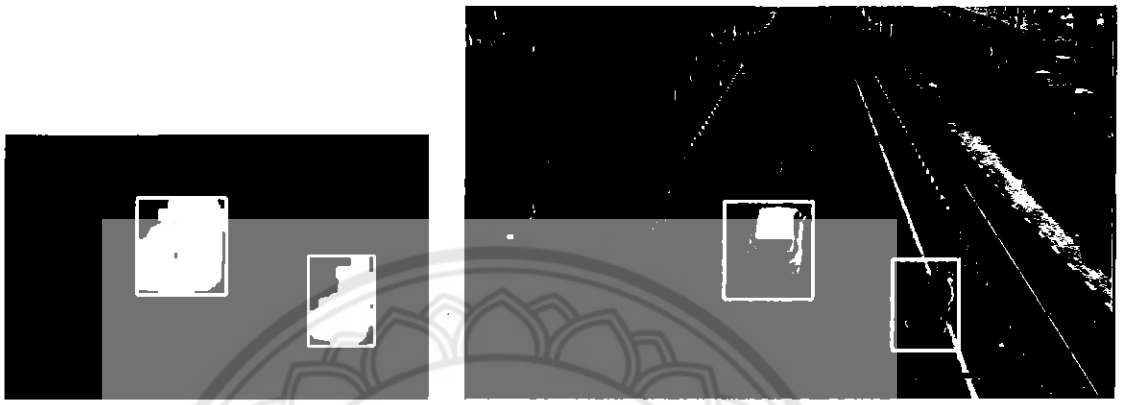
จากรูปที่ 3.9 (ก) โดยประมาณค่าจากเส้นกรอบในพื้นที่สนใจ (เส้นสีน้ำเงิน) ทำให้ได้ภาพตรงจุดที่สนใจโดยประมาณดังรูปที่ 3.9 (ข) และได้ภาพไบนารีดังรูปที่ 3.9 (ค) เป็นการนำพื้นที่ ที่สนใจเท่านั้นมาคำนวณจะไม่ได้นำทั้งภาพมาคำนวณ และตัดส่วนอื่นที่ไม่สนใจไม่นำมาคำนวณ

3.2.9 การหาพื้นที่ที่ติดกัน

ขั้นตอนนี้จะเป็นการหาบริเวณที่มีการเชื่อมติดกัน (Connected Component Labeling) ของพิกเซลและแยกบริเวณต่างๆออกจากกันในภาพไบนารี โดยใน OpenCV ใช้วิธีการหา contours เพื่อให้ได้วัตถุที่เป็นสีขาวออกเป็นกลุ่มๆ แล้วจะนำแต่ละกลุ่มก้อนไปใช้ในขั้นตอนต่อไป

3.2.10 การตีกรอบรอบวัตถุ

เมื่อได้แยกวัตถุออกเป็นกลุ่มๆสีขาวแล้ว เราจะทำการตีกรอบล้อมรอบวัตถุ เพื่อที่จะแสดงว่าสิ่งนั้นคือวัตถุที่เราสนใจจะตรวจจับดังรูปที่ 3.10



(ก) (ข)

รูปที่ 3.10 แสดงการตีกรอบรอบวัตถุที่สนใจ
(ก) รูปภาพแสดงการตีกรอบวัตถุบนพื้นที่ตรวจสอบจากภาพไบนารี
(ข) รูปภาพแสดงการตีกรอบวัตถุไปที่ภาพผลลัพธ์

โดยในขั้นตอนที่หา connected component labeling มาแล้ว จะทำการตีกรอบทุกพื้นที่ที่เป็นกลุ่มสีขาว จากรูปที่ 3.10 (ก) จะตีกรอบรอบวัตถุและจะมีตำแหน่งของกรอบ ในการตีกรอบไปที่ภาพผลลัพธ์ จะต้องบวกค่าตำแหน่งดังนี้

ตำแหน่ง x + 100

ตำแหน่ง y + 100

เนื่องจากพื้นที่ที่นำมาคำนวณนั้น จากข้อ 3.2.8 มีจุดเริ่มต้นของขอบภาพพื้นที่ตรวจสอบ (x , y) = (100 , 100) แสดงว่าต้องบวกค่าตำแหน่งของ (x , y) เพื่อไปแสดงผลการตีกรอบรอบวัตถุที่ตำแหน่งเดียวกันที่ภาพผลลัพธ์ เมื่อคำนวณแล้วจะได้ตำแหน่งการตีกรอบวัตถุแสดงผลออกที่ภาพผลลัพธ์เป็นตำแหน่งเดียวกับการตีกรอบล้อมรอบขอบวัตถุ ดังรูปที่ 3.10 (ข)

3.2.11 การกำหนดพื้นที่ตรวจสอบและเส้นตรวจสอบทิศทาง

การกำหนดพื้นที่ตรวจสอบและเส้นตรวจสอบทิศทางนั้นมีขั้นตอนการทำดังนี้ กำหนดเส้นแบ่งเลนซ้ายขวา (เส้นสีฟ้าตรงกลาง) ให้อยู่กลางถนนตรงเส้นแบ่งเลนของถนนจริง โดยการทดลองนี้มีแกน x = 270

เส้นตรวจสอบ (เส้นสีแดงบน) โดยให้อยู่ใน ½ ของภาพในแนวแกน y นั้นคือแกน y = 150 และมีแกน x จากจุดกลาง มาทางซ้ายและขวา 110 พิกเซล จะได้ ตำแหน่ง (x , y) ของเส้นวาดจากจากขอบด้านซ้ายถึงด้านขวา ตำแหน่งด้านซ้ายคือ (160 , 150) และตำแหน่งด้านขวาคือ (380 , 150)

เส้นตรวจสอบ (เส้นสีแดงล่าง) โดยมีระยะห่างอยู่ที่ 50 พิกเซลจากเส้นตรวจสอบเส้นบนลงมา นั้นคือแกน y = 200 และมีแกน x จากจุดกลางมาทางซ้ายและขวา 130 พิกเซล จะได้ตำแหน่ง (x , y) ของเส้นวาดจากขอบด้านซ้ายถึงด้านขวา ตำแหน่งด้านซ้ายคือ (140 , 200) และตำแหน่งด้านขวาคือ (400 , 200)

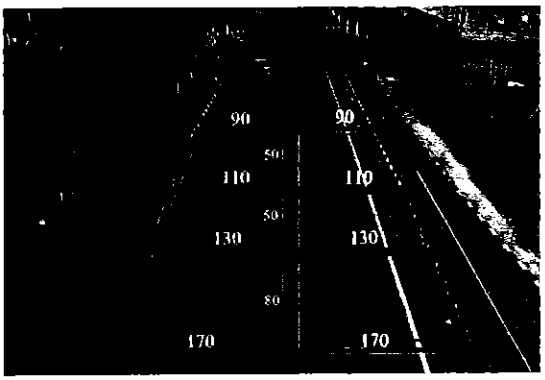
เส้นกรอบพื้นที่ที่สนใจ (เส้นสีฟ้าบน) ระยะห่างจากเส้นตรวจสอบบน 50 พิกเซล นั้นคือแกน y = 100 และมีแกน x จากจุดกลาง มาทางซ้ายและขวา 90 พิกเซล จะได้ ตำแหน่ง (x , y) ของเส้นวาดจากขอบด้านซ้ายถึงด้านขวา ตำแหน่งด้านซ้ายคือ (180 , 100) และตำแหน่งด้านขวาคือ (360 , 100)

เส้นกรอบพื้นที่ที่สนใจ (เส้นสีฟ้าล่าง) ระยะห่างจากเส้นตรวจสอบล่าง 80 พิกเซล นั้นคือแกน y = 280และมีแกน x จากจุดกลาง มาทางซ้ายและขวา 170 พิกเซล จะได้ ตำแหน่ง (x , y) ของเส้นวาดจากขอบด้านซ้ายถึงด้านขวา ตำแหน่งด้านซ้ายคือ (100 , 280) และตำแหน่งด้านขวาคือ (440 , 100)

เส้นขอบสีฟ้า (เส้นซ้าย) จะได้ ตำแหน่ง (x , y) ของเส้นวาดจากขอบด้านบนถึงด้านล่าง ตำแหน่งด้านบนคือ (180 , 100) และตำแหน่งด้านล่างคือ (100 , 280)

เส้นขอบสีฟ้า (เส้นขวา) จะได้ ตำแหน่ง (x , y) ของเส้นวาดจากขอบด้านบนถึงด้านล่าง ตำแหน่งด้านบนคือ (360 , 100) และตำแหน่งด้านล่างคือ (440 , 280)

จากขั้นตอนทั้งหมดจะได้เส้นพื้นที่ตรวจสอบสีน้ำเงินและเส้นตรวจสอบสีแดงแสดงผลลัพธ์ออกทางภาพผลลัพธ์ดังรูปที่ 3.11



รูปที่ 3.11 แสดงการกำหนดพื้นที่ตรวจสอบและเส้นตรวจสอบ

3.2.12 การตรวจจับรถย้อนศร

เป็นการตรวจสอบว่ารถเคลื่อนที่ไปในทิศทางที่ถูกต้องหรือไม่ เริ่มต้นจาก เมื่อรถเข้ามาในพื้นที่ ที่สนใจโดยจะคำนวณหาจุดศูนย์กลางของกรอบสี่เหลี่ยมที่ตีกรอบรถ คำนวณได้จาก

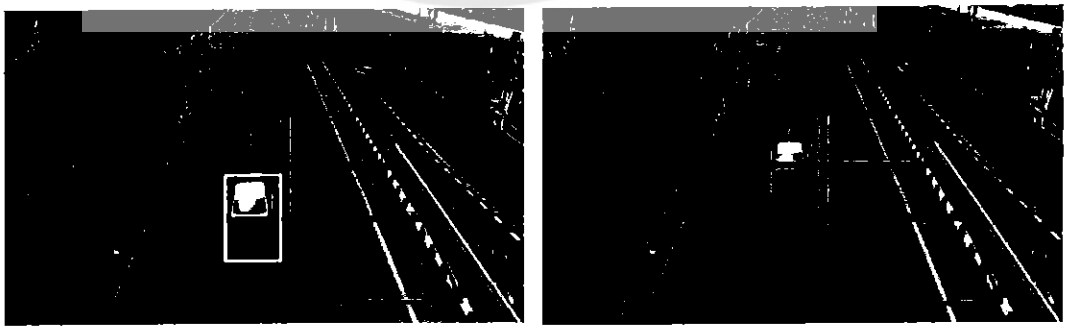
Width / 2

High / 2

หลังจากคำนวณแล้วจะได้ตำแหน่งจุดศูนย์กลาง (x , y) มาจากนั้นกำหนดให้จุดศูนย์กลางที่เข้ามาในเฟรมแรกของการตรวจสอบเก็บค่าตำแหน่ง (x , y) เริ่มต้นไว้ โปรแกรมจะมีตัวแปร Source (x , y) เก็บค่าจุดเริ่มต้นของแต่ละกรอบสี่เหลี่ยมที่ตีกรอบรถอยู่ ถ้ารถเคลื่อนที่ไปในทิศทางใดก็ตามจะมีตัวแปร Destination (x , y) ไว้เก็บค่าตำแหน่ง (x , y) ปัจจุบันของรถคันดังกล่าว

เมื่อเราทราบตำแหน่งเริ่มต้นและตำแหน่งปัจจุบันของรถแล้ว เราจะนำตำแหน่งปัจจุบันของรถมาตรวจสอบกับเส้นตรวจสอบว่ารถมีการเคลื่อนที่ไปในทิศทางที่ถูกต้องหรือย้อนศร ถ้าตำแหน่งปัจจุบันเคลื่อนที่ไปที่ 145 > y > 155 ผ่านเส้นตรวจสอบที่ 1 เส้นจะเปลี่ยนเป็นสีเขียว ถ้าตำแหน่งปัจจุบันเคลื่อนที่ไปที่ 195 > y > 205 ผ่านเส้นตรวจสอบที่ 2 เส้นจะเปลี่ยนเป็นสีเขียว เมื่อผ่านเส้นตรวจสอบแต่ละเส้นจะเก็บสถานะของการผ่านเส้นแต่ละเส้นเอาไว้ตรวจสอบ

การตรวจจับรถที่ขับถูกทิศทางการเดินรถนั้นทำได้โดยตรวจสอบเงื่อนไขว่าเมื่อเข้ามาทิศทางที่ถูกต้องโดยมีแกน x ในการแบ่งเลนถนน และผ่านเส้นตรวจสอบเงื่อนไขทั้งสองเส้น แสดงว่าเป็นรถที่ขับถูกทิศทาง นั่นคือเมื่อจุดเริ่มต้นของรถเข้ามาถูกทิศทางจะแสดงการตีกรอบรถเป็นสีเหลือง และจะมีตำแหน่งปัจจุบันของรถคันดังกล่าวเคลื่อนที่ผ่านเส้นตรวจสอบที่ 1 ดังรูปที่ 3.12 (ก) และผ่านเส้นตรวจสอบที่ 2 ก่อนจะแสดงการตีกรอบรถเป็นสีเขียวดังรูปที่ 3.12 (ข) แสดงว่าเป็นรถที่เคลื่อนที่ถูกทิศทางการเดินรถ



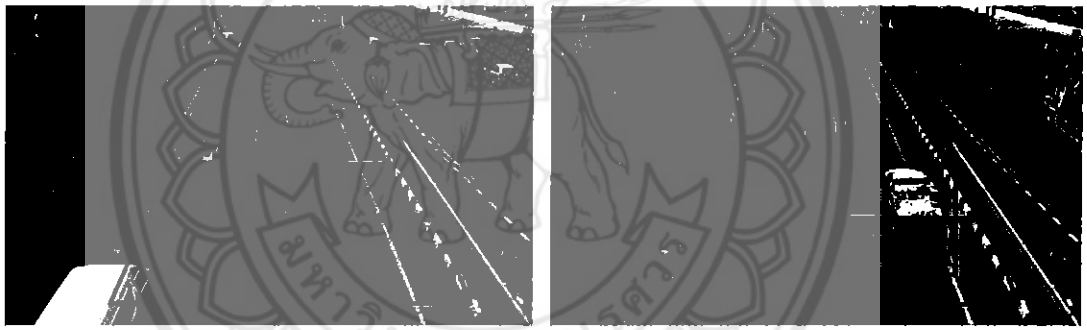
(ก) (ข)

รูปที่ 3.12 แสดงการเคลื่อนที่ของรถในเลนถนนที่ถูกต้องผ่านเส้นตรวจสอบ

(ก) รูปภาพแสดงการเคลื่อนที่ของรถผ่านเส้นตรวจสอบเส้นแรก ในเลนของถนนที่ถูกต้องของเลนซ้าย

(ข) รูปภาพแสดงการเคลื่อนที่ของรถผ่านเส้นตรวจสอบเส้นที่ 2 ในเลนของถนนที่ถูกต้องของเลนซ้าย กรอบจะเป็นสีเขียวแสดงว่าขับถูกทิศทางการเดินรถ

การตรวจจับรถที่เคลื่อนที่ผิดทิศทางการเดินรถนั้นทำได้โดย ตรวจสอบเงื่อนไขว่าเมื่อรถเข้ามาทิศทางที่ผิดโดยมีแกน x ในการแบ่งเลนถนน และผ่านเส้นตรวจสอบเงื่อนไขเส้นแรก แสดงว่าเป็นรถที่ขับขี่ผิดทิศทางการเดินรถ ถ้าตำแหน่งเริ่มต้นมีค่า $x < 165$ แล้วมีตำแหน่งการเคลื่อนที่ของรถในทิศทางที่ผิดผ่านเส้นตรวจสอบเส้นแรกที่ผ่าน แสดงว่าขับผิดทิศทางการเดินรถ ตีกรอบเป็นสีแดงดังรูปที่ 3.13 (ก) และถ้าตำแหน่งเริ่มต้นมีค่า $x > 166$ แล้วมีตำแหน่งการเคลื่อนที่ของรถในทิศทางที่ผิดผ่านเส้นตรวจสอบเส้นแรกที่ผ่าน แสดงว่าขับผิดทิศทางการเดินรถ ตีกรอบเป็นสีแดงดังรูปที่ 3.13 (ข)



(ก) (ข)

รูปที่ 3.13 แสดงการเคลื่อนที่ของรถที่ผิดเลนถนนผ่านเส้นตรวจสอบ

(ก) รูปภาพแสดงการเคลื่อนที่ของรถในทิศทางที่ผิดของเลนซ้ายถนนผ่านเส้นตรวจสอบเส้นแรกกรอบจะเป็นสีแดง แสดงว่าขับผิดทิศทาง

(ข) รูปภาพแสดงการเคลื่อนที่ของรถในทิศที่ผิดของเลนขวาถนนผ่านเส้นตรวจสอบเส้นแรกกรอบจะเป็นสีแดง แสดงว่าขับผิดทิศทาง

บทที่ 4
ผลการทดลอง

การทดลองตรวจจับการขับรถผิดทิศทางการเดินรถด้วยการประมวลผลภาพนั้นจะเป็นการตรวจจับวัตถุที่เป็นรถยนต์หรือรถจักรยานยนต์ก็ตามที่ขับในทิศทางการเดินรถได้ถูกต้องกับการตรวจจับรถที่ขับผิดทิศทางการเดินรถ(ย้อนศร) ในการทดลองนี้ผู้จัดทำได้ทำการติดตั้งกล้องและบันทึกวีดีโอบริเวณสะพานลอยหน้ามหาวิทยาลัยนเรศวร ซึ่งมีจำนวนรถที่ใช้ทดลอง 336 คัน แบ่งเป็นรถขับผิดทิศทาง 26 คัน รถขับถูกทิศทาง 310 คัน โดยตัดสินจากสายตา และมีผลการทดลองดังนี้

4.1 ผลการทดลองตรวจจับรถที่ขับผิดทิศทางการเดินรถ

ในการทดลองตรวจจับรถที่ขับผิดทิศทางการเดินรถ ผู้จัดทำได้ทำการทดลองโดยการปรับขนาดของพื้นที่ที่ใช้อ้างอิงในการตีกรอบรอบวัตถุ (boundingRect(contours[i]).area()) เพื่อใช้ในการติดตามรถที่เราต้องการจะติดตาม ก่อนจะนำไปตรวจสอบว่ารถคันดังกล่าว ขับผิดทิศทางหรือไม่ โดยเราต้องการทราบว่าการปรับขนาดของวัตถุดังกล่าวจะมีผลทำให้โปรแกรมตรวจจับวัตถุมีความถูกต้องแค่ไหน เพื่อใช้ตรวจสอบรถที่ขับผิดหรือถูกทิศทางการเดินรถ ซึ่งกำหนดให้ค่า (boundingRect(contours[i]).area()) มีค่ามากกว่า 0, 500, 1000, 1500, 2000, 2500 และ 3000 เพื่อใช้ตรวจสอบ โดยสามารถแบ่งออกได้ 4 กรณีดังนี้

1. True Positive (TP) คือ รถที่ขับผิดทิศทาง และโปรแกรมระบุว่าผิดทิศทาง
2. False Negative (FN) คือ รถที่ขับผิดทิศทาง แต่โปรแกรมไม่สามารถระบุได้หรือโปรแกรมทำนายผิด
3. False Positive (FP) คือ รถที่ขับถูกทิศทาง แต่โปรแกรมไม่สามารถระบุได้หรือโปรแกรมทำนายผิด
4. True Negative (TN) คือ รถที่ขับถูกทิศทาง และโปรแกรมระบุว่าถูกทิศทาง

ตารางที่ 4.1 รูปแบบคอนฟิวชั่นเมทริกซ์ (Confusion matrix)

Actual class (%)	Predicted class (%)	
	Yes	No
Yes	TP	FN
No	FP	TN

จากตารางที่ 4.1 ค่า Accuracy คือ ค่าความถูกต้องที่โปรแกรมสามารถทำได้ หาได้จากสมการ

$$Accuracy = (TP+TN)/(TP+TN+FP+FN) \quad (9)$$

ตารางที่ 4.2 การทดสอบจากอัลกอริทึมที่ boundingRect(contours[i]).area() > 0

ค่าอ้างอิง	โปรแกรมตรวจจับได้	
	รถขับผิดทิศทาง	รถขับถูกทิศทาง
รถขับผิดทิศทาง	15	11
รถขับถูกทิศทาง	57	253

ผลการทดลองตรวจจับรถที่ขับผิดทิศทางการเดินรถที่ boundingRect(contours[i]).area() > 0 **ให้ความถูกต้อง 79.76 %**

ตารางที่ 4.3 การทดสอบจากอัลกอริทึมที่ boundingRect(contours[i]).area() > 500

ค่าอ้างอิง	โปรแกรมตรวจจับได้	
	รถขับผิดทิศทาง	รถขับถูกทิศทาง
รถขับผิดทิศทาง	15	11
รถขับถูกทิศทาง	51	259

ผลการทดลองตรวจจับรถที่ขับผิดทิศทางการเดินรถที่ boundingRect(contours[i]).area() > 500 **ให้ความถูกต้อง 81.54 %**

ตารางที่ 4.4 การทดสอบจากอัลกอริทึมที่ boundingRect(contours[i]).area() > 1000

ค่าอ้างอิง	โปรแกรมตรวจจับได้	
	รถขับผิดทิศทาง	รถขับถูกทิศทาง
รถขับผิดทิศทาง	16	10
รถขับถูกทิศทาง	40	270

ผลการทดลองตรวจจับรถที่ขับผิดทิศทางการเดินรถที่ boundingRect(contours[i]).area() > 1000 **ให้ความถูกต้อง 85.11 %**

ตารางที่ 4.5 การทดสอบจากอัลกอริทึมที่ boundingRect(contours[i]).area() > 1500

ค่าอ้างอิง	โปรแกรมตรวจจับได้	
	รถขับผิดทิศทาง	รถขับถูกทิศทาง
รถขับผิดทิศทาง	16	10
รถขับถูกทิศทาง	48	262

ผลการทดลองตรวจจับรถที่ขับผิดทิศทางการเดินรถที่ boundingRect(contours[i]).area() > 1500 **ให้ความถูกต้อง** 82.73 %

ตารางที่ 4.6 การทดสอบจากอัลกอริทึมที่ boundingRect(contours[i]).area() > 2000

ค่าอ้างอิง	โปรแกรมตรวจจับได้	
	รถขับผิดทิศทาง	รถขับถูกทิศทาง
รถขับผิดทิศทาง	9	17
รถขับถูกทิศทาง	62	248

ผลการทดลองตรวจจับรถที่ขับผิดทิศทางการเดินรถที่ boundingRect(contours[i]).area() > 2000 **ให้ความถูกต้อง** 76.48 %

ตารางที่ 4.7 การทดสอบจากอัลกอริทึมที่ boundingRect(contours[i]).area() > 2500

ค่าอ้างอิง	โปรแกรมตรวจจับได้	
	รถขับผิดทิศทาง	รถขับถูกทิศทาง
รถขับผิดทิศทาง	8	18
รถขับถูกทิศทาง	113	197

ผลการทดลองตรวจจับรถที่ขับผิดทิศทางการเดินรถที่ boundingRect(contours[i]).area() > 2500 **ให้ความถูกต้อง** 61.01 %

ตารางที่ 4.8 การทดสอบจากอัลกอริทึมที่ boundingRect(contours[i]).area() > 3000

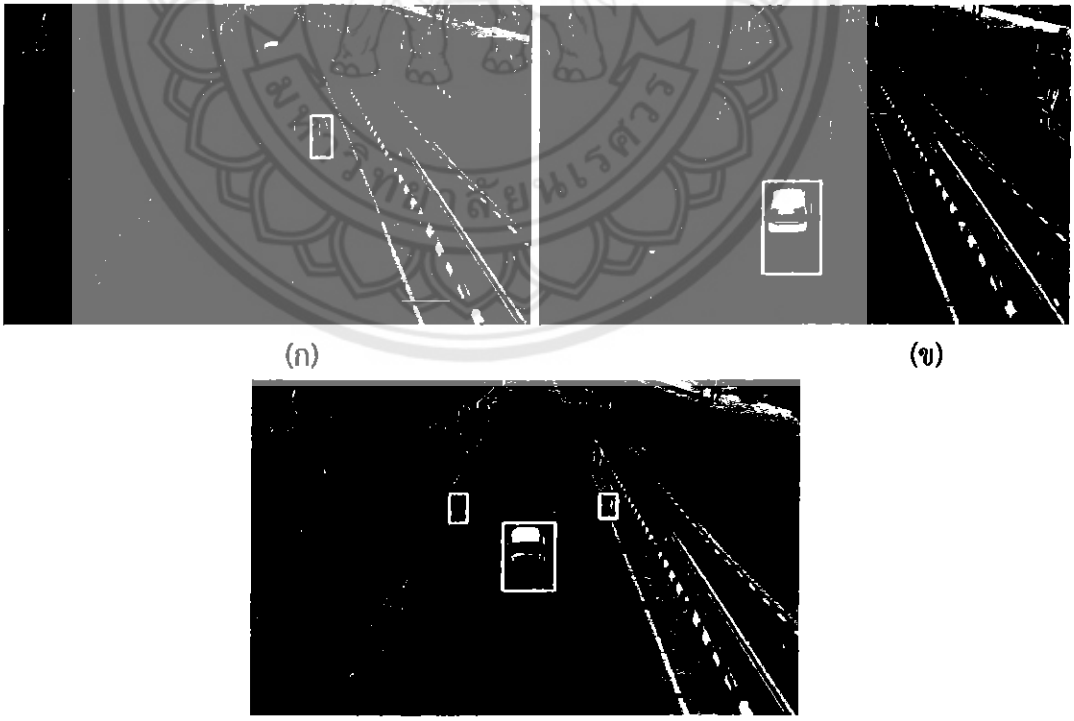
ค่าอ้างอิง	โปรแกรมตรวจจับได้	
	รถขับผิดทิศทาง	รถขับถูกทิศทาง
รถขับผิดทิศทาง	7	19
รถขับถูกทิศทาง	142	168

ผลการทดลองตรวจจับรถที่ขับผิดทิศทางการเดินรถที่ boundingRect(contours[i]).area() > 3000 **ให้ความถูกต้อง** 52.08 %

จากการทดลองปรับค่า (boundingRect(contours[i]).area()) ทั้งหมดนั้นได้ค่าที่ดีที่สุดสำหรับการทดลองนี้นั่นคือการปรับค่า (boundingRect(contours[i]).area() > 1000) ซึ่งให้ความถูกต้องที่สุดจากค่าทั้งหมด มีความถูกต้องถึง 85.11 %

4.2 ผลการทดลองแสดงผลจากการทดสอบด้วยวีดีโอ

4.2.1 ผลการตรวจจับรถที่เข้ามาในพื้นที่ตรวจสอบ



(ก) (ข)

(ค)

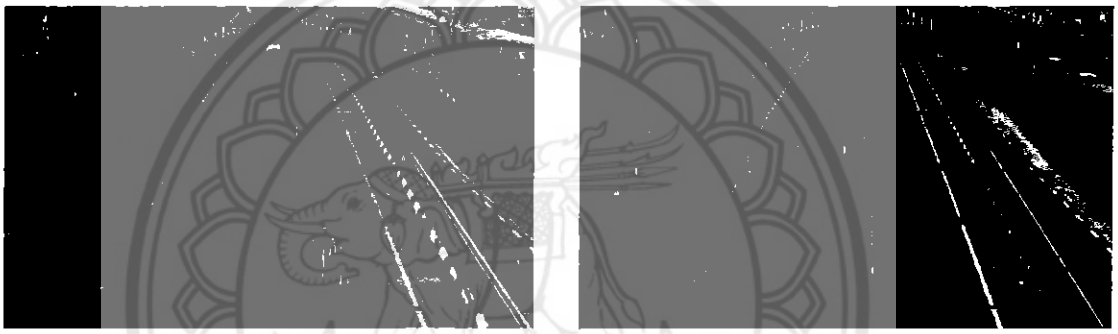
รูปที่ 4.1 แสดงผลลัพธ์ของการตรวจจับรถที่เข้ามาในพื้นที่ตรวจสอบ

(ก) แสดงผลการตรวจจับรถจักรยานยนต์ 1 คัน ที่เข้ามาในพื้นที่ตรวจสอบพิจารณา โดยจะตีกรอบสีเหลืองให้รู้ว่ากำลังจะทำการติดตามและตรวจสอบ

(ข) แสดงผลการตรวจจับรถยนต์ 1 คัน ที่เข้ามาในพื้นที่ตรวจสอบพิจารณาโดยจะตีกรอบสีเหลืองให้รู้ว่ากำลังจะทำการติดตามและตรวจสอบ

(ค) แสดงผลการตรวจจับรถที่เข้ามาในพื้นที่ตรวจสอบมากกว่า1คัน โดยจะตีกรอบสีเหลืองทั้ง 3 คัน ให้รู้ว่ากำลังจะทำการติดตามและตรวจสอบ

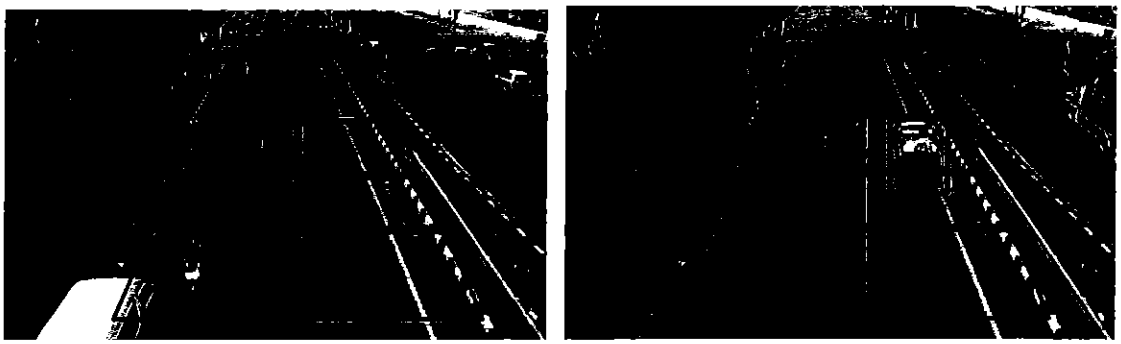
4.2.2 ผลการตรวจจับรถที่ขับขี่ถูกทิศทางการเดินรถ



(ก) (ข)

รูปที่ 4.2 แสดงผลลัพธ์ของการตรวจจับรถที่ขับขี่ถูกทิศทางการเดินรถ
(ก) แสดงผลการตรวจสอบรถที่ขับขี่ถูกทิศทางการเดินรถทางเลนขวาของถนน เมื่อผ่านเส้นตรวจสอบสีแดง จะตีกรอบสี่เหลี่ยมสีเขียวที่รถ
(ข) แสดงผลการตรวจสอบรถที่ขับขี่ถูกทิศทางการเดินรถทางเลนซ้ายของถนน เมื่อผ่านเส้นตรวจสอบสีแดง จะตีกรอบสี่เหลี่ยมสีเขียวที่รถ

4.2.3 ผลการตรวจจับรถที่ขับขี่ผิดทิศทางการเดินรถ



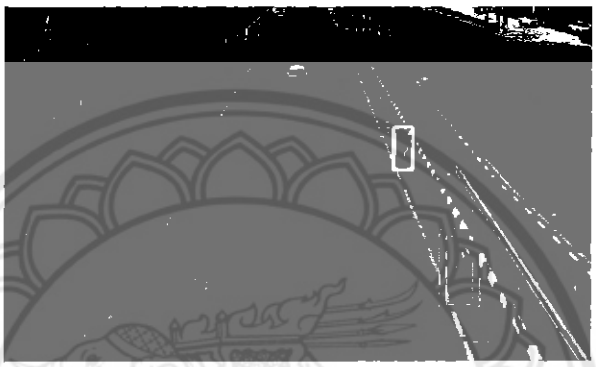
(ก) (ข)

รูปที่ 4.3 แสดงผลลัพธ์ของการตรวจจับรถที่ขับขี่ผิดทิศทางการเดินรถ

(ก) แสดงผลการตรวจสอบรถที่ขับขี่ผิดทิศทางการเดินรถทางเลนซ้ายของถนน เมื่อผ่านเส้นตรวจสอบสีแดง จะตีกรอบสี่เหลี่ยมสีแดงที่รถ

(ข) แสดงผลการตรวจสอบรถที่ขับขี่ผิดทิศทางการเดินรถทางเลนขวาของถนน เมื่อผ่านเส้นตรวจสอบสีแดง จะตีกรอบสี่เหลี่ยมสีแดงที่รถ

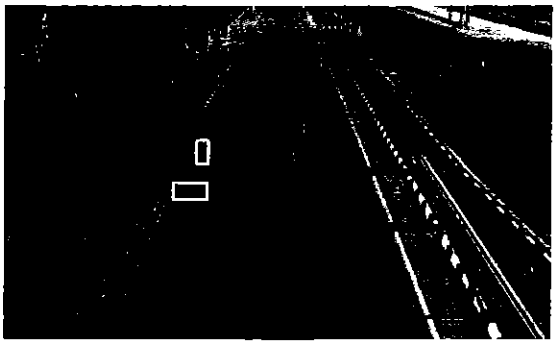
4.2.4 ผลการตรวจจับรถในพื้นที่ตรวจสอบมากกว่า 1 คัน



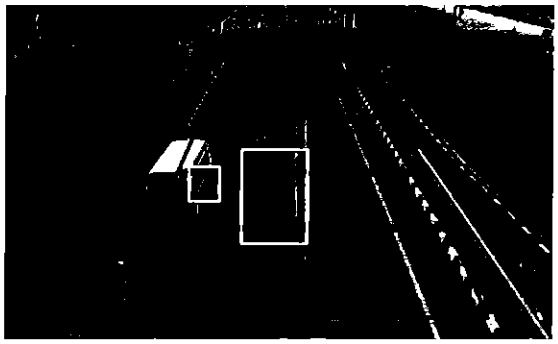
รูปที่ 4.4 แสดงผลลัพธ์ของการตรวจจับรถที่เข้ามาในพื้นที่ตรวจสอบมากกว่า 1 คัน

จากรูปที่ 4.4 เป็นผลการแสดงการตรวจจับรถที่เข้ามาในพื้นที่ตรวจสอบมากกว่า 1 คัน โดยจะแสดงผลตามที่รถแต่ละคันนั้นขับมาถูกต้องหรือผิดทิศทางการเดินรถตามที่ได้โปรแกรมไว้ จากภาพนั่นคือตีกรอบสีแดงรถที่ขับผิดทิศทางการเดินรถทางเลนซ้าย ตีกรอบสีเขียวรถที่ขับถูกต้องทางเลนขวา ตีกรอบสีเหลืองรถที่กำลังเข้ามาในพื้นที่ตรวจสอบทางเลนขวาถนน

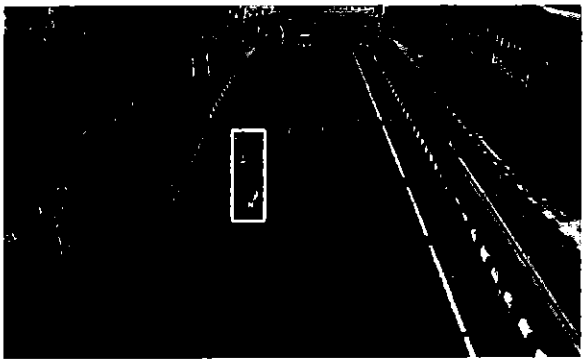
4.2.5 ผลการตรวจจับที่ผิดพลาด



(ก)



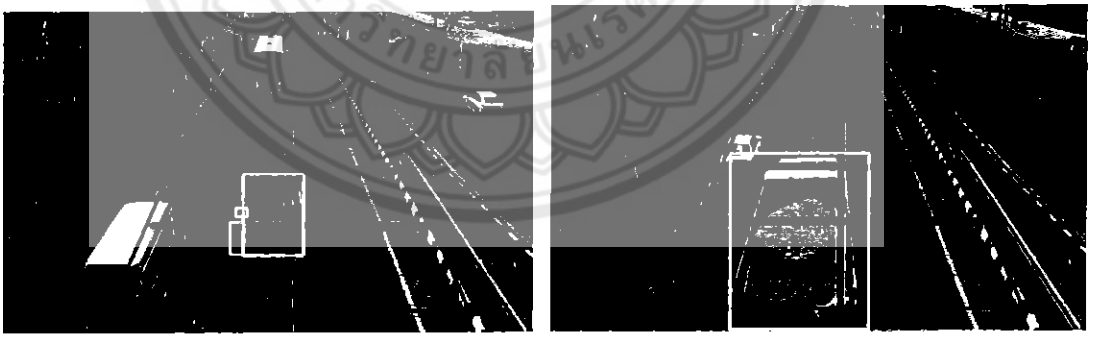
(ข)



(ค)

รูปที่ 4.5 แสดงผลลัพธ์ของการตรวจจับที่ผิดพลาด
(ก) แสดงผลการตรวจจับเงาของใบไม้ เนื่องจากมีลมพัดให้เงาของใบไม้เคลื่อนที่จึงเกิดการตรวจจับที่ผิดพลาด
(ข) แสดงผลการตรวจจับเงาในกระจกรถที่จอดอยู่ในพื้นที่ตรวจสอบ เนื่องจากรถที่ขับผ่านเกิดเงาสะท้อนในกระจกรถที่จอด จึงทำให้เกิดเงาที่กระจกและมีการเคลื่อนที่
(ค) แสดงผลการตรวจจับรถที่ใกล้กันและตีกรอบเป็นกรอบเดียวกัน

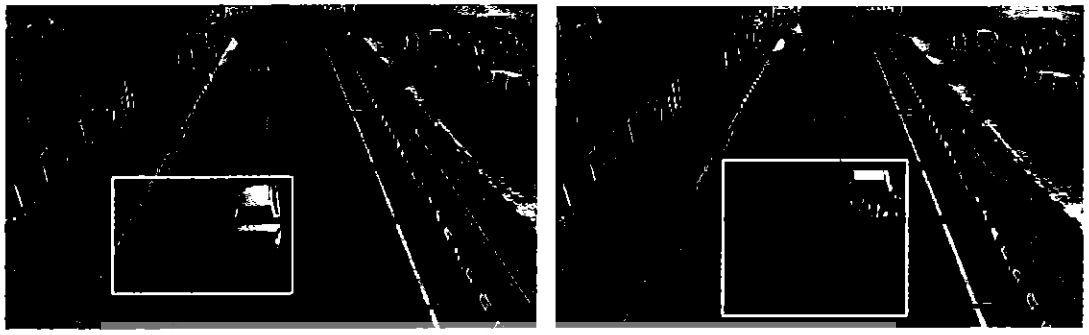
4.2.6 ผลการตรวจจับรถที่มีขนาดใหญ่เกินพื้นที่ตรวจสอบ



(ก) (ข)

รูปที่ 4.6 แสดงผลลัพธ์ของรถที่มีขนาดใหญ่เกินพื้นที่ตรวจสอบ
(ก) แสดงผลการตรวจจับรถเมล์ที่ใหญ่เกินพื้นที่ตรวจสอบ ทำให้ไม่สามารถตรวจสอบได้
(ข) แสดงผลการตรวจจับรถบรรทุกที่ใหญ่เกินพื้นที่ตรวจสอบ ทำให้ไม่สามารถตรวจสอบได้

4.2.7 ผลการตรวจจับรถที่มีเงาขนาดใหญ่



(ก) (ข)

รูปที่ 4.7 แสดงผลลัพธ์ของการตรวจจับรถที่มีเงาขนาดใหญ่
(ก) แสดงผลการตรวจจับรถที่มีเงาขนาดใหญ่
(ข) แสดงผลการตรวจจับรถผิดทิศทางที่มีเงาขนาดใหญ่ ทำให้ไม่สามารถตรวจสอบได้

4.3 สรุปผลการทดลองด้วยวิดีโอ

จากผลการทดลองการตรวจจับวัตถุที่เป็นรถนั้นจะเห็นได้ว่ารถที่มีขนาดใหญ่เกินกว่าพื้นที่ตรวจสอบนั้นจะไม่สามารถตรวจสอบและติดตามได้ และการปรับขนาด area ของ contour นั้นมีผลในการตรวจจับรถเนื่องมาจากการตรวจจับวัตถุต่างๆที่เคลื่อนที่อยู่ในเฟรมวีดีโอไม่ว่าจะเป็นรถยนต์ รถจักรยานยนต์ รถจักรยาน เงา ใบไม้ หรือวัตถุใดๆก็ตามที่มีการเคลื่อนที่และมีขนาดต่างกัน ผู้จัดทำได้ปรับขนาด area ของ contour ที่เหมาะสมนั่นคือ (boundingRect(contours[i]).area() > 1000) เพื่อที่จะลดสัญญาณรบกวน (noise) เหล่านั้นเพื่อทำให้โปรแกรมทำงานมีประสิทธิภาพมากขึ้น

บทที่5
สรุปผลการการดำเนินงานและแนวทางพัฒนา

ในบทนี้จะเป็นการกล่าวสรุปผลการทดลองและปัญหาที่ได้พบของโครงงานการตรวจจับการขับรถผิดทิศทางการเดินรถด้วยการประมวลผลภาพ (Wrong Traffic Direction Detection using Image Processing) และข้อเสนอแนะในการแก้ปัญหาเพื่อใช้ในการพัฒนาโครงงานต่อในอนาคต

5.1 สรุปผลการทดลอง

การตรวจจับการขับรถผิดทิศทางการเดินรถด้วยการประมวลผลภาพนั้น จะมีวิธีการทำงานโดยการจะตรวจสอบรถที่เข้ามาในพื้นที่ตรวจสอบที่เราสนใจ จากนั้นจะทำการแยกวัตถุที่เคลื่อนที่ออกจากพื้นหลัง โดยวัตถุที่เคลื่อนที่ในที่นี้ส่วนใหญ่จะเป็นรถ และจึงทำการวิเคราะห์ว่าการเคลื่อนที่ของรถว่ามีการเคลื่อนที่ไปในทิศทางใดผลที่ได้คือโปรแกรมสามารถระบุได้ว่ารถคันที่เราสนใจจะทำการตรวจสอบการเคลื่อนที่ถูกทิศทางหรือผิดทิศทางซึ่งจะระบุได้จากการเริ่มแรกรถเข้ามาในพื้นตรวจสอบจะทำการตีกรอบรอบรถด้วยกรอบสีเหลือง และถ้าหากมีการเคลื่อนที่ไปในทิศทางที่ถูกต้อง กรอบสีเหลืองรอบรถจะเปลี่ยนเป็นสีเขียว แต่ถ้าเคลื่อนที่ไปในทิศทางที่ไม่ถูกต้อง กรอบสีเหลืองจะเปลี่ยนเป็นสีแดง

จากผลการทดลองการตรวจจับการขับรถผิดทิศทางการเดินรถด้วยการประมวลผลภาพนั้น เราได้กำหนดค่าคอนทัวร์ (Contour) เพื่อกำหนดขนาดของวัตถุที่เราจะทำการตีกรอบล้อมรอบแล้วทำการติดตามเพื่อที่จะนำไปใช้ในการตรวจสอบ โดยค่าคอนทัวร์ที่กำหนดให้มีขนาดเล็กก็จะสามารถทำการติดตามวัตถุได้เกือบทั้งหมด แต่ถ้าหากปรับค่าให้มีขนาดใหญ่มากขึ้น ก็จะไม่สามารถทำการติดตามวัตถุที่มีขนาดเล็กได้ เพราะฉะนั้นเราจึงได้ทำการทดลองหาค่าที่เหมาะสม ที่จะนำมาใช้ในการติดตามวัตถุและทำการตรวจสอบการขับรถผิดทิศทางต่อไป

5.2 ปัญหาที่พบระหว่างการทำงาน

5.2.1 พื้นที่ที่ใช้ในการทดสอบรถที่ขับผิดทิศทางมีน้อย

5.2.2 มุมมองในการรับภาพ เริ่มต้นได้ทดลองใช้มุมด้านข้าง เพราะภาพในมุมด้านข้างวัตถุจะไม่มีการเปลี่ยนแปลงขนาดทำให้มีขนาดเท่าเดิม แต่มีความสูงไม่พอ ดังรูปที่ 5.1

รูปที่ 5.1 ภาพถ่ายมุมด้านข้าง

5.2.3 รถที่เข้าใกล้กันมาก จะทำให้ตรวจจับเป็นรถคันเดียวกัน

5.2.4 เงาของรถที่มีขนาดใหญ่ทำให้ตรวจจับเป็นรถที่มีขนาดใหญ่

5.2.5 รถคันใหญ่ที่เข้ามาเต็มพื้นที่ตรวจสอบไม่สามารถทำการตรวจสอบได้

5.3 แนวทางการแก้ไขและข้อเสนอแนะ

5.3.1 สำรวจและวางแผนการเลือกใช้พื้นที่เพื่อใช้ในการทดลอง

5.3.2 จากปัญหาการวางตำแหน่งกล้องมุมสูงในด้านข้างของถนนที่ไม่สามารถทำได้ ทำให้เราใช้วิธีการตั้งกล้องบนสะพานลอยและเปลี่ยนมุมของภาพเป็นด้านตรง

5.3.3 ใช้วิธีการอื่นในการติดตามรถ เช่น Optical flow, Kalman filter, Camshift

5.3.4 ใช้วิธีอื่นที่ทำการติดตามเฉพาะตัวรถ โดยไม่สนใจเงาที่เกิดขึ้นจากรถ

5.3.5 รถที่มีขนาดใหญ่อาจจะให้ตีกรอบใหญ่เป็นกรอบเดียว และเมื่อเข้ามาในพื้นที่ตรวจสอบ ก็ทำการระบุว่าเป็นรถคันใหญ่ 1 คัน

5.4 แนวทางการพัฒนาในอนาคต

5.4.1 ปรับปรุงให้การตรวจจับรถที่เข้ามาในพื้นที่การตรวจสอบมีความถูกต้องมากยิ่งขึ้นและลดสัญญาณรบกวนต่างๆให้น้อยลง

5.4.2 ปรับปรุงและแก้ไขให้โปรแกรมมีประสิทธิภาพมากขึ้น

5.4.3 พัฒนาเพื่อให้โปรแกรมสามารถใช้งานได้จริง

5.4.4 เพิ่มหน้าต่างโปรแกรม (Graphic User Interface) ให้สะดวกต่อการใช้งานมากยิ่งขึ้น

5.4.5 ประยุกต์ใช้ร่วมกับกล้อง CCTV

5.4.6 พัฒนาโปรแกรมให้สามารถตรวจจับเวลากลางคืนได้

เอกสารอ้างอิง

[1] Casey Hynes. (Feb 25, 2014). **Thailand's roads 2nd most dangerous in the world.** Retrieved August 20, 1992, from http://asiancorrespondent.com/119892/study-thailand-roads-2nd-most-dangerous-in-the-world/

[2] **Image Processing. เทคโนโลยีการประมวลผลภาพ.** สืบค้นเมื่อ 20 สิงหาคม 2557, จาก http://jaratcyberu.blogspot.com/2009/10/image-processing.html

[3] OpenCV คืออะไร. สืบค้นเมื่อ 20 สิงหาคม 2557, จาก https://th.answers.yahoo.com/question/index?qid=20090523084409AAgq3Rw

[4] RGB Color Model. สืบค้นเมื่อ 25 สิงหาคม 2557, จาก http://en.wikipedia.org/wiki/RGB_color_model

[5] HSL & HSV. สืบค้นเมื่อ 25 สิงหาคม 2557, จาก http://en.wikipedia.org/wiki/HSL_and_HSV

[6] Gray Scale and Color Image. สืบค้นเมื่อ 10 กันยายน 2557, จาก http://imageprocessingindelphi.blogspot.com/2008/08/rgb-to-gray-scale-conversion-using.html

[7] Binary Image. สืบค้นเมื่อ 10 กันยายน 2557, จาก google.co.th/books?id=ttNuj2mmt4sC&printsec=copyright&hl=th#v=onepage&q&f=false

[8] Morphological Operation. สืบค้นเมื่อ 15 กันยายน 2557, จาก www.msit.mut.ac.th/newweb/textword/download/research-proposal.doc

[9] Basic Thresholding Operations. สืบค้นเมื่อ 30 กันยายน 2557, จาก http://docs.opencv.org/doc/tutorials/imgproc/threshold/threshold.html

[10] Connected Component Labeling. สืบค้นเมื่อ 30 กันยายน 2557, จาก http://en.wikipedia.org/wiki/Connected-component_labeling

[11] ฐานิศร์ เฉลิมบุญ. (2550), **Motion Detection by Optical Flow,** ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะ วิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์

[12] ชลธิชา เวศโอสถ, นิคม สุวรรณวร. (2556), **การพัฒนาอัลกอริทึมเพื่อตรวจนับปริมาณรถบนถนนด้วยการประมวลผลภาพจากกล้องวิดีโอ.** วารสารมหาวิทยาลัยนราธิวาสราชนครินทร์, ปีที่ 5 ฉบับที่ 1, หน้าที่ 1-13, ISSN: 1906-5681.

[13] Raúl Ignacio Ramos García and Dule Shu. Vision based Vehicle Tracking using a high angle camera (2010) Monterrey Institute of Technology and Higher Education (ITESM)

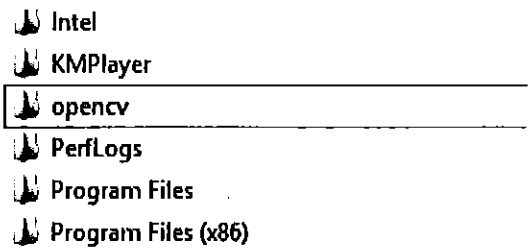
ภาคผนวก (ก)

การติดตั้งการใช้งาน Library Opencv

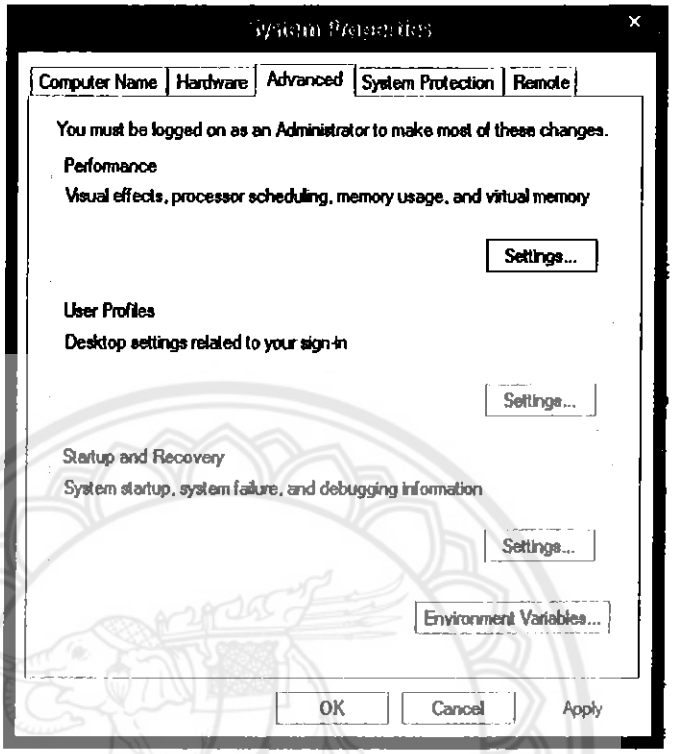
1. ติดตั้งโปรแกรม Microsoft Visual Studio 2012 ให้เรียบร้อยพร้อมใช้งาน
2. ติดตั้ง Library Opencv สามารถโหลดได้จาก http://opencv.org/downloads.html ให้เลือก version ที่ต้องการ ดังรูป

3. เลือกดาวน์โหลด version ของ opencv ที่ต้องการ ในที่นี้ทางผู้จัดทำได้เลือก version 3.0 BETA เลือก Opencv for Windows ดังรูป

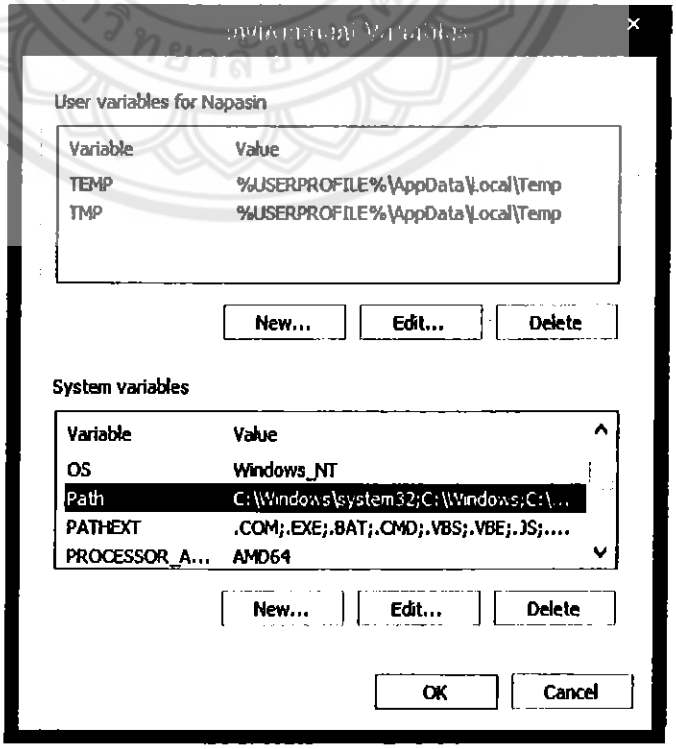
4. ทำการติดตั้งให้เรียบร้อย ที่ Drive C



5. ทำการ Set part โดย ไปที่ control panel > System and Security > System > Advanced system settings เลือก Environment Variable ดังรูป

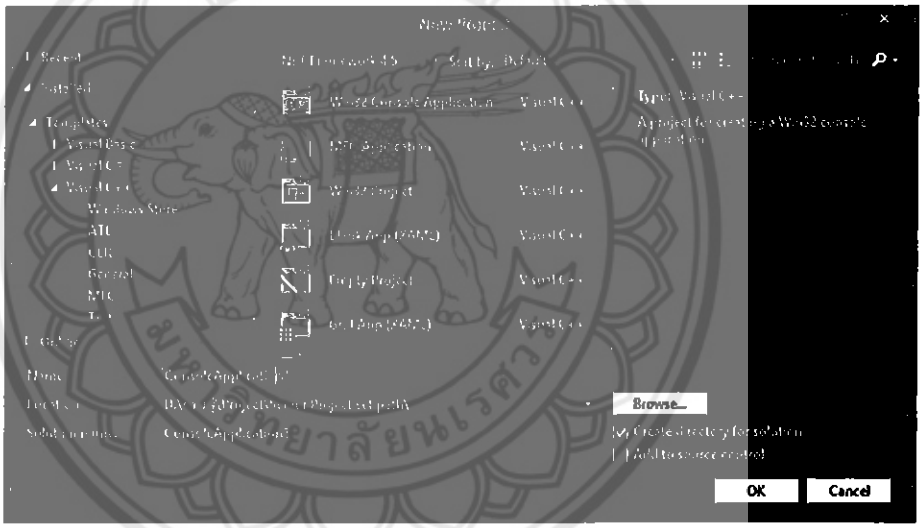


6. ที่ System variables ให้เลือก path จากนั้น กด Edit

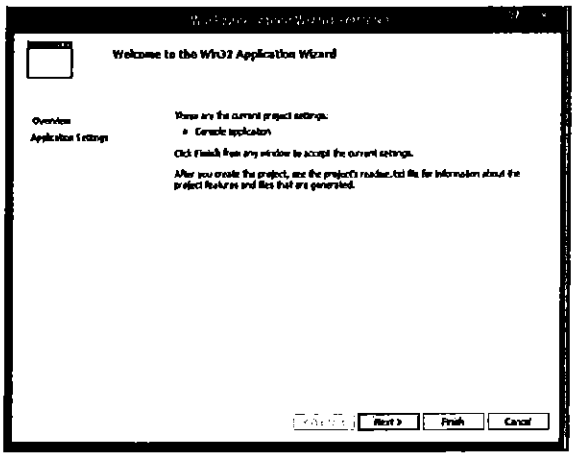


7. เพิ่ม path ของ opencv ดังนี้ C:\opencv\build\x86\vc11\bin; (ที่ขีดเส้นใต้ path ของ opencv ที่ทำการติดตั้งไว้ตอนแรก) จากนั้น กด ปุ่ม OK

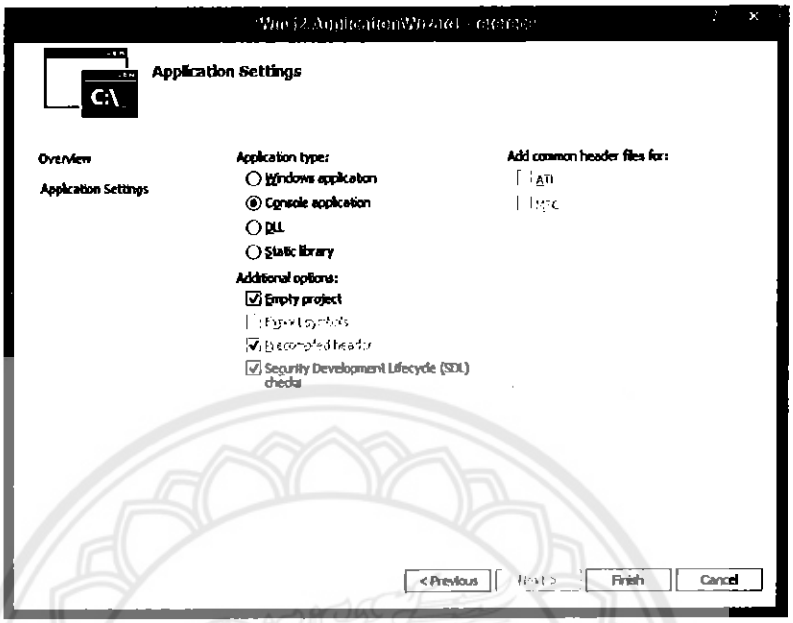
8. เปิดโปรแกรม Microsoft Visual Studio ทำการสร้าง New Project เลือก Visual C++ > Win32 Console Application > ตั้งชื่อ > เลือก Location กด ปุ่ม OK



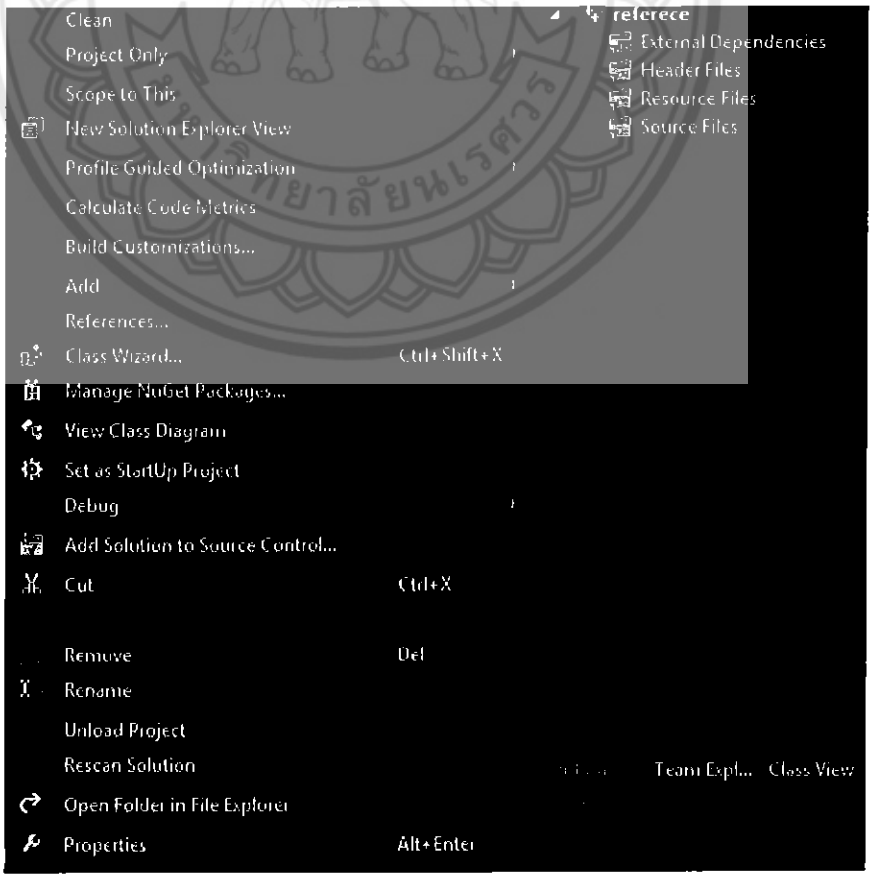
9. กดปุ่ม Next >



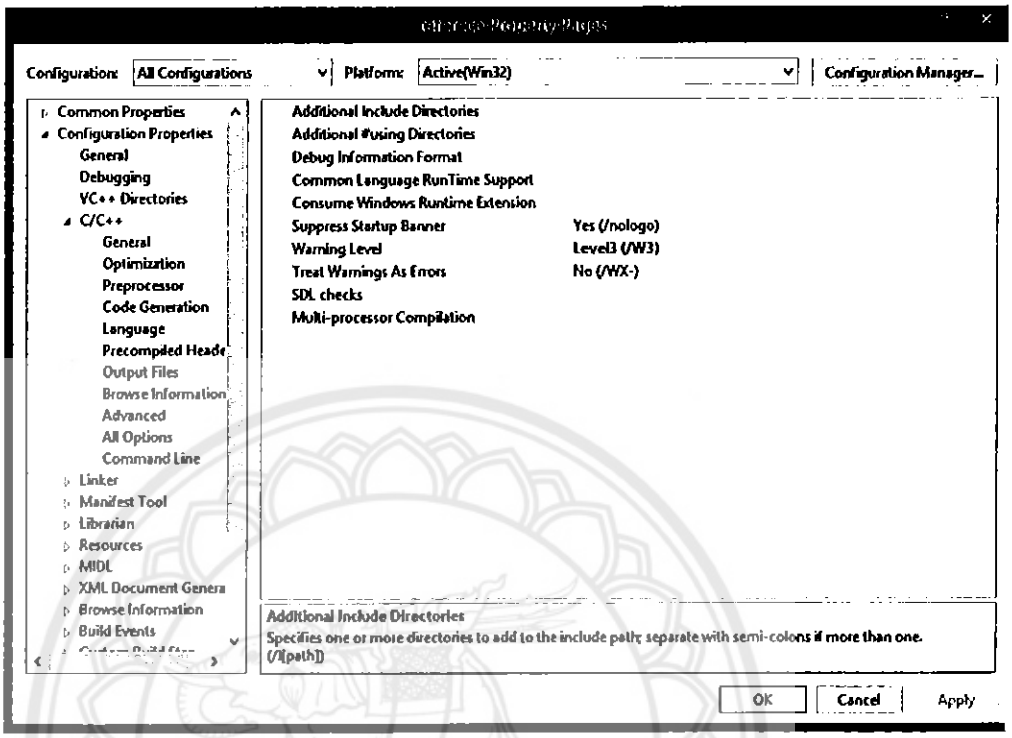
10. ที่ Application type ให้เลือก Console application และที่ Additional options ให้เลือก Empty project จากนั้นกดปุ่ม Finish



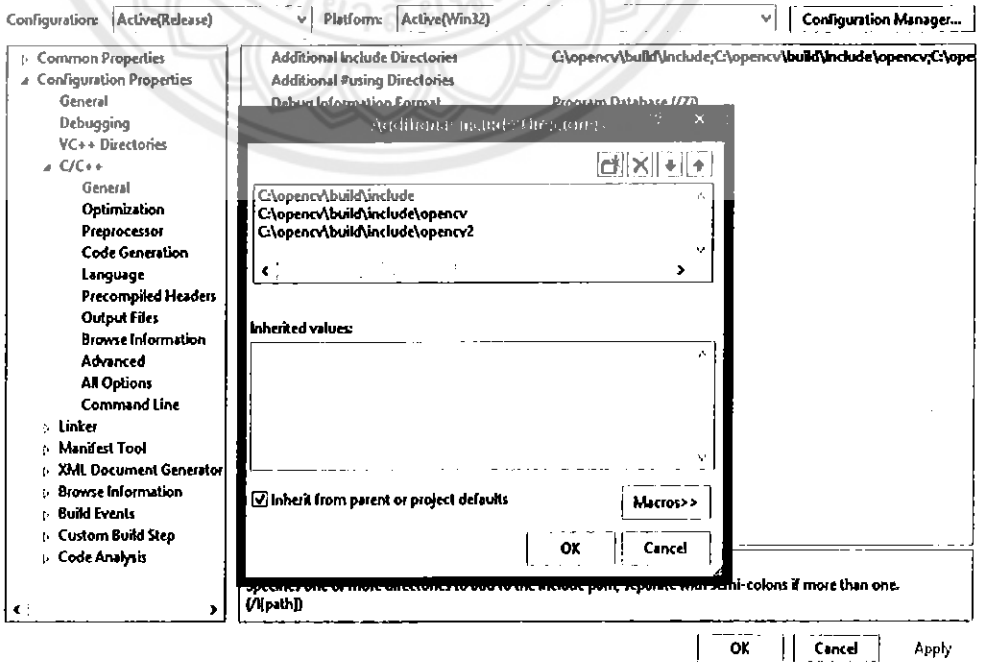
11. หน้าต่าง Solution Explorer ให้คลิกขวาที่ชื่อ project เลือก Properties



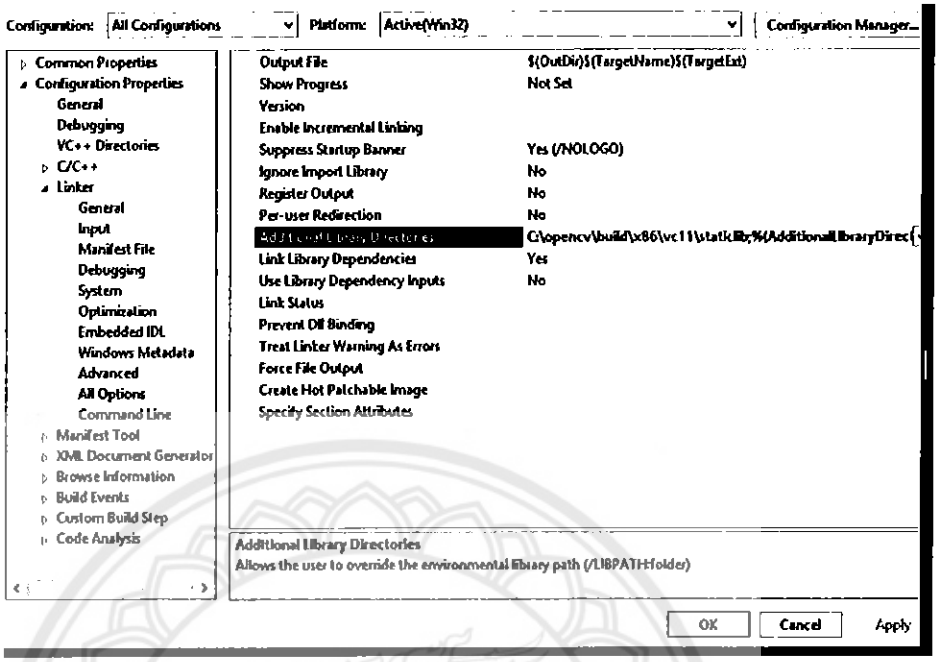
12. ที่ Configuration เลือก All Configurations จากนั้น เลือก C/C++ เลือก Additional Include Directories กด Edit



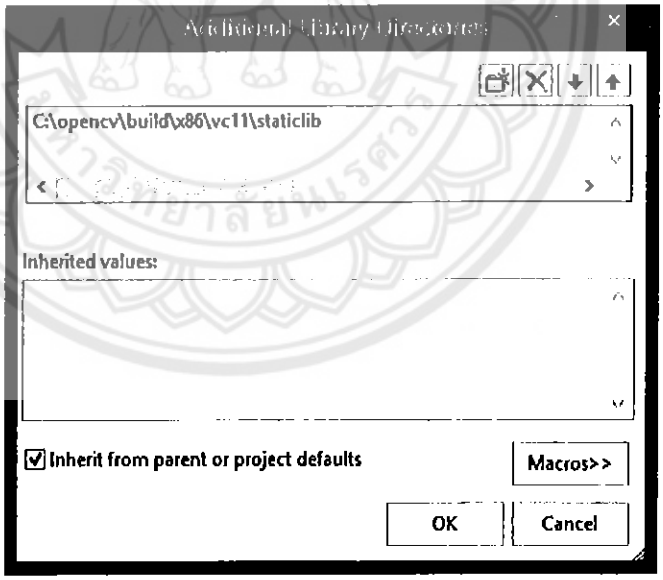
13. เพิ่มโฟลเดอร์ C:\opencv\build\include, C:\opencv\build\include\opencv, C:\opencv\build\include\opencv2 เข้ามาแล้วกด OK



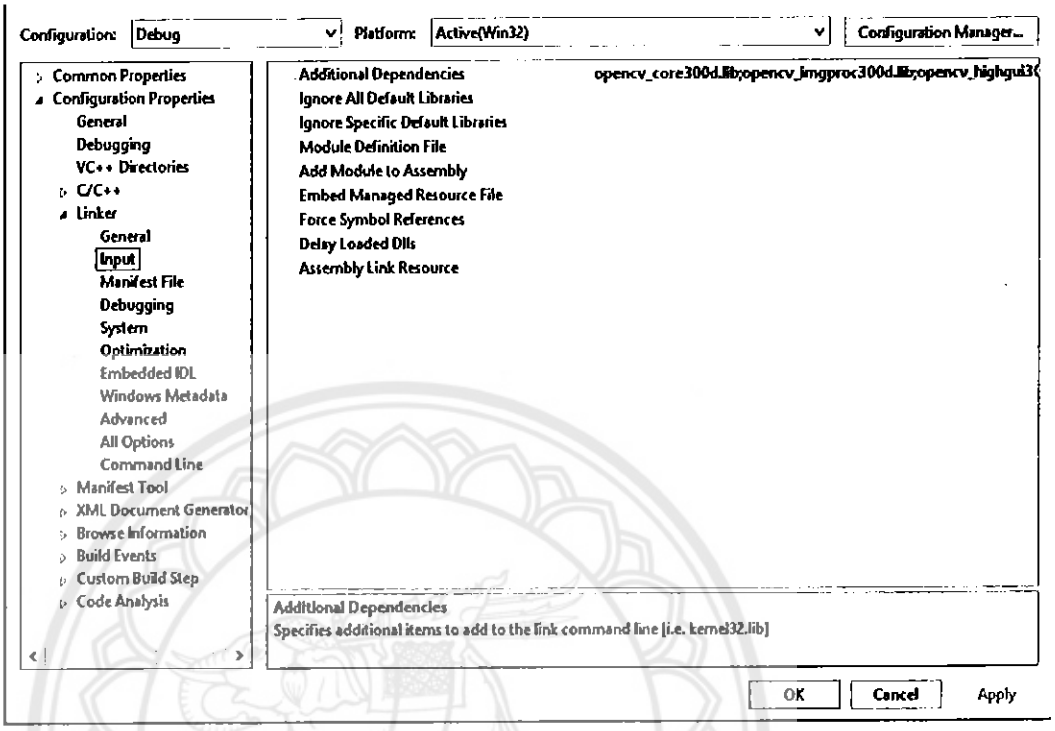
14. เลือก Linker > General > Additional Library Directories กดEdit



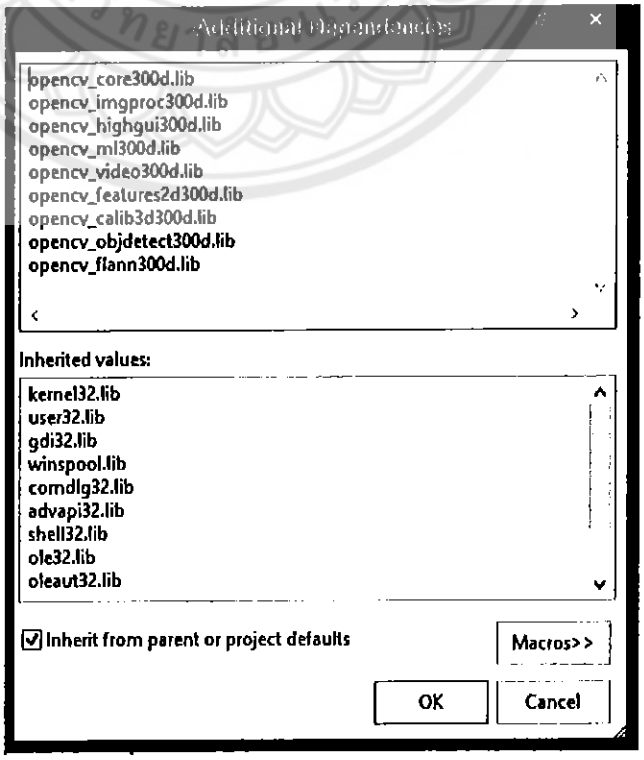
15. เพิ่มโฟลเดอร์ C:\opencv\build\x86\vc11\staticlib กด OK



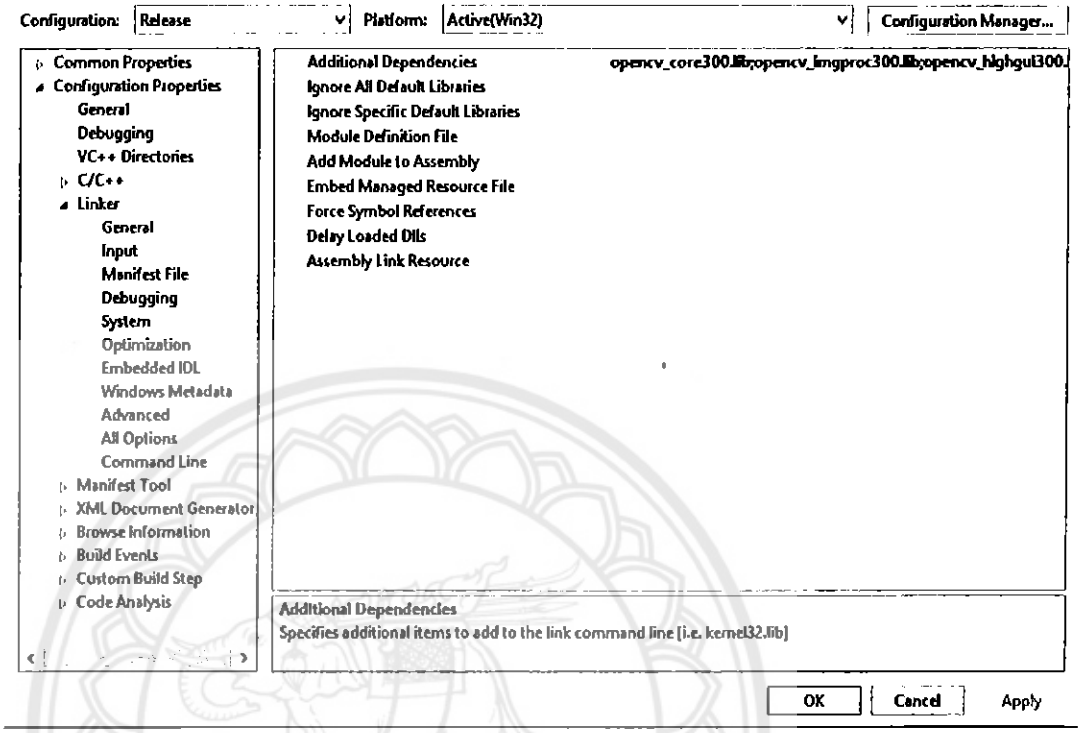
16. ที่ Configuration เลือก Debug เลือก Linker > Input > Additional Dependencies กด Edit



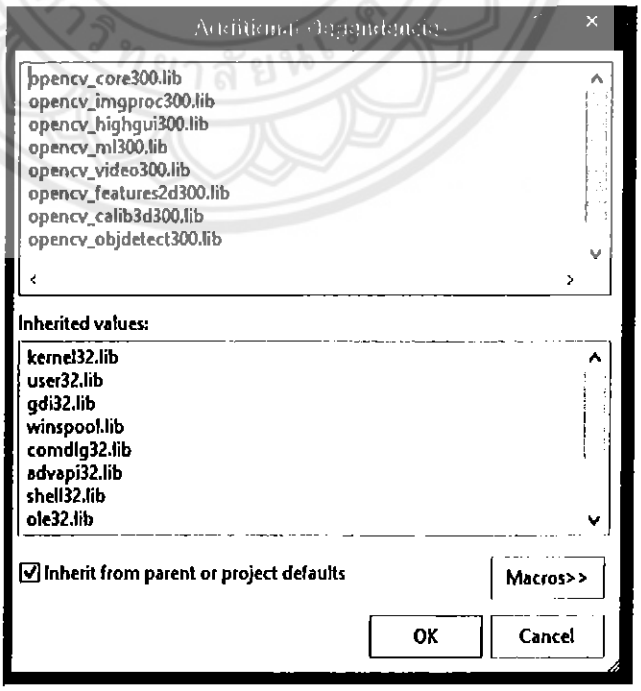
17. เพิ่ม .lib ดังรูป (opencv_core300d.lib) ที่ขีดเส้นใต้เป็นตัวเลข version ของ opencv ตามด้วย d คือ debug mode



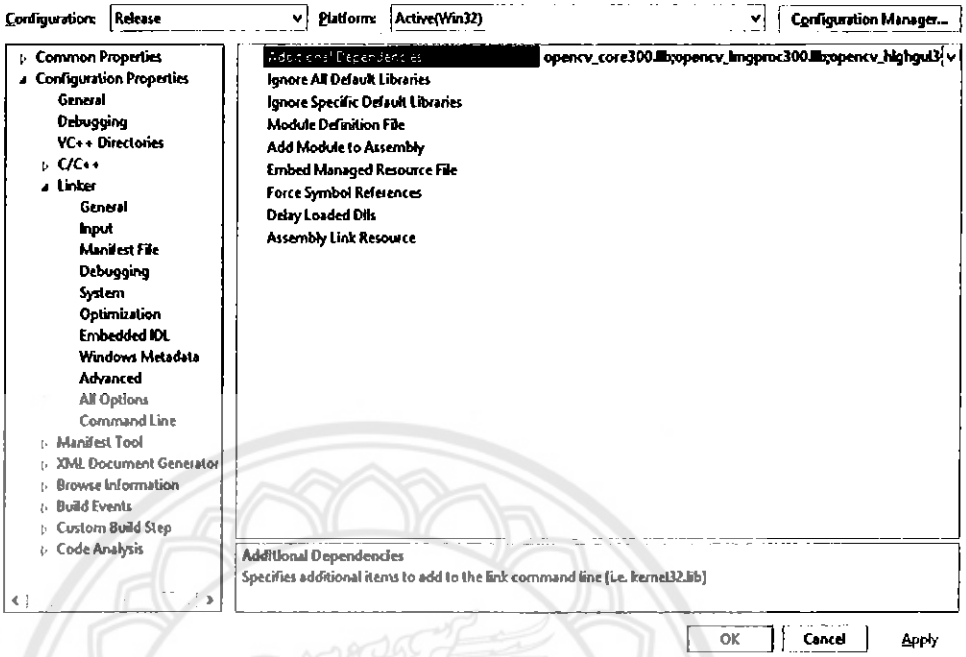
18. ที่ Configuration เลือก Release เลือก Linker > Input > Additional Dependencies กด Edit



19. เพิ่ม .lib ดังรูป (opencv_core300.lib) ที่ขีดเส้นใต้เป็นตัวเลข version ของ OpenCV

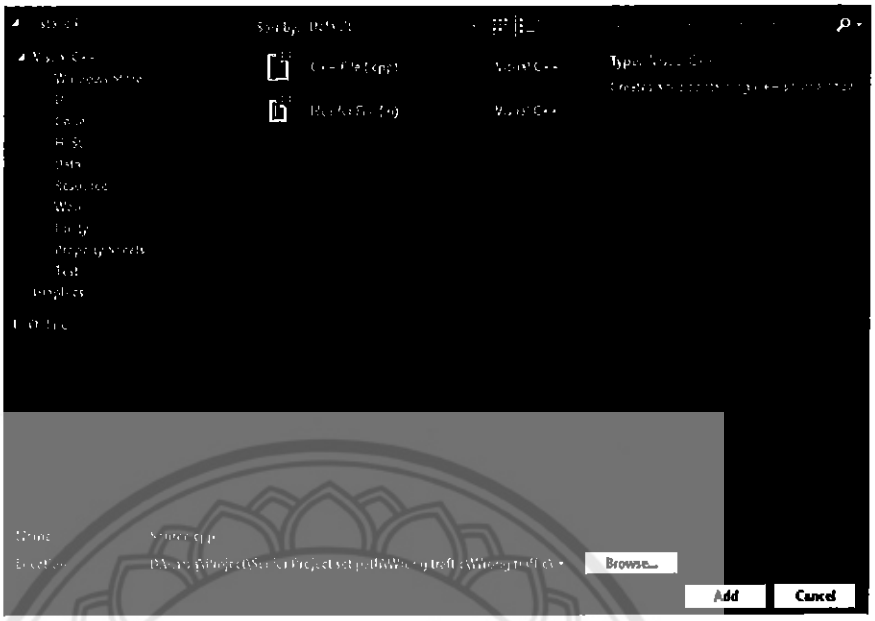


20. ตั้งแต่เสร็จเรียบร้อย กดปุ่ม OK



21. ที่หน้าต่าง Solution Explorer คลิกขวาที่ Source File > Add >New Item

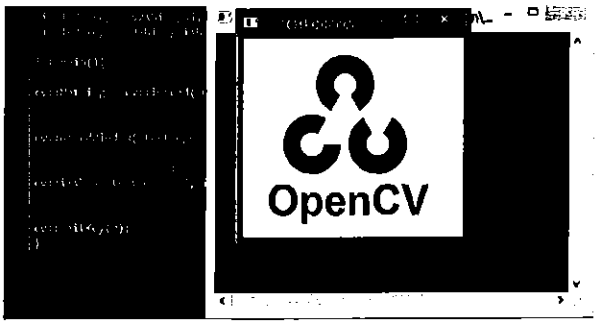
22. เลือก Visual C++ > C++ File (.cpp) > ตั้งชื่อ > Add



23. ทดลองใส่ Code เพื่อทดสอบการทำงาน

```
#include <opencv2\highgui.hpp>
 #include<opencv2\highgui\highgui.hpp>
int main(){
// Read image from file
cv::Mat img = cv::imread("C://opencv.png");
// Create a window
cv::namedWindow("test opencv");
// Display image in window
cv::imshow("test opencv", img);
// Wait for user to press a key in window
cv::waitKey(0);
}
```

เมื่อใส่Code เสร็จ ผลการรันจะออกมาดังภาพ แสดงว่าทำการติดตั้ง opencv สำเร็จ



ภาคผนวก (ข)
โค้ดของโปรแกรม

```
#include <opencv2\highgui\highgui.hpp>
#include <opencv2\imgproc\imgproc.hpp>
#include <iostream>
#include <sstream>
#include <math.h>
#include <opencv\cv.h>
#include <opencv\highgui.h>
#include <opencv2\objdetect\objdetect.hpp>
#include <opencv2/opencv.hpp>
#include <vector>

// กำหนดที่อยู่ Video หรือ ใช้กล้อง
#define VIDEO "C://test//3.mp4"
//#define VIDEO 1
// กำหนดค่าสี
#define red Scalar(0,0,255)
#define green Scalar(0,255,0)
#define blue Scalar(255,0,0)
#define white Scalar(255,255,255)
#define yellow Scalar(0,255,255)

using namespace std;
using namespace cv;

// เก็บสถานะรถ
struct checkCar{
        bool use;                       //รถที่ใช้ตรวจสอบอยู่
        bool line1LD;                   //ผ่านเส้น1
        bool line2LT;                   //ผ่านเส้น2
        bool wrong;                     //เช็คสถานะผิดหรือถูก
        int neverFound;                         //ถ้าไม่เจอรถกี่เฟรม ถ้าเกินก็ลบทิ้ง
        int indexContour;               //รถอยู่คอนทัวร์ที่เท่าไร
        Point source;                   //จุดเริ่มต้นที่เจอรถ
        Point destination;              //จุดที่ปัจจุบัน
};
```

```
// วนหารถคันใหม่
void loopCar(checkCar cCar[])
{
	for (int i = 0; i < 20; i++)
	{
		if (cCar[i].use == true)
		{
			cCar[i].neverFound++;
		}
	}
}

// กำหนดค่าเริ่มต้นของรถทั้งหมด
void initialcCar(checkCar cCar[])
{
	for (int i = 0; i < 20; i++)
	{
		cCar[i].use = false;
		cCar[i].line1LD = false;
		cCar[i].line2LT = false;
		cCar[i].wrong = true;
	}
}

// กำหนดสถานะเริ่มต้นของรถ
void createNewCar(checkCar cCar[],Point newPoint,int indexContour)
{
	bool checked = false;
	for (int i = 0; i < 20; i++)
	{
		if (cCar[i].use == false)
		{
			cCar[i].source.x = cCar[i].destination.x = newPoint.x;
			cCar[i].source.y = cCar[i].destination.y = newPoint.y;
			cCar[i].use = true;
			checked = true;
			cCar[i].neverFound = 0;
			cCar[i].indexContour = indexContour;
```

```
                return;
            }

            if (!checked)
            {
                cCar[i].neverFound++;
            }
        }
    }

// ลบรถออกจากพื้นที่ตรวจสอบ
void deleteCar(checkCar cCar[])
{
    for (int i = 0; i < 20; i++)
    {
        if (cCar[i].source.y <= 10 && cCar[i].destination.y >= 178 )
        {
            cCar[i].use = false;
            cCar[i].line1LD = false;
            cCar[i].line2LT = false;
        }
        else if (cCar[i].source.y >= 170 && cCar[i].destination.y <= 1)
        {
            cCar[i].use = false;
            cCar[i].line1LD = false;
            cCar[i].line2LT = false;
        }
    }
}

// ตรวจสอบรถคันเก่า
bool checkOldCar(checkCar cCar[],Point nPoint,int indexContour)
{
    for (int i = 0; i < 20; i++)
    {
        if (cCar[i].use)
        {
```

```
                        if (nPoint.x >= cCar[i].destination.x - 30 && nPoint.x <=
                        cCar[i].destination.x + 30 && nPoint.y >= cCar[i].destination.y -
                        30
                         && nPoint.y <= cCar[i].destination.y + 30)
                        {
                                cCar[i].destination.x = nPoint.x;
                                cCar[i].destination.y = nPoint.y;
                                cCar[i].indexContour = indexContour;
                                return true;
                        }
                }
        }
        return false;
}

// แปลงจำนวนเต็มเป็นตัวอักษร
 string intToString(int number){
        std::stringstream ss;
        ss << number;
        return ss.str();
}
cv::Point init;

// Main ของโปรแกรม
int main(){
VideoCapture cap = VideoCapture(VIDEO);
//VideoCapture cap = VideoCapture(0);
Mat cameraFrame,cameraFrame1, bgsub, bgsub2;
 //Ptr< BackgroundSubtractor> pMOG; //MOG
Ptr< BackgroundSubtractor> pMOG2; //MOG2
Ptr<BackgroundSubtractorKNN> pKNN; //KNN
//pMOG = createBackgroundSubtractorMOG2();
pMOG2= createBackgroundSubtractorMOG2();
pKNN = createBackgroundSubtractorKNN();
bool isInit = true;
int count = 0;
int oldCount =0;
vector<cv::Point> init(100);
```

```
checkCar cCar[10];
initialcCar(cCar);

// Loop video กด q เพื่อหยุดทำงาน
while (waitKey(15) != 'q')
{
        cap >> cameraFrame1;
        if ((cameraFrame1.rows != 0)) //เช็คว่า video หมดหรือยัง
        {
                //imshow("input",cameraFrame1);
                Size s = cv::Size(500, 300);
                resize(cameraFrame1, cameraFrame, s);
                // Show input
                imshow("(1)Video Input", cameraFrame);
                Mat temp = cameraFrame(Rect(100,100,320,200)).clone();
                //Mat tmp = binaryImg(Rect(125,90,270,200)).clone();

                Mat temp2 = temp.clone();
                //imshow("area interest input", temp);
                Mat resize_blur_img;
                Mat binaryImg;
                Mat ContourImg;

                // resize  เช่น Size(c.s.w/2,c.s.h/3)
                resize(cameraFrame, resize_blur_img, Size(cameraFrame.size().width,
                cameraFrame.size().height) );

                // Blur
                blur(resize_blur_img, resize_blur_img, Size(5,5) );
                //imshow("(2)Blur", resize_blur_img);

                // Video to gray scale
                Mat image_gray;
                cvtColor(resize_blur_img, image_gray, CV_BGR2GRAY);
                //imshow("(3)GrayScale",image_gray);

                // Background subtraction
                pMOG2->apply(image_gray,bgsub,-1);
```

```
//imshow("(4)BackGround Subtraction",bgsub);

// threshold
threshold(bgsub, binaryImg, 100, 255, THRESH_BINARY);
//imshow("(5)threshold binaryImg", binaryImg);

// Morphological
Mat morp;
Mat eroded;
Mat element2(3,3,CV_8U,cv::Scalar(1));
erode(binaryImg,eroded,element2);
//imshow("emorphological", eroded);

Mat dilated;
Mat element3(8,8,CV_8U,cv::Scalar(1));
dilate(eroded,morp,element3);

//Mat opened;
//Mat element4(2,2,CV_8U,cv::Scalar(1));
//morphologyEx(binaryImg,opened,cv::MORPH_OPEN,element4);
//Mat closed;
//Mat element5(8,8,CV_8U,cv::Scalar(1));
//morphologyEx(opened,morp,cv::MORPH_OPEN,element5);
//imshow("morphological", morp);

// กำหนดพื้นที่ตรวจสอบ
ContourImg = morp.clone();
Mat tmp = morp(Rect(100, 100, 320, 200)).clone();
//Mat tmp = binaryImg(Rect(125,90,270,200)).clone();
Mat tmp2 = tmp.clone();
//imshow("Binary IMG interest", tmp2);
// วาดเส้นพื้นที่ตรวจสอบ (สีน้ำเงิน)
//cv::line(cameraFrame, cv::Point(250, 1), cv::Point(250, 299), white, 1, 1);
cv::line(cameraFrame, cv::Point(270, 100), cv::Point(270, 280), blue, 4, 4);//แนวตั้ง
cv::line(cameraFrame, cv::Point(180, 100), cv::Point(360, 100), blue, 2, 4);//บน
cv::line(cameraFrame, cv::Point(100, 280), cv::Point(440, 280), blue, 2, 4);//ล่าง
cv::line(cameraFrame, cv::Point(180, 100), cv::Point(100, 280), blue, 2, 4);//ซ้าย
cv::line(cameraFrame, cv::Point(360, 100), cv::Point(440, 280), blue, 2, 4);//ขวา
```

```
// หา connected component labeling
vector< vector< Point> > contours;
findContours(tmp,
        contours,
        RETR_EXTERNAL,
        CHAIN_APPROX_NONE);
vector<vector<cv::Point> > contours_poly( contours.size() );
vector<Point2f>center( contours.size() );
vector<float>radius( contours.size() );
vector<Rect>  boundRect(contours.size());
vector<cv::Moments> mu(contours.size() );
vector<cv::Point2f> mc(contours.size() );

bool passLine1LD = 0,passLine2LT = 0;
int count = 0;
//cout << contours.size() << endl;
//imshow("con",tmp);
loopCar(cCar);

for( int i = 0; i < contours.size(); i++ )
{
        mu[i] = moments( contours[i], true );
        mc[i] = cv::Point2f( mu[i].m10/mu[i].m00 , mu[i].m01/mu[i].m00 );
        approxPolyDP( Mat(contours[i]), contours_poly[i], 3, true );
        boundRect[i] = boundingRect( Mat(contours_poly[i]) );
        minEnclosingCircle( (Mat)contours_poly[i], center[i], radius[i] );
        int x = mc[i].x;
        int y = mc[i].y;
        //cout <<contours.size() << ": " << i << ": " <<
        boundingRect(contours[i]).area() << endl;

        approxPolyDP(Mat(contours[i]), contours_poly[i], 3, true);
        boundRect[i] = boundingRect(Mat(contours_poly[i]));
        rectangle(tmp2, boundRect[i], white, 2, 8, 0);
        //imshow("binaryImg clone",tmp2);
```

```
                    //putText(cameraFrame,".",Point(boundRect[i].x +
                    boundRect[i].width/2 + 100, boundRect[i].y +
                    boundRect[i].height/2 + 100),1,1,red,3);
                    //putText(tmp,".",Point(boundRect[i].x + boundRect[i].width/2,
                    boundRect[i].y + boundRect[i].height/2),1,1,white,3);

if (!checkOldCar(cCar, Point(boundRect[i].x + boundRect[i].width/2, boundRect[i].y +
boundRect[i].height/2),i))
        {
            if(boundRect[i].y + boundRect[i].height/2 > 5 && boundRect[i].y +
            boundRect[i].height/2 < 175)
        // ปรับขนาดพื้นที่ของ contours
        if ( boundingRect(contours[i]).area() > 1000){
        cout << boundingRect(contours[i]).area() ;
        createNewCar(cCar, Point(boundRect[i].x + boundRect[i].width/2,
        boundRect[i].y + boundRect[i].height/2),i);
            }
        }
        cout << boundingRect(contours[i]).area() ;
        }//end for
        deleteCar(cCar);
        for (int j = 0; j < 5; j++)
        {
        if(contours.size() > 0 ) //contour.size คือ มีจำนวนเท่าไร
            if (cCar[j].use == true)
            {
            //cv::line(tmp, cCar[j].source, cCar[j].destination, white, 1, CV_AA);
            //cv::line(cameraFrame,
            Point(cCar[j].source.x+100,cCar[j].source.y+100),
            Point(cCar[j].destination.x+100,cCar[j].destination.y+100),white, 1,
            CV_AA);
// ตรวจสอบการเคลื่อนที่ของรถ
if(cCar[j].destination.y+100 <= 205 && cCar[j].destination.y+100 >= 195) // 195 > y >
205
        {
        passLine1LD = 1;
        cCar[j].line1LD = true;
        }
```

```
if(cCar[j].destination.y+100 <= 155 && cCar[j].destination.y+100 >= 145) // 145 > y >
155
        {
        passLine2LT = 1;
        cCar[j].line2LT = true;
        }
if(cCar[j].source.x <= 160 && !cCar[j].line1LD && cCar[j].line2LT)
        {
        cCar[j].wrong = false;
        }
if(cCar[j].source.x >= 185 && cCar[j].line1LD && !cCar[j].line2LT)
        {
        cCar[j].wrong = false;
        }
}
}
// ตีกรอบสี่เหลี่ยมรอบรถ
for (int j = 0; j < 5; j++)
{
cout << j << ":" <<cCar[j].use <<"  S:"<< cCar[j].source   <<"  D:" << cCar[j].destination
<< " down: " << cCar[j].line1LD <<" up: "<< cCar[j].line2LT<< endl;
//putText(cameraFrame,intToString(cCar[j].wrong),
Point(cCar[j].destination.x+100,cCar[j].destination.y+100),1,1,white,3);
        if(cCar[j].use)
        {
        if(contours.size() > 0)
        {
                if(!cCar[j].wrong) // ตีสี่เหลี่ยมรอบรถผิด (กรอบสีแดง)
                {
                //putText(cameraFrame,"Wrong",
                Point(cCar[j].destination.x+100,cCar[j].destination.y+100),1,1,Scalar(0,0,2
                55),3);
                rectangle(cameraFrame,Point(boundRect[cCar[j].indexContour].x + 100,
                boundRect[cCar[j].indexContour].y100),
                Point(boundRect[cCar[j].indexContour].x + 100 +
                boundRect[cCar[j].indexContour].width,
                boundRect[cCar[j].indexContour].y + 100 +
                boundRect[cCar[j].indexContour].height),red, 2, 8, 0);
```

```
                }
        else if(cCar[j].line1LD && cCar[j].line2LT ) // ตีสี่เหลี่ยมรอบรถถูก (กรอบสีเขียว)
                {
                //putText(cameraFrame,"True",
                Point(cCar[j].destination.x+100,cCar[j].destination.y+100),1,1,Scalar(0,25
                5,0),3);
                rectangle(cameraFrame, Point(boundRect[cCar[j].indexContour].x +
                100, boundRect[cCar[j].indexContour].y + 100),
                Point(boundRect[cCar[j].indexContour].x + 100 +
                boundRect[cCar[j].indexContour].width,
                boundRect[cCar[j].indexContour].y + 100 +
                boundRect[cCar[j].indexContour].height),green, 2, 8, 0);
                }
        else
                {// ตีสี่เหลี่ยมรอบรถที่กำลังพิจารณา (กรอบสีเหลือง)
                rectangle(cameraFrame, Point(boundRect[cCar[j].indexContour].x +
                100, boundRect[cCar[j].indexContour].y + 100),

                Point(boundRect[cCar[j].indexContour].x + 100 +
                boundRect[cCar[j].indexContour].width,
                boundRect[cCar[j].indexContour].y + 100 +
                boundRect[cCar[j].indexContour].height),      yellow, 2, 8, 0);
                }
        }
    }
}

// เส้นตรวจสอบ (เส้นสีแดง)
        if(passLine1LD)
        cv::line(cameraFrame, cv::Point(140,200), cv::Point(400, 200), green, 3, CV_AA);
//เส้นล่าง
        else
        cv::line(cameraFrame, cv::Point(140,200), cv::Point(400, 200), red, 3, CV_AA);
//เส้นล่าง
        if(passLine2LT)
        cv::line(cameraFrame, cv::Point(160,150), cv::Point(380, 150), green, 3, CV_AA);
//เส้นบน
        else
```

```
            cv::line(cameraFrame, cv::Point(160,150), cv::Point(380, 150), red, 3, CV_AA);
              //เส้นบน
            if (contours.size() == 0)
            {
                    initialcCar(cCar);
            }
            //imshow("Contour",tmp);
            //imshow("(6)contour",ContourImg);
            //waitKey(20);
            imshow("(7) Output",cameraFrame);

//int key = cv::waitKey(33);
//if (key == 0x1b) break;
//while (key != ' ')
//{
//        key = cv::waitKey(33);
//}
}else{//video หมด ให้ใส่ใหม่
cap = VideoCapture(VIDEO);
 }
}//end waitkey(15)
}//end main
```