ระบบอัตโนมัติสำหรับจัดการที่จอดรถ

An Automatic System for Car Parking

โดย

นายประสงค์สุข อนุวงศ์

นายปรีชา วชิรภราดร

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมสารสนเทศ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2549

AN AUTOMATIC SYSTEM FOR CAR PARKING

BY

MR. PRASONGSUK ANUWONG

MR.PRECHA WACHIRAPARADON

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENT FOR THE DEGREE OF

BACHELOR IN DEPARTMENT OF INFORMATION ENGINEERING

FACULTY OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2006

หัวข้อปริญญานิพนธ์	ระบบอัตโนมัติสำหรับจัดการที่จอดรถ
นักศึกษา	นายประสงค์สุข อนุวงศ์ รหัสนักศึกษา 46010417
	นายปรีชา วชิรภราดร รหัสนักศึกษา 46010432
อาจารย์ที่ปรึกษา	ดร.พิทักษ์ ธรรมวาริน
ระดับการศึกษา	ปริญญาตรี วิศวกรรมศาสตรบัณฑิต
	สาขาวิศวกรรมสารสนเทศ
ภาควิชา	วิศวกรรมสารสนเทศ
ปีการศึกษา	2549

ปริญญานิพนธ์ฉบับนี้ได้รับการอนุมัติเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต
คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

.....

(ดร.พิทักษ์ ธรรมวาริน)
อาจารย์ที่ปรึกษา

หัวข้อปริญญานิพนธ์ ระบบอัตโนมัติสำหรับจัดการที่จอดรถ

ชื่อนักศึกษา นายประสงค์สุข อนุวงศ์ รหัสนักศึกษา 46010417

นายปรีชา วชิรภราดร รหัสนักศึกษา 46010432

อาจารย์ที่ปรึกษา ดร. พิทักษ์ ธรรมวาริน

ระดับการศึกษา ปริญญาตรี วิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมสารสนเทศ

ภาควิชา วิศวกรรมสารสนเทศ

ปีการศึกษา 2549

บทคัดย่อ

ในปริญญานิพนธ์ฉบับนี้ได้นำเสนอ ระบบอัตโนมัติสำหรับค้นหาตำแหน่งว่างสำหรับที่จอดรถ การค้นหาตำแหน่งว่างสำหรับที่จอดรถกระทำได้โดย ใช้ภาพถ่ายสถานที่จอดรถมาทำการตรวจสอบด้วยโปรแกรมคอมพิวเตอร์ จากนั้นแสดงตำแหน่งที่จอดรถบนหน้าจอคอมพิวเตอร์สำหรับให้ผู้ใช้ตรวจสอบก่อนนำรถเข้าไปจอดในที่จอดรถ นอกจากนั้นในระบบที่ได้พัฒนายังมีระบบคิดค่าจอดรถ ระบบจองตำแหน่งจอดรถเพื่อให้การจัดสรรที่จอดรถ เป็นไปอย่างมีประสิทธิภาพ โดยประสิทธิภาพของระบบสามารถแสดงได้โดยนำระบบที่ได้พัฒนามา ทดลองใช้กับที่จอดรถแบบจำลองที่ได้สร้างขึ้นมา และแสดงผลไว้ในส่วนผลการทดลอง

Thesis Title An Automatic System for Car Parking.

Student Mr. Prasongsuk Anuwong ID. 46010417

Mr. Precha Wachiraparadon ID. 46010432

Advisor Dr. Pitak Thummawarin

Graduate Level Bachelor Degree of Information Engineering

Department Information Engineering

Academic Year 2006

ABSTRACT

This project presents an automatic system for car parking. In the proposed system, the available parking lot is searched by computer using an image obtained from digital camera. Then the position of the available spaces are shown to the user before entering the parking place. Moreover, the system also provide the parking fee and booking the parking lot. The simulation of the car parking system are given to show the effectiveness of the proposed system.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงได้อย่างดี ด้วยคำแนะนำและคำปรึกษา จากอาจารย์พิทักษ์ ธรรมวาริน ซึ่งเป็นอาจารย์ที่ปรึกษา ข้าพเจ้ารู้สึกซาบซึ้งใจ ในความอนุเคราะห์จากท่านอาจารย์ และขอขอบพระคุณเป็นอย่างสูง

ขอกราบขอบพระคุณคณาจารย์ภาควิชาวิศวกรรมสารสนเทศ และคณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมกล้าเจ้าคุณทหารลาดกระบัง ทุกๆ ท่านที่ได้ประสิทธิ์ประสาทวิชาให้กับข้าพเจ้า ขอขอบคุณเพื่อนๆ พี่ๆ น้องๆ ในภาควิชาวิศวกรรมสารสนเทศ ทุกคนที่ให้กำลังใจ และคอยให้คำแนะนำต่างๆเสมอมา

สุดท้ายนี้ข้าพเจ้าขอกราบขอบพระคุณ บิดา มารดา และครอบครัวของข้าพเจ้าที่เป็นกำลังใจและได้การสนับสนุนในทุกเรื่องๆ ทำให้ข้าพเจ้าสามารถทำปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงด้วยดี คุณค่าและประโยชน์อันพึงมาจากปริญญานิพนธ์ฉบับนี้ ข้าพเจ้าขอมอบแด่ผู้มีพระคุณทุกท่าน

นายประสงค์สุข อนุวงศ์
นายปรีชา วชิรภารดร
8 กุมภาพันธ์ 2550

สารบัญ

สารบัญ(ต่อ)

สารบัญรูปภาพ

เรื่อง **หน้า**

สารบัญรูปภาพ(ต่อ)

เรื่อง **หน้า**

บทที่1
บทนำ

1.1 ที่มาของโครงงาน

ในปัจจุบันปัญหาการจัดการที่จอดรถนับว่าเป็นปัญหาที่พบได้ทั่วไป เพราะว่าสถานที่จอดรถในปัจจุบันมีขนาดใหญ่และซับซ้อน เพื่อให้การจัดการที่จอดรถเป็นไปอย่างมีประสิทธิภาพ จึงจำเป็นต้องมีการพัฒนาระบบช่วยในการค้นหาที่จอดรถ

จึงได้มีแนวความคิดที่จะทำเทคโนโลยี มาประยุกต์เพื่อแก้ปัญหาดังกล่าวโดยอาศัยทฤษฎี Image Processing และโปรแกรม Visual Basic โดยนำทฤษฎี Image Processing มาตรวจสอบเปรียบเทียบภาพจากกล้องที่ติดตั้ง แล้วประมวลผลโดยโปรแกรม Visual Basic แสดงผลออกทางหน้าจอมอนิเตอร์ อีกทั้งในโครงงานยังมีระบบคิดค่าจอดรถ ระบบจองตำแหน่งจอดรถเพื่อป้องกันการแย่งกันจอดรถการเก็บสถิติการจอดรถเพื่อนำไปวิเคราะห์ในส่วนอื่นๆของสำนักอาคารที่จอดรถ

1.2 วัตถุประสงค์

- เพื่อตรวจสอบตำแหน่งว่ามีรถจอดอยู่หรือไม่ โดยวิเคราะห์จากภาพ แล้วแสดงผลทางมอนิเตอร์
- เพื่อศึกษาการวิเคราะห์และการออกแบบระบบ
- เพื่อศึกษาการเขียนโปรแกรม Application โดยอาศัยโปรแกรม Visual Basic และโปรแกรมประยุกต์อื่นๆ
- เพื่อลดต้นทุนจากการใช้เซนเซอร์ โดยใช้การวิเคราะห์จากภาพแทน
- สามารถนำความรู้ที่ได้ศึกษามา ก่อประโยชน์สูงสุด

1.3 ขอบเขตของโครงงาน

- ระบบสามารถตรวจหาตำแหน่งที่จอดรถได้
- ระบบสามารถจองตำแหน่งเพื่อป้องกันการเลือกที่จอดรถซ้ำซ้อน
- ระบบสามารถคำนวณค่าจอดรถ
- ระบบสามารถเก็บสถิติการจอดรถในแต่ละวันได้

1.4 วิธีการดำเนินงาน

- ศึกษาระบบจอดรถที่มีอยู่ในปัจจุบัน นำมาวิเคราะห์ปัญหาของระบบเดิม แล้วแก้ปัญหาระบบเดิมให้มีประสิทธิภาพยิ่งขึ้น
- ศึกษาทฤษฎี Image Processing เช่น การได้มาของภาพ การปรับปรุงภาพ เป็นต้น และศึกษาโปรแกรม Visual Basic ซึ่งเป็นโปรแกรมที่ได้รับความนิยม และง่ายต่อการทำความเข้าใจ และโปรแกรมประยุกต์อื่นๆที่ใช้งานกับภาพ
- ลงมือเขียนโปรแกรม ทดลอง และหาปัญหาที่เกิดจากการทดลอง พร้อมทั้งแก้ปัญหาและปรับปรุงให้โปรแกรมมีประสิทธิภาพมากยิ่งขึ้น

1.5 ประโยชน์ที่คาดว่าจะได้รับ

- เพื่อช่วยหาที่จอดรถได้เร็วขึ้น ทำให้ประหยัดเวลา และประหยัดเชื้อเพลิง
- ได้ความรู้เกี่ยวกับเรื่อง Image Processing
- ได้รับความรู้ในการเขียนโปรแกรมด้วยภาษา Visual Basic
- ได้รับความรู้ในการวิเคราะห์ระบบและออกแบบฐานข้อมูล
- ทำให้ระบบเดิมมีประสิทธิภาพมากยิ่งขึ้น
- ผู้ศึกษาได้รับความรู้ แนวความคิด และทักษะการเขียนโปรแกรม

บทที่ 2
ทฤษฎีและหลักการ

การพัฒนาระบบที่จอดรถให้มีประสิทธิภาพมากยิ่งขึ้น โดยเพิ่มระบบตรวจหาตำแหน่งที่ว่างสำหรับจอดรถ โดยโครงงานอาศัยภาพที่ได้จากกล้องวงจรปิดมาวิเคราะห์ด้วยกระบวนการต่างๆทาง Image Processing ซึ่งในที่นี้จะกล่าวเฉพาะเนื้อหาที่ควรรู้เท่านั้นที่เกี่ยวกับโครงงาน

2.1 พื้นฐานการประมวลผลภาพดิจิตอล

การประมวลผลภาพดิจิตอล (Image Processing) เป็นกระบวนการที่นำข้อมูลภาพดิจิตอลเข้ามาทำการประมวลผล และให้ข้อมูลออกมาเป็นข้อมูลภาพใหม่ ซึ่งการประมวลผลส่วนใหญ่จะเน้นในด้านการปรับปรุงภาพให้ดีขึ้น (Image Enhancement) การวิเคราะห์ภาพ (Image Analysis) การแบ่งภาพออกเป็นส่วนๆ (Image Segmentation)

2.2 การได้มาของภาพ (Image Acquisition)

การได้มาของภาพมาจากกล้องวงจรปิด โดยCaptureภาพนิ่งที่ได้ จะเก็บในรูปแบบ Digital Image เก็บลงในหน่วยความจำเพื่อนำไปประมวลผลภาพในขั้นต่อไป

2.3 พิกเซล (Pixels) จุดกำเนิดภาพในจอและชนิดของรูปภาพคอมพิวเตอร์

ภาพในคอมพิวเตอร์เกิดจากการนำจุดสีที่เรียกว่า "พิกเซล (Pixels)" มาวางเรียงต่อๆ กันจนดูต่อเนื่อง ภาพจะดูคมชัดเท่าใดก็ขึ้นอยู่กับว่าภาพนั้นมีพิกเซลอยู่มากเท่าใดเรามักจะนับความละเอียดของภาพโดยนับจำนวนพิกเซลต่อความยาวภาพหนึ่งนิ้วหรือพิกเซลต่อนิ้ว (Pixels/Inch) ยิ่งมีจำนวนพิกเซลต่อนิ้วมากภาพก็ยิ่งละเอียด

ภาพยิ่งละเอียดก็ยิ่งดูเหมือนจริง ความละเอียดของภาพเรียกว่า "เรโซลูชั่น(Resolution)" ภาพที่ละเอียดมากๆจึงเรียกว่าภาพที่มีเรโซลูชั่นสูงส่วนภาพที่หยาบๆ ก็เรียกว่า ภาพที่มีเรโซลูชั่นต่ำ

ชนิดของรูปภาพบนเครื่องคอมพิวเตอร์

รูปภาพที่ปรากฏและใช้งานบนเครื่องคอมพิวเตอร์นั้น มีอยู่ด้วยกันสองชนิดคือ รูปภาพแบบเวคเตอร์ (Vecter Image) และรูปภาพแบบบิตแมป (Bitmap Image)

- **ภาพแบบเวคเตอร์ (Vecter Image)**

ภาพชนิดเวคเตอร์ (Vecter) เป็นภาพที่มีการบันทึกส่วนต่างๆ ของรูปภาพไม่ว่าจะเป็นเส้นตรงเส้นโค้ง รูปทรง และสีต่างๆ ด้วยสมาการทางคณิตศาสตร์ เมื่อต้องการนำรูปภาพนี้แสดงออกทางจอภาพ หรือพิมพ์ออกทางเครื่องพิมพ์ โปรแกรมที่ใช้ในการเปิดรูปภาพดังกล่าวจะต้องนำสมการต่างๆ ที่บันทึกไว้มาคำนวณ และสร้างรูปทรงต่างๆ ขึ้นมาใหม่ จุดเด่นที่ถือว่าเป็นข้อดีของภาพแบบเวคเตอร์คือ ไม่ว่าจะขยายขนาดภาพให้ใหญ่โตแค่ไหนก็ตามเครื่องคอมพิวเตอร์จะคำนวณอัตราส่วนของสมการใหม่ทำให้ภาพที่เกิดใหม่นั้นมีความคมชัดทุกครั้ง

- **ภาพแบบบิตแมป** (Bitmap Image)

ภาพแบบบิตแมป (Bitmap Image) เป็นภาพที่เกิดจากการเรียงเม็ดสีขนาดเล็กประกอบกันขึ้นมาเป็นภาพใหญ่ที่เรามองเห็นกัน เม็ดสีเล็กๆ หรือจุดสีเล็กๆ ดังกล่าว จะบรรจุด้วยสีหนึ่งสี ซึ่งจุดนี้มีชื่อเรียกว่า พิกเซล (Pixels) ในการแก้ไขภาพแบบบิตแมปนั้นหมายความว่าเรากำลังแก้ไขเม็ดสีดังกล่าวเพื่อให้ภาพเปลี่ยนไปตามต้องการ

เนื่องจากภาพแบบบิตแมปนั้นบรรจุไปด้วยเม็ดสีในจำนวนคงที่ดังนั้นเมื่อขยายภาพขึ้นมามากๆ จะทำให้รายละเอียดของภาพสูญเสียไป

2.4 ประเภทการกระทำการภาพ (Types of Image Operation)

การประมวลผลภาพดิจิตอลจะเป็น กระบวนการที่ทำการกระทำการ (Operation) ภาพอย่างใดอย่างหนึ่งต่อภาพนำเข้า (Input Image) เพื่อให้ได้ภาพผลลัพธ์ (Output Image) มีลักษณะของภาพเป็นไปตามที่ต้องการ ซึ่งการกระทำการภาพที่ใช้ในการประมวลผลภาพดิจิตอลมีอยู่มากมากหลายแบบ ความเข้าใจเกี่ยวกับคุณลักษณะและการแยกแยะประเภทของการกระทำภาพ จะช่วยให้เราสามารถคาดคะเนภาพผลลัพธ์ที่จะได้จากการกระทำการแต่ละแบบ หรือการประมาณความซับซ้อนของการกระทำการภาพที่จะใช้

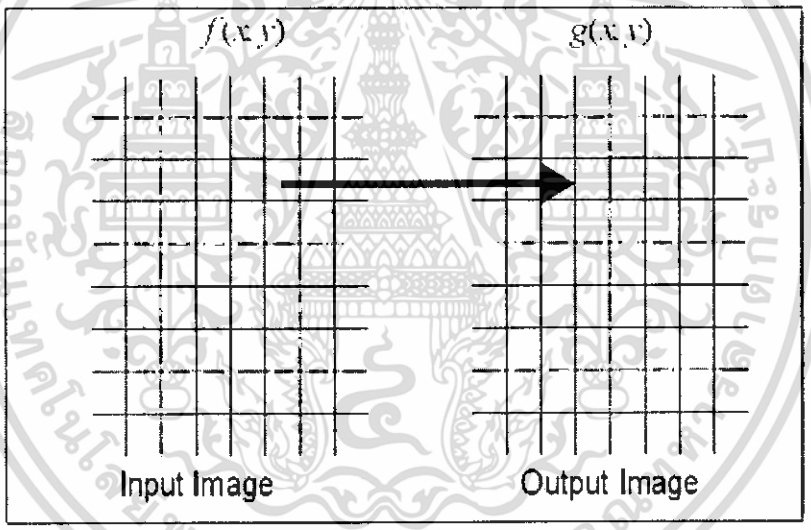
การกระทำการภาพในการประมวลผลภาพดิจิตอลสามารถแบ่งออกได้เป็นประเภทใหญ่ 3 ประเภท คือ

2.4.1 การกระทำการภาพจุดต่อจุด (Point Operations)

การกระทำแบบนี้ ค่าความเข้มแสงในแต่ละพิกเซลของภาพผลลัพธ์ จะขึ้นกับค่าความเข้มแสงของพิกเซลในภาพนำเข้า ณ ตำแหน่งที่สมนัยกันดังรูปที่ 1 ลักษณะการกระทำการภาพประเภทนี้ได้แก่ การปรับความสว่าง หรือ ความคมชัดของภาพดิจิตอล การ บวก ลบ คูณ และ หาร ภาพดิจิตอล หรือการกระทำการทางตรรกศาสตร์ต่างๆเป็นต้น

ถ้า f(x,y) และ g(x,y) เป็นภาพนำเข้าและภาพผลลัพธ์ตามลำดับ ค่าของพิกเซล g(x,y) จะมีค่าดังนี้

$$g(x_i,y_i) = £[f(x_i,y_i)] \quad \text{:เมื่อ } £ \text{ เป็นการกระทำภาพใดๆ}$$



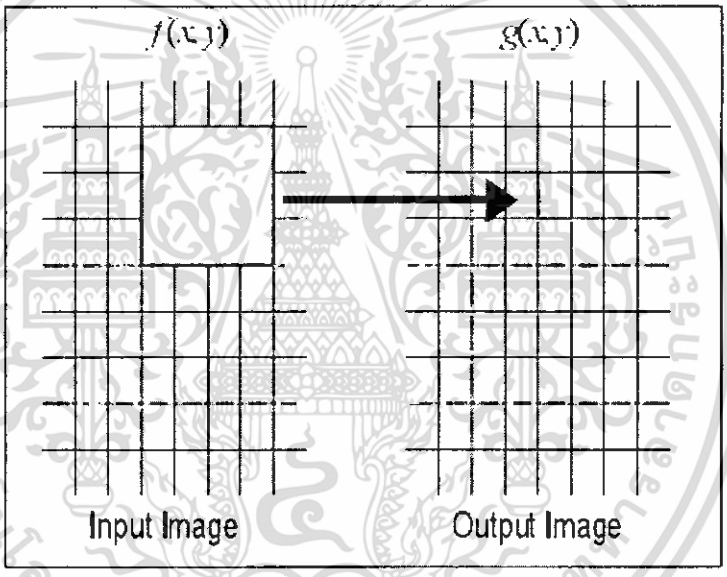
รูปที่ 2.1 การกระทำการจุดต่อจุด (Point Operations)

2.4.2 การกระทำการเฉพาะบริเวณ (Local Operation)

สำหรับการกระทำการแบบนี้ค่าความเข้มแสงของพิกเซลแต่ละจุดในภาพผลลัพธ์จะขึ้นกับค่าความเข้มแสงของกลุ่มพิกเซลที่อยู่บริเวณเดียวกัน (Neighborhood Pixels) ในภาพนำเข้า ดังรูปที่2 ลักษณะการกระทำการภาพประเภทนี้ได้แก่การหาขอบ (Edge Detection) การกรองสัญญาณในโดเมนระยะทาง (Spatial Filtering) เป็นต้น

ถ้า f(x,y) และ g(x,y) เป็นภาพนำเข้าและภาพผลลัพธ์ตามลำดับ ค่าของพิกเซล g(x,y) จะมีค่าดังนี้

$$g(x_i,y_i) = \text{£}[\text{neighborhood of } f(x_i,y_i)]$$



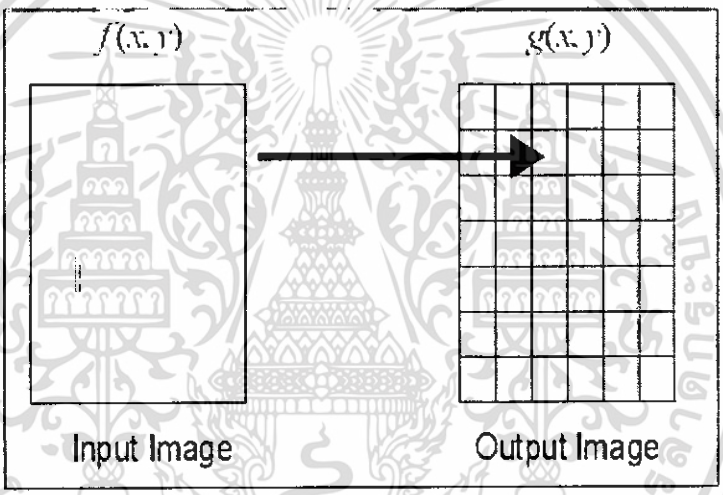
รูปที่ 2.2 การกระทำการเฉพาะบริเวณ (Local Operation)

2.4.3 การกระทำการทั้งหมด (Global Operations)

การกระทำแบบนี้ ค่าความเข้มแสดงในแต่ละพิกเซลของภาพผลลัพธ์ (Output Image) จะขึ้นกับค่าความเข้มของแสงของพิกเซลทุกตัวในภาพนำเข้า ดังรูปที่ 2.3 ลักษณะการกระทำการภาพประเภทนี้ได้แก่ การเทรสโฮลดิ้ง (Thresholding) การทำฮีสโทรแกรม (Histogram) เป็นต้น

ถ้า f(x,y) และ g(x,y) เป็นภาพนำเข้าและภาพผลลัพธ์ตามลำดับ ค่าของพิกเซล g(x,y) จะมีค่าดังนี้

$$g(x_i,y_i) = \pounds[f(x_i,y_i) \text{ for all } i]$$



รูปที่ 2.3 การกระทำการทั้งหมด (Global Operation)

2.5 **Color Gray Images**

ในโครงงานนี้ จะใช้ภาพ Color Gray มาใช้ในการทำงาน ซึ่งมีข้อดีหลายข้อ คือ ภาพ Color Gray เพียงพอที่จะนำมาทำการตรวจสอบการเปลี่ยนแปลงของภาพ ก็คือการเปลี่ยนแปลงของความเข้มของแสง และภาพ Color Gray ใช้หน่วยความจำในการเก็บข้อมูลน้อยกว่าภาพสี ทำให้ประหยัดหน่วยความจำ เมื่อเทียบกับการเก็บข้อมูลของภาพสี ทำให้ภาพ Color Gray ใช้เวลาในการประมวลผลเร็วกว่าภาพสี

Gray Level คือระดับความเข้มของเม็ดสียิ่งค่าความเข้มสูงสีที่แสดงผลออกทางจอคอมพิวเตอร์จะเป็นสีขาวและระดับความเข้มต่ำสีที่แสดงผลออกทางจอคอมพิวเตอร์จะเป็นสีดำ ซึ่งระดับGray Level มีหลายระดับดัง รูปที่ 4

รูปที่ 2.4 Gray Level ระดับต่างๆ

ในโครงงานได้เลือก Gray Level 256 level เพราะมีความละเอียดเพียงพอที่จะนำมาใช้ในการวิเคราะห์ภาพ

2.6 การหาค่าความสว่างของภาพ (Image Brightness)

การหาค่า Brightness คือการหาค่าความสว่างของภาพ โดยหาได้จากหาค่าผลรวมของ gray level ทั้งหมดหารด้วยค่าของจำนวนพิกเซลทั้งหมด ดังสมการ

$$\text{Brightness} = \overline{X} = \sum_{i=0}^{N} \frac{Xi}{N} \qquad 2.1$$

N = total of pixels

Xi = gray level of pixels

2.7 การปรับปรุงภาพ (Image Enhancement)

คือกระบวนการเปลี่ยนคุณภาพของภาพให้อยู่ในระดับที่ดีขึ้น ที่จะสามารถนำไปวิเคราะห์ได้ การปรับปรุงภาพที่ใช้ คือวิธีการ Histogram Equalization เป็นการกระทำการทั้งหมด (Global Operations) มีวิธีการดังนี้

2.7.1 หาจำนวน Pixel ของแต่ละ Gray level ($P_i(j)$)

2.7.2 ทำ Accumulated Sum มีสมาการดังนี้

$$A_i(k) = \sum_{j<=k} P_i(j)$$

$A_i(k)$ = Accumulated Sum

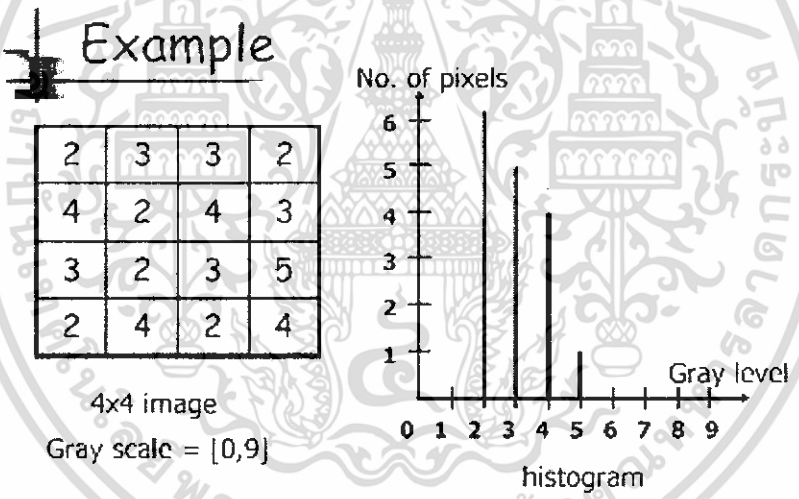
$P_i(j)$ = จำนวน Pixel ของแต่ละ Gray level

2.7.3 ทำ Normalized มีสมการดังนี้

$$P_l(k) = A_l(k) / N$$

N = จำนวน Pixel ทั้งหมด

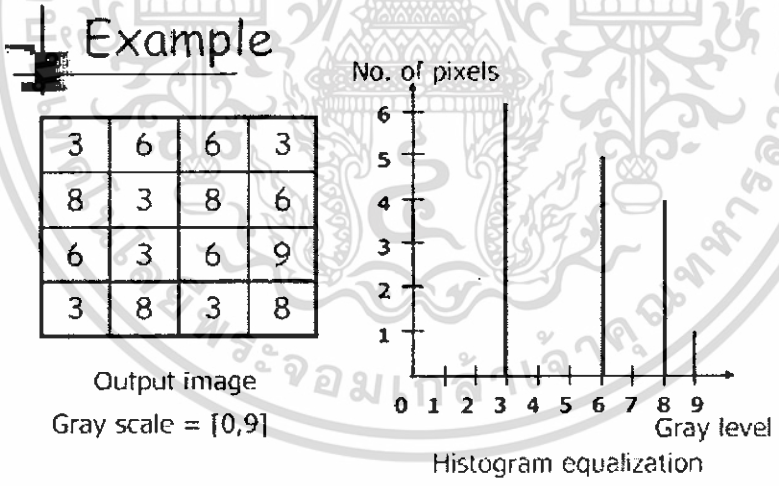
2.7.4 นำค่า max ของ gray level คูณค่า $P_l(k)$



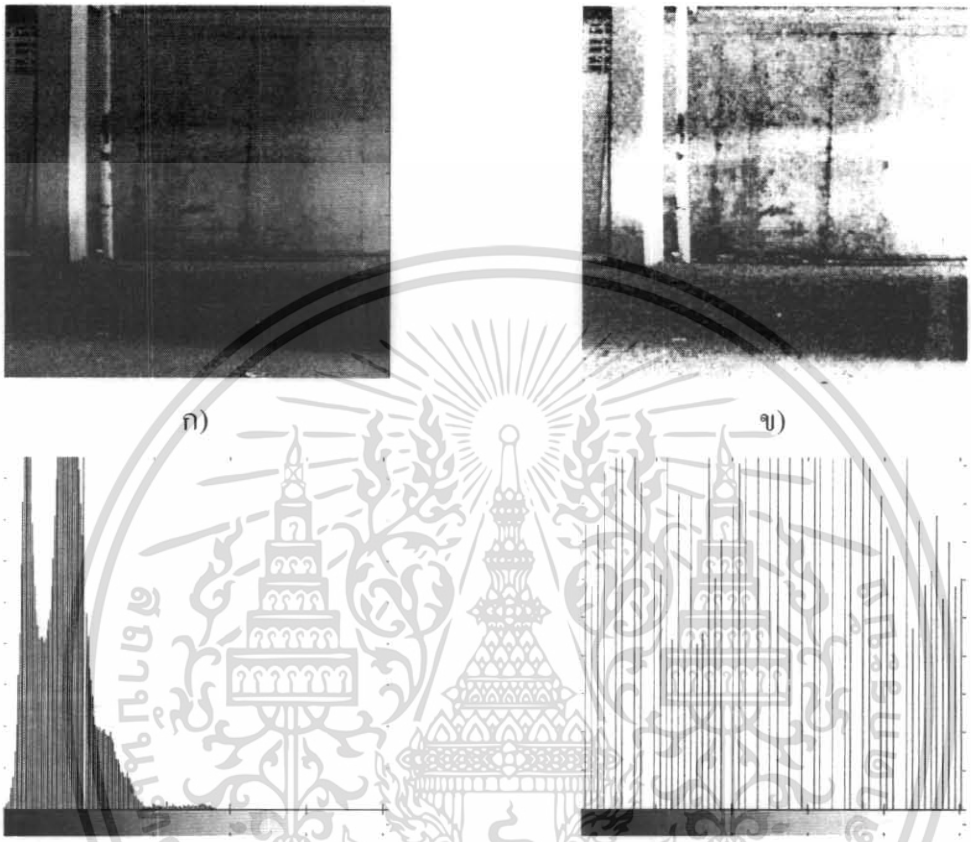
รูปที่ 2.5 ตัวอย่างการทำ วิธีการ Histogram Equalization

Gray Level(j)	0	1	2	3	4	5	6	7	8	9
No. of pixels	0	0	6	5	4	1	0	0	0	0
$\sum_{i=0}^{j} n_i$	0	0	6	11	15	16	16	16	16	16
$s \sum_{i=0}^{j} \frac{n_i}{n}$	0	0	6 / 16	11 / 16	15 / 16	16 / 16	16 / 16	16 / 16	16 / 16	16 / 16
$s \times 9$	0	0	3.3 ≈3	6.1 ≈6	8.4 ≈8	9	9	9	9	9

รูปที่ 2.6 ตัวอย่างการทำ วิธีการ Histogram Equalization



รูปที่ 2.7 ตัวอย่างการทำ วิธีการ Histogram Equalization



ก)

ข)

ค)

ง)

รูปที่ 2.8 ก) ภาพที่จอดรถขณะไม่มีรถจอดก่อนทำ Histogram Equalization

ข) ภาพที่จอดรถขณะ ไม่มีรถจอดหลังทำ Histogram Equalization

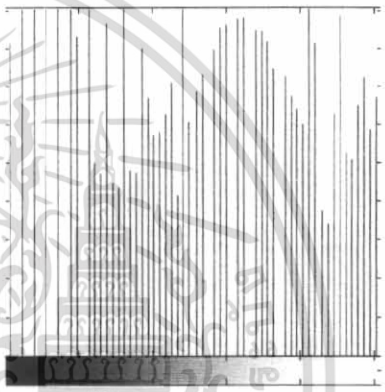
ค) ภาพ Histogram ขณะไม่มีรถจอด

ง) ภาพ Histogram หลังทำ Histogram Equalization ขณะไม่มีรถจอด

ก)

ข)

ค)



ง)

รูปที่ 2.9 ก) ภาพที่จอดรถขณะมีรถจอดก่อนทำ Histogram Equalization

ข) ภาพที่จอดรถขณะมีรถจอดหลังทำ Histogram Equalization

ค) ภาพ Histogram ขณะมีรถจอด

ง) ภาพ Histogram หลังทำ Histogram Equalization ขณะมีรถจอด

2.8 **การทำ Thresholding**

เป็นกระบวนการที่ใช้ในการแยกวัตถุที่มีความแตกต่างจากพื้นหลังของภาพ การจำแนกแต่ละกลุ่มของพิกเซลว่าเป็นวัตถุเดียวกันหรือว่าเป็น ภาพพื้นหลัง สามารถกำหนดค่าให้กับภาพต้นฉบับได้ถ้าอยู่ในช่วงที่เรากำหนด

การทำThresholdทั่วไป จะเป็นแบบตรงไปตรงมา กำหนดให้ $a \in R^x$ ของภาพต้นฉบับและให้ช่วงที่ต้องการทำThresholdเป็น [h,k] Thresholdของภาพ b {0,1}x ให้

$$b(x) = \begin{cases} 1 & \text{ถ้า } h \le a \le k \\ 0 & \text{อื่นๆ} \end{cases}$$

ทุกค่าของ x มีสองกรณีที่ต้องให้ความสนใจคือ ค่าของภาพ b มีค่าสูงเพียงอย่างเดียว หรือมีค่าต่ำเพียงอย่างเดียว ในกรณีแรกของ Threshold ภาพ b จะให้

$$b(x) = \begin{cases} 1 & \text{ถ้า } a \ge k \\ 0 & \text{อื่นๆ} \end{cases}$$

กรณีที่สอง

$$b(x) = \begin{cases} 1 & \text{ถ้า } a < k \\ 0 & \text{อื่นๆ} \end{cases}$$

2.9 **Sobel Edge Detection**

เป็นวิธีการหาขอบของวัตถุในรูปภาพ โดยใช้ Mask ของ Sobel ไปทำการ Convolution เพื่อให้ได้ขอบของวัตถุในแนวแกนนอน และแนวแกนตั้ง

ซึ่ง Mask ของ Sobel มีดังนี้

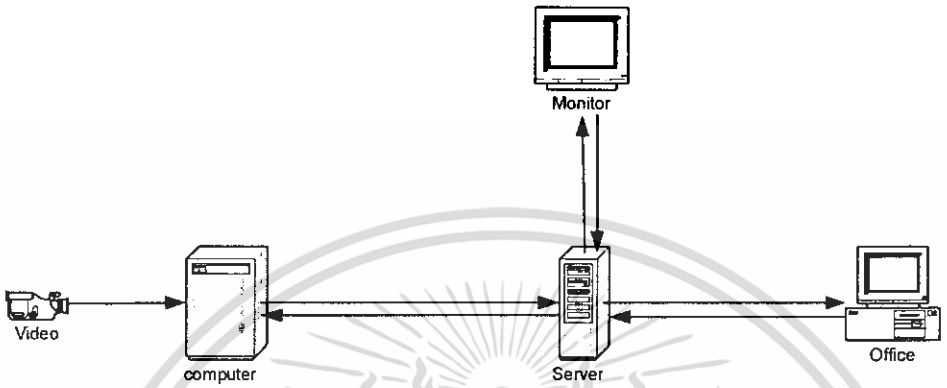
$$A = \begin{bmatrix} -1 & 0 & 1 \\ -2 & \boxed{0} & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} -1 & -2 & -1 \\ 0 & \boxed{0} & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Mask หาขอบแกนตั้ง Mask หาขอบแนวนอน

รูปที่ 2.10 ภาพก่อนและหลังการทำ Sobel Edge Detection

บทที่ 3
การออกแบบและการเขียนโปรแกรม

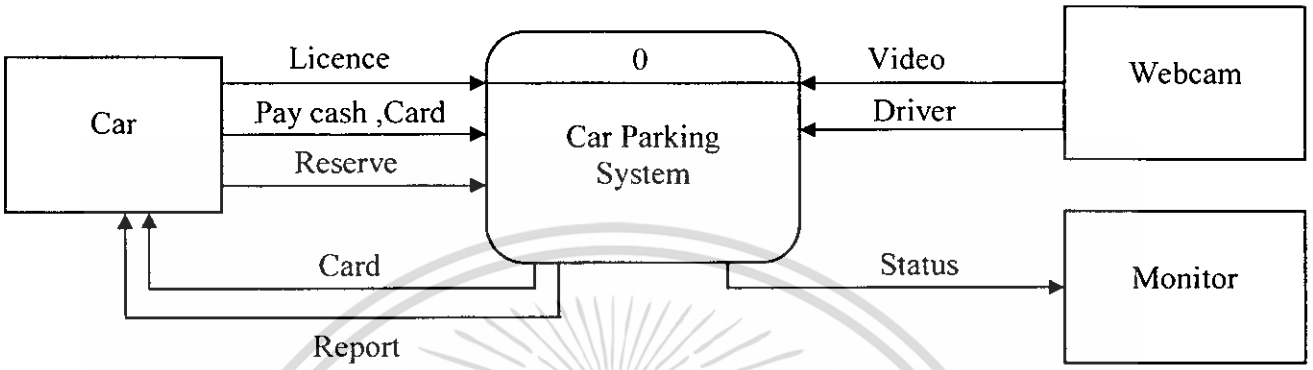
3.1 การออกแบบ



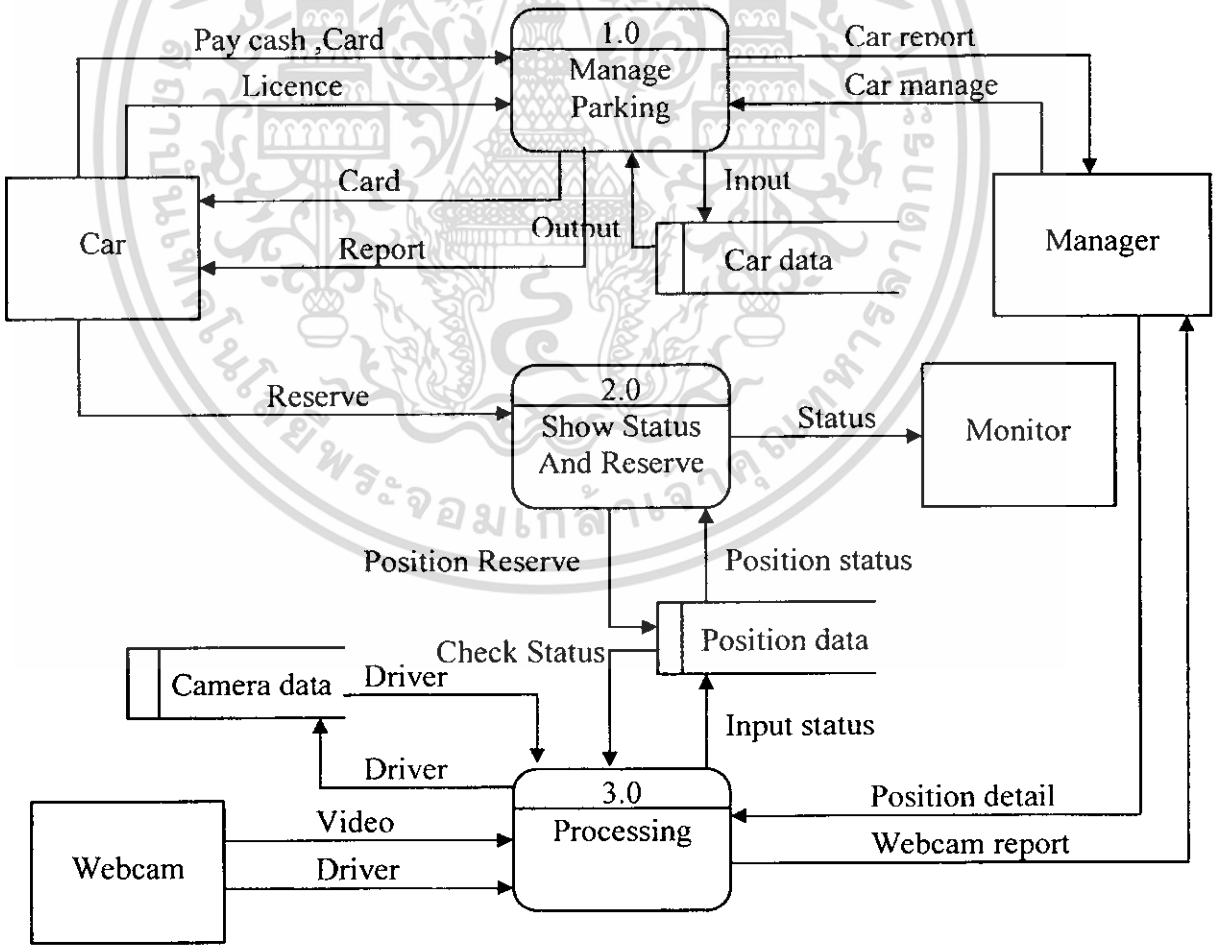
รูปที่ 3.1 รูปแบบการทำงานของระบบ CarParking

จากรูป video จะเป็นตัวเว็บแคมที่จับภาพในที่จอดรถ ซึ่งจะส่งภาพไปให้คอมพิวเตอร์ประมวลผล ซึ่งจะทำการประมวลผล ทุกๆ 15 วินาที เมื่อประมวลผลแล้วจะทำการส่งข้อมูลไปเก็บที่ Server ซึ่งจะเป็นตัวที่จะแสดงผลบนจอมอนิเตอร์ และที่มอนิเตอร์นี้จะมีการจอง ที่จอดรถได้ โดยเมื่อมีการจอง จะส่งตำแหน่งที่จองกลับมาเก็บยัง Server และเมื่อมีการมีการประมวลผลครั้งถัดไป คอมพิวเตอร์ที่ทำการวิเคราะห์ภาพ จะทำการดึงข้อมูลว่าได้มีการจองที่ตำแหน่งใดบ้าง แล้วจะทำการยกเลิกการประมวลผลที่ ตำแหน่งที่ได้มีการจองเป็นเวลา 5 นาที เพื่อให้รถที่จองได้เข้าไปจอดแต่ถ้ารถเข้าไปจอดไม่ตรงกับช่องที่จอง หลังจาก 5 นาทีแล้ว status ก็จะได้กลับมาเป็นที่ว่างเหมือนเดิมให้คนอื่นจองได้ต่อไป ทางด้าน Office นั้นจะเป็นในส่วนที่ ยามจะแจกบัตรจอดรถ โดยที่จะเก็บข้อมูล เลขทะเบียนรถ,วันเวลาเข้าและออก,หมายเลขบัตรจอดรถ,และในส่วนคิดค่าจอดรถ และสามารถเรียกดู สถิติต่างๆได้

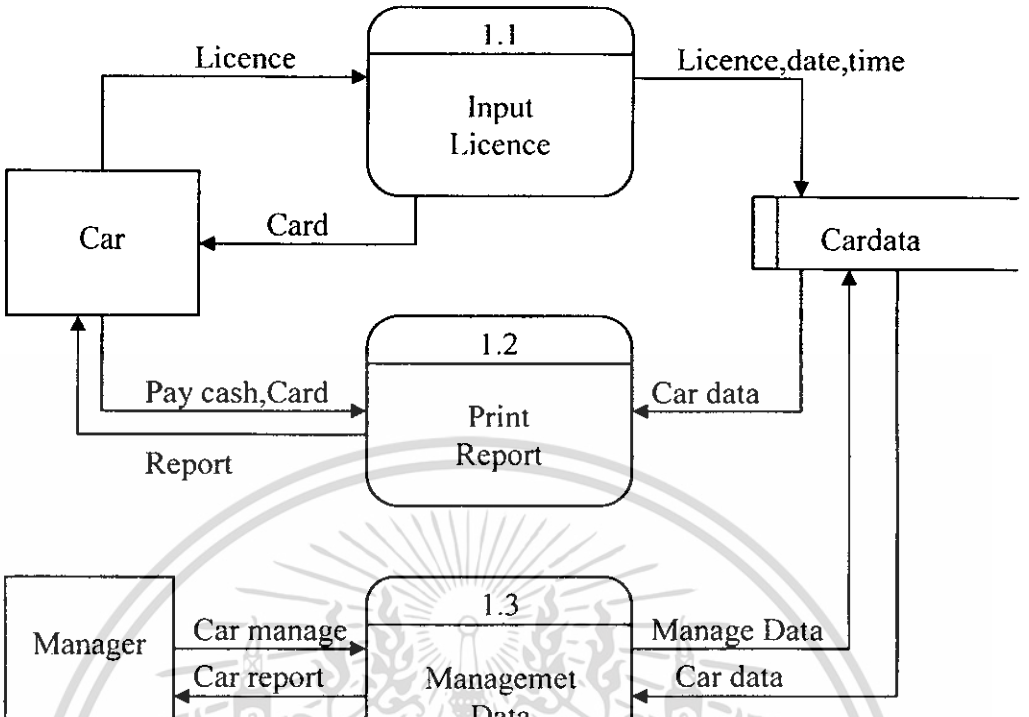
3.2 **Data Flow Diagram ของระบบ**



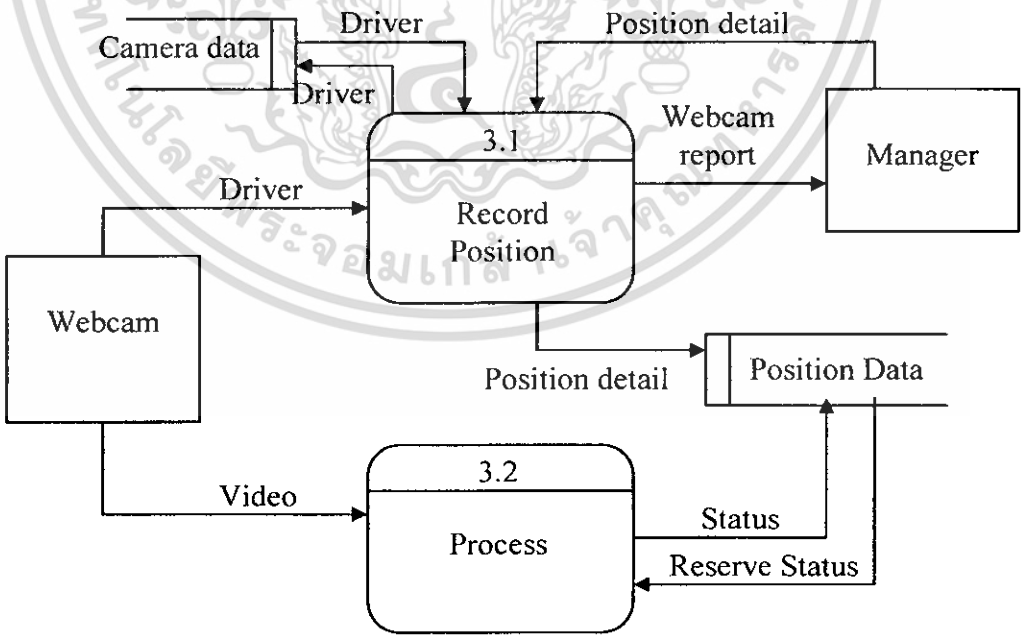
รูปที่ 3.2 Context Diagram ของระบบ Car Parking



รูปที่ 3.3 DFD level 0 ของระบบ Car Parking



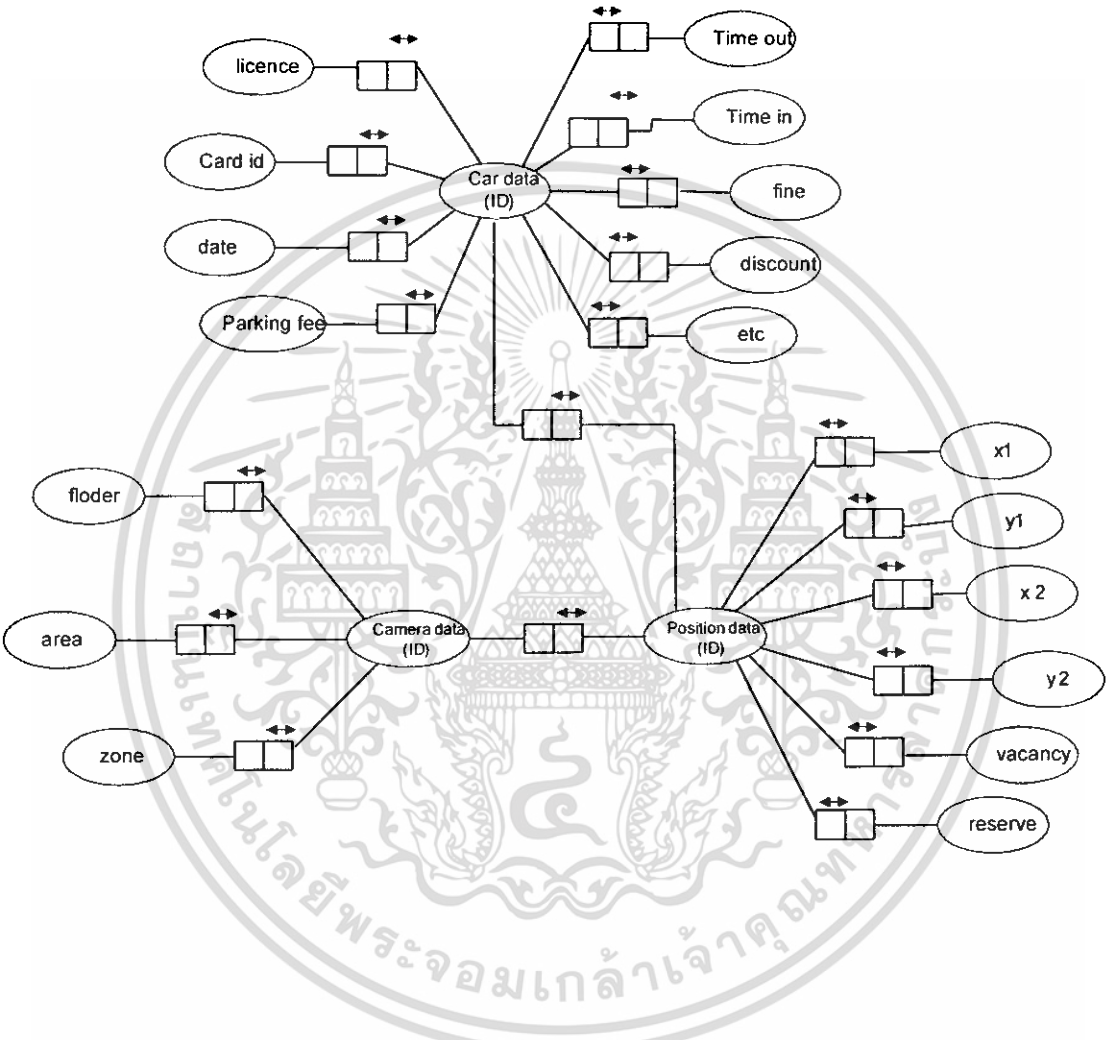
รูปที่ 3.4 DFD Level 1 of Process 1.0



รูปที่ 3.5 DFD Level 1 of Process 3.0

3.3 การออกแบบฐานข้อมูล

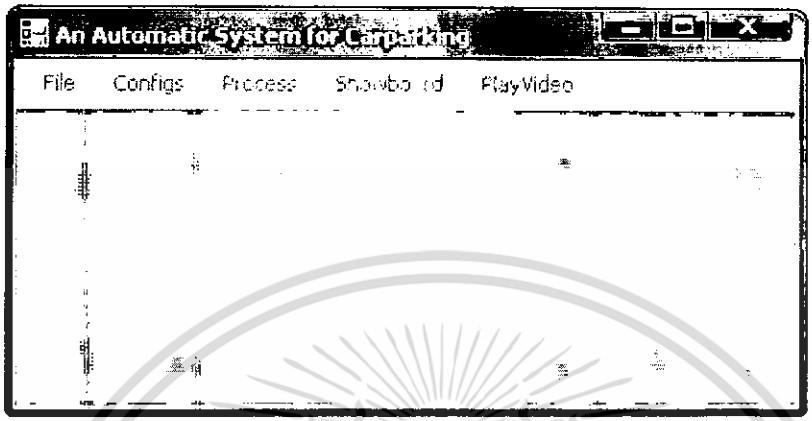
จากการออกแบบ Data Flow Diagram จะได้ว่ามี ทั้งหมด 3 ตาราง โดยมี ดังแผนภาพ NIAM



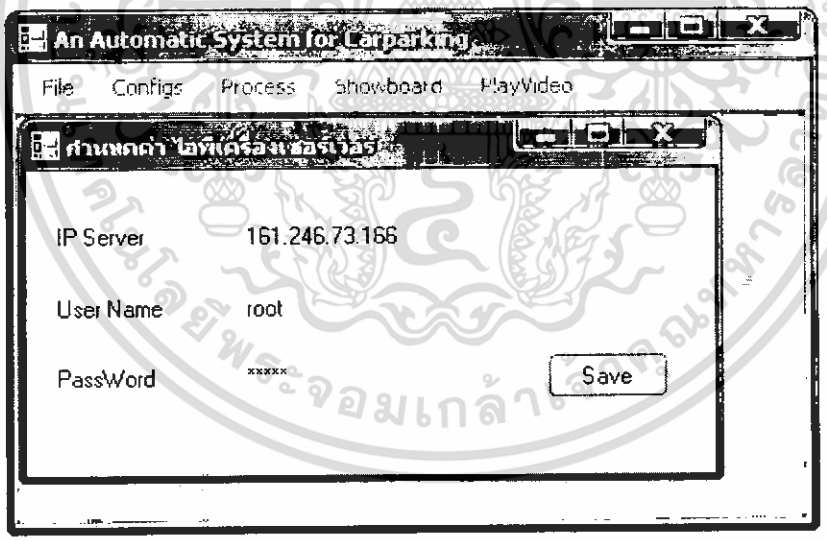
รูปที่ 3.6 แสดงแผนภาพ NIAM ของฐานข้อมูล

3.4 **การออกแบบโปรแกรม**

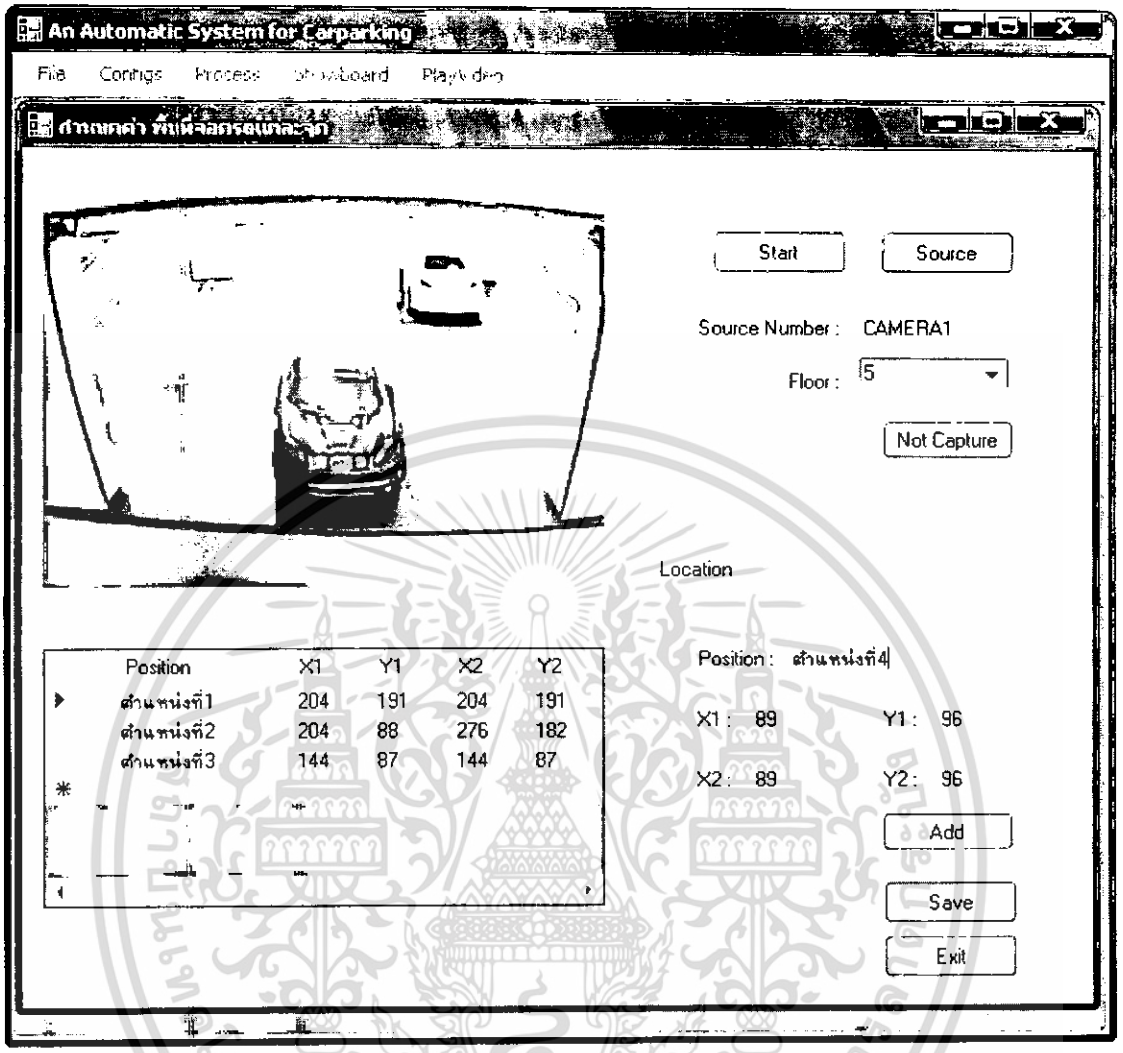
ในตอนเริ่มต้นเขียนโปรแกรมจำเป็นจะต้องออกแบบ Graphic User Interface ก่อนเพื่อง่ายในการเขียนโปรแกรม



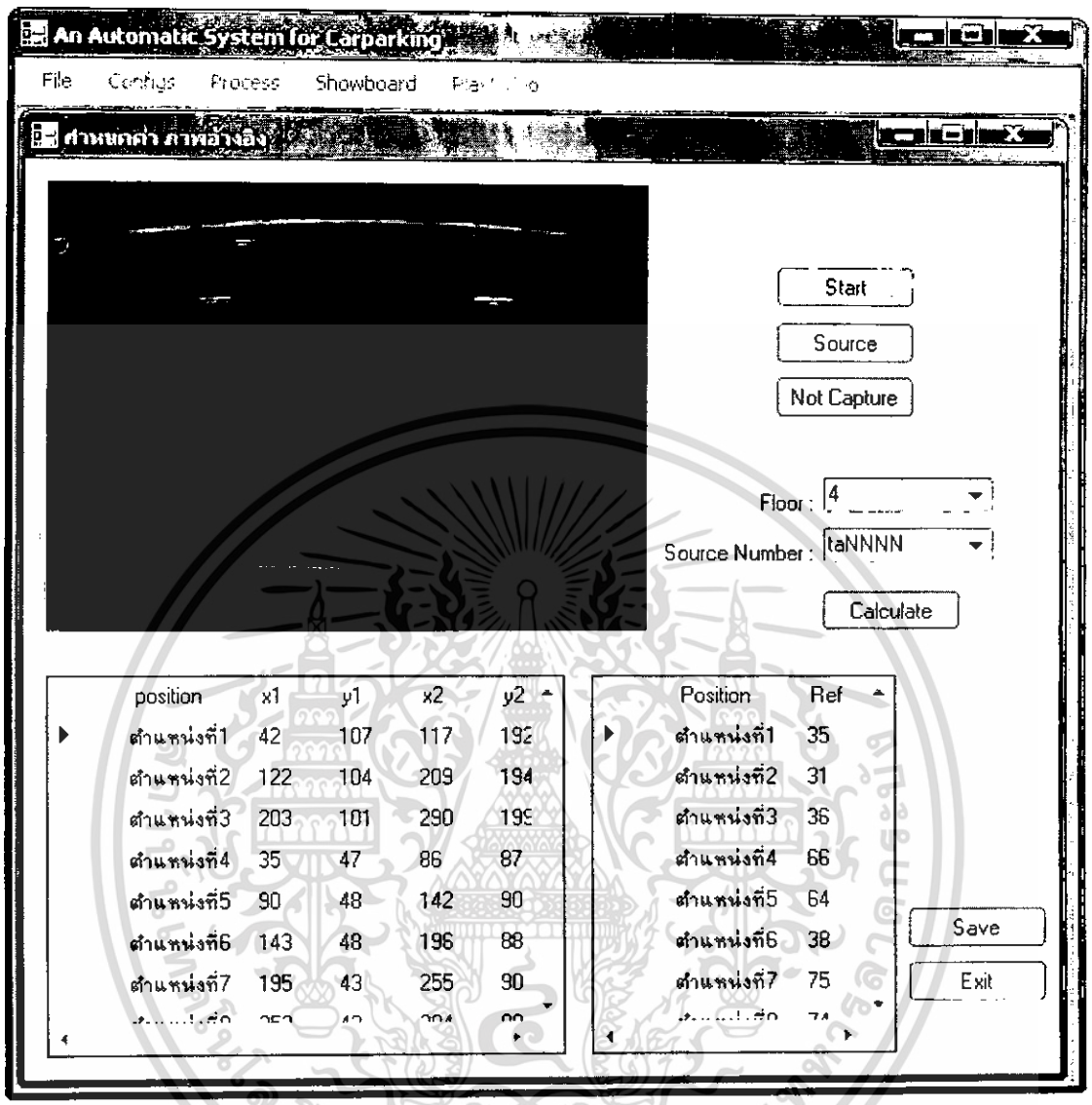
รูปที่3.7 หน้าจอหลักโปรแกรม



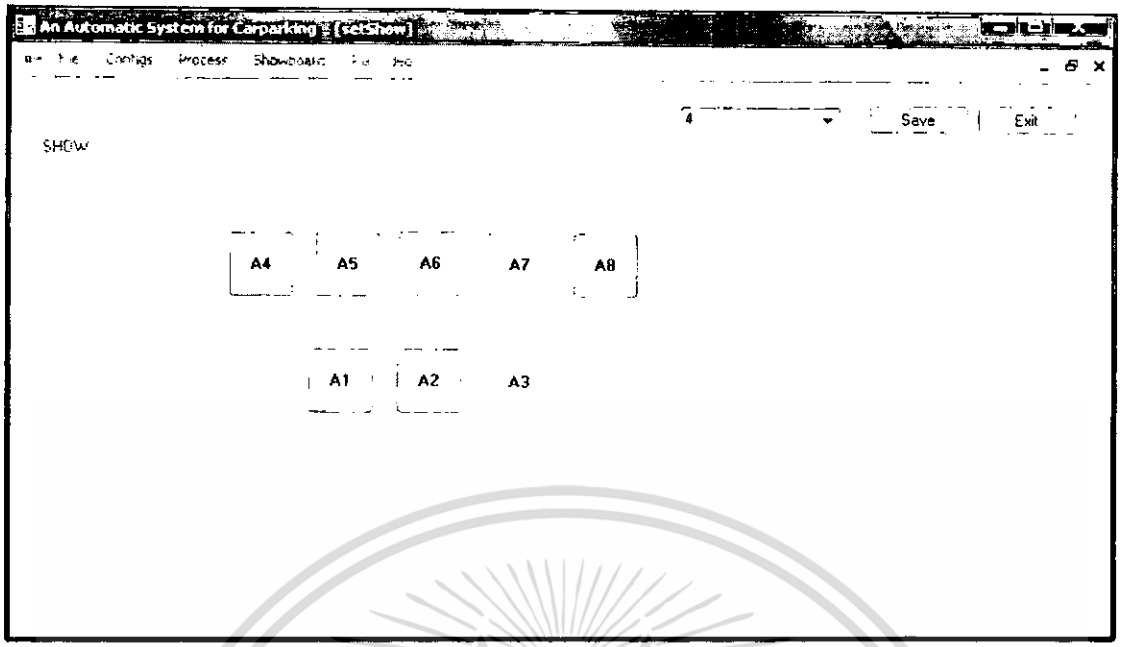
รูปที่3.8 หน้าจอกำหนดค่า IP Server



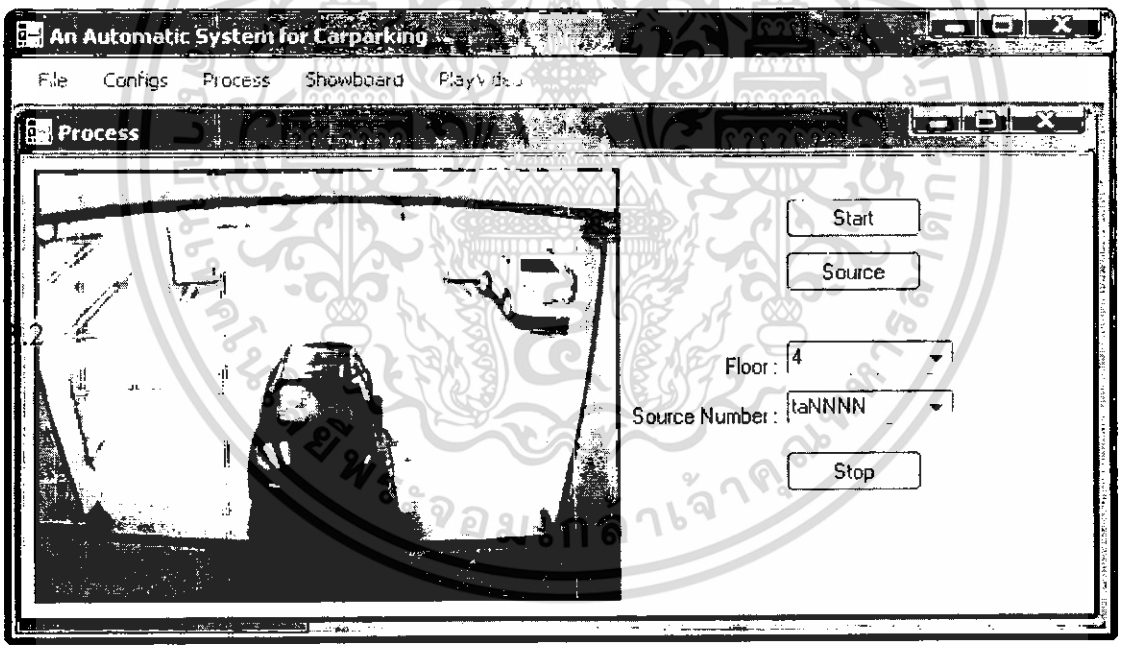
รูปที่3.9 หน้าจอกำหนดค่าพื้นที่จอดรถแต่ละจุด



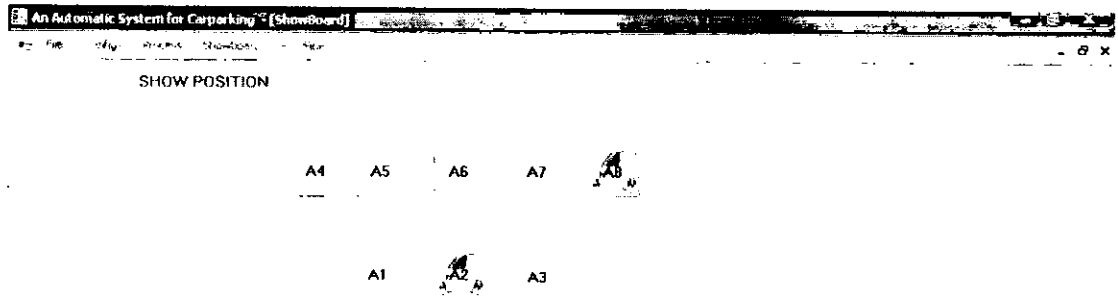
รูปที่3.10 หน้าจอกำหนดค่าอ้างอิง



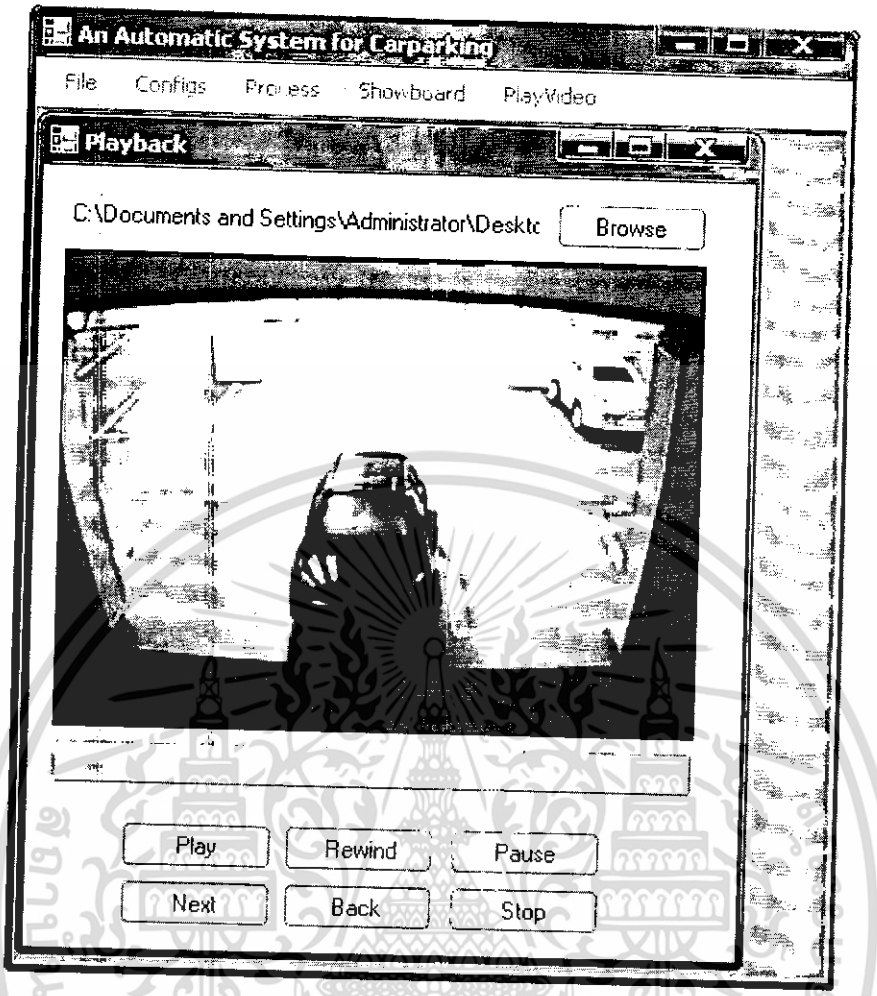
รูปที่3.11 หน้าจอกำหนดแผนผังที่จอดรถ



รูปที่3.12 หน้าจอแสดงส่วนประมวลผล

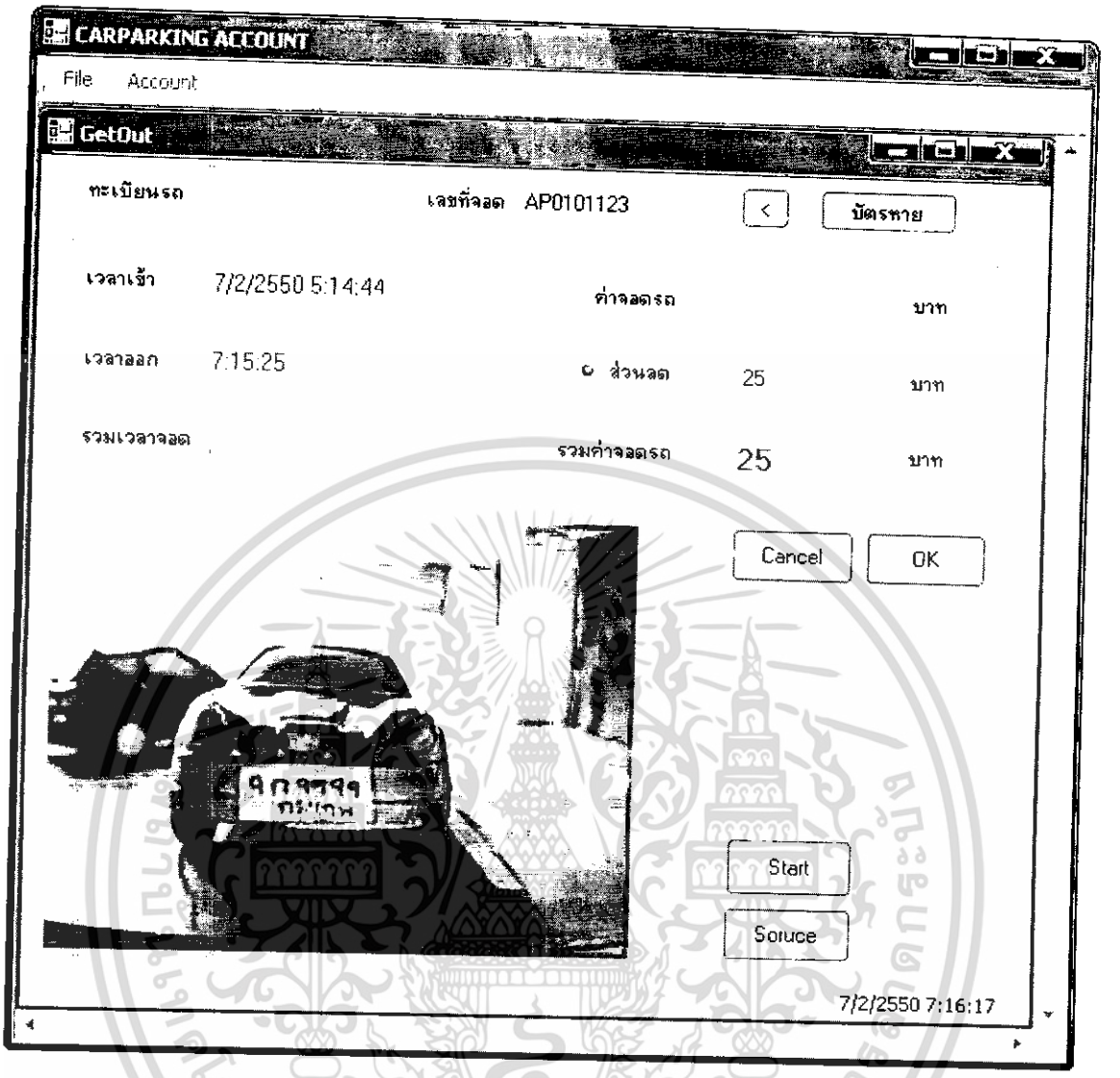


รูปที่3.13 หน้าจอแสดงผลทางมอนิเตอร์



รูปที่3.14 หน้าจอแสดง ไฟล์ วีดีโอ

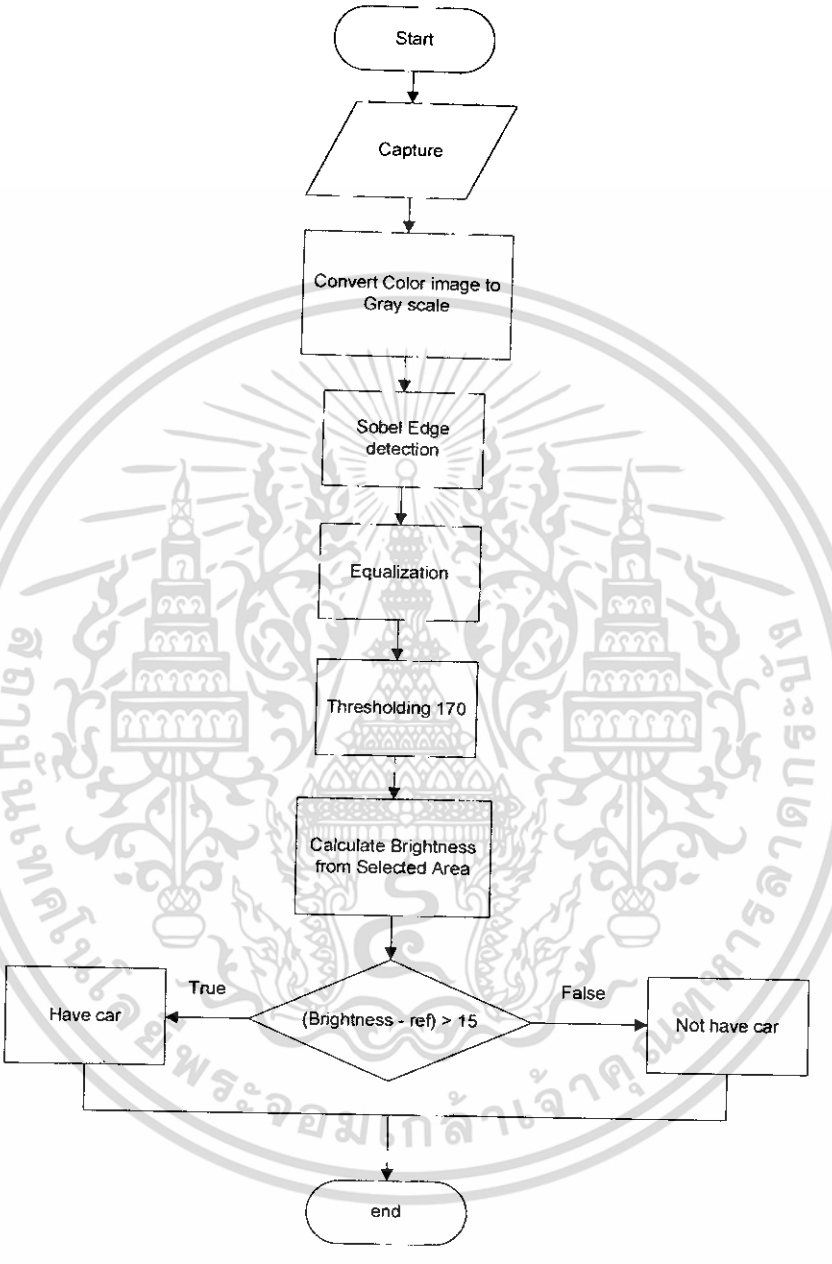
รูปที่3.15 หน้าจอเก็บข้อมูลลูกค้าทางเข้า



รูปที่3.16 หน้าจอเก็บข้อมูลลูกค้าทางออก

3.5 **Flowchart การทำงานของระบบ**

เป็นลำดับขั้นตอนในการประมวลผลว่าภาพที่จอดรถนั้นว่างหรือ ไม่



รูปที่3.17 Flowchart การทำงานแยกแยะว่ามีรถหรือไม่

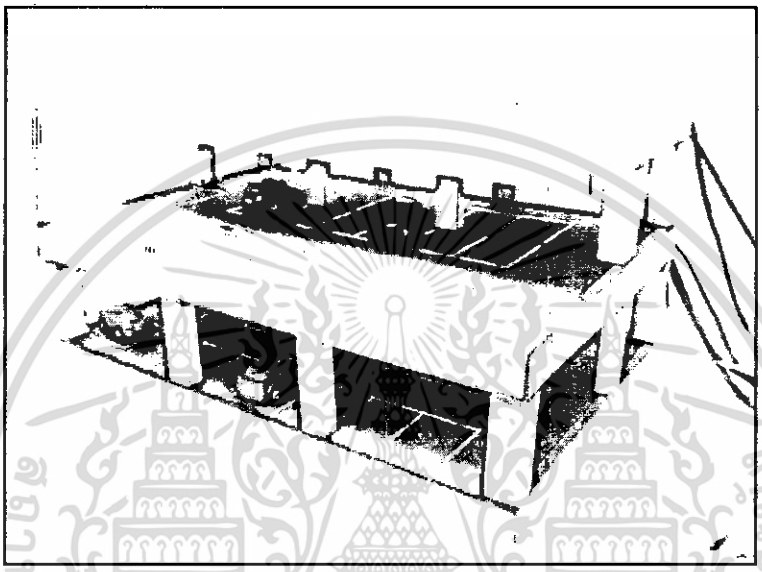
ชื่อขั้นตอน	คำอธิบาย
1. Capture	เก็บภาพจากกล้องเป็นรูปแบบภาพสี เพื่อใช้ในการวิเคราะห์
2. Convert Color Image To Gray Scale	เปลี่ยนภาพ ที่ได้จากข้อ 1 เป็นภาพ Gray Scale ซึ่งใช้ในการวิเคราะห์
3. Sobel Edge Detection	นำภาพที่ได้จากข้อ 2 มาหาขอบของภาพโดยวิธีการ Sobel Edge Detection
4. Equalization	นำภาพที่ได้ จากข้อ 3 มาทำการปรับปรุงภาพเพื่อให้ ได้ขอบภาพที่ชัดเจนยิ่งขึ้น โดยวิธี Equalization
5. Thresholding 170	ทำการ Thresholding ที่ระดับ 170 เพื่อ เปลี่ยนภาพจาก Gray Scale เป็นภาพ Binary โดยเลือกเอาเฉพาะขอบของภาพ
6. Calculate Brightness From Select Area	นำภาพที่ได้จากข้อ 5 มาคำนวณหาค่า Brightness ของภาพ โดยหาเฉพาะบริเวณที่ช่องที่จอดรถแต่ละช่อง
7. Brightness - ref	นำค่าที่ได้จากข้อ 6. ทำการ Normalize เพื่อให้ได้ค่าที่อยู่ในมาตรฐานเดียวกันจะได้ใช้ค่าในการขีดแบ่งค่าเดียวกัน เท่ากับ 15 โดยถ้ากรณีที่ค่าที่ได้จากการลบกัน มีค่ามากกว่า 15 จะถือว่ามีรถจอด และกรณีที่ลบกันแล้วได้ค่าน้อยกว่า 15 ถือว่าไม่มีรถจอด
8. Have Car	จากข้อ 7. ถ้าเป็นกรณีที่มากกว่า 15 จะถือว่ามีรถจอดจะทำการเก็บค่า Status เป็น 1 คือ มีรถจอด
9. Not Have Car	จากข้อ 7. ถ้าเป็นกรณีที่น้อยกว่า 15 จะถือว่าไม่มีรถจอดจะทำการเก็บค่า Status เป็น 0 คือ ไม่มีรถจอด

จากตารางจะอธิบายถึงแต่ละขั้นตอนของการทำงานในการประมวลผลว่ามีรถจอดอยู่หรือ ไม่มีรถจอด ซึ่งหลังจากการประมวลผลจะทำการเก็บค่า Status ลงยังฐานข้อมูลเพื่อนำผลที่ได้ไปโชว์ที่ว่างต่อไป

บทที่ 4

การทดลองและผลการทดลอง

การทดลองโปรแกรมในส่วนนี้เป็นการทดลองหาตำแหน่งที่จอดรถ แล้วแสดงผลทางมอนิเตอร์ โดยในการทดลองนี้ได้จำลองโมเดลที่จอดรถ เพื่อง่ายในการทดลองและควบคุมตัวแปรที่สำคัญคือ แสงและสีของรถยนต์



รูปที่4.1 แบบจำลองที่จอดรถ

ขั้นตอนการทดลอง

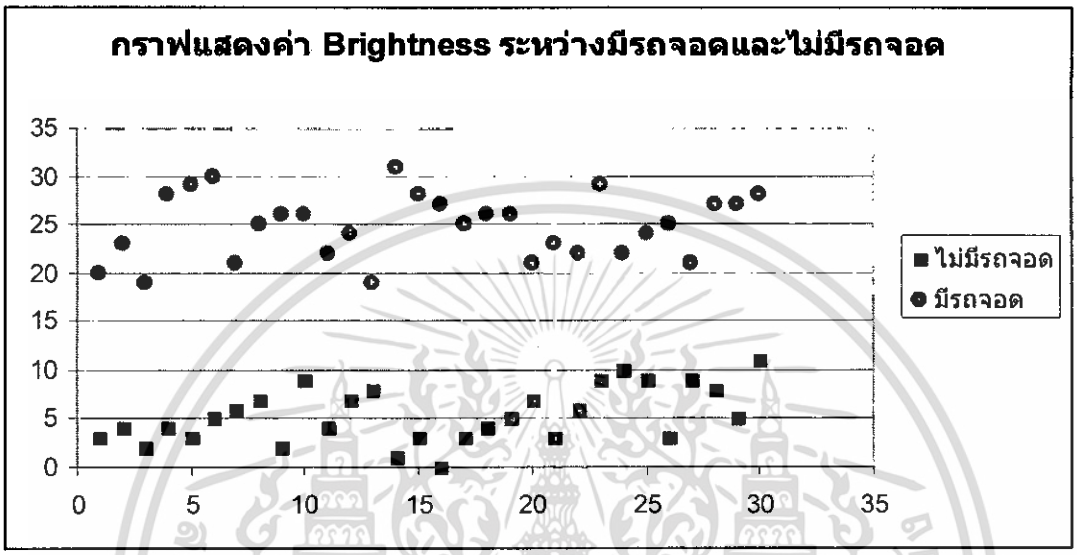
1.หาค่า Brightness ที่เหมาะสม ที่จะนำไปใช้ Threshold เพื่อแยกแยะว่าภาพนั้นมีรถจอดหรือไม่

2.ทดลองโปรแกรมตรวจหาตำแน่งที่ว่างโดยใช้สีของรถคงที่ และแสงเปลี่ยนแปลง

3.ทดลองโปรแกรมตรวจหาตำแหน่งที่ว่างโดยใช้แสงคงที่ และสีรถปลี่ยนแปลง

4.1 การทดลองเพื่อหาค่า Brightness ที่เหมาะสม

ในการทดลองนี้ได้ทดลองวัดค่า Brightness ของตำแหน่งต่างๆ ระหว่างมีรถจอดและไม่มีรถจอดแล้วนำมาพล็อตกราฟ ได้ผลดังนี้ ซึ่งค่า Brightness หาได้ตามหัวข้อที่ 2.6

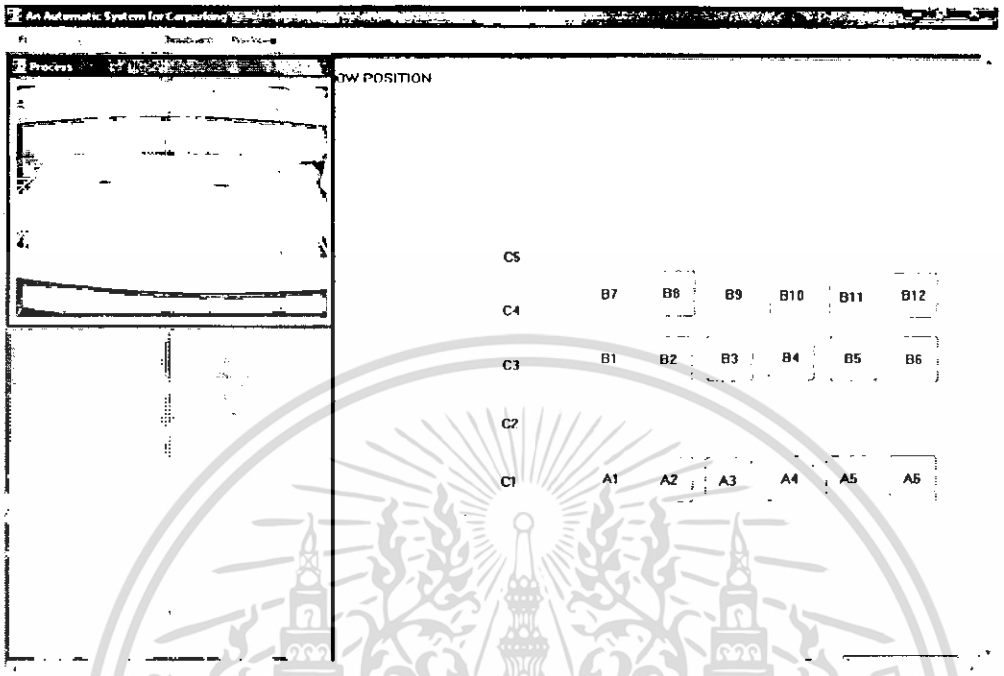


รูปที่4.2 กราฟแสดงค่าBrightness ของภาพระหว่างมีรถจอดและไม่มีรถจอด

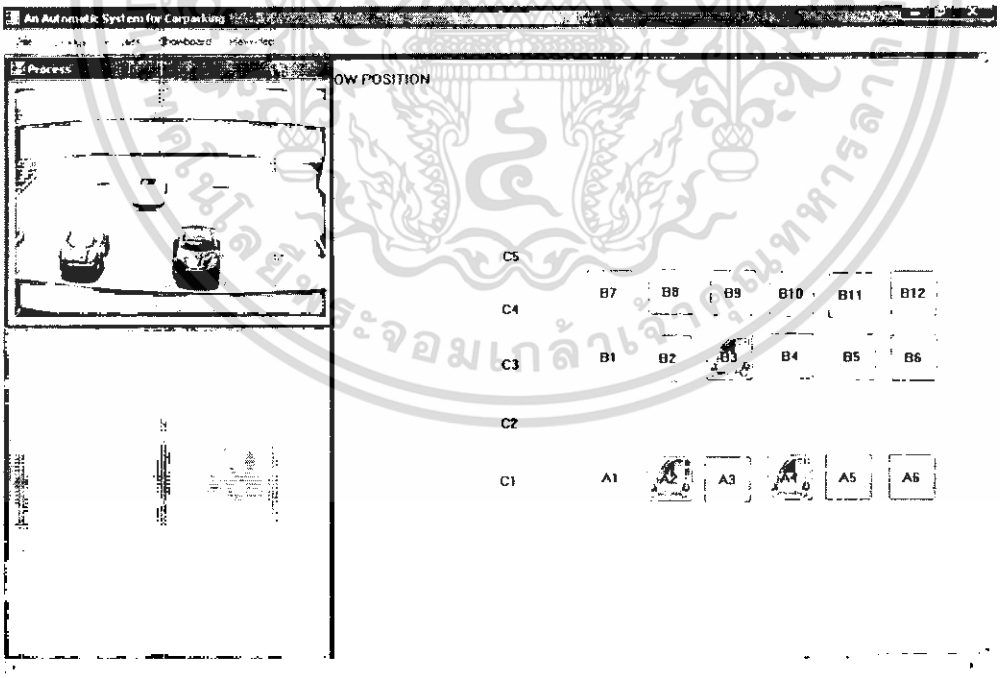
จากกราฟ สังเกตุได้ว่าค่าของ Brightness ระหว่างมีรถจอดและไม่มีรถจอดสามารถจัดกลุ่มแยกได้ โดยใช้ค่า Brightness = 15 สำหรับเป็นค่าขีดแบ่งเพื่อแยกแยะว่ามีรถจอดหรือไม่มีรถจอด

4.2 การทดลองโปรแกรมตรวจหาตำแน่งที่ว่างโดยใช้สีของรถคงที่ และแสงเปลี่ยนแปลง

4.2.1 กรณีแสงสว่าง

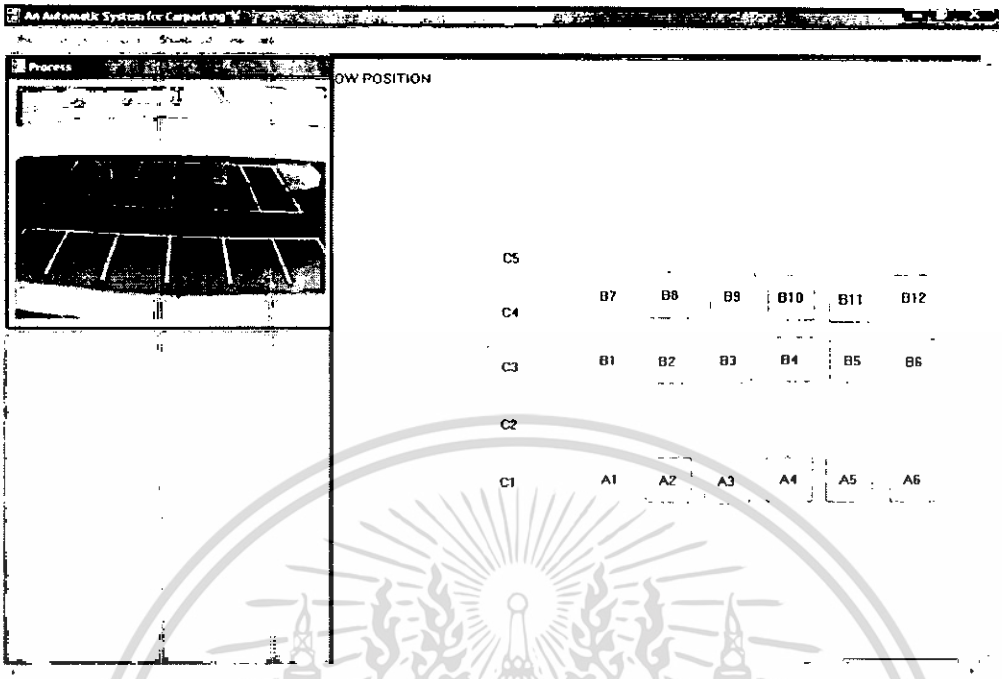


รูปที่4.3 ภาพที่จอดรถและจอมอนิเตอร์ก่อนมีรถจอดกรณีแสงสว่าง

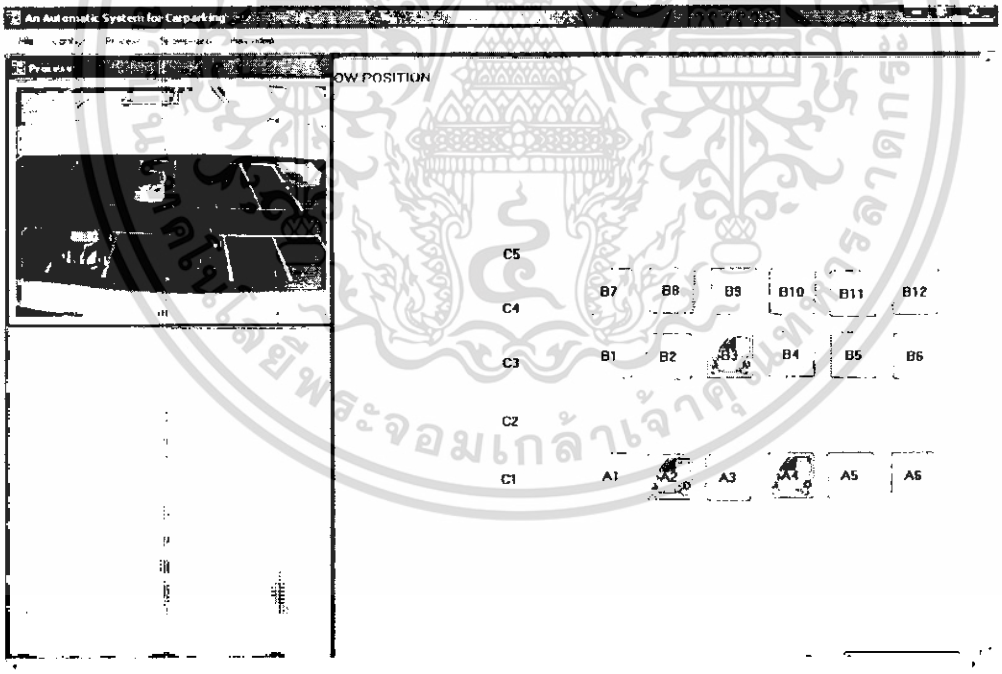


รูปที่4.4 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถจอดกรณีแสงสว่าง

4.2.2 กรณีแสงปกติ

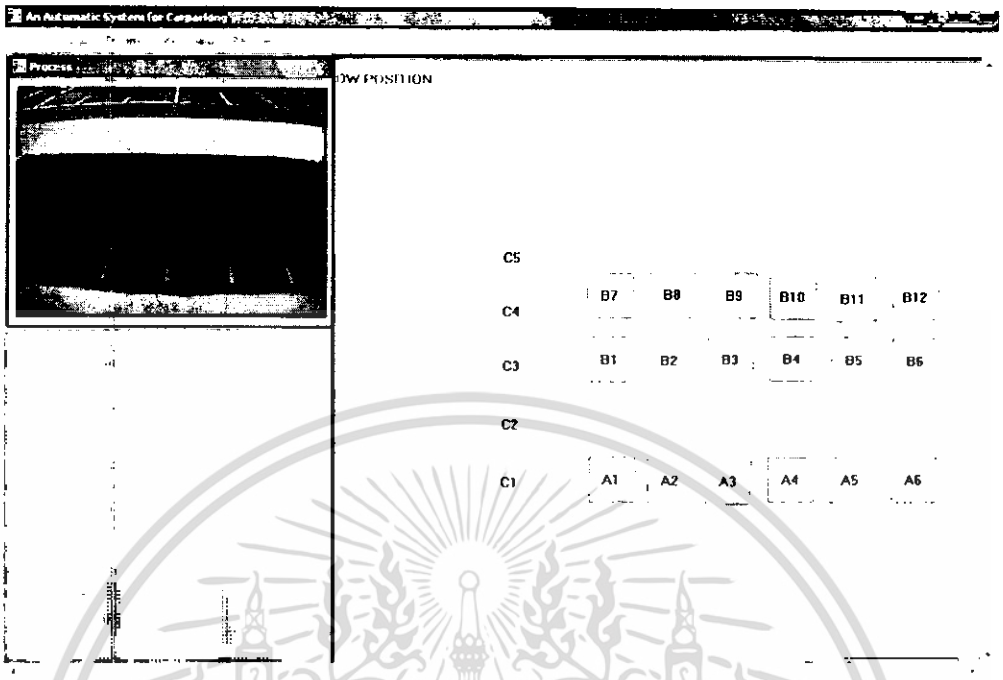


รูปที่4.5 ภาพที่จอดรถและจอมอนิเตอร์ก่อนมีรถจอดกรณีแสงปกต

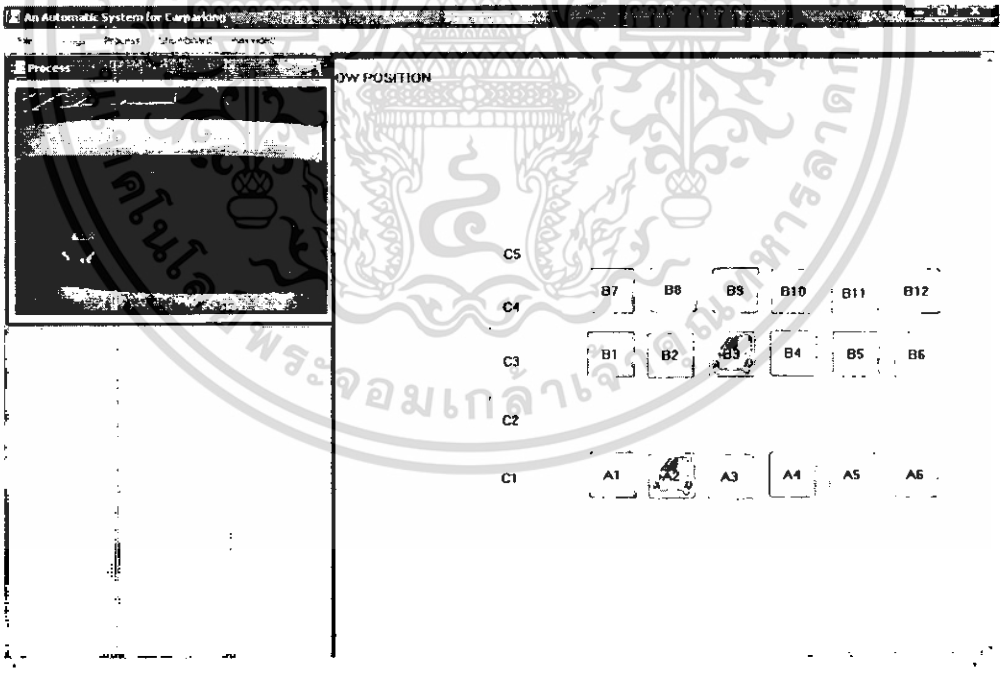


รูปที่4.6 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถจอดกรณีแสงปกติ

4.2.3 กรณีแสงน้อยมาก

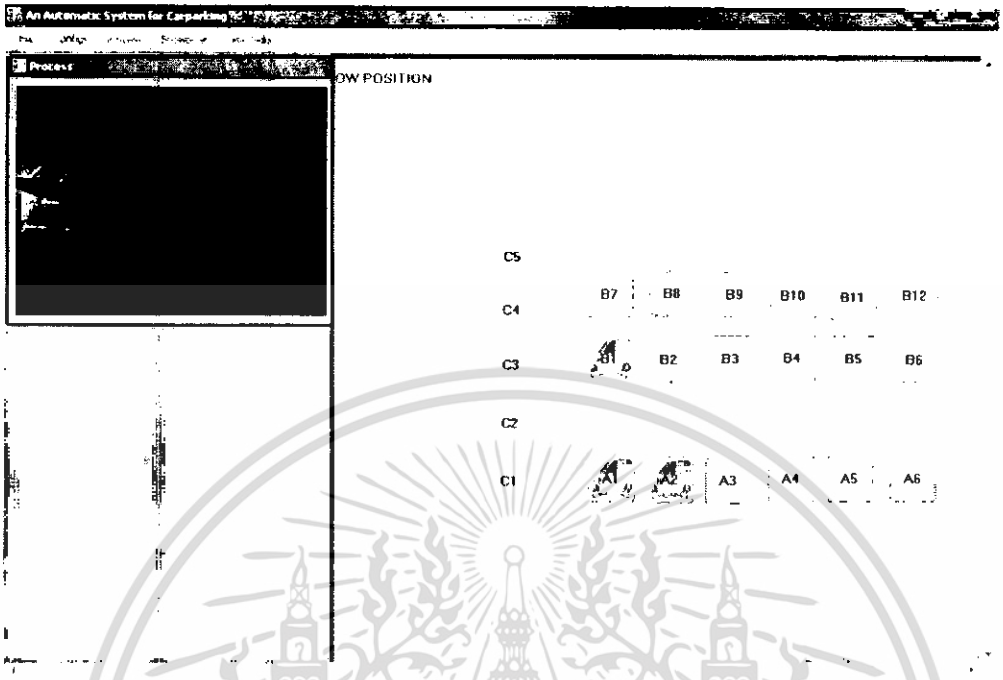


รูปที่4.7 ภาพที่จอดรถและจอมอนิเตอร์ก่อนมีรถจอดกรณีแสงน้อยมาก

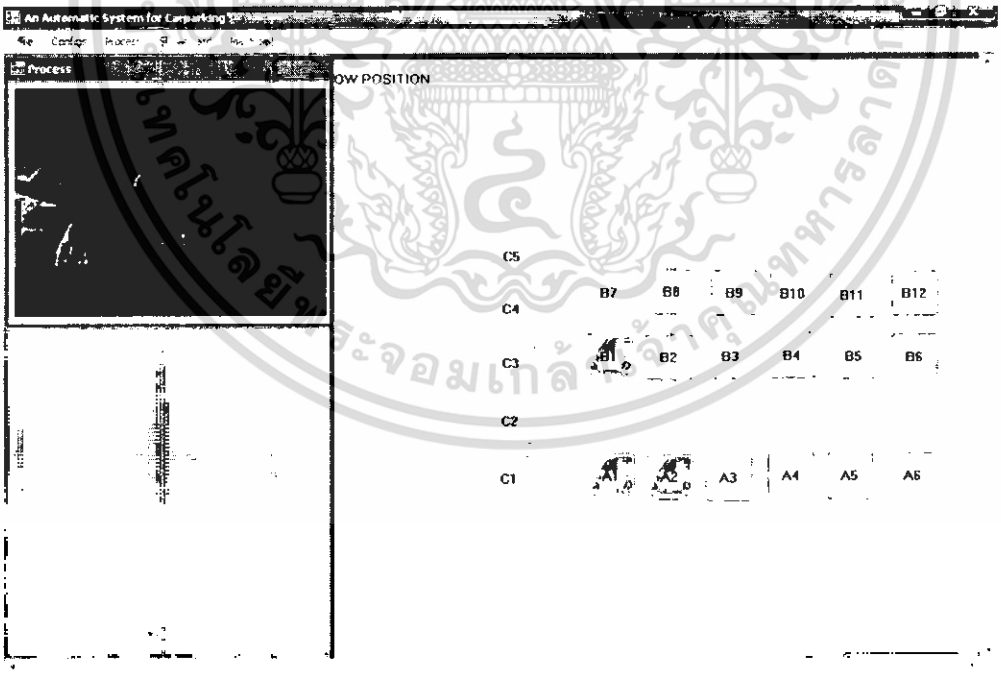


รูปที่4.8 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถจอดกรณีแสงน้อยมาก

4.2.4 กรณีแสงมืด



รูปที่4.9 ภาพที่จอดรถและจอมอนิเตอร์ก่อนมีรถจอดกรณีแสงมืด

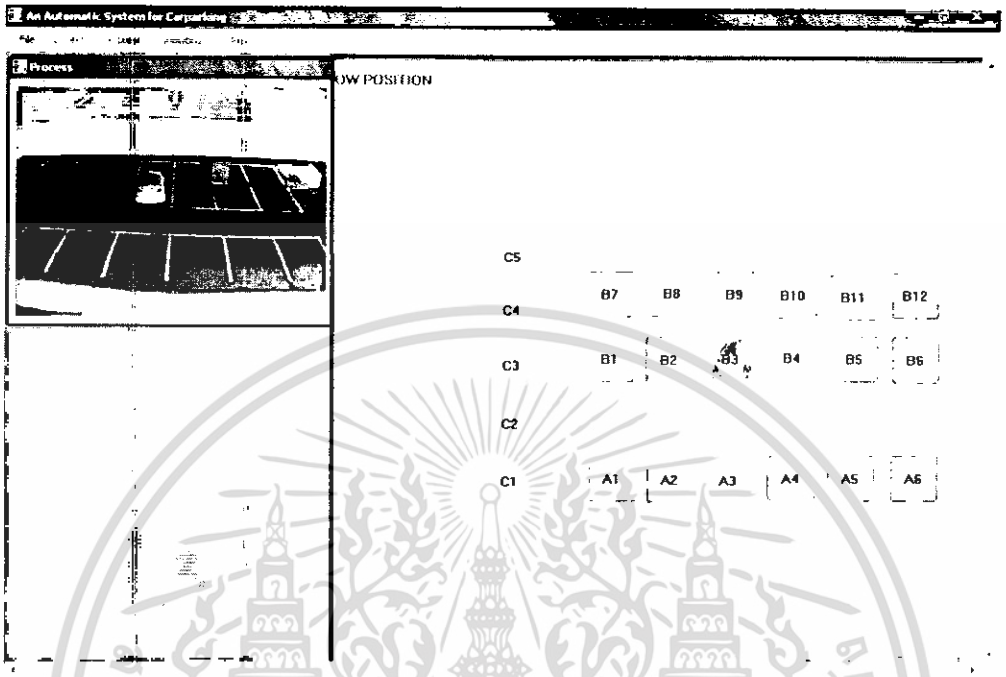


รูปที่4.10 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถจอดกรณีแสงมืด

4.3 การทดลองโปรแกรมตรวจหาตำแหน่งที่ว่างโดยใช้แสงคงที่ และสีรถปลี่ยนแปลง

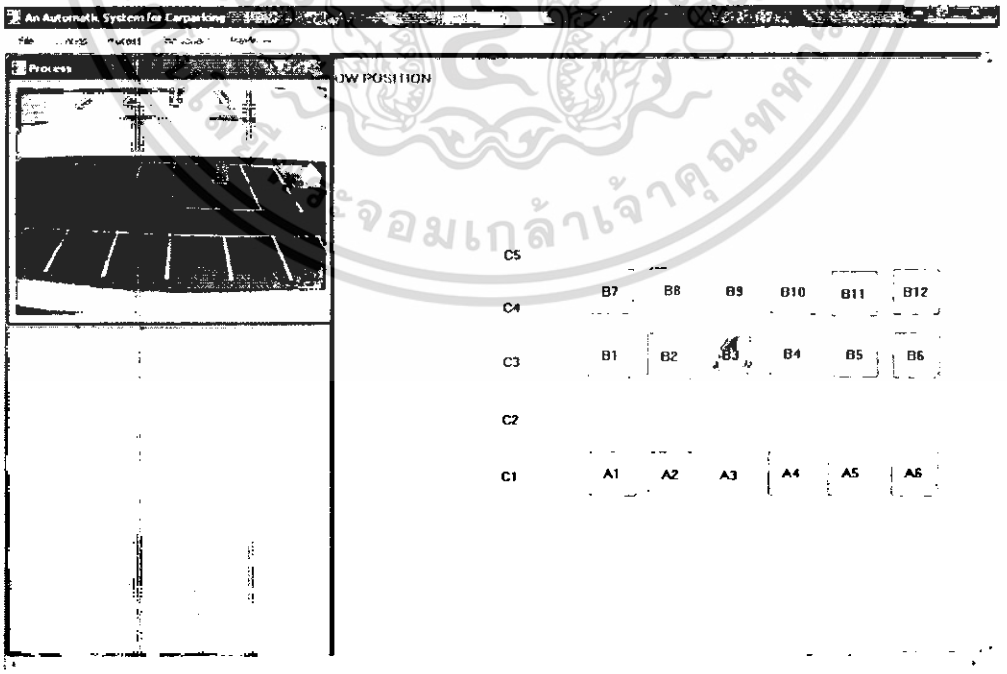
4.3.1 กรณีแสงปกติ

4.3.1.1 รถสีขาว



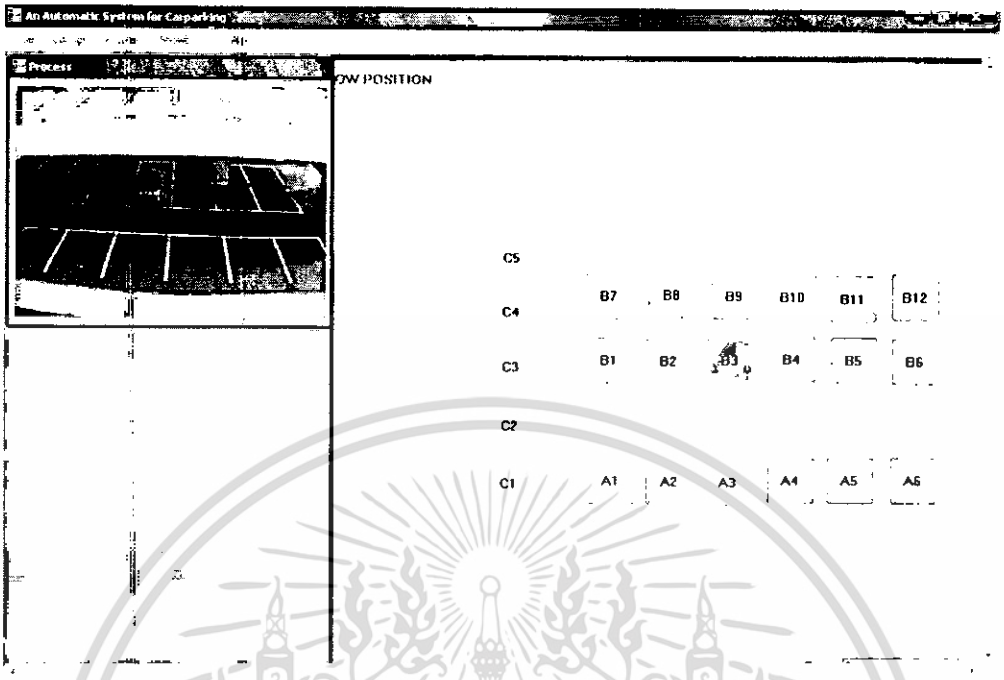
รูปที่ 4.11 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีขาวจอด

4.3.1.2 รถสีแดง



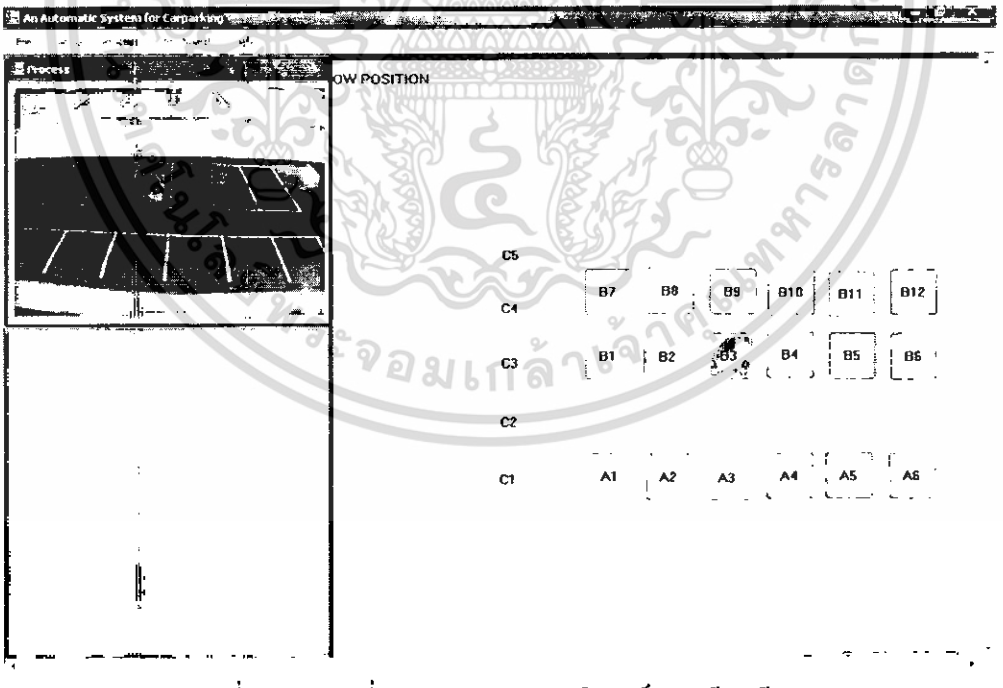
รูปที่ 4.12 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีแดงจอด

4.3.1.3 รถสีฟ้า



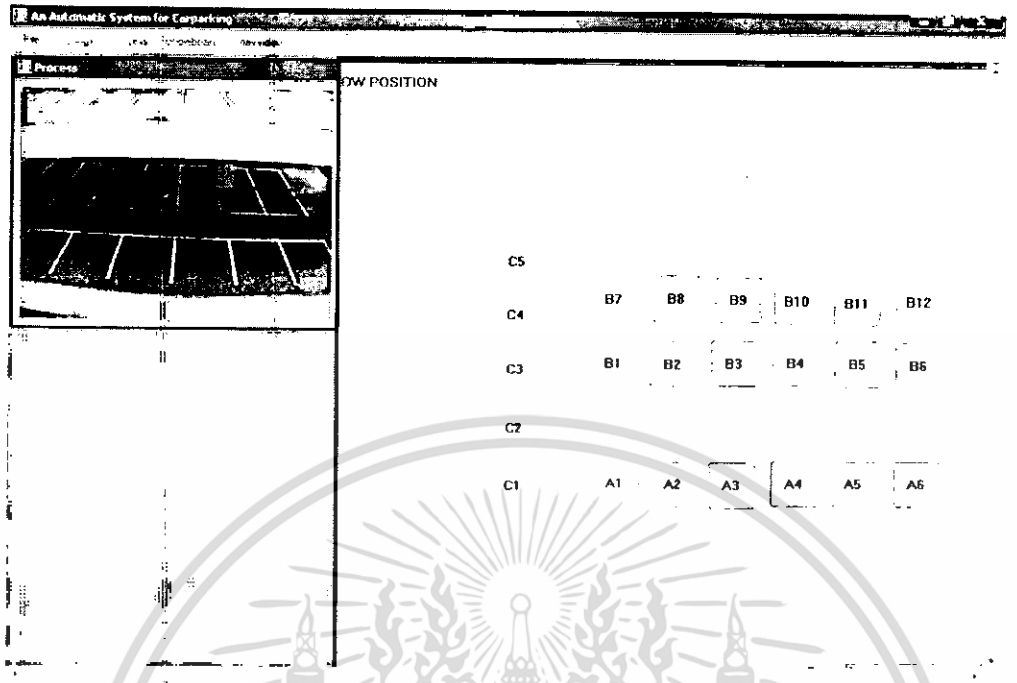
รูปที่ 4.13 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีฟ้าจอด

4.3.1.4 รถสีเทา



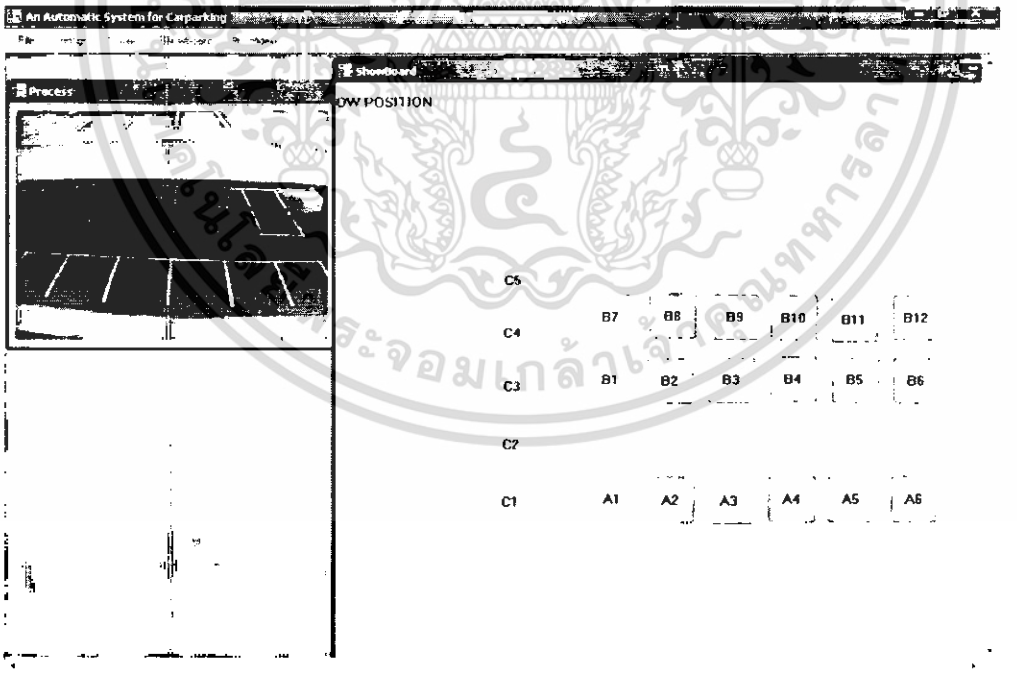
รูปที่ 4.14 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีเทาจอด

4.3.1.5 รถสีดำ



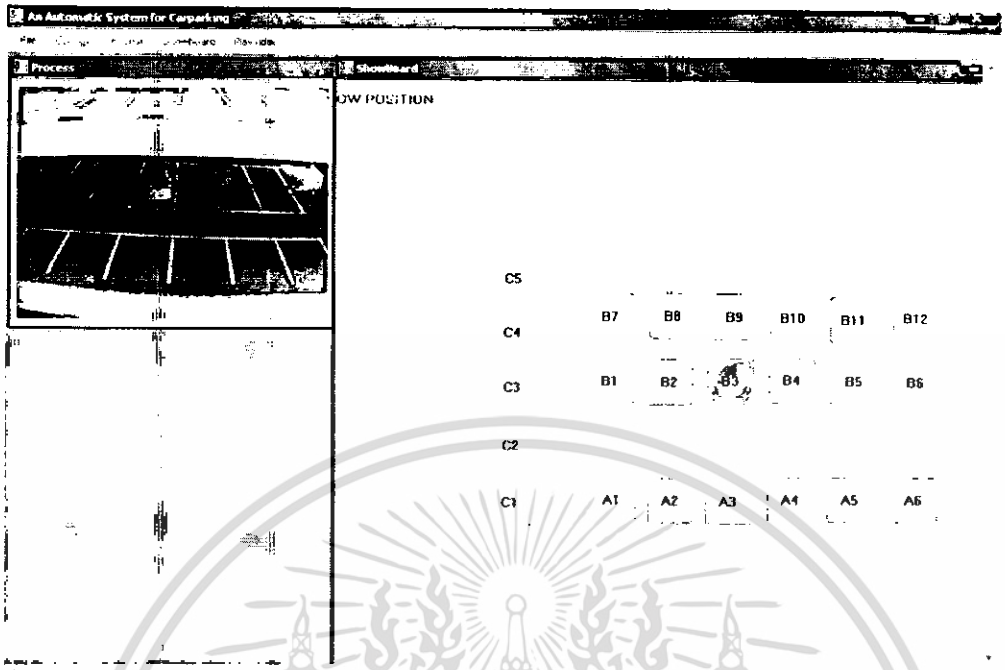
รูปที่ 4.15 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีดำจอด

4.3.1.6 รถสีเขียว



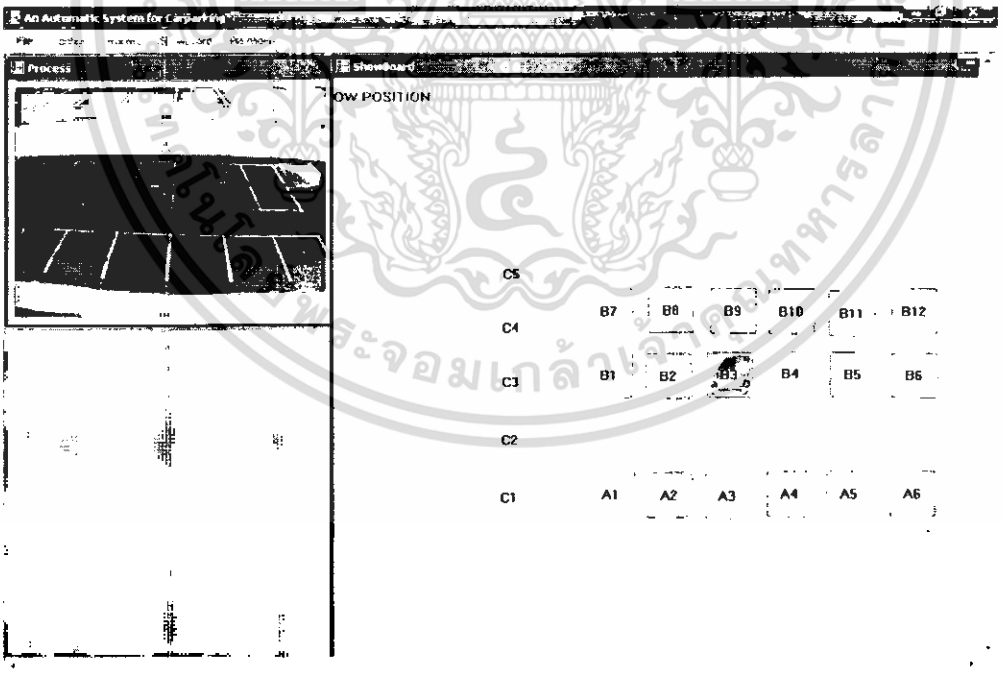
รูปที่ 4.16 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีเขียวจอด

4.3.1.7 รถสีเหลือง



รูปที่ 4.17 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีเหลืองจอด

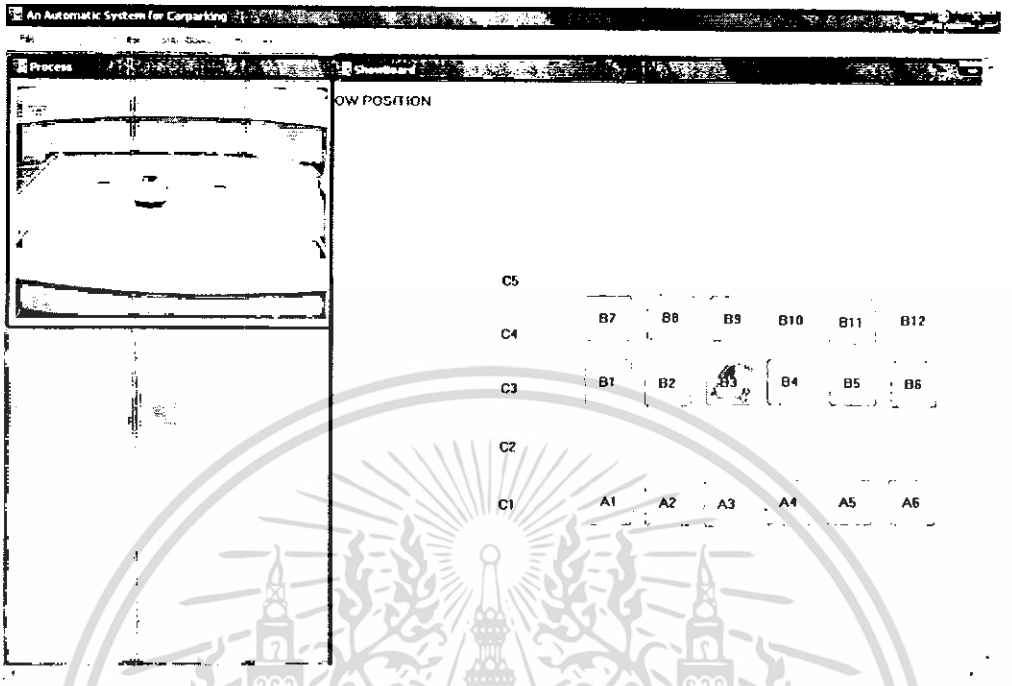
4.3.1.8 รถสีน้ำเงิน



รูปที่ 4.18 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีน้ำเงินจอด

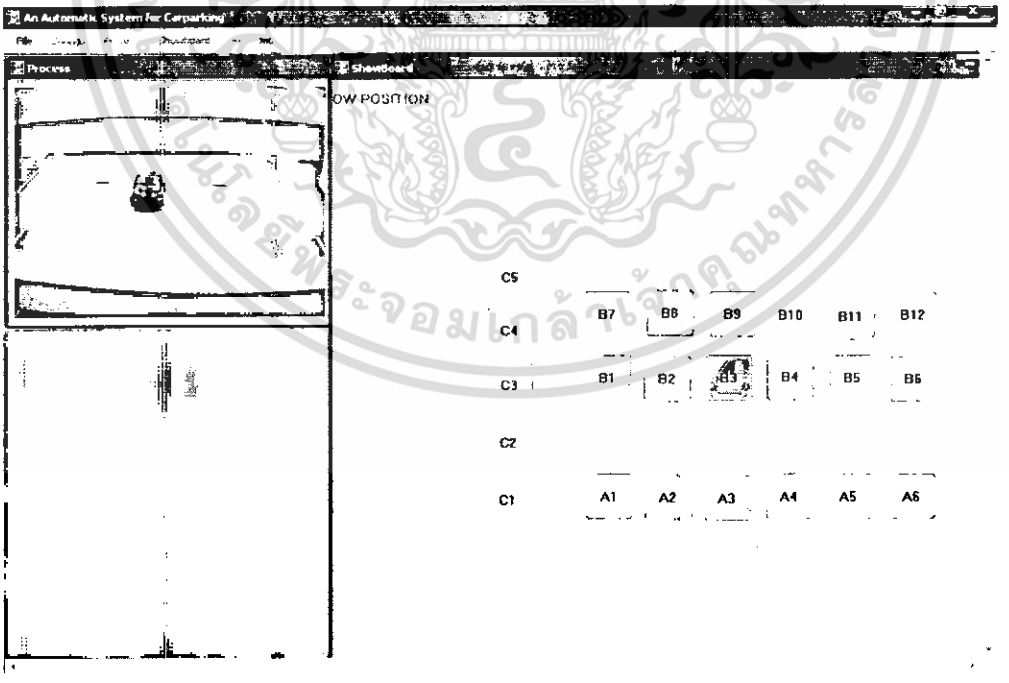
4.3.2 กรณีแสงสว่าง

4.3.2.1 รถสีขาว



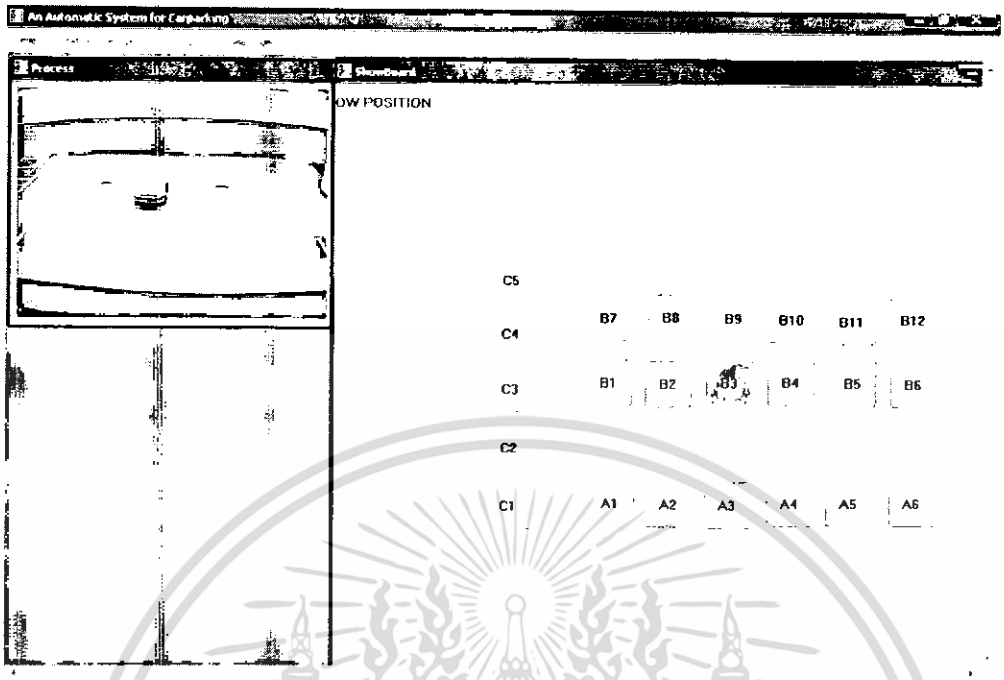
รูปที่ 4.19 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีขาวจอด

4.3.2.2 รถสีแดง



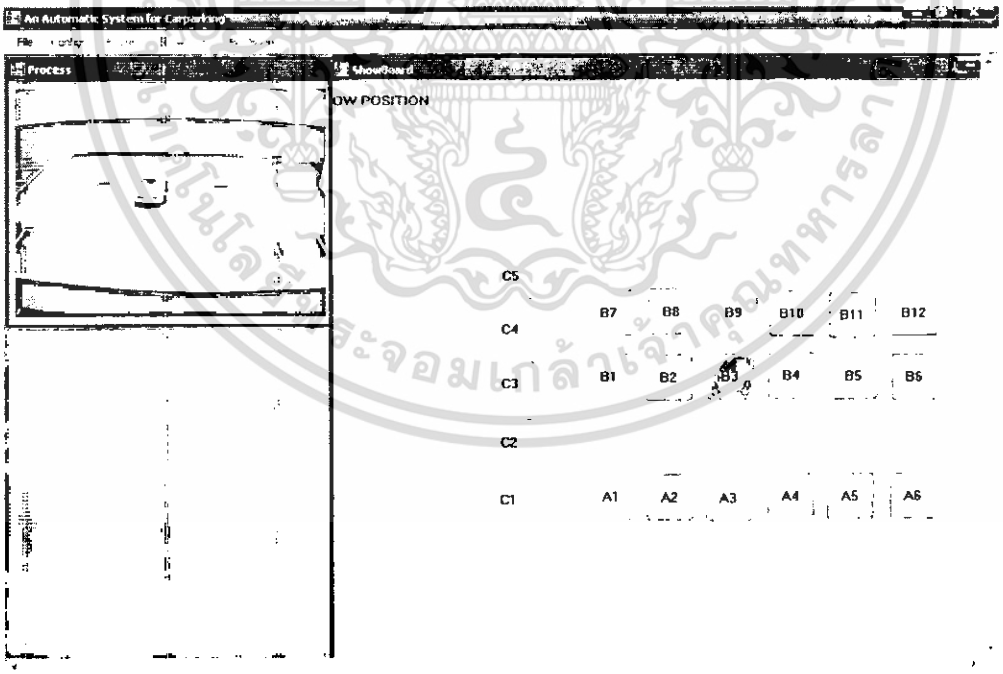
รูปที่ 4.20 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีแดงจอด

4.3.2.3 รถสีฟ้า



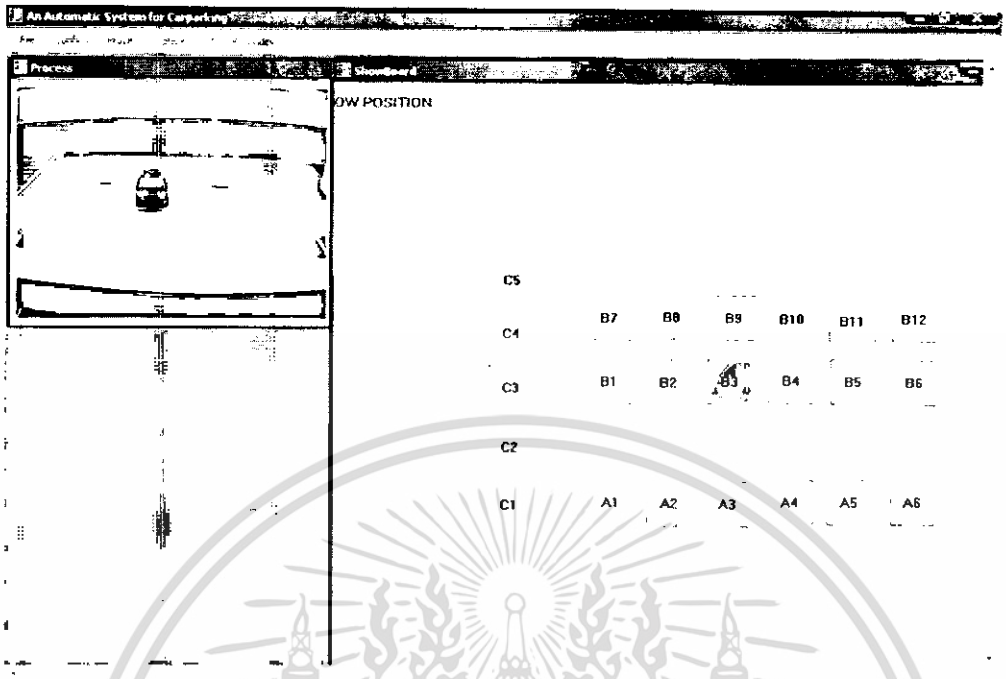
รูปที่ 4.21 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีฟ้าจอด

4.3.2.4 รถสีเทา



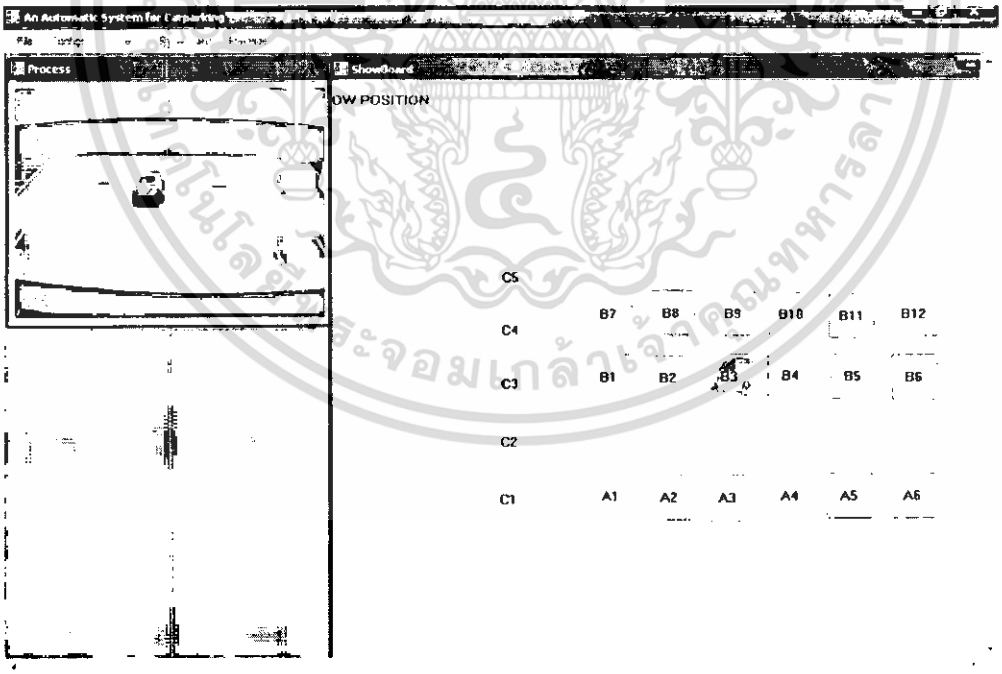
รูปที่ 4.22 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีเทาจอด

4.3.2.5 รถสีดำ



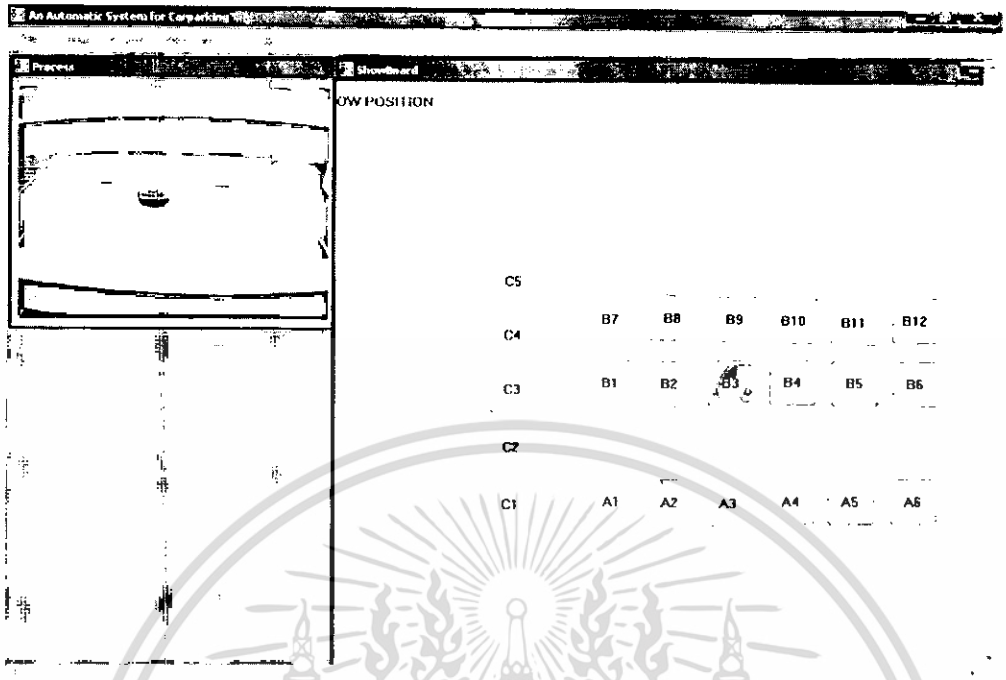
รูปที่ 4.23 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีดำจอด

4.3.2.6 รถสีเขียว



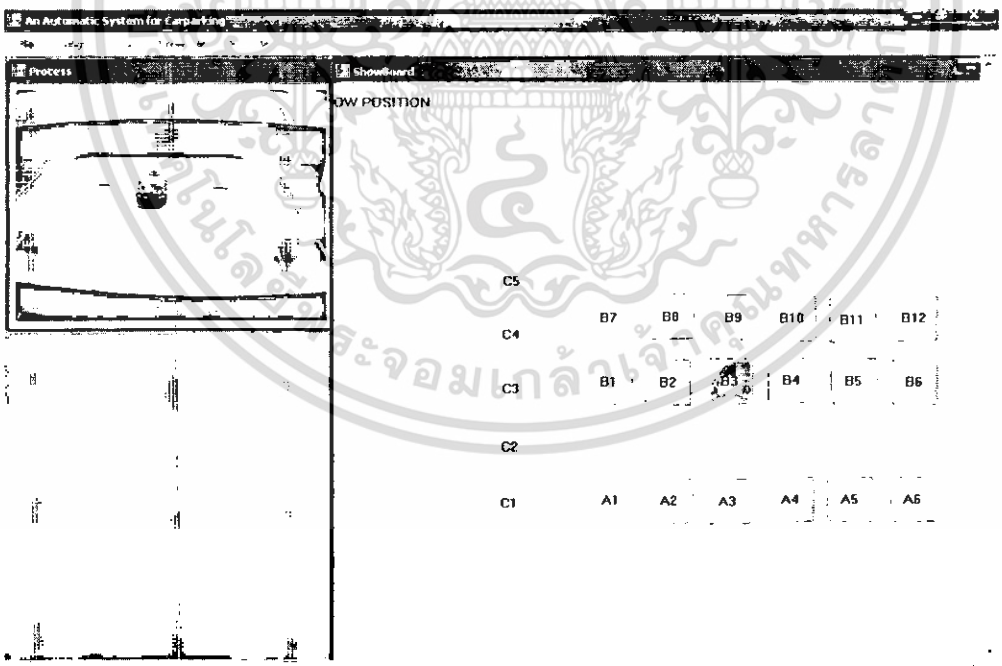
รูปที่ 4.24 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีเขียวจอด

4.3.2.7 รถสีเหลือง



รูปที่ 4.25 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีเหลืองจอด

4.3.2.8 รถสีน้ำเงิน



รูปที่ 4.26 ภาพที่จอดรถและจอมอนิเตอร์ขณะมีรถสีน้ำเงินจอด

บทที่ 5
บทวิจารณ์และสรุป

5.1 บทวิจารณ์และสรุป

จากการทดลองที่ได้แบ่งเป็น 3 ตอน โดยในตอนที่ 1 ได้สรุปผลของค่า Brightness ที่จะนำไปใช้ในการแยกแยะภาพว่ามีรถจอดหรือไม่มีรถจอดดังนี้

$$\text{Status} = \begin{cases} 1 \text{ (มีรถจอด)} & ; \text{Brightness} >= 15 \\ 0 \text{ (ไม่มีรถจอด)} & ; \text{Brightness} < 15 \end{cases}$$

ในตอนที่ 2 ได้ทำการทดลองโปรแกรมโดยให้แสงมีการเปลี่ยนแปลง 4 กรณี คือ แสงสว่าง แสงปกติ แสงน้อย และ แสงมืด โดยควบคุมให้สีของรถคงที่ ผลการทดลองพบว่าแสงมีผลต่อการแยกแยะภาพว่ามีรถจอดหรือไม่ โดยปัญหาที่พบคือแสงที่ทำให้โปรแกรมไม่สามารถแยกแยะภาพได้ว่ามีรถจอดหรือไม่คือ แสงน้อยและแสงมืด

ในตอนที่ 3 ได้แบ่งการทดลองย่อยเป็น 2 ตอน คือ ตอนที่ 1 ได้เลือกใช้แสงปกติและควบคุมแสงให้คงที่ หลังจากนั้นจึงนำรถที่มีสีต่างๆมาทดสอบ ได้แก่รถสี ขาว แดง ฟ้า เทา ดำ เขียว เหลือง และ น้ำเงิน ผลการทดลองพบว่า สามารถแยกแยะได้ว่ามีรถจอดหรือไม่ แต่มีรถสีเขียวและดำที่ไม่สามารถแยกแยะได้ว่ามีรถจอดหรือไม่ ตอนที่2 ได้เปลี่ยนไปใช้แสงสว่างและควบคุมให้คงที่เช่นเดียวกัน ผลการทดลองพบว่า สามารถแยกแยะรถได้ทุกสีว่ามีรถจอดหรือไม่

จากการทดลองทั้ง 3 ตอนทำให้สรุปได้ว่าในการที่จะแยกแยะรถได้ว่ามีรถจอดหรือไม่มีรถจอด จะต้องคำนึงถึงปัจจัย 2 อย่างด้วยกันคือ แสงในที่จอดรถ หากแสงไม่เพียงพอก็จะทำให้ไม่สามารถแยกแยะว่ามีรถจอดหรือไม่ อย่างที่สองคือ สีของรถ ถ้าหากเป็นรถสีมืดและนำไปจอดในที่มืดก็จะไม่สามารถแยกแยะได้เช่นกันว่ามีรถจอดหรือไม่ ซึ่งได้มีการแก้ปัญหาโดยการปรับปรุงภาพด้วยวิธีการ Histogram Equalization ตามหัวข้อที่ 2.7 ซึ่งจากการทดลองพบว่าวิธี Histogram Equalization สามารถแก้ปัญหาเรื่องความแตกต่างของแสง ในที่จอดรถได้อย่างมีประสิทธิภาพ ทำให้ประสิทธิภาพของโปรแกรมดียิ่งขึ้น

5.2 **ปัญหาและอุปสรรคที่พบ**

- ปัญหาเรื่องแสง ซึ่งเป็นตัวแปรที่เปลี่ยนไปตามสิ่งแวดล้อมตามช่วงเวลาซึ่งถ้าแสงน้อยมากจะทำให้ประสิทธิภาพของระบบลดลง
- ปัญหาเรื่องสีของรถที่เข้ามาจอด ซึ่งถ้าแสงน้อยและรถที่จอดมีสีดำหรือสีมืดจะทำให้กลมกลืนกับฉากหลัง ทำให้ประสิทธิภาพของระบบลดลง
- ปัญหาการวางมุมกล้อง การวางมุมกล้องที่สูงจะทำให้ประสิทธิภาพดีกว่าการวางมุมกล้องที่ต่ำ เพราะจะลดการบังกล้องในที่จอดรถ

5.3 **แนวทางการศึกษาต่อ**

- นำไปทดสองติดตั้งกับสถานที่จริง เพื่อหาค่าพารามิเตอร์ต่างๆที่เหมาะสมเพื่อให้โปรแกรมมีความสมบูรณ์มากยิ่งขึ้น
- เปลี่ยนจากกล้อง Web Cam ไปเป็นกล้องวงจรปิดเพื่อให้ได้ความคงทนและความรวดเร็วในการตอบสนองการใช้งาน
- ศึกษาหาวิธีอื่นที่ทำให้สามรถตรวจสอบวัตถุที่เป็นรถได้
- เพิ่มเติมในส่วนการใช้งานการเก็บค่าจอดรถให้เป็นระบบอัตโนมัติมากยิ่งขึ้น

บรรณานุกรม

Alasdair McAndrew . **Introduction to Digital Image Processing with Matlab** .United States of America:Course Technology.

พร้อมเลิศ หล่อวิจิตร .2549.**คู่มือเรียน Visual Basic 2005 .** กรุงเทพฯ : โปรวิชั่น.

พงศธร เก้ารัตน์ และ พัฒนวัฒน์ เรืองวัฒนไพศาล " ระบบลานจอดรถยนต์ไร้ผู้ควบคุม" วิทยานิพนธ์วิศกรรมศาตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง 2548.

ภาคผนวก

ภาคผนวก ก.

กล้อง

กล้องที่เลือกใช้เป็นกล้อง Webcam เพราะมีราคาถูกและสามารถใช้ในการเก็บภาพได้คุณภาพดีพอสมควร ยี่ห้อที่เลือกใช้คือ OKER ดังรูป

รูปที่ ก-1 รูปกล้องยี่ห้อ OKER

คุณสมบัติของกล้อง

- High-quality VGA CMOS sensor
- Manual focus
- Video capture: up to 640*480 pixels
- Still image capture: up to 640*480 pixels
- Frame rate: up to 30 frame per second
- Price : 550

คุณสมบัติที่เลือกใช้

- Video capture: 320*240 pixels
- Frame rate: 5 frame per second
- Picture Grayscal 256 level

ActiveX Control (VideoOCX และ VideoOCXTools)

ในปัจจุบันกล้อง web cam ถือได้ว่าเป็นอุปกรณ์ที่ได้รับความนิยมจากผู้ใช้เป็นจำนวนมาก จึงได้มีผู้พัฒนาเครื่องมือต่างๆที่ช่วยในการเขียนโปรแกรมที่ใช้ในการติดต่อกับกล้อง web cam ออกมามากมาย เพื่อตอบสนองต่อความต้องการของผู้ใช้ ซึ่งในโครงงานนี้ได้เลือกใช้เครื่องมือ VideoOCX ซึ่งเป็น ActiveX control ที่ช่วยให้ผู้เขียนโปรแกรมนั้น สามารถเขียนโปรแกรมติดต่อกับกล้อง web cam เพื่อใช้ในการจับภาพและทำ image processing ได้อย่างง่ายดาย โดยเพียงกำหนดค่าตัวแปรให้กับ VideoOCX และเรียกใช้งานผ่าน method ที่มีให้มา ตัวอย่าง method ที่สำคัญได้แก่

Method ที่สำคัญ

Syntax : BOOL *object*.**Start**()

Description : initiate the background capture process

Parameter : none

Return Type : true if successful

Syntax : BOOL *object*.**Stop**()

Description : end the internal capturing process

Parameter : none

Return Type : true if the internal capture process was stopped

Syntax : BOOL *object*.**ShowDriverDlg**()

Description : open dialog that enables the user to choose a video driver

Parameter : none

Return Type : true if successful

Syntax : void *object*.**SetErrorMessages**(BOOL *flag*)

Description : display every error that occurs

Parameter :true to turn error messages on

Return Type : none

Syntax : BOOL *object*.**Init**()

Description : initialize VideoOCX and connect to the specified image source

Parameter : none

Return Type : true if successful

Syntax : LPCTSTR *object*.**GetLastErrorString**()

Description : returns the message of the last occured error

Parameter : none

Return Type : error message

Syntax : long *object*.**GetGrayImageHandle**()

Description : receive an empty grayscale image

Parameter : none

Return Type : handle to the new allocated image

Syntax : long *object*.**GetColorImageHandle**()

Description : receive an empty color image

Parameter : none

Return Type : handle to the new allocated image

Syntax : BOOL *object*.**ToGray**(*long colorimagehandle, long grayimagehandle*)

Description : convert a color image to grayscale

Parameter : color image handle and grayscale image handle

Return Type : true if successful

Syntax : long *object*.**GetDataPointer**(*long imagehandle*)

Description : get a pointer to the image data

Parameter : image handle

Return Type : address of image data

Syntax : long *object*.**Capture**(*long imagehandle*)

Description : capture the current video image

Parameter : current video image is copied into the handle

Return Type : address to image data array

Syntax : VOID *object*.**Show**(*long imagehandle*)

Description : display any image in the control

Parameter : handle to the image

Return Type : none

Syntax : VARIANT *object*.**GetMatrix**(*long imagehandle*)

Description : get dimensional array to the image pixels

Parameter : handle to an image

Return Type : 2 or 3 dimensional matrix containing the image pixels

Syntax : void *object*.**ReleaseMatrixToImageHandle**(*long imagehandle*)

Description : release memory for the last matrix

Parameter : handle to and image

Return Type : none

Syntax : void *object*.**ReleaseImageHandle**(*long imagehandle*)

Description : release the memory for an image

Parameter : handle to an image

Return Type : none

Syntax : BOOL *object*.**Close**()

Description : closes the current connection

Parameter : none

Return Type : true if successful

Syntax : BOOL *object*.**ToPicture**(*long imagehandle*)

Description : assign image handle any picture to a picture box

Parameter : imagehandle Handle to an image

Return Type : True if successfull

Syntax : BOOL *object*.**ShowSourceDlg**()

Description : user to choose the video source and to adjust some image parameters

Parameter : none

Return Type : true if successful

Syntax : BOOL *object*.**AVISaveMovieInit**(*String filename*)

Description : Initialize continuoud capture into an AVI movie

Parameter : String filename

Return Type : True if SaveMovie mode was initialized

Syntax : BOOL *object*.**AVISaveMovieStart**()

Description : Start contunous capture into an AVI file

Parameter : None

Return Type : True if Capture was started

Syntax : BOOL *object*.**AVISaveMovieStop**()

Description : Stop contunous capture into an AVI file

Parameter : None

Return Type : True if Capture was stopped

Syntax : BOOL *object*.**AVISaveMovieClose**()

Description : Close continuous AVI-Movie capture mode

Parameter : None

Return Type : True if AVI capture was closed successfully

Syntax : BOOL *object*.**AVIPlaybackMovieInit**(*String filename*)

Description : Initialize AVI Movie playback mode

Parameter : String filename

Return Type : True if AVI movie was opened successfully

Syntax : BOOL *object*.**AVIPlaybackMovieStart**()

Description : Start to playback an AVI movie.

Parameter : None

Return Type : True if movie started successfully

Syntax : BOOL *object*.**AVIPlaybackMovieRewind**()

Description : Allows you to restart AVI movie playback (seek to the first frame).

Parameter : None

Return Type : True if successfully

Syntax : BOOL *object*.**AVIPlaybackMovieStop**()

Description : Stop AVI playback started with

Parameter : None

Return Type : True if movie was stopped

Syntax : BOOL *object*.**AVIPlaybackMovieGetLen**()

Description : Returns the lenght (in number of frames) of the current AVI movie opened

Parameter : None

Return Type : Long Number of frames in the AVI

Syntax : BOOL *object*.**SobelMagnitude**(*ImageHandle image, ImageHandle magnitude*)

Description : Use **SobelMagnitude** to detect edges

Parameter : image The original image (has to be grayscale)

Magnitude Sobel magnitude (has to be grayscale)

Return Type : True if successfully

Syntax :

BOOL *object*.**Threshold** (*ImageHandle image, long threshold, ImageHandle binary*)

Description : The **Threshold** method sets all pixel in *image* to 0 all 255

Parameter : image The original image (has to be grayscale)

Threshold Threshold value

Binary The binary result image (has to be grayscale)

Return Type : True if successfully