ระบบจดจำทะเบียนรถยนต์

Car Plate Recognition System

โดย

นางสาวอรพินท์ ฐิตรัตนพล

นายเอกวัฒน์ เบญจพรกุลนิจ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต

สาขาวิชาอิเล็คทรอนิกส์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2544

ระบบจดจำทะเบียนรถยนต์

Car Plate Recognition System

โดย

นางสาวอรพินท์ ฐิตรัตนพล เลขประจำตัว 41014523

นายเอกวัฒน์ เบญจพรกุลนิจ เลขประจำตัว 41014551

อาจารย์ที่ปรึกษา

รศ.ดร.สุรพันธ์ เอื้อไพบูลย์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต

สาขาวิชาอิเล็คทรอนิกส์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2544

ปริญญานิพนธ์ปีการศึกษา2544

ภาควิชาอิเล็กทรอนิกส์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบจดจำป้ายทะเบียนรถยนต์ (Car Plate Recognition System)

ผู้จัดทำ

1. นางสาวอรพินท์ ฐิตรัตนพล รหัส 41014523
2. นายเอกวัฒน์ เบญจพรกุลนิจ รหัส 41014551

.....

(รศ.ดร.สุรพันธ์ เอื้อไพบูลย์)

อาจารย์ที่ปรึกษา

ปริญญานิพนธ์เรื่อง ระบบจดจำทะเบียนรถยนต์

Car Plate Recognition System

จัดทำโดย นางสาวอรพินท์ ฐิตรัตนพล รหัส 41014523

นายเอกวัฒน์ เบญจพรกุลนิจ รหัส 41014551

อาจารย์ที่ปรึกษา รศ.ดร. สุรพันธ์ เอื้อไพบูลย์

ปริญญานิพนธ์นี้ได้ผ่านการตรวจสอบโดยอาจารย์ที่ปรึกษาแล้ว

ลงชื่อ.....

(รศ.ดร. สุรพันธ์ เอื้อไพบูลย์)

อาจารย์ที่ปรึกษา

ระบบจดจำทะเบียนรถยนต์

นางสาวอรพินท์ ฐิตรัตนพล
นายเอกวัฒน์ เบญจพรกุลนิจ
รศ.ดร. สุรพันธ์ เอื้อไพบูลย์ (อาจารย์ที่ปรึกษา)
ปีการศึกษา 2544

บทคัดย่อ

ในรายงานฉบับนี้กล่าวถึงหลักการและทฤษฎีต่างๆ เกี่ยวกับการประมวลผลภาพ และการจดจำรูปแบบตัวอักษร ได้กล่าวถึงรายละเอียดของโครงงานที่ได้ทำ นั่นคือโปรแกรมแยกหมายเลขทะเบียนรถยนต์ออกจากภาพ ซึ่งประกอบไปด้วยส่วนของการหาขอบภาพ การกำจัดสิ่งรบกวนของภาพ การแปลงภาพสีเป็นภาพขาวดำ การหาขอบเขตของป้ายทะเบียน การหาขอบเขตของหมายเลขทะเบียน การจดจำรูปแบบของตัวอักษรโดยวิธีตัดและจำแนก และวิธีเทียบจุดภาพ รวมทั้งการทดลองและสรุปผลการทดลอง

Carplate Recognition System

Miss. Orapin Titarattanaphol

Mr. Ekkawat Benjapornkunlanij

Assot. Prof. Dr. Surapan Airpaiboon (Advisor)

Educational Year 2001

ABSTRACT

This report concerns about principles and theory of digital image processing and character recognition and including the detail of designing the "Car plate Recognition System" which consist of edge detection, noise reduction, binarization, plate segmentation, character segmentation and character recognition system using cut and classify method and matching method. We also test the program and conclude the results of the test.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้เป็นอย่างดีด้วยคำแนะนำและคำปรึกษาจากอาจารย์ สุรพันธ์ เอื้อไพบูลย์ ซึ่งเป็นอาจารย์ที่ปรึกษาในการทำงาน ผู้จัดทำรู้สึกซาบซึ้งในความอนุเคราะห์จากท่านและกราบขอบพระคุณอย่างสูง

ขอบพระคุณเพื่อนๆทุกๆคนที่ช่วยเหลือ ให้คำแนะนำต่างๆ และยังให้กำลังใจตลอดมา

คุณค่าและประโยชน์อันพึงมีจากปริญญานิพนธ์นี้ ผู้จัดทำขอมอบแด่ผู้มีพระคุณทุกๆท่าน

อรพินท์

.....

(นางสาวอรพินท์ ฐิตรัตนพล)

เอกวัฒน์

.....

(นายเอกวัฒน์ เบญจพรกุลนิจ)

สารบัญ

สารบัญรูป

หน้า

บทที่ 1
บทนำ

ในปัจจุบันรถยนต์มีจำนวนเพิ่มมากขึ้น ทำให้เป็นการไม่สะดวกและสิ้นเปลืองบุคลากรในการบันทึกข้อมูลทะเบียนรถยนต์เหล่านั้นในงานต่างๆ เช่น การเก็บค่าผ่านทางด่วน การให้เช่าสถานที่จอดรถ เป็นต้น จึงได้มีแนวความคิดที่จะพัฒนาระบบจดจำทะเบียนรถยนต์โดยอัตโนมัติขึ้น โดยใช้หลักการเปลี่ยนข้อมูลที่อยู่ในลักษณะรูปภาพ(Image Data) ให้อยู่ในรูปการจัดเก็บข้อมูลในรูปตัวหนังสือ(Text Data) เพื่อนำข้อมูลที่ได้ไปใช้กับระบบฐานข้อมูล(Database) หรือประยุกต์ใช้กับงานอื่นๆ ได้ โดยอาศัยหลักการประมวลผลภาพ(Image Processing) และการจดจำรูปแบบ(Pattern Recognition) เป็นหลักสำคัญในการออกแบบ

1.1 วัตถุประสงค์ของโครงงาน

- เพื่อศึกษาหลักการประมวลผลภาพ
- เพื่อศึกษาหลักการจดจำรูปแบบ
- เพื่อศึกษาหลักการเขียนโปรแกรมประยุกต์ใช้งานด้วยภาษาเดลไฟ

1.2 ขอบเขตของโครงงาน

ระบบจดจำป้ายทะเบียนที่นำเสนอนี้ จะนำข้อมูลภาพของทะเบียนรถยนต์เป็นข้อมูลเพื่อใช้ในการหาตำแหน่งของหมายเลขทะเบียนรถยนต์โดยอาศัยขั้นตอนการประมวลผลภาพ เช่น การกำจัดสัญญาณรบกวน การทำไบนาไรเซชัน(Image Binarization) การจำแนกภาพ(Image Segmentation) แล้วนำผลลัพธ์ที่ได้เข้าสู่กระบวนการจดจำรูปแบบของหมายเลขทะเบียนรถยนต์ เป็นการทำให้คอมพิวเตอร์สามารถรู้จักและเข้าใจภาพได้(Recognition)

ปริญญานิพนธ์ฉบับนี้ได้เสนอวิธีการแยกหมายเลขทะเบียนรถยนต์จากข้อมูลภาพเพื่อเตรียมข้อมูลเข้าสู่ขั้นตอนการจดจำรูปแบบโดยนำข้อมูลภาพด้านหน้าและด้านหลังรถยนต์ที่มีการเก็บข้อมูลในการวิเคราะห์เพื่อแยกเฉพาะส่วนของป้ายทะเบียนออกจากข้อมูลภาพทั้งหมด จากนั้นจึงนำป้ายทะเบียนที่ได้มาแยกเฉพาะส่วนของตัวอักษร โดยมีขั้นตอนต่างๆ ดังรูปที่1.1

หลังจากผ่านขั้นตอนของการแยกหมายเลขทะเบียนรถยนต์แล้ว ข้อมูลภาพของหมายเลขทะเบียนรถยนต์จะถูกนำเข้าสู่กระบวนการจดจำรูปแบบ เพื่อทำการประมวลผลข้อมูลและให้ผลลัพธ์

รูปที่ 1.1 ขั้นตอนการทำงานของโปรแกรม

ออกมาเป็นค่าของตัวเลขหรือตัวอักษร ในรายงานฉบับนี้ได้เสนอวิธีการจดจำตัวอักษรโดยใช้หลักการ 2 แบบ คือ การตัดและจำแนก และการเทียบค่าจุดภาพ ซึ่งรายละเอียดการทำงานดังจะอธิบายต่อไป

1.3 การประยุกต์ใช้งานของระบบจดจำป้ายทะเบียนรถยนต์

- ระบบเก็บค่าใช้บริการทางด่วน

ระบบจดจำเลขทะเบียนรถยนต์สามารถนำไปประยุกต์ใช้กับฐานข้อมูลของระบบค่าใช้บริการทางด่วน โดยเจ้าของรถยนต์จะมีบัญชีเพื่อหักค่าบริการทางด่วน เมื่อรถยนต์ผ่านเข้าไปใช้บริการทางด่วน ระบบจดจำเลขทะเบียนรถยนต์จะประมวลผลเพื่อระบุเลขทะเบียนรถยนต์ แล้วจัดการกับข้อมูลของรถยนต์คันนั้นในฐานข้อมูล เช่น การหักบัญชีค่าบริการทางด่วนจากเจ้าของรถยนต์โดยอัตโนมัติ ซึ่งทำให้สามารถลดจำนวนพนักงานเก็บค่าทางด่วนและเพิ่มความสะดวกสบายให้กับผู้ใช้บริการทางด่วน

- ระบบป้องกันการโจรกรรมรถยนต์

ในสถานที่ที่มีความเสี่ยงต่อการโจรกรรมรถยนต์ เช่น ในศูนย์การค้าสามารถใช้ระบบจดจำเลขทะเบียนรถยนต์เพื่อตรวจเช็คการเข้าออกของรถยนต์ โดยเมื่อมีรถยนต์เข้ามาระบบจะระบุหมายเลขรถยนต์ และออกบัตรจอดรถที่มีหมายเลขทะเบียนตรงกัน ดังนั้นรถยนต์จะสามารถออกไปได้เมื่อมีบัตรตรงกับหมายเลขทะเบียนรถยนต์เท่านั้น

นอกจากนี้ยังสามารถประยุกต์ใช้ร่วมกับงานของเจ้าหน้าที่ตำรวจในการติดตามรถยนต์ที่ต้องการ เช่น รถยนต์ที่ถูกโจรกรรม หรือรถยนต์ที่ผู้กระทำผิดใช้โดยติดตั้งระบบตามด่านตรวจต่างๆ เพื่อให้สามารถติดตามรถยนต์ได้อย่างรวดเร็ว

- ระบบการควบคุมการเข้าออกสถานที่สำคัญ

ในสถานที่สำคัญ เช่น เขตทหารจะไม่สามารถให้รถยนต์ที่ไม่ได้รับอนุญาตเข้าไปในสถานที่นั้นได้ ดังนั้นในระบบควบคุมการเข้าออกสถานที่จะใช้ระบบจดจำทะเบียนรถยนต์เพื่อตรวจสอบ

หมายเลขทะเบียนของรถยนต์นั้นว่าตรงกับหมายเลขทะเบียนของรถยนต์ในระบบฐานข้อมูลที่สามารถเข้าไปในสถานที่นั้นได้หรือไม่

บทที่ 2
การประมวลผลภาพ

2.1 การประมวลผลภาพเชิงตัวเลข(Digital Image Processing)

การประมวลผลภาพเชิงตัวเลข หมายถึง การนำภาพที่พบทั่วไปมาประมวลผลด้วยเครื่องคอมพิวเตอร์ โดยภาพที่นำมาประมวลผลด้วยเครื่องคอมพิวเตอร์นี้จะถูกแทนที่ให้อยู่ในรูปแบบของเมตริกซ์ แต่ภาพที่ได้โดยส่วนมากแล้วจะเป็นภาพที่ได้รับจากตัวรับสัญญาณ ซึ่งอยู่ในรูปของฟังก์ชัน f(x,y) ที่ต่อเนื่องในระนาบสองมิติ โดยจะเป็นสัดส่วนกับค่าความสว่างหรือความเข้มของภาพที่ตำแหน่ง (x,y) ซึ่งเรียกว่า ระดับสีเทา (Gray Level)

2.1.1 การแทนภาพด้วยข้อมูลแบบดิจิตอล

ภาพข้อมูลแบบดิจิตอล(Digital Image) เป็นภาพที่ถูกแปลงมาจากอนาลอก อยู่ในรูปของตัวเลข โดยภาพอนาลอกถูกแบ่งเป็นพื้นที่สี่เหลี่ยมเล็กๆ ที่เรียกว่า พิกเซล(Pixels) ในแต่ละพิกเซลจะถูกระบุตำแหน่งโดย (x,y) และค่าระดับสีเทาของพิกเซล โดยเราสามารถแปลงภาพเป็นข้อมูลแบบดิจิตอลได้ โดยมีขั้นตอนและวิธีการดังนี้

เมื่อเรานำสัญญาณอนาลอกที่ต้องการประมวลผลมาผ่านส่วนที่เรียกว่า ดิจิไตเซอร์ (Digitizor) ซึ่งจะมีหน้าที่ในการเปลี่ยนสัญญาณอนาลอกให้เป็นสัญญาณดิจิตอล อุปกรณ์ส่วนนี้ได้แก่ กล้องโทรทัศน์ดิจิไตเซอร์ จากนั้นทำการ ควอนไตซ์(Quantizing) เพื่อที่จะประมวลสัญญาณด้วยระบบคอมพิวเตอร์ ฟังก์ชันของภาพ f(x,y) จะถูกทำให้เป็นสัญญาณไม่ต่อเนื่องทั้งระนาบของภาพ ซึ่งเราเรียกว่า การสุ่มภาพ(Image Sampling) ของฟังก์ชันที่ได้เรียกว่า การ ควอนไตเซชัน(Gray Level Quantization) ก็จะได้ข้อมูลที่เป็นดิจิตอล

สมมุติว่าสัญญาณภาพต่อเนื่อง f(x,y) ถูกดิจิไตซ์ในระนาบ X และ Y เป็นช่วงเท่าๆ กัน เราสามารถจัด f(x,y) ให้อยู่ในรูปของเมตริกซ์ขนาด N×N ได้ดังสมการที่ 2.1

$$f(x,y) = \begin{cases} f(0,0) & f(0,1) & f(0,2) & \cdots\cdots & f(0,N-1) \\ f(1,0) & f(1,1) & f(1,2) & \cdots\cdots & f(1,N-1) \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ f(N-1,0) & f(N-1,1) & f(N-1,2) & \cdots\cdots & f(N-1,N-1) \end{cases} \quad \text{.....(2.1)}$$

ซึ่งทางขวาของสมการ จะเรียกได้ว่า ภาพดิจิตอล และทุกๆ สมาชิกของเมตริกซ์ จะเรียกว่า พิกเซล จากขบวนการสร้างภาพดิจิตอลข้างต้น จะเห็นได้ว่าสามารถทราบขนาดของความละเอียดของภาพ N×N พิกเซล และจำนวนระดับของเกรย์สเกล ในทางปฏิบัติการทำควอนไตเซชันในระบบภาพดิจิตอล จะมีค่าดังสมการที่ 2.2

$$B = N \times N \times M \text{ บิต} \qquad \text{.....(2.2)}$$

เมื่อ B = ขนาดของข้อมูลภาพที่เป็นดิจิตอล

G = จำนวนของเกรย์สเกลที่ต้องการใช้ในการเก็บข้อมูลภาพ

M = จำนวนบิตที่ใช้ในการแทนข้อมูลภาพ 1 พิกเซล

โดย M สามารถหาได้จาก

$G = 2^M$

2.1.2 ลักษณะการจัดเก็บข้อมูลภาพแบบดิจิตอล

โดยทั่วไปแล้ว ข้อมูลภาพจะมีค่าความเข้มตั้งแต่ 2 ระดับขึ้นไป แต่ที่ใช้กันมากจะใช้กันที่ค่าระดับความเข้มของจุดภาพเท่ากับ 256 ระดับ ซึ่งจะทำให้ค่าของจุดภาพอยู่ในช่วง 0-255 โดยใช้เนื้อที่การเก็บข้อมูลภาพขนาด 1 ไบต์ หรือ 8 บิต สำหรับข้อมูล 1 จุดภาพ ($2^8 = 256$) ในกรณีที่ต้องการภาพที่มีความละเอียดของระดับความเข้มสูงๆ อาจจะต้องการจำนวนบิตสำหรับการเก็บข้อมูลมากกว่า 8 บิต คือ อาจจะเป็น 16 หรือ 24 บิต โดยค่าความเข้มของจุดภาพจะเท่ากับ 2^{16} และ 2^{24} โดยจะแยกให้เห็นชัดเจนดังนี้

1. ภาพ 2 ระดับ คือ มีเพียงแค่จุดขาวกับจุดดำเท่านั้น โดยแต่ละจุดภาพเป็นข้อมูลขนาด 1 บิต
2. ภาพ 16 ระดับ คือ ในแต่ละจุดภาพจะมีขนาดของข้อมูล 4 บิต ซึ่งทำให้สามารถแสดงได้ 16 ระดับสี หรือ 16 เกรย์สเกล ขึ้นอยู่กับภาพนั้นเป็นภาพสีหรือภาพสีหรือภาพขาวดำ
3. ภาพ 256 ระดับ คือ ในแต่ละจุดภาพจะมีขนาดของข้อมูล 8 บิต ซึ่งทำให้สามารถแสดงภาพได้ 256 ระดับสี หรือ 256 เกรย์สเกล ขึ้นอยู่กับภาพนั้นเป็นภาพสีหรือภาพขาวดำ
4. ภาพทรูคัลเลอร์(True color) คือ ในแต่ละจุดภาพจะมีขนาดของข้อมูล 24 บิต ทำให้สามารถแสดงผลภาพได้เหมือนภาพจริงที่สุด เพราะสามารถแสดงสีได้ถึง16,777,216สี ภาพทรูคัลเลอร์สามารถแสดงได้เฉพาะภาพสีเท่านั้น ไม่สามารถแสดงผลภาพขาวดำได้

โดยทั่วไปวิธีการประมวลผลภาพเชิงตัวเลขที่ทำให้คอมพิวเตอร์สามารถรู้จักวัตถุภายในภาพได้นั้น พอจะแบ่งได้สองระดับด้วยกันคือ การประมวลผลภาพในระดับต่ำ (Low Level Image Processing) และการประมวลผลภาพในระดับสูง (High Level Image Processing) การประมวลผลในระดับต่ำจะเป็นการประมวลผลในเชิงตัวเลขเกือบทั้งหมด เพื่อหาตัวแปรต่างๆ มาอธิบายข้อมูลภาพ โดยมีจุดประสงค์เพื่อนำตัวแปรเหล่านั้นไปประมวลผลในระดับสูงต่อไป โดยทั่วไปแล้วการประมวลผลภาพระดับต่ำจะประกอบด้วย การประมวลผลภาพก่อน (Preprocessing) การกำจัดสัญญาณรบกวน หรือการทำให้ภาพคมชัด การหาขอบภาพ เป็นต้น

การประมวลผลในระดับสูงเป็นการนำผลลัพธ์ หรือสัญลักษณ์ที่ได้จากการประมวลผลระดับต่ำมาตีความ หรือประมวลผลเพื่อให้คอมพิวเตอร์สามารถรู้จักและเข้าใจภาพได้ สำหรับความแตกต่างของการประมวลผลระดับต่ำและระดับสูงนั้นคือ ข้อมูลที่นำมาใช้ในการประมวลผลภาพ โดยที่การประมวลผลภาพระดับต่ำจะใช้ค่าความสว่างของจุดโดยตรง ส่วนการประมวลผลภาพในระดับสูงนั้นข้อมูลของภาพที่นำมาประมวลผลจะถูกแสดงในรูปของสัญลักษณ์ ซึ่งสัญลักษณ์เหล่านี้จะแสดงถึงสิ่งต่างๆ ที่มีอยู่ในภาพ เช่น ขนาดวัตถุ รูปร่าง และความสัมพันธ์กันระหว่างวัตถุภายในภาพ

2.2 ไฟล์ข้อมูลภาพชนิดบิตแมป

2.3.1 รูปแบบของไฟล์ข้อมูลภาพชนิดบิตแมป

รูปแบบของไฟล์ข้อมูลชนิดบิตแมป เป็นฟอร์แมตของวินโดว์บิตแมป ซึ่งเป็นมาตรฐานสำหรับไฟล์กราฟฟิกบนวินโดว์ ซึ่งจะใช้ในการตัดต่อ หรือสำเนาภาพต่างๆ ลงบนคลิปบอร์ด (Clipboard) เมื่อเวลาจัดเก็บไฟล์ที่มีสกุล BMP ซึ่งฟอร์แมตนี้ยังสามารถใช้เป็นวอลเปเปอร์ (Wallpaper) ของวินโดว์ได้อีกด้วย

2.3.2 โครงสร้างของไฟล์ของข้อมูลภาพชนิดบิตแมป

โครงสร้างของไฟล์ข้อมูลภาพชนิดบิตแมป จะประกอบด้วย 3 ส่วนคือ

1. ข้อมูลเฮดเดอร์ (Header)
2. ข้อมูลจานสี (Palette)
3. ข้อมูลภาพ (Data)

1. **ข้อมูลเฮดเดอร์** คือ ข้อมูลที่อยู่บริเวณส่วนหัวของไฟล์ ซึ่งประกอบด้วยข้อมูลที่บอกรายละเอียดต่างๆ ของภาพ เช่น ความกว้าง ความยาวของภาพ จำนวนสี จำนวนบิต ความละเอียด เป็นต้น

2. **ข้อมูลจานสี** คือ ข้อมูลที่บอกถึงชุดของจานสีที่เกิดจากการผสมแม่สีทั้งสาม คือ แดง เขียว และน้ำเงิน มาผสมกันได้เป็นสีต่างๆ ตามจำนวนสีของภาพ เช่น รูปขนาด 4 บิตจะมี 16 ระดับสี รูป 8 บิตจะมีขนาด 256 ระดับสี เป็นต้น ซึ่งถ้ามีจำนวนสีน้อยๆ ก็จะมีการเก็บค่าจานสีนี้ลงไฟล์ด้วย แต่ถ้ารูปประเภท 24 บิตจะไม่มีค่าจานสี แต่จะใช้วิธีการเก็บค่าแม่สีทั้งสามลงไปเป็นข้อมูลแทน เพราะถ้าเก็บค่าจานสีที่มีถึง 16.7 ล้านสีลงไปด้วยจะเปลืองพื้นที่มาก ข้อแตกต่างที่สำคัญของบิตแมปขนาดนี้คือ ไฟล์บิตแมปจะเก็บค่าจานสีชุดละ 4 ไบต์ แต่ก็ใช้แค่ 3 ไบต์เช่นกันคือ แดง เขียว และน้ำเงิน

3. **ข้อมูลภาพ** คือ ข้อมูลสีของภาพแต่ละจุดที่มาประกอบกันเป็นรูปภาพ ซึ่งค่าที่เก็บนี้จะเป็นค่าที่ใช้ในการชี้ตาราง Palette หมายเลขอะไร เช่น จุดแรกมีค่าเป็น 10 ก็ให้ไปเปิดตาราง Palette หมายเลข 10 สมมุติว่าของแม่สีเป็น R = 0, G = 0, และ B = 100 ก็จะได้จุดนี้เป็นสีน้ำเงิน ซึ่งถ้าเป็นกรณีของรูป 24 บิต จะเป็นการอ่านข้อมูลขึ้นมา 3 ค่า เป็นค่าของแม่สี RGB แล้วนำไปผสมบนจอภาพแทน

2.3.3 การจัดเก็บไฟล์ข้อมูลชนิดบิตแมป

การจัดเก็บไฟล์ข้อมูลชนิดบิตแมป มีการเก็บอยู่ 2 แบบ คือ

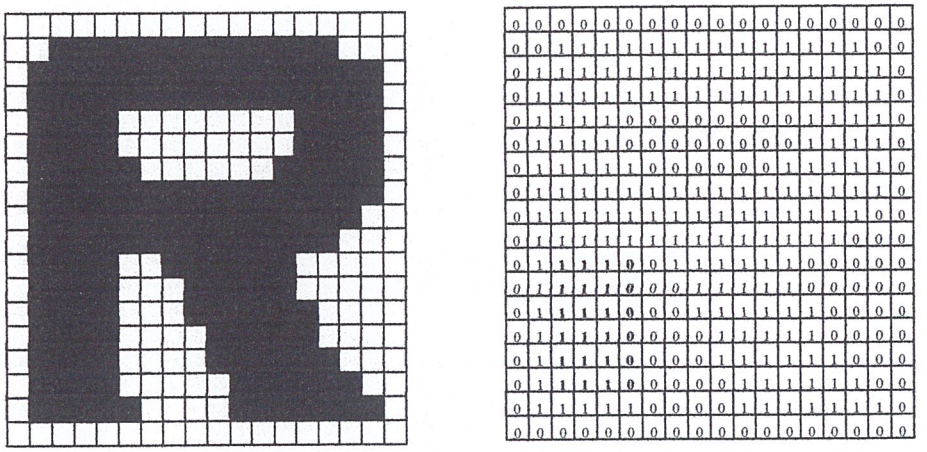
- แบบบีบอัดข้อมูล
  - RLE 4 เป็นการบีบอัดข้อมูลแบบ Run-length Encoder แบบ 4 บิต
  - RLE 8 เป็นการบีบอัดข้อมูลแบบ Run-length Encoder แบบ 8 บิต
- แบบไม่ได้บีบอัดข้อมูล

เป็นการเก็บข้อมูลจริงของสีของพิกเซล ซึ่งทำให้ขนาดไฟล์ค่อนข้างใหญ่ แต่จะทำการแสดงภาพได้เร็วกว่า เพราะไม่ต้องเสียเวลาในการคลายข้อมูล

2.4 การสร้างภาพไบนารี

อุปกรณ์ที่มีความสามารถในการแสดงผลได้แค่ 2 ระดับ หรือ 2 สี คือ สีขาวกับสีดำยังมีการใช้กันอย่างแพร่หลาย เช่น เครื่องพิมพ์(Printer) เครื่องโทรสาร(Fax) จอภาพแสดงผลแบบโมโนโครม(Monochrome Monitor) เป็นต้น เนื่องจากอุปกรณ์เหล่านี้เป็นอุปกรณ์ที่มีราคาถูก ดังนั้นการที่จะแสดงผลหรือพิมพ์รูปภาพที่มีระดับความเข้มของภาพหลายระดับซึ่งมีมากกว่าความสามารถในการแสดงผลของอุปกรณ์เหล่านั้นที่มีเพียงแค่ 2 ระดับเท่านั้น

จะเห็นได้ว่าการที่จะแก้ปัญหาการแสดงผลภาพที่มีความเข้มหลายระดับบนอุปกรณ์ที่สามารถแสดงผลได้ 2 ระดับนั้น จะต้องทำการแปลงข้อมูลภาพที่มีระดับความเข้มหลายระดับ (Multi Level Image) ให้เป็นภาพที่มีระดับความเข้มเพียง 2 ระดับ นั่นคือ 1 จุดภาพมีได้ 2 ค่าเท่านั้น คือ 0 กับ 1 โดยจุดภาพที่แทนด้วย 0 จะหมายถึงจุดภาพที่มีสีดำ ส่วนจุดที่แทนด้วย 1 จะหมายถึงจุดภาพที่มีสีขาว เมื่อทำการแปลงเป็นภาพไบนารีแล้วจึงนำภาพนั้นไปแสดงผลบนอุปกรณ์เหล่านั้น จะเห็นได้ว่าการแปลงข้อมูลภาพหลายระดับเป็นภาพไบนารีจึงมีความจำเป็นและมีประโยชน์มากในการแสดงผลภาพที่มีระดับความเข้มของภาพหลายระดับบนอุปกรณ์ที่มีความสามารถในการแสดงผลได้ 2 ระดับ สำหรับประโยชน์อีกประการหนึ่งในการแปลงข้อมูลภาพนั้นเป็นภาพไบนารี คือการลดเนื้อที่การเก็บข้อมูลภาพจะใช้ในการเก็บ 8 บิต เมื่อสร้างเป็นภาพไบนารีแล้วสามารถลดลงได้ถึง 8 เท่า นั่นคือ 1 จุดภาพจะใช้เนื้อที่ในการเก็บ 1 บิต อีกทั้งยังสามารถนำไปประยุกต์ใช้งานได้อย่างแพร่หลาย เช่น นำไปประยุกต์ใช้ในการวิเคราะห์เอกสารในขั้นตอนที่เรียกว่า การประมวลผลขั้นต้น (Preprocessing) เป็นต้น



รูปที่ 2.1 ภาพไบนารีและข้อมูลของแต่ละพิกเซล

ในการสร้างภาพไบนารี สามารถทำได้โดยใช้เทคนิคการทำเทรชโฮล(Thresholding Technique) โดยพิจารณาว่าจุดภาพใดควรจะเป็นจุดขาวหรือจุดดำจะกระทำโดยการเปรียบเทียบระหว่างจุดภาพเริ่มต้นกับค่าคงที่ค่าหนึ่งที่เรียกว่า "ค่าเทรชโฮล" (Threshold Value) เทคนิคนี้ใช้กันมากในกรณีที่ข้อมูลภาพมีลักษณะแตกต่างกันระหว่างวัตถุ (Object) และพื้นหลัง (Background) โดยค่าของจุดภาพใดๆ ที่มีค่าน้อยกว่าค่าเทรชโฮลจะถูกกำหนดให้เป็น0 (จุดดำ) และถ้าค่ามากกว่าค่าเทรชโฮลจะถูกกำหนดให้เป็น 1 (จุดขาว) ซึ่งการทำงานสามารถแสดงได้ดังสมการที่ 2.3

$$b(x,y) = \begin{cases} 0 & ; g(x,y) < Thr \\ 1 & ; g(x,y) \geq Thr \end{cases} \qquad \text{.....(2.3)}$$

b(x,y) ข้อมูลภาพผลลัพธ์เป็นภาพไบนารี

g(x,y) ข้อมูลภาพอินพุทที่มีระดับความเข้ม 0 ถึง 1 ระดับ

Thr ค่าเทรชโฮลเป็นค่าคงที่ที่อยู่ระหว่าง 0 ถึง L ระดับ

0 จุดดำ

1 จุดขาว

โดยที่ L ระดับความเข้มของจุดภาพสูงสุด

ในการสร้างภาพไบนารีโดยใช้เทคนิคเทรชโฮลเพื่อให้ได้ผลลัพธ์ที่ได้เหมาะสมและคมชัด สิ่งที่สำคัญที่สุดคือ ค่าเทรชโฮล เมื่องจากถ้าเลือกค่าเทรชโฮลไม่เหมาะสม (ค่าเทรชโฮลที่มีค่าน้อยเกินไปหรือมีค่ามากเกินไป) ภาพที่ได้อาจไม่เหมาะสม ขาดความคมชัดและรายละเอียดบางส่วนขาดหายไป กล่าวคือภาพที่ได้อาจจะมืดเกินไป (จุดดำมากเกินไป) หรือสว่างเกินไป (จุดขาวมากเกินไป) หรือภาพที่ได้มีสิ่งรบกวนเกิดขึ้น (Noise) เกิดขึ้น อันเป็นผลให้ภาพผลลัพธ์ที่ได้ไม่สวยงามเท่าที่ควร ดังนั้นปัญหาของการสร้างภาพไบนารีโดยวิธีเทรชโฮลนี้คือ ทำอย่างไรจึงจะสามารถคำนวณหาค่าเทรชโฮลที่เหมาะสมสำหรับแต่ละภาพที่จะนำมาทำการสร้างภาพไบนารี ซึ่งมีวิธีการคำนวณหาค่าเทรชโฮลหลายวิธี โดยแต่ละวิธีเหมาะสมสำหรับการทำงานที่ต่างไป เช่นการหาค่าเทรชโฮลโดยการกำหนดค่าลวงหน้า (Preassigned Threshold Value) การหาค่าเทรชโฮลจากค่ากลาง (Mid-Range Threshold Value) แต่ละวิธีอธิบายได้ดังนี้

การหาค่าเทรชโฮลโดยการกำหนดค่าล่วงหน้า (Preassigned Threshold Value) การหาค่าเทรชโฮลโดยวิธีการกำหนดล่วงหน้านี้เป็นวิธีที่ง่ายที่สุด เป็นการคำนวณค่าเทรชโฮลโดยการกำหนดเองจากผู้ใช้ ซึ่งการกำหนดนี้ขึ้นอยู่กับประสบการณ์ของผู้ใช้นั้นๆ โดยการเลือกค่าคงที่ค่าหนึ่ง ซึ่งเรียกค่านั้นว่า ค่าเทรชโฮล โดยค่าที่เลือกมานี้จะเป็นค่าที่อยู่ระหว่างค่าต่ำสุดและค่าสูงสุดของระดับความเข้มของข้อมูลภาพอินพุท เช่น ภาพข้อมูลอินพุทมีเกรย์สเกล 256 ระดับ จะมีค่าเกรย์สเกลได้ตั้งแต่ 0-255 เมื่อเลือกค่าเทรชโฮลได้แล้วสามารถสร้างภาพไบนารีได้ดังสมการ 2.3

การหาค่าเทรชโฮลจากค่ากลาง (Mid-Range Threshold Value) การหาค่าเทรชโฮลโดยพิจารณาจากค่ากลาง เป็นการหาค่าเทรชโฮลที่แตกต่างจากการหาค่าเทรชโฮลวิธีแรก สำหรับวิธีนี้จะเป็นการคำนวณหาค่าโดยอัตโนมัติ โดยไม่ต้องให้ผู้ใช้เป็นผู้กำหนด การหาค่าเทรชโฮลวิธีนี้ได้อาศัยการคำนวณพื้นฐานทางสถิติในเรื่องของการหาค่ากลางหรือค่าเฉลี่ย(Mean) มาประยุกต์ใช้ ค่าเทรช

โฮลที่คำนวณได้จะเป็นค่าที่ได้จากค่ากึ่งกลางที่อยู่ระหว่างค่าระดับความเข้มสูงสุดและค่าความเข้มต่ำสุดของข้อมูลภาพอินพุท สำหรับการคำนวณหาค่ากึ่งกลางนี้สามารถคำนวณได้จากสมการที่ 2.4

$$Thr = \frac{\text{Maximum}(g(x,y)) + \text{Minimum}(g(x,y))}{2} \quad \ldots.(2.4)$$

โดยที่ Thr : ค่าเทรชโฮล

g(x,y) : ข้อมูลภาพอินพุทที่มีระดับความเข้ม 0 ถึง L ระดับ

Maximum(g(x,y)) : ค่าสูงสุดเกรย์สเกลของข้อมูลอินพุท

Minimum(g(x,y)) : ค่าต่ำสุดเกรย์สเกลของข้อมูลอินพุท

เมื่อทำการคำนวณค่าเทรชโฮลได้แล้ว ก็สามารถสร้างภาพไบนารีได้โดยนำค่าเทรชโฮลที่ได้มาแทนค่าในสมการ 2.3

การหาค่าเทรชโฮลจากค่าเฉลี่ยเลขคณิต หาได้จากสมการที่ 2.5

$$Thr = \frac{\sum_{i=0}^{NxN} g_i(x,y)}{NxN} \quad \ldots..(2.5)$$

2.5 การแบ่งส่วนภาพ (Image Segmentation)

บางครั้งการวิเคราะห์ภาพจะเน้นที่วัตถุ (Object) บนภาพ ซึ่งวัตถุเหล่านั้นจะถูกแยกแยะออกได้ด้วยวิธีการที่หลากหลายตั้งแต่วิธีการง่ายๆ เช่นการหาส่วนที่เป็นแนวของวัตถุ หรือส่วนที่เป็นขอบของวัตถุ ไปจนกระทั่งวิธีการที่สลับซับซ้อน สำหรับวิธีการง่ายๆในการแบ่งส่วนภาพได้แก่

2.5.1 การตรวจรู้แนวเส้น (Line Detection)

สำหรับวิธีนี้เป็นการเปรียบเทียบค่าระดับสีเทาระหว่างจุดภาพที่พิจารณา กับจุดภาพข้างเคียง ทั้งนี้จะเพิ่มน้ำหนักหรือให้ความสำคัญของจุดภาพข้างเคียงที่มีการเรียงตัวร่วมกับจุดภาพที่พิจารณาเป็นเส้นตรงรวม 4 แบบ คือ เส้นตรงในแนวตั้ง เส้นตรงในแนวนอน เส้นตรงในแนวทแยงจากล่างซ้ายขึ้นไปทางบนขวา หรือเรียกว่าเส้นทแยง $+45^{\circ}$ และเส้นตรงในแนวทแยงจากบนขวาลงไปทางล่างซ้าย หรือเรียกว่าเส้นทแยง -45° หมายความว่าแนวเส้นที่จะตรวจรู้เป็นแนวเส้นตั้ง แนวเส้นนอน และเส้นเอียงทำมุม $+45^{\circ}$ หรือ -45° เท่านั้น

กระบวนการตรวจรู้จะทำได้โดยนำหน้ากาก (mask) ขนาด 3 x 3 (อาจใช้ขนาดใหญ่กว่านี้ เช่น 5 x 5 หรือ 7x7 ก็ได้ แต่จะใช้เวลาในการประมวลผลเพิ่มมากขึ้น) ไปทาบกับจุดภาพในตำแหน่งที่จะทำการประมวลผล จากนั้นจะคำนวณหาค่าตามสมการที่ (2.6)

$$R = \sum_{k=1}^{9} W_k Z_k \qquad \text{.....(2.6)}$$

โดยที่ W_k เป็นค่าน้ำหนักของแต่ละจุดบนหน้ากาก ดังรูปที่ 2.2

Z_k เป็นค่าระดับสีเทาของจุดภาพในตำแหน่งที่ตรงกับจุดบนหน้ากาก กล่าวคือ

$Z_1 = f(x-1,y-1)$ $Z_2 = f(x,y-1)$ $Z_3 = f(x+1,y-1)$

$Z_4 = f(x-1,y)$ $Z_5 = f(x,y)$ $Z_6 = f(x+1,y)$

$Z_7 = f(x-1,y+1)$ $Z_8 = f(x,y+1)$ $Z_9 = f(x+1,y+1)$

W_1	W_2	W_3
W_4	W_5	W_6
W_7	W_8	W_9

ก) หน้ากากทั่วไป

-1	-1	-1
2	2	2
-1	-1	-1

ข) หน้ากากแนวนอน

-1	2	-1
-1	2	-1
-1	2	-1

ค) หน้ากากแนวตั้ง

-1	-1	2
-1	2	-1
2	-1	-1

ง) หน้ากากแนว +45°

2	-1	-1
-1	2	-1
-1	-1	2

จ) หน้ากากแนว –45°

รูปที่ 2.2 หน้ากากที่ใช้ในการหาแนวเส้น

ในการตรวจหาแนวเส้นของภาพ จะนำหน้ากากทั้ง 4 แบบ ไปทาบบนจุดภาพต่างๆ จนครบทั้งภาพ แล้วคำนวณตามสมการที่ 2.6 จะได้ค่า R มา 1 ค่า คือค่า R_1แล้วเปลี่ยนหน้ากากไปเรื่อยๆ ตามรูปที่ 2.2 จนครบ 4 อัน จะได้ค่า R_2 , R_3 และ R_4 ตามลำดับ จากนั้นทำการเปรียบเทียบค่า R ทั้ง 4 ค่า ถ้า $|R_i| > |R_j|$ ที่ทุกค่าของ $j \neq i$จุดภาพนั้นจะถูกตัดสินว่าเป็นส่วนหนึ่งของแนวเส้นตามแบบของหน้ากากที่ i

บางครั้ง มีความต้องการหาแนวเส้นลักษณะใดลักษณะหนึ่ง ก็อาจทำได้โดยเลือกใช้หน้ากากแนวที่ต้องการทาบบนจุดภาพ และหาผลรวมตามสมการที่ 2.6 จากนั้นนำค่าที่ได้ไปเปรียบเทียบกับค่า เทรชโฮลด์ที่ได้กำหนดไว้แล้ว ถ้าค่าที่ได้สูงกว่าค่าเทรชโฮลด์ จุดภาพนั้นจะถูกตัดสินว่าเป็นส่วนหนึ่งของแนวเส้น

2.5.2 การตรวจหาขอบ (Edge detection)

องค์ประกอบหรือวัตถุที่ถูกบันทึกภาพจะมีตำแหน่ง การจัดวาง ที่ทำให้เกิดการสะท้อนแสงที่ต่างกัน ค่าความเข้มแสงหรือค่าระดับสีเทาที่ปรากฏบนภาพ จะต้องมีขอบเขตองค์ประกอบนั้นๆ แสดงให้เห็นชัดเจนหรือค่อนข้างชัดเจน เนื่องจากบริเวณดังกล่าวมีค่าระดับสีเทาแตกต่างกัน ดังนั้น การตรวจหาขอบของวัตถุบนภาพ จึงใช้หลักการพิจารณาการเปลี่ยนแปลงค่าระดับสีเทา ที่มี 2 แนวทาง คือ การพิจารณาอัตราหรือค่าความชันในการเปลี่ยนแปลงค่าระดับสีเทา ซึ่งเรียกว่าวิธีเกรเดียนธ์ (Gradient Method) และการพิจารณาอัตราการเปลี่ยนแปลงค่าความชันในการเปลี่ยนแปลงค่าระดับสีเทา ซึ่งเราเรียกว่าวิธีลาปลาเซียน (Laplacian Method)

2.5.2.1 วิธีเกรเดียนต์ (Gradient Method)

จะใช้กับภาพที่มีองค์ประกอบส่วนใหญ่มีค่าระดับสีเทาไม่สม่ำเสมอตลอดทั้งภาพได้ แต่ความแตกต่างของค่าระดับสีเทาในวัตถุใดๆจะมีไม่มาก คือ มีการเปลี่ยนแปลงของระดับสีเทาอย่างช้าๆ ในวัตถุใดวัตถุหนึ่ง การตัดสินใจว่าบริเวณใดเป็นขอบก็ขึ้นอยู่กับค่าความชันในการเปลี่ยนแปลงค่าระดับสีเทา ซึ่งจะต้องมีค่ามากกว่าค่าเทรชโฮลด์ (Threshold) ซึ่งภาพที่ได้จะหาขอบได้ดีหรือไม่ก็ขึ้นกับการกำหนดค่าเทรชโฮลด์นั่นเอง

เนื่องจากวิธีเกรเดียนต์ใช้หลักในการพิจารณาการเปลี่ยนแปลงของค่าระดับสีเทาของภาพ ถ้ามีการเปลี่ยนแปลงมาก ก็จะถูกกำหนดให้เป็นขอบของวัตถุ ทั้งนี้ค่าเกรเดียนต์ของภาพ f(x,y) ณ ตำแหน่ง (x,y) จะเป็นค่าเวกเตอร์

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \qquad \text{.....(2.7)}$$

โดยที่ขนาดของค่าเวกเตอร์ ∇f เป็นปริมาณที่มีความสำคัญในการหาขอบของวัตถุ โดยทั่วๆไปจะเรียกว่าค่าเกรเดียนต์ และใช้สัญลักษณ์ ∇f ที่มีค่า

$$\nabla f = mag(\nabla f) = \sqrt{G_x^2 + G_y^2} \qquad \text{.....(2.8)}$$

ในทางปฏิบัติ เพื่อให้ง่ายต่อการคำนวณ จะทำการประมาณค่าเกรเดียนต์ด้วยค่าสัมบูรณ์

$$\nabla f \approx |G_x| + |G_y| \qquad \text{....(2.9)}$$

การหาค่าอนุพันธ์ย่อย G_x และ G_y ทำได้โดยการใช้หน้ากาก (mask)หรือ หน้าต่าง (window) หรือตัวดำเนินการ (operator) ขนาด 3x3 ที่มีค่าสัมประสิทธิ์ ดังแสดงในรูปที่ 2.3ก เข้าช่วยปรับค่าระดับสีเทา ณ จุดภาพที่สอดคล้องกัน ซึ่งตัวดำเนินการมีหลายแบบดังนี้

2.5.2.1.1 *ตัวดำเนินการแบบโรเบิร์ตครอสส์เกรเดียนต์ (Robert-cross gradient operator)*

ตัวดำเนินการแบบโรเบิร์ตครอสส์เกรเดียนต์ อาจพิจารณาให้เป็นหน้ากากขนาด 2x2 ดังรูปที่ 2.3 ข.1 หรือเป็นหน้ากากขนาด 3x3 ดังรูปที่ 2.3 ข.2 ที่ให้ค่าอนุพันธ์ย่อยเป็น

$$G_x = |Z_5 - Z_9| \quad \ldots(2.10ก)$$

$$G_y = |Z_6 - Z_8| \quad \ldots(2.10ข)$$

2.5.2.1.2 ตัวดำเนินการแบบโซเบล (Sobel Operator)

ตัวดำเนินการแบบโซเบล มีหน้ากากขนาด 3x3 ดังรูปที่ 2.3 ค ทำให้ได้ค่าอนุพันธ์ย่อยเป็น

$$G_x = (Z_7 + 2Z_8 + Z_9) - (Z_1 + 2Z_2 + Z_3) \quad \ldots(2.11ก)$$

$$G_y = (Z_3 + 2Z_6 + Z_9) - (Z_1 + 2Z_4 + Z_7) \quad \ldots(2.11ข)$$

2.5.2.1.3 *ตัวดำเนินการแบบพรีวิตต์ (Prewitt Operator)*

ตัวดำเนินการแบบพรีวิตต์ มีหน้ากากขนาด 3x3 เช่นเดียวกับตัวดำเนินการโซเบลแต่มีค่าสัมประสิทธิ์แตกต่างกัน ดังรูปที่ 2.3ง ทำให้ได้ค่าอนุพันธ์ย่อยเป็น

$$G_x = (Z_7 + Z_8 + Z_9) - (Z_1 + Z_2 + Z_3) \quad \ldots(2.12ก)$$

$$G_y = (Z_3 + Z_6 + Z_9) - (Z_1 + Z_4 + Z_7) \quad \ldots(2.12ข)$$

W_1	W_2	W_3
W_4	W_5	W_6
W_7	W_8	W_9

ก) หน้ากากขนาด 3x3 ที่จะใช้เป็นตัวดำเนินการ

1	0
0	-1

0	1
-1	0

หน้ากากที่ใช้คำนวณค่า G_x และ G_y

0	0	0
0	1	0
0	0	-1

0	0	0
0	0	1
0	-1	0

หน้ากากที่ใช้คำนวณค่า G_x และ G_y

ข) ตัวดำเนินการแบบโรเบิร์ตครอสส์เกรเดียนต์

-1	-2	-1
0	0	0
1	2	1

หน้ากากที่ใช้คำนวณค่า G_x

-1	0	1
-2	0	2
-1	0	1

หน้ากากที่ใช้คำนวณค่า G_y

ค) ตัวดำเนินการแบบโซเบล

-1	-1	-1
0	0	0
1	1	1

หน้ากากที่ใช้คำนวณค่า G_x

-1	0	1
-1	0	1
-1	0	1

หน้ากากที่ใช้คำนวณค่า G_y

ง) ตัวดำเนินการแบบพรีวิตต์

รูปที่ 2.3 ตัวดำเนินการที่ใช้ในการคำนวณหาค่าเกรเดียนต์

2.5.2.2 วิธีลาปลาเซียน (Laplacian Method)

วิธีลาปลาเซียน ใช้หลักการพิจารณาการเปลี่ยนแปลงค่า Gradient ของค่าระดับสีเทาอีกต่อหนึ่ง และจะกำหนดตำแหน่งที่มี Zero-Crossing เกิดขึ้นเป็นตำแหน่งขอบของวัตถุ ทั้งนี้ การหาอนุพันธ์ย่อยอันดับสองของภาพ f(x.y) คือการหาค่า Laplacian แบบ 2 มิติ เป็นต้น

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \qquad \ldots(2.13ก)$$

หน้ากากของวิธีลาปลาเซียนขนาด 3x3 จะมีค่าสัมประสิทธิ์แตกต่างกันหลายแบบ ดังรูปที่ 2.4 และตัวอย่างการใช้ตัวดำเนินการแบบ 4-neighbor เพื่อหาค่าอนุพันธ์ย่อยอันดับสองของจุดภาพ f(x,y) คือ

$$\nabla^2 f = 4Z_5 - (Z_2+Z_4+Z_6+Z_8) \quad \ldots(2.13ข)$$

0	-1	0
-1	4	-1
0	-1	0

ก) แบบ 4-neighborhood

-1	-1	-1
-1	8	-1
-1	-1	-1

ข) แบบ 8-neighborhood

-2	1	-2
1	4	1
-2	1	-2

ค) แบบ seperable 8-neighborhood

รูปที่ 2.4 หน้ากากที่ใช้ในตัวดำเนินการลาปลาเซียน

2.5.2.3 *การหาขอบแบบแคนนี (Canny edge detection)*

การหาขอบแบบแคนนี เป็นการดัดแปลงวิธีเกรเดียนต์ให้เหมาะสมกับการตรวจหาขอบขององค์ประกอบในภาพที่มีสัญญาณรบกวน แต่อยู่บนเงื่อนไข 3 ประการ ดังนี้

ก) ไม่เกิดความผิดพลาดในการหาขอบที่สำคัญ และไม่ควรมีการหาขอบที่ผิดเกิดขึ้น

ข) การบอกตำแหน่งของขอบ จะต้องมีระยะระหว่างตำแหน่งจริงและตำแหน่งที่ห่างขอบสั้นที่สุด

ค) ถ้ามีค่าหลายค่าในการหาขอบของขอบใดขอบหนึ่ง ค่าที่น้อยที่สุดจะเป็นค่าของขอบที่แท้จริง เพื่อแก้ปัญหาสัญญาณรบกวน ซึ่งจะทำให้ได้ขอบที่ไม่เรียบ

สำหรับขั้นตอนวิธีการหาขอบแบบแคนนี มี 4 ขั้นตอน คือ

ขั้นที่ 1 ใช้ตัวกรองไบโนเมียล (Binomial Filter) เพื่อลดทอนสัญญาณรบกวน ดังสมการที่ 2.14

$$f(x,y) = \left[\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \right]^n \quad \ldots(2.14)$$

โดยที่ n = 2 เท่าของความแปรปรวนของการกระจายแบบเกาส์ (Gaussian distribution)

ขั้นที่ 2 ทำการคำนวณเกรเดียนต์ตามสมการที่ (2.10) และใช้ตัวดำเนินการแบบพรีวิต ดังรูปที่ 2.3ง จะได้

$$\nabla f = |(Z_7+Z_8+Z_9)-(Z_1+Z_2+Z_3)| + |(Z_3+Z_6+Z_9)-(Z_1+Z_4+Z_7)| \quad \ldots(2.15)$$

ขั้นที่3 ทำการหาว่าจุดภาพมีค่า เกรเดียนต์เฉพาะถิ่นสูงสุด (Maximum local gradient) หรือไม่ สมมติค่าเกรเดียนต์ของแต่ละจุดภาพในกรอบขนาด 3x3 มีค่าเป็น

P1 P2 P3
P4 P5 P6
P6 P7 P8

ทำการพิจารณาเปรียบเทียบค่าเกรเดียนต์ของจุดภาพกลางกับจุดภาพอื่นๆ ในแนวเส้นตรงต่างๆ ประกอบด้วย แนวตั้ง แนวนอน แนวแทยง $+45^{\circ}$ และ -45°

ถ้า P5 > Pk ทุกค่า $k \neq 5$ และ P5 > ค่าเทรชโฮลด์ต่ำ (low threshold value) T_L แสดงว่าค่า P5 เป็นค่าเกรเดียนต์เฉพาะถิ่นสูงสุด และจะนำค่าดังกล่าวไปคำนวณตามขั้นตอนที่ 4 ต่อไป

ขั้นที่ 4 การทำเทรชโฮลด์แบบฮีสเทอรีซิส (Hysteresis Threshold)

ขั้นตอนสุดท้ายนี้ เป็นขั้นตอนการตัดสินใจว่าจุดภาพหนึ่งๆ เป็นขอบขององค์ประกอบหรือไม่ มีเงื่อนไข 3 ประการ ดังนี้

- จุดภาพที่มีขนาดของเกรเดียนต์ใหญ่กว่าค่าเทรชโฮลด์สูง (High threshold value) T_H จุดภาพนั้นจะเป็นขอบขององค์ประกอบ
- ทุกจุดภาพที่จะประกอบกันเป็นแนวขอบแนวหนึ่ง จะต้องมีอย่างน้อย 1 จุดภาพในแนวนั้นที่มีค่าเกรเดียนต์ > T_H
- จุดภาพที่มีค่าเกรเดียนต์ < T_H และไม่สอดคล้องกับเงื่อนไขที่สอง จะถูกตัดออกไป

2.6 ทฤษฎีการแยกตัวอักขระออกจากภาพ

2.6.1 การหากรอบตัวอักขระโดยวิธี Line Crossing

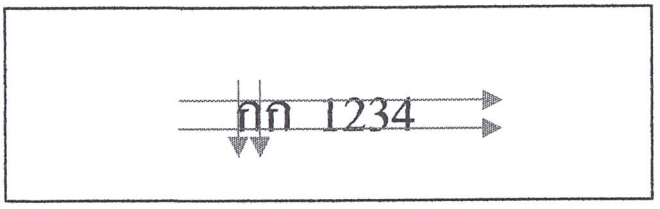
เทคนิคนี้จะมีหลักการง่ายๆ คือ พยายามแบ่งอักขระแต่ละตัวออกมาให้ได้มากที่สุด โดยเริ่มจากการสแกนเป็นแนวตรง ทั้งแนวนอนและแนวตั้ง ตัดไปตามช่องว่างทำให้เกิดเป็นตารางครอบตัวอักขระ ซึ่งตารางที่เล็กที่สุดที่ครอบตัวอักขระอยู่ถือเป็นการ segment ได้หนึ่งอักขระดังรูปที่ 2.5

แนวคิดของวิธีการ Line Crossing ดังรูปที่ 2.5 มีขั้นตอนดังต่อไปนี้

1. สแกนเป็นเส้นตรงตามแนวนอน
2. สแกนเป็นเส้นตรงตามแนวตั้ง

(ขั้นตอนที่ 1 กับ 2 สามารถที่จะทำขั้นตอนใดก่อนก็ได้)

3. คัดลอกข้อมูลที่อยู่ในกรอบที่มีอักขระเก็บไว้ใน buffer เพื่อนำไปแสดงผล



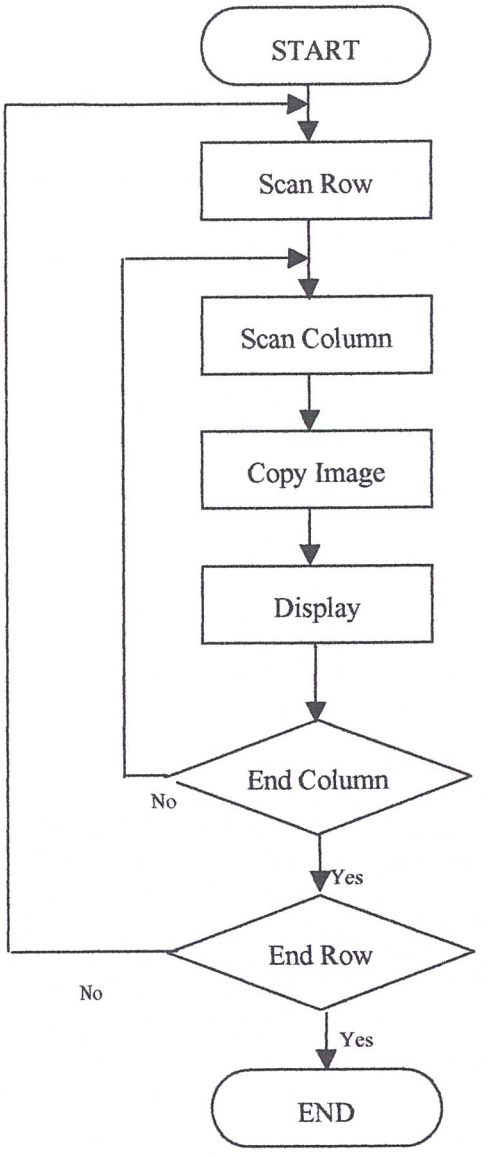
รูปที่ 2.5 แสดงการแยกตัวอักขระด้วยวิธี Line Crossing

อธิบาย Flowchart การแยกตัวอักขระด้วยวิธีการ Line Crossing จากรูปที่ 2.6

1. Scan Row : หาช่องว่างในแนวนอน
2. Scan Column : หาช่องว่างในแนวตั้ง
3. Copy Image : คัดลอกข้อมูลของส่วนที่ตัดได้
4. Display : แสดงรูปที่ทำการตัดได้
5. End Column : การ scan ข้อมูลในแนวตั้งครบหมด
    - Yes : End Row
    - No : Scan Column
6. End Row : การ scan ข้อมูลในแนวนอนครบหมด
    - Yes : End Column
    - No : Scan Row

ข้อดีของการทำ Line Crossing ก็คือ ง่ายต่อการเขียนโปรแกรม และมีประสิทธิภาพในการแยกอักขระกับแบบอักขระที่มีข้อผิดพลาดน้อย ตัวอักขระไม่ติดกันมาก และถ้าเป็นการ Segment ตัวอักขระภาษาอังกฤษจะได้ผลลัพธ์ที่ดีกว่าตัวอักขระภาษาไทย

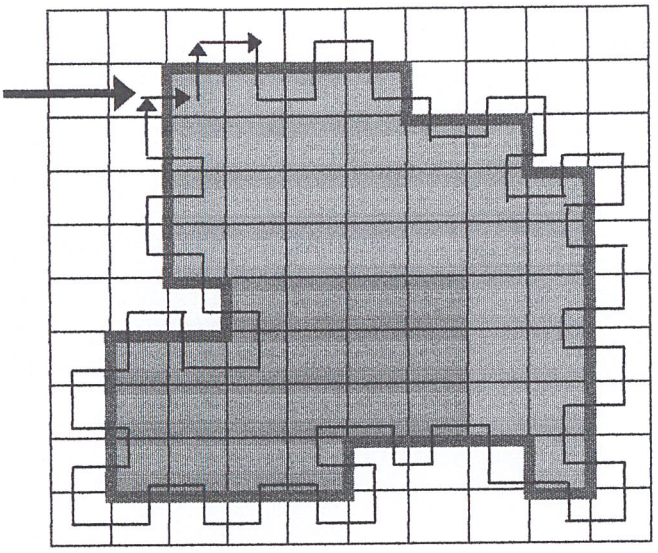
ข้อเสียของวิธีการนี้ที่เห็นได้ชัดเจนก็คือ ในการทำการแยกตัวอักขระที่มีการเหลื่อมล้ำกันจะไม่สามารถใช้เทคนิคในการแยกนี้ได้



รูปที่ 2.6 แสดง Flow Chart การแยกตัวอักขระด้วยวิธี Line Crossing

2.6.2 เทคนิคการตามรอยขอบภาพ (Contour Following)

เทคนิคการตามรอยขอบภาพจะช่วยให้เราทำการ Segment ตัวอักษรได้ง่ายขึ้น โดยไม่ต้องคำนึงถึงรูปร่างของวัตถุที่จะทำการ Segment เมื่อพิจารณาขอบของวัตถุจะมีค่าสีที่มีความแตกต่างกันกับสีของพื้นหลัง เช่น ภาพตัวอักขระสีดำที่วางอยู่บนพื้นกระดาษสีขาว



รูปที่ 2.7 แสดงการหาขอบด้วยวิธี Contour Following

การทำงานของเทคนิคนี้ในการทำ Contour Following ตามรูปที่ 2.7 ทำได้ตามขั้นตอนต่อไปนี้

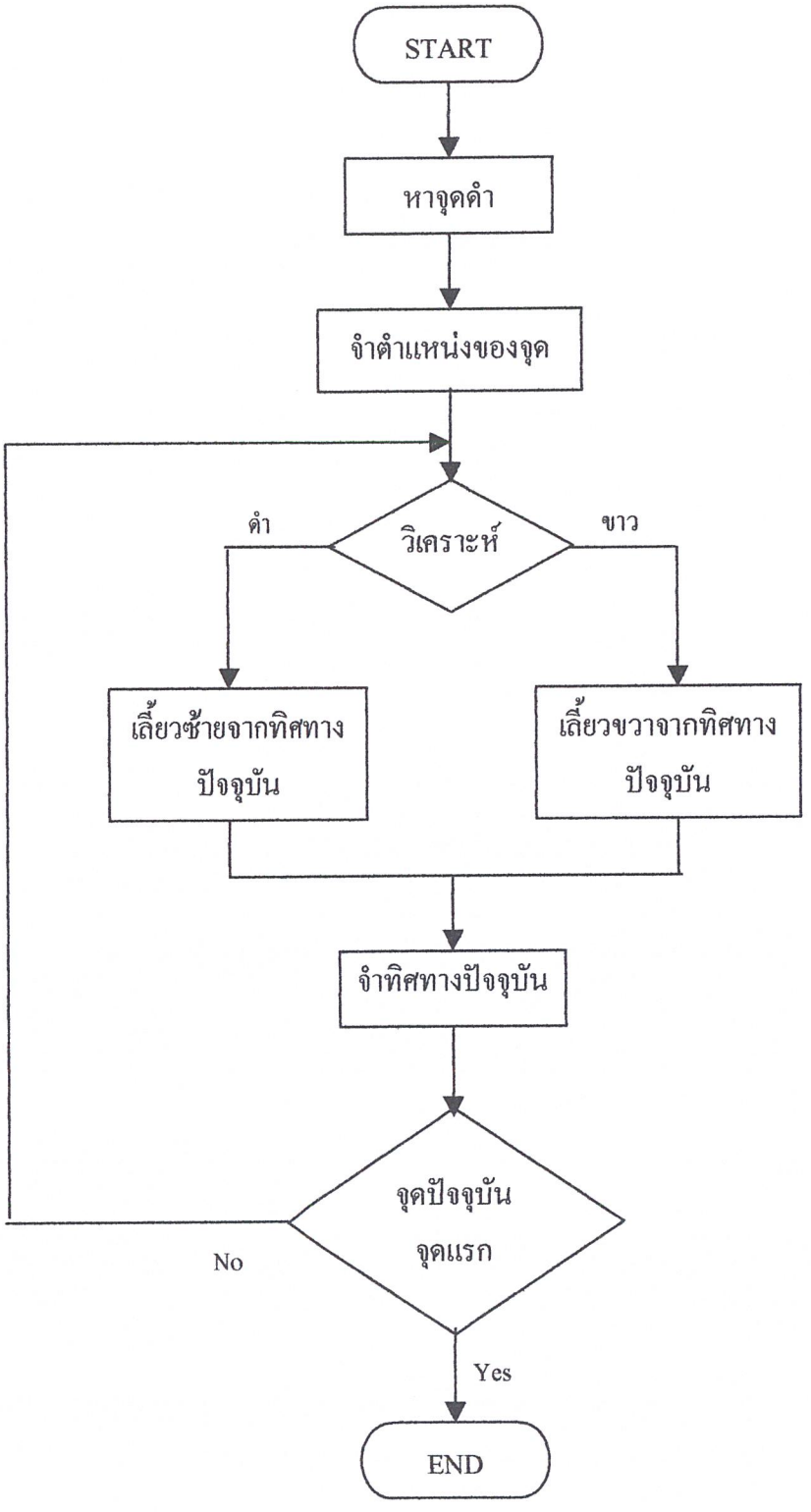
1. จุดภาพปัจจุบันมีข้อมูลเป็นดำ ให้เลี้ยวซ้ายจากทิศทางปัจจุบัน
2. จุดภาพปัจจุบันมีข้อมูลเป็นสีขาว ให้เลี้ยวขวาจากทิศทางปัจจุบัน
3. สิ้นสุดเมื่อจุดภาพปัจจุบันอยู่ตำแหน่งเดียวกับจุดเริ่มต้นพอดี

อธิบาย Flowchart การแยกอักขระด้วยวิธีการ Contour Following จากรูปที่ 2.8

1. หาจุดดำ : เป็นการหาจุดดำจุดแรกที่พบ โดยการ scan ตามแนวนอน
2. จำตำแหน่งของจุด : เป็นการจำตำแหน่งของจุดดำที่พบ
3. วิเคราะห์จุด : มี 2 กรณี
    - ขวา : เลี้ยวขวาจากทิศทางปัจจุบัน
    - ดำ : เลี้ยวซ้ายจากทิศทางปัจจุบัน
4. จำทิศทางปัจจุบัน : จดจำทิศทางและตำแหน่งปัจจุบัน
5. จุดปัจจุบัน = จุดแรก : เป็นการเปรียบเทียบตำแหน่งปัจจุบันกับจุดเริ่มต้น
    - Yes : End
    - No : วิเคราะห์จุด

ข้อดีของการทำวิธี Contour Following ก็คือ เราไม่ต้องสนใจว่ารูปร่างของตัวอักขระจะมีลักษณะเป็นอย่างไร เมื่อเราวิ่งไปตามขอบของภาพ เราก็จะได้ขอบของตัวอักขระออกมา

ข้อเสียของการทำวิธีการนี้ก็คือ ถ้าอักขระมีส่วนที่ติดกันอยู่จะไม่สามารถใช้วิธีนี้ในการแยกตัวอักขระออกจากกันได้



รูปที่ 2.8 แสดง Flow Chart การหาขอบด้วยวิธี Contour Following

2.6.3 เทคนิคการแยกโดยใช้วิธีการ Contour with matrix

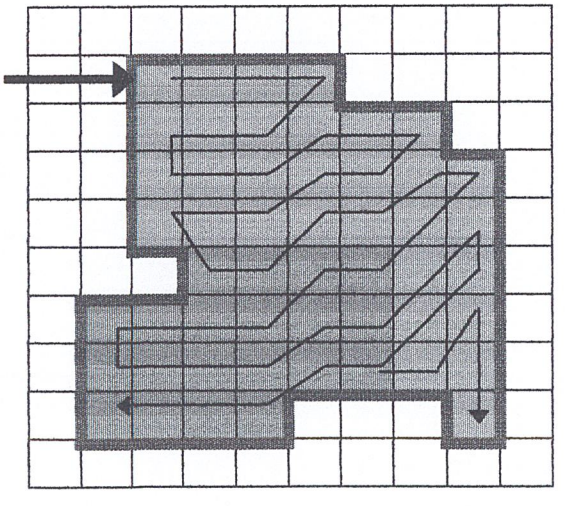
เทคนิคนี้คล้ายกับการทำ Contour Following แต่จะต่างกันตรงที่เทคนิคนี้จะมีการดึงเอาข้อมูลภาพที่อยู่ในขอบเขตของการทำ Contour ไปใช้ด้วย ทำให้เราได้ตัวอักขระทั้งตัวไปใช้งาน และการใช้งานไม่เพียงแต่ดูว่าข้อมูลตรงจุดนั้นเป็น 0 หรือ 1 แต่จะใช้ matrix ช่วยตรวจสอบแทน matrix ที่ใช้มีขนาด 3×3 ดังรูปที่ 2.9

P1	P2	P3
P4	P5	P6
P7	P8	P9

รูปที่ 2.9 เมตริกซ์ ขนาด 3 x 3

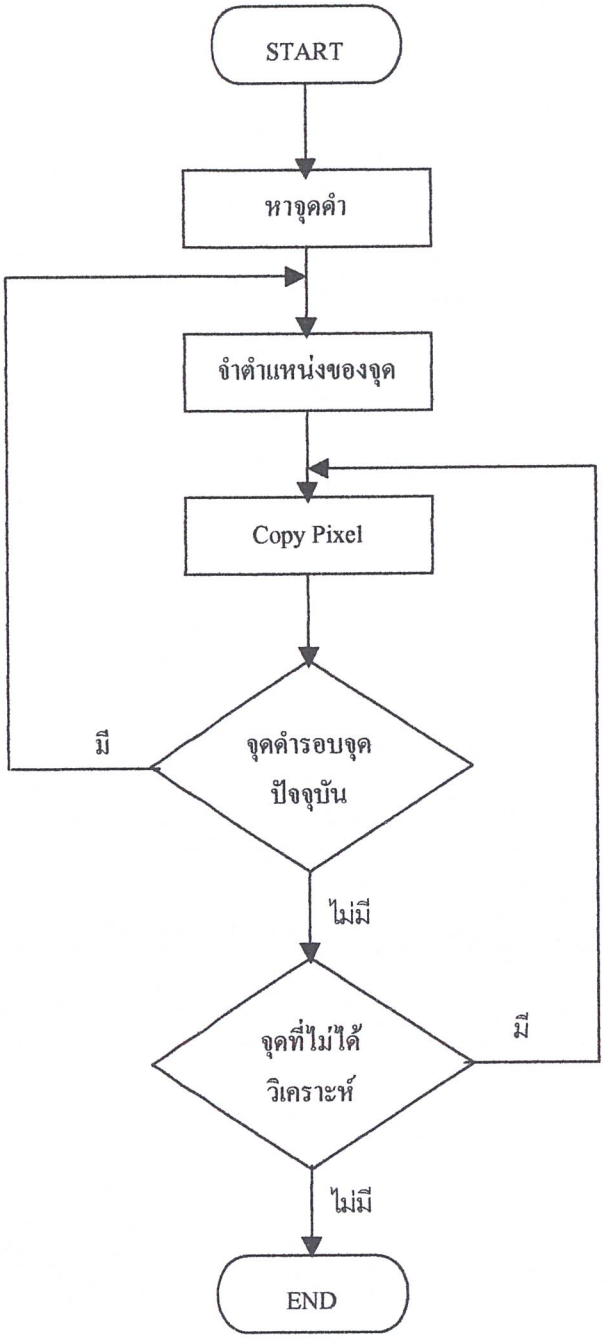
แนวคิดของการทำ Contour with matrix ดังรูปที่ 2.10 มีขั้นตอนการทำงานดังต่อไปนี้

1. ให้กำหนดจุดดำที่พบเป็นจุดกลางของ matrix(P5) คัดลอกจุดนี้ลงใน buffer แล้วลบจุดกลางนั้นออกจากรูปภาพ
2. หาจุดกลางถัดไป โดยตรวจสอบข้อมูลรอบๆ จุดกลางปัจจุบัน(P5) ซึ่งเป็นจุดสีดำ คือ ข้อมูลตัวอักขระ การตรวจสอบจะเริ่มจากด้านบนซ้ายก่อน คือ ตำแหน่ง P1,P2,P3,P4,P6,P7,P8, และ P9 ตามลำดับ กำหนดจุดดำที่พบจุดแรกให้เป็นจุดกึ่งกลาง matrix
3. เมื่อรอบๆ จุดกลางนั้นไม่มีจุดดำแล้ว จะทำการ return กลับไปยังจุดกลางตัวก่อน
4. การทำงานจะเป็นเช่นนี้ไปเรื่อยๆ จนกระทั่งไม่มีจุดดำรอบจุดกลางทุกตัว



รูปที่ 2.10 แสดงการหาขอบโดยวิธี Contour with metrix

ตัวอย่างการทำงานด้วยวิธีนี้แสดงดังรูปที่ 2.11 เส้นทางการวิ่งจะเป็นไปตามลูกศรที่แสดงในรูป ซึ่งรูปตัวอย่างเป็นรูปเดียวกับตัวอย่างที่แสดงในวิธีการตามรอยขอบภาพ (Contour Following)



รูปที่ 2.11 แสดง Flow Chart แสดงการหาขอบโดยวิธี Contour with matrix

อธิบาย Flowchart การแยกอักขระด้วยวิธีการ Contour with matrix จากรูปที่ 2.11

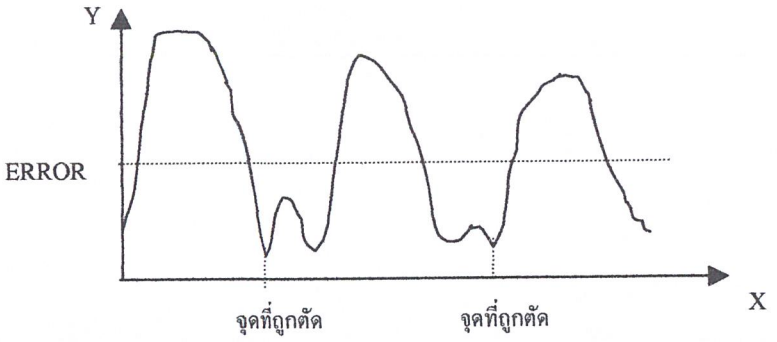
1. หาจุดดำ : เป็นการหาจุดดำจุดแรกที่พบโดยการ scan ตามแนวนอน
2. จำตำแหน่งของจุด : จำตำแหน่งจุดดำที่พบ
3. Copy Pixels : จะทำการคัดลอกจุดที่พบไว้ใน buffer และทำการลบจุดที่พบออกจากรูปภาพ
4. จุดดำรอบจุดปัจจุบัน : ตรวจว่ารอบจุดปัจจุบันมีจุดสีดำล้อมรอบหรือไม่
   - มี : จำตำแหน่งของจุด
   - ไม่มี : จุดที่ไม่ได้วิเคราะห์
5. จุดที่ไม่ได้วิเคราะห์ : ตรวจว่ามีจุดที่ยังไม่ได้ทำการวิเคราะห์หรือไม่
   - มี : Copy Pixels
   - ไม่มี : End

ข้อดีของวิธี Contour with matrix ก็คือ สามารถดึงเอาส่วนที่เป็นเนื้อตัวอักขระออกมาได้ทั้งหมด เป็นประโยชน์ต่อการนำไปใช้ในกระบวนการอื่นๆ ที่เนื้อของข้อมูลเป็นสิ่งจำเป็น

ข้อเสียของวิธีนี้ก็คือ เมื่อเราต้องดึงเอาข้อมูลทุกส่วนของตัวอักขระออกมาก็จะทำให้เราต้องใช้เวลานานกับการคัดลอก และลบข้อมูล วิธีการนี้ทำงานแบบ recursive ถ้ามีการ backtracking มากเกินไปอาจทำให้ stack overflow ได้

2.6.4 เทคนิคการแยกตัวอักขระที่ติดกันโดยใช้ Histogram

Histogram ใช้เพื่อวัดความหนาแน่นของข้อมูลภาพในช่วงที่กำหนด โดยค่าความหนาแน่นที่ใช้เป็นจำนวนจุดดำ เพื่อช่วยในการวิเคราะห์ภาพตัวอักขระ โดยเก็บเป็นค่าเปอร์เซ็นต์ Error เพื่อนำไปใช้เปรียบเทียบกับตัวอักขระแต่ละตัว เพื่อหาตัวอักขระที่คาดว่าจะมีการติดกันอยู่ และการใช้ histogram จะช่วยให้โปรแกรมสามารถตัดสินใจได้ว่า ควรที่จะตัดส่วนที่ติดกันของอักขระที่ตรงส่วนไหน เพื่อให้สามารถแยกตัวอักขระที่ติดกัน ให้ออกมาเป็นตัวอักขระเดี่ยวให้ได้มากที่สุด รูปที่ 2.12 แสดงการเลือกจุดที่จะทำการแยกตัวอักขระออกจากกัน



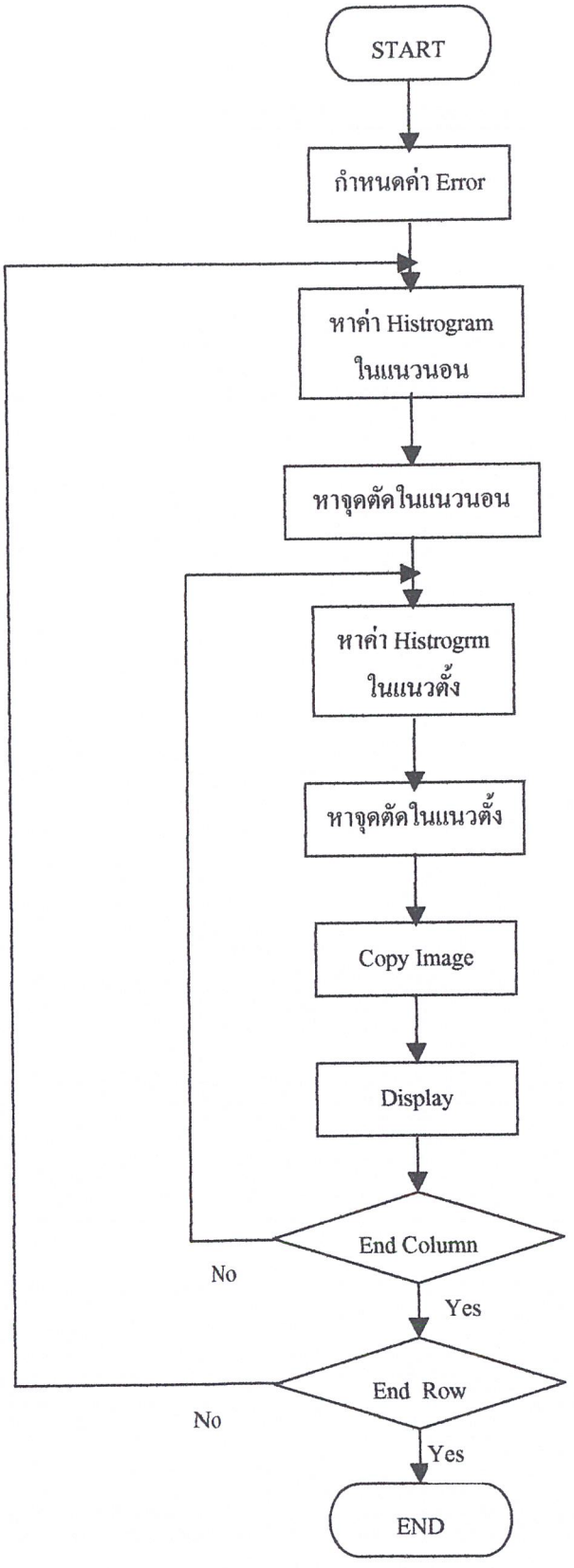
รูปที่ 2.12 แสดงการกำหนดจุดตัดด้วย Histogram

การแยกส่วนที่ติดกันของตัวอักขระ โดยใช้ histogram มีวิธีการดังนี้

1. ทำการหาค่า histogram ของตัวอักขระที่สามารถแยกออกมาจากรูปภาพได้
2. กำหนดค่าเปอร์เซ็นต์ error เพื่อใช้ในการแยกตัวอักขระออกจากกัน
3. ทำการแยกตัวอักขระตรงส่วนที่มีค่าความหนาแน่นเท่ากับ หรือต่ำกว่าค่าเปอร์เซ็นต์ error ที่ได้กำหนดไว้ โดยจะเลือกแยกตัวอักขระตรงส่วนที่มีค่าความหนาแน่นต่ำที่สุด
4. ถ้าค่าความหนาแน่นที่ต่ำที่สุดมีหลายตำแหน่ง จะใช้ตำแหน่งที่อยู่ตรงกลาง

อธิบาย Flowchart การแยกตัวอักขระที่ติดกันด้วยการใช้ histogram จากรูปที่ 2.10

1. กำหนดค่า Error : เป็นการกำหนดค่าเปอร์เซ็นต์ error เพื่อใช้ในการแยกอักขระ
2. หาค่า histogram ในแนวนอน : หาค่าของ histogram โดยหาค่าความหนาแน่นในแนวนอน
3. หาจุดตัดในแนวนอน : เลือกตำแหน่งในการแยกตัวอักขระออกจากกันในแนวนอน
4. หาค่า histogram ในแนวตั้ง : หาค่าของ histogram โดยหาค่าความหนาแน่นในแนวตั้ง
5. หาจุดตัดในแนวตั้ง : เลือกตำแหน่งในการแยกตัวอักขระออกจากกันในแนวตั้ง
6. Copy Image : คัดลอกข้อมูลของส่วนที่ตัดได้
7. Display : แสดงรูปที่ทำการตัดได้
8. End Column : การ scan ข้อมูลในแนวตั้งครบหมด
    - Yes : End Row
    - No : หาค่า Histogram ในแนวตั้ง
9. End Row : การ scan ข้อมูลในแนวนอนครบหมด
    - Yes : End
    - No : หาค่า Histogram ในแนวนอน



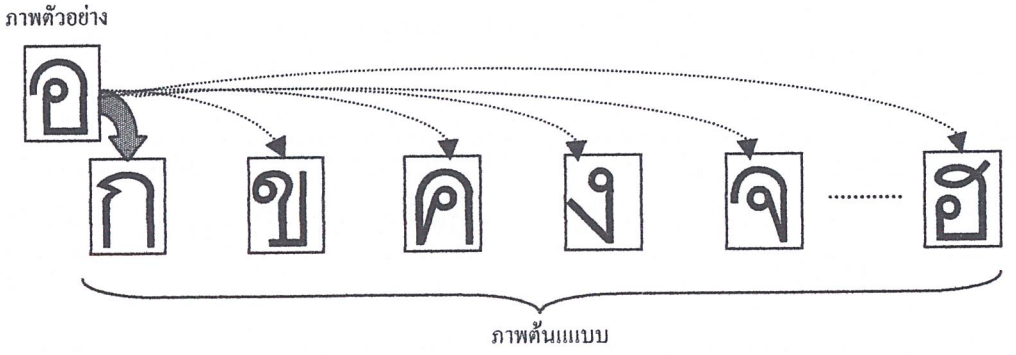
รูปที่ 2.13 แสดง Flow Chart การแยกตัวอักขระที่ติดกันโดยการใช้ Histogram

บทที่ 3
การรู้จำตัวอักขระ

ในการรู้จำตัวอักขระจากข้อมูลภาพนั้นมีวิธีที่ใช้ได้มากมายหลายวิธี เช่น การใช้เทคโนโลยีโครงข่ายนิวรอล (Neural Network) , การเทียบค่าจุดภาพ(Matching) , การตัดและจำแนก (Cut and Classification) ซึ่งแต่ละวิธีก็มีข้อดี และข้อเสียแตกต่างกันออกไปแล้วแต่วัตถุประสงค์ของการใช้งาน ในที่นี้จะกล่าวถึงวิธีเทียบค่าจุดภาพ และการตัดและจำแนกดังต่อไปนี้

3.1 การเทียบค่าจุดภาพ (Matching)

เป็นการจำแนกตัวอักษรโดยการนำภาพตัวอักษรซึ่งเป็นภาพแบบขาวดำมาเปรียบเทียบกับภาพต้นแบบทีละจุด จนครบทั้งภาพแล้วนับจำนวนจุดภาพที่เหมือนกับต้นแบบ โดยจะต้องมีภาพต้นแบบเป็นภาพตัวอักษรที่มีลักษณะชัดเจนครบทุกตัว เมื่อภาพตัวอักษรนั้นเหมือนต้นแบบตัวใดมากที่สุด หรือมีจำนวนจุดภาพที่เหมือนกันมากที่สุดก็แสดงว่าตัวอักษรนั้นคือตัวเดียวกับต้นแบบตัวนั้น โดยภาพตัวอย่างและภาพต้นแบบจะต้องมีขนาดที่เท่ากัน จึงจะทำการเทียบจุดภาพได้



รูปที่ 3.1 แสดงวิธีการเทียบจุดภาพ

ข้อจำกัดของการรู้จำตัวอักขระแบบนี้คือ ถ้าหากข้อมูลภาพของตัวอักขระเอียงก็จะทำให้การรู้จำตัวอักขระมีความผิดพลาดได้ อีกทั้งภาพต้นแบบจะต้องมีความชัดเจนมาก และใช้เวลาในการประมวลผลนานเพราะทำการพิจารณาค่าจุดภาพของทั้งภาพ

3.2 การตัดและจำแนก (Cut and Classify)

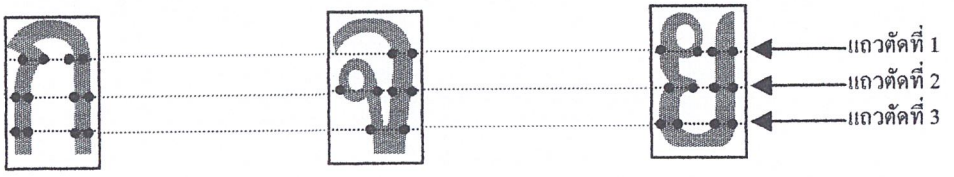
หลักการของการตัดคือการกราดตรวจไปตามแถวหรือหลักเพื่อหาจุดขอบหรือจุดบ่งชี้ที่มีการเปลี่ยนแปลงจากขาวเป็นดำ และดำเป็นขาว ดังรูปที่ 3.2 และ 3.3

รูปที่ 3.2 ตัวอย่างการหาจุดบ่งชี้ตามแนวนอน

รูปที่3.3 ตัวอย่างการหาจุดบ่งชี้ตามแนวตั้ง

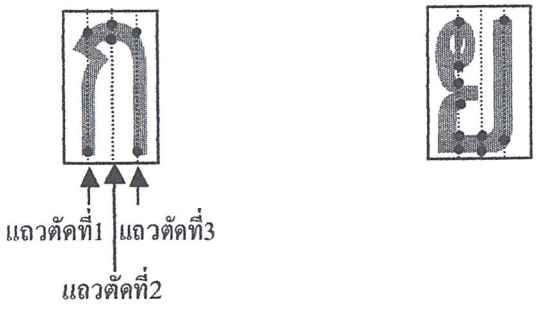
การจำแนกตัวอักขระใช้ลักษณะการค้นหาแบบโครงสร้างต้นไม้ (Tree Structure Search) โดยเริ่มทำการแบ่งกลุ่มตัวอักษรจากจำนวนจุดบ่งชี้ในแถวตัดแรกในแนวนอนทั้งสามเป็นหลัก เนื่องจากการตัดเริ่มต้นมีเพียงสามแถว ผลการจำแนกขั้นแรกจะสามารถแบ่งกลุ่มตัวอักษรได้เป็นกลุ่มใหญ่ๆ จากแต่ละกลุ่มใหญ่ๆ จะทำการตัดต่อไปทางแนวตั้งเพื่อการจำแนกตัวอักษรให้ละเอียดยิ่งขึ้น วิธีการนี้สามารถตรวจสอบความถูกต้องของการจำแนกได้โดยการตัดและจำแนกต่อไปอีก แต่ผลที่ตามมาคือ โครงสร้างต้นไม้ที่มีขนาดใหญ่และซับซ้อนขึ้นเรื่อยๆ

ตัวอย่างการจำแนกตัวอักขระอธิบายคร่าวๆได้ดังนี้ สมมติให้ตัวอักขระที่ต้องการแยกมีเพียง 3 ตัวคือ ก,จ และ ย การจำแนกตัวอักขระเริ่มต้นเมื่อได้จุดบ่งชี้ใน 3 แถวแรก แล้วเริ่มแบ่งกลุ่มแต่ละกรณีจากจำนวนจุดบ่งชี้ในแถวตัดทั้งสาม อาทิเช่น ถ้ามีจำนวนจุดบ่งชี้ในแถวตัดที่ 1 เท่ากับ 4 จุด กลุ่มที่น่าจะเป็นไปได้คือ ก และ ย ส่วน จ มีจุดบ่งชี้ในแถวนี้คือ 2 จุด ก็จะทำการแยกออกมาก่อน



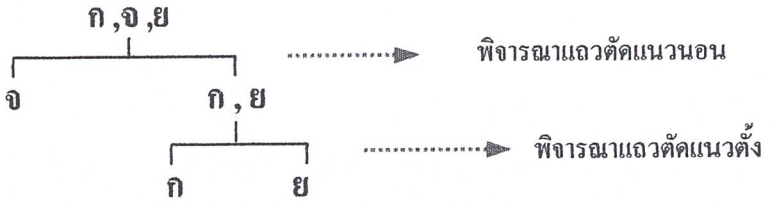
รูปที่ 3.4 การรู้จำตัวเลขโดยเริ่มพิจารณาจากแถวตัดแนวนอน

จากนั้นก็พิจารณาแถวตัดที่2และ3 ต่อ พบว่า ก และ ย มีจุดบ่งชี้ 4 จุดเหมือนกัน จึงนำมาทำการตัดในแนวตั้งต่อไป



รูปที่ 3.5 การรู้จำตัวเลขโดยเริ่มพิจารณาจากแถวตัดแนวตั้ง

จากรูปจะเห็นได้ว่า ก มีจุดตัดในแถวตัดตามแนวตั้งที่1 เพียง 2 จุด แต่ ย มีจุดตัดถึง 4 จุด จึงจะทำให้สามารถจำแนก ก และ ย ออกจากกันได้ ซึ่งเมื่อนำมาเขียนโครงสร้างต้นไม้ จะสามารถเขียนได้ดังรูปที่ 3.6



รูปที่ 3.6 แสดงโครงสร้างต้นไม้ของการตัดและจำแนกตัวอักษร ก,จ,ย

ในการจำแนกพยัญชนะนั้นจะมีความซับซ้อนมากกว่าตัวเลข เนื่องจากมีการใช้พยัญชนะในป้ายทะเบียน ดังนั้นจึงมีกรณีที่ต้องแบ่งมากกว่าการรู้จำตัวเลข ซึ่งในการจำแนก อาจทำให้ต้องใช้จำนวนแถวที่ตัดมากกว่าการรู้จำตัวเลข

ข้อดีของการรู้จำตัวอักขระแบบตัดและจำแนกคือ ใช้เวลาในการประมวลผลน้อย เนื่องจากพิจารณาเพียงแค่แถวหรือหลักที่ต้องการเท่านั้น ไม่จำเป็นต้องพิจารณาทั้งภาพ

ส่วนข้อจำกัดของการรู้จำแบบนี้คือ เมื่อภาพของตัวอักษรมี Noise เกิดขึ้นระหว่างการแปลงภาพสีเป็นภาพขาวดำ หรือตัวอักษรเอียงก็จะทำให้ผลที่ได้ไม่มีความแม่นยำ

3.3 ลักษณะของป้ายทะเบียนรถยนต์

ป้ายทะเบียนรถยนต์ในประเทศไทยนั้น จะสามารถแบ่งได้เป็น 2 แบบตามลักษณะของตัวอักขระในป้ายทะเบียน ได้แก่

- ป้ายทะเบียนแบบเก่า มีรูปแบบคือ ตัวอักขระตัวแรกเป็นตัวเลข (อาจมีหรือไม่มีก็ได้) ถัดมาเป็นตัวพยัญชนะ แล้วตามด้วยขีด และตามด้วยตัวเลขอีก 4 หลักดังรูปที่ 3.7
- ป้ายทะเบียนแบบใหม่ มีรูปแบบคือ ตัวพยัญชนะ 2 ตัว แล้วตามด้วยตัวเลขอีก 4 หลัก ซึ่งตัวเลข 4 หลักนี้ อาจมีไม่ครบก็ได้ (เว้นเป็นช่องว่างไว้) ดังรูปที่ 3.8

รูปที่ 3.7 ป้ายทะเบียนรถยนต์แบบเก่า

รูปที่ 3.8 ป้ายทะเบียนรถยนต์แบบใหม่

ซึ่งในการรู้จำตัวอักขระในป้ายทะเบียนนั้นอาจต้องแยกทำเป็นสองส่วน ส่วนที่หนึ่งคือส่วนตัวเลขและพยัญชนะ และส่วนที่สองคือส่วนตัวเลขอย่างเดียว ดังรูปที่ 3.9

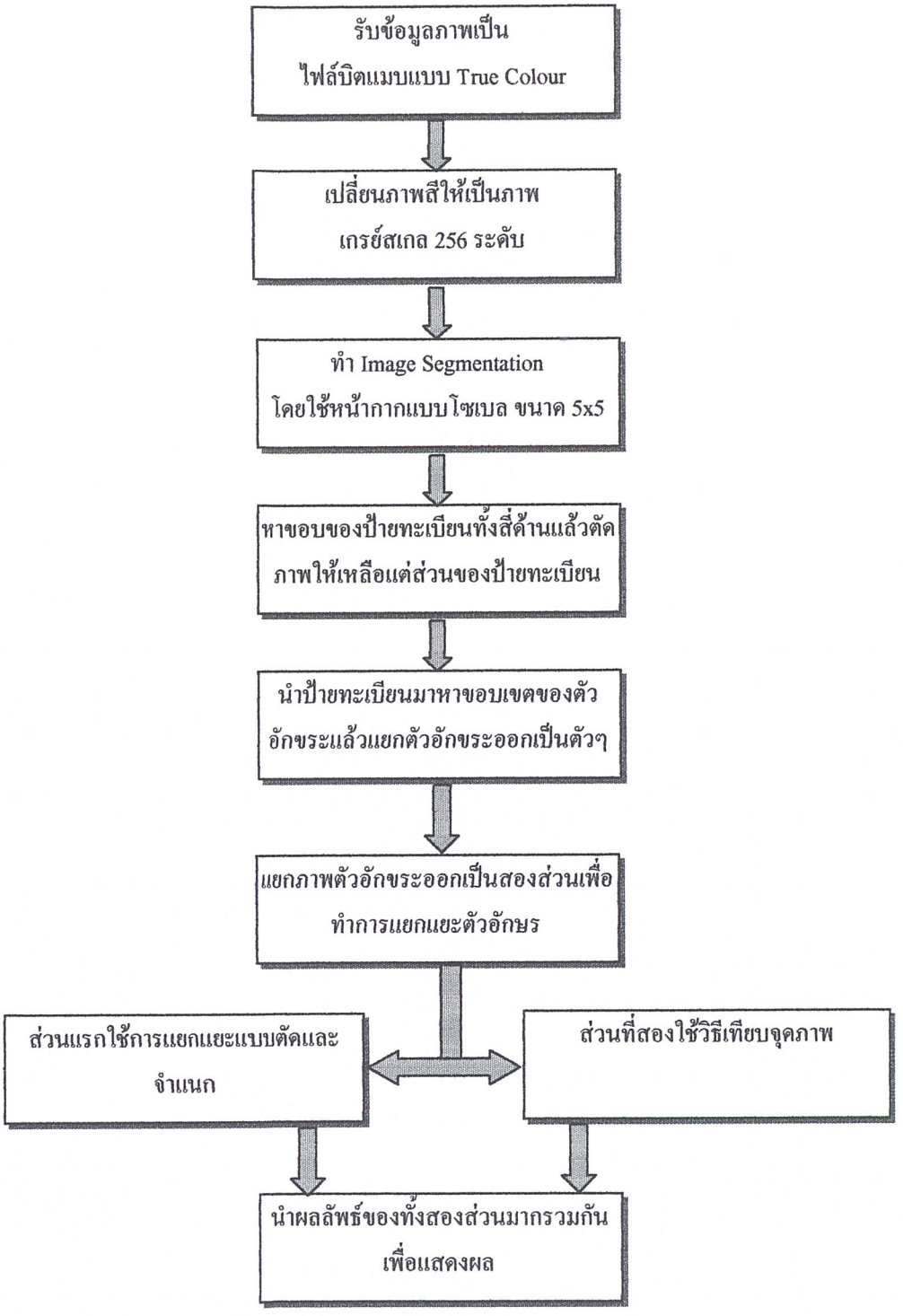
รูปที่ 3.9 แสดงการแบ่งส่วนของตัวอักขระในป้ายทะเบียนเพื่อทำการรู้จำตัวอักขระ

ในการแยกพิจารณาเป็นส่วนๆนี้จะทำให้ลดความผิดพลาดในการวิเคราะห์ตัวอักขระได้ในส่วนที่สอง เนื่องจากตัวเลขมีเพียง 10 ตัว จึงพิจารณาเพียงกรณีของตัวเลขซึ่งก็คือ 10 กรณีเท่านั้น

บทที่ 4
การทำงานของโปรแกรม

4.1 โครงสร้างของระบบจดจำทะเบียนรถยนต์

ระบบจดจำเลขทะเบียนรถยนต์มีขั้นตอนดังต่อไปนี้ คือ ในขั้นแรกจะเป็นการแปลงภาพที่รับเข้ามาเป็นภาพสี ทำการแปลงให้เป็นภาพเกรย์สเกล 256 ระดับ จากนั้นนำมาหาขอบภาพ (Edge Detection) โดยใช้วิธีเกรเดียนต์ (Gradient Operator) โดยใช้หน้ากาก (Mask) แบบโซเบล ขนาด 5x5 แล้วทำการ Binarization โดยกำหนดค่า Threshold ให้เหมาะสม เมื่อผ่านขั้นตอนนี้แล้วก็จะได้ภาพที่สีดำและขอบวัตถุที่มีสีขาว แล้วจึงนำมาหาส่วนที่น่าจะเป็นทะเบียนรถยนต์ โดยการวิเคราะห์จุดสีขาวที่มีการเรียงตัวในแนวแกน X และ แกน Y มีความยาวในระดับที่น่าจะเป็นทะเบียนรถยนต์ได้ จากนั้นนำมาแยกแยะขอบเขตของตัวอักษรในป้ายทะเบียนต่อไปโดยใช้วิธีพิจารณาการเปลี่ยนของค่าจุดภาพจากขาวไปดำ หรือ ดำไปขาว เมื่อได้ขอบเขตของตัวอักขระที่แน่นอนแล้วก็นำภาพตัวอักขระมาทำการแยกแยะว่าเป็นตัวอักษรใด โดยใช้วิธีการรู้จำแบบเทียบจุดภาพ และวิธีตัดและจำแนก แสดงได้ดังรูปที่ 4.1



รูปที่ 4.1 ขั้นตอนการทำงานของโปรแกรม

4.2 การเตรียมข้อมูลภาพ

4.2.1 การแปลงเป็นภาพแบบเกรย์สเกล

ภาพที่รับเข้าเป็นภาพสีแบบ True Colour ซึ่งมีขนาดข้อมูลใหญ่ จึงต้องทำการเปลี่ยนภาพให้เป็นแบบเกรย์สเกล 256 ระดับก่อน เพื่อสะดวกต่อการประมวลผลภาพ โดยจะใช้ค่าความเข้มของสีน้ำเงิน สีแดง และสีเขียว มาหาค่าเฉลี่ย แล้วใช้ค่าเฉลี่ยนี้เป็นค่าความเข้มของสีน้ำเงิน สีแดง และสีเขียวแทน จะทำให้ได้ภาพเกรย์สเกล 256 ระดับ แล้วนำภาพนี้ไปทำการประมวลผลภาพต่อไป

รูปที่ 4.2 ตัวอย่างภาพทะเบียนรถยนต์แบบ True Colour

รูปที่ 4.3 ตัวอย่างภาพทะเบียนรถยนต์เมื่อเปลี่ยนเป็น Gray Scale 256 ระดับ

4.2.2 การทำ Edge detection

ในขั้นตอนนี้จะใช้วิธี Gradient โดยใช้หน้ากากแบบ Sobel ขนาด 5x5 ดังแสดงในรูปที่ 4.4 ทาบลงบนภาพ แล้วคำนวณหาขนาดของเกรเดียนต์ โดยในรูปที่ 4.4ก เป็นหน้ากากที่ใช้ตรวจหาขอบตามแนวแกน Y ซึ่งจะได้ค่า G_Y และใช้หน้ากากตามรูปที่ 4.4ข เพื่อหาขอบตามแนวแกน X ซึ่งจะได้ค่า G_X ซึ่งขนาดของเกรเดียนต์ มีค่าตามสมการที่ 4.1

$$\text{Mag}(\nabla f) = |G_X| + |G_Y| \qquad \ldots(4.1)$$

-1/4	-1/3	0	1/3	1/4
-1/3	-1/2	0	1/2	1/3
-1/2	-1	0	1	1/2
-1/3	-1/2	0	1/2	1/3
-1/4	-1/3	0	1/3	1/4

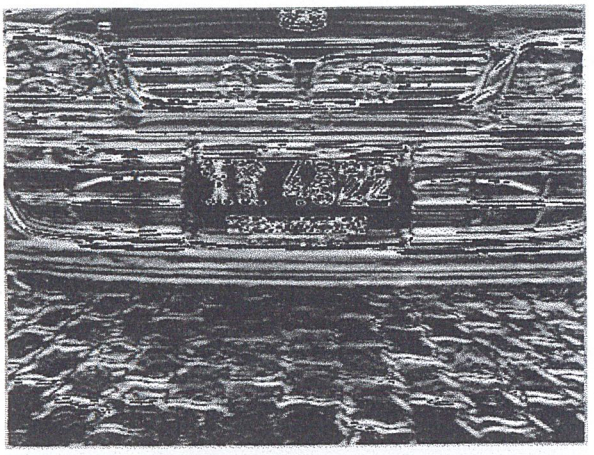
ก) หน้ากากที่ใช้ตรวจหาขอบแนวตั้ง

-1/4	-1/3	-1/2	-1/3	1/4
-1/3	-1/2	-1	-1/2	-1/3
0	0	0	0	0
1/3	1/2	1	1/2	1/3
1/4	1/3	1/2	1/3	1/4

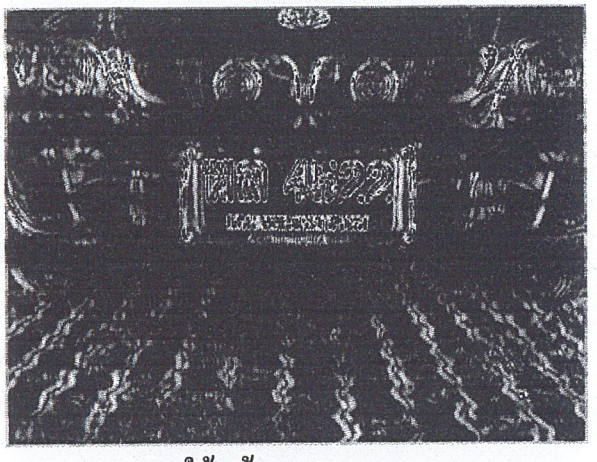
ข) หน้ากากที่ใช้ตรวจหาขอบแนวนอน

รูปที่ 4.4 หน้ากากแบบโซเบล ขนาด 5x5

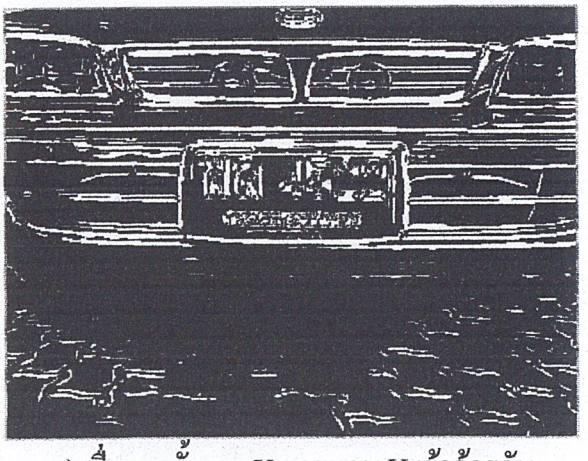
ภาพที่ได้ทำการ ตรวจหาขอบแล้วจะมีลักษณะดังรูปที่ 4.5 จะเห็นว่า ถ้าใช้หน้ากากแบบแนวตั้ง ตามรูป 4.4 กนั้น เมื่อใช้ค่า Threshold ที่เหมาะสม ภาพที่ได้จะแสดงส่วนที่เป็นขอบของวัตถุที่เป็นแนวนอนเป็นสีขาว ส่วนอื่นๆของภาพจะเป็นสีดำดังแสดงในรูปที่ 4.5 ก ส่วนถ้าใช้หน้ากากแบบแนวนอนนั้น ภาพที่ได้จะแสดงส่วนที่เป็นขอบของวัตถุที่เป็นแนวตั้งเป็นสีขาว ส่วนอื่นๆ ของภาพจะเป็นสีดำ ดังแสดงในรูปที่ 4.5 ข และเมื่อนำมารวมกันทั้งสองแกน ก็จะได้ภาพดังรูปที่ 4.5 ค ซึ่งจะแสดงทั้งส่วนที่เป็นขอบของวัตถุทั้งแกนตั้งและแกนนอน



ก) ใช้หน้ากากตามแกน X



ข) ใช้หน้ากากตามแกน Y

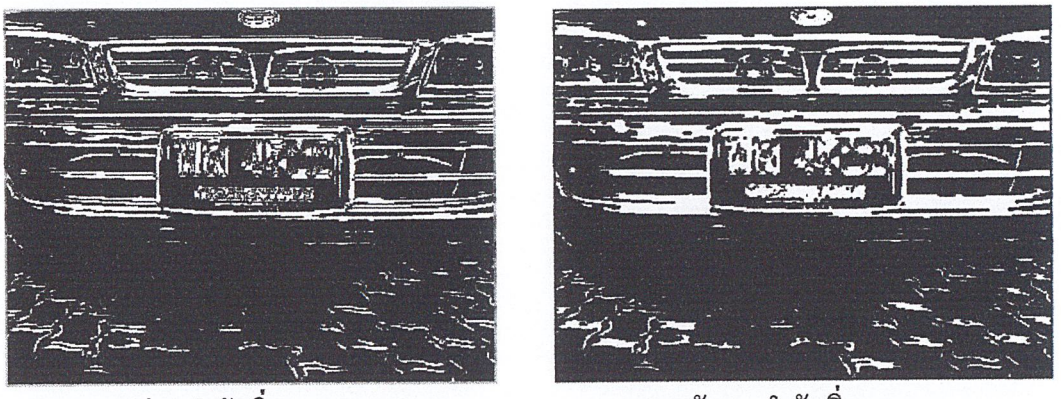


ค) เมื่อรวมทั้งแกนX และแกน Y เข้าด้วยกัน

รูปที่ 4.5 ตัวอย่างภาพทะเบียนรถยนต์เมื่อผ่านการ ตรวจหาขอบ(Edge detection) ตามวิธีเกรเดียนต์ โดยใช้หน้ากากแบบโซเบล ขนาด 5x5

4.2.3 การกำจัดสิ่งรบกวน (Noise Reduction)

เมื่อภาพผ่านการตรวจหาขอบ โดยใช้หน้ากากแบบโซเบลนั้น อาจมีบางส่วนที่เป็นจุดสีดำเดี่ยวๆ จำนวนมาก ซึ่งจะทำให้มีผลต่อการประมวลผลภาพในขั้นต่อไปได้ เช่น การหาขอบของทะเบียนรถยนต์ ดังนั้นจึงต้องกำจัดพิกเซลที่เป็นสิ่งรบกวนนี้ออกจากภาพ โดยการหาตำแหน่งของพิกเซลที่มีสีดำและมีจุดข้างเคียงเป็นสีขาวเกิน 4 จุด แล้วเปลี่ยนให้เป็นสีขาวแทน ผลของการกำจัดพิกเซลที่เป็นสิ่งรบกวนออกแล้วแสดงได้ในรูปที่ 4.6ข เมื่อเทียบกับตอนที่ยังไม่กำจัดสิ่งรบกวน จะเห็นได้ชัดเจนว่า ภาพมีขอบสีขาวที่ชัดเจนยิ่งขึ้น



ก) ก่อนกำจัดสิ่งรบกวน ข) หลังจากกำจัดสิ่งรบกวน

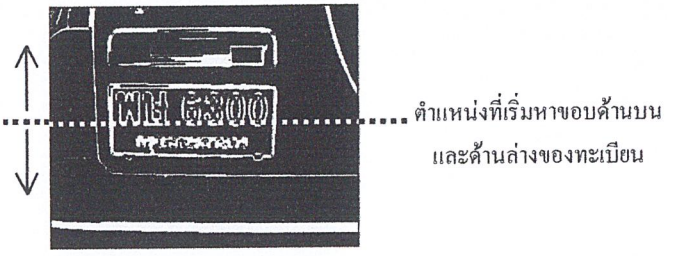
รูปที่ 4.6 ตัวอย่างภาพทะเบียนรถยนต์ก่อนและหลังจากกำจัดสิ่งรบกวนแล้ว

4.2.4 การหาตำแหน่งของทะเบียนรถยนต์

หลังจากได้ภาพที่ผ่านการตรวจหาขอบ และ Noise Reduction แล้ว จากนั้นนำภาพมาหาตำแหน่งของทะเบียนรถยนต์ โดยการหาจำนวนจุดสีขาวที่ติดกันเป็นแถบ ในขั้นแรกจะหาขอบแนวตั้งของทะเบียนรถยนต์ก่อน จะเริ่มหาจากคอลัมน์กึ่งกลางของรูปไล่ไปทางซ้ายของรูปเพื่อหาขอบด้านซ้าย และไล่ไปทางด้านขวาของรูปเพื่อหาขอบด้านขวา โดยหาจุดสีขาวที่ติดกันมีค่าอยู่ในช่วงที่น่าจะเป็นขอบด้านข้างได้ เมื่อเจอแล้วก็จะตรวจหาขอบตามแนวนอนต่อไป

รูปที่ 4.7 แสดงการหาขอบด้านข้างของป้ายทะเบียน

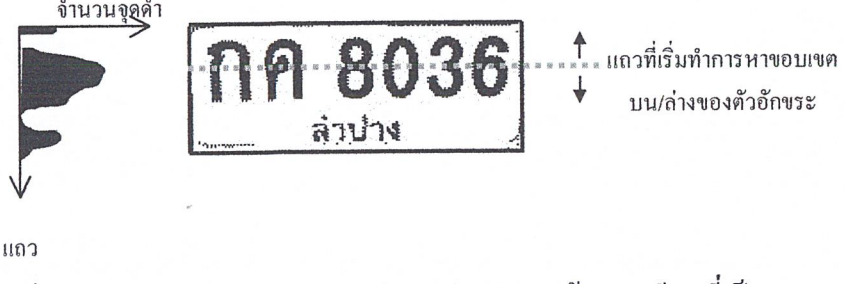
ส่วนการหาขอบตามแนวตั้งจะเริ่มจากตำแหน่งแถวกึ่งกลางของขอบด้านข้างของทะเบียนที่ตรวจเจอมาก่อนหน้านี้ โดย Scan ไปด้านบนและด้านล่างทีละแถวตามแนวนอน เมื่อเจอแถบสีขาวที่มีความยาวอยู่ในช่วงที่น่าจะเป็นขอบด้านบนและด้านล่างได้ ก็จะถือว่าแถวนั้นเป็นขอบด้านบนและด้านล่างของป้ายทะเบียน เมื่อได้ขอบด้านบนและด้านล่างแล้ว ก็จะนำภาพมาแยกแยะหาหมายเลขทะเบียนต่อไป



รูปที่4.8 แสดงการหาขอบด้านข้างของป้ายทะเบียน

4.2.5 การหาตำแหน่งของหมายเลขทะเบียน

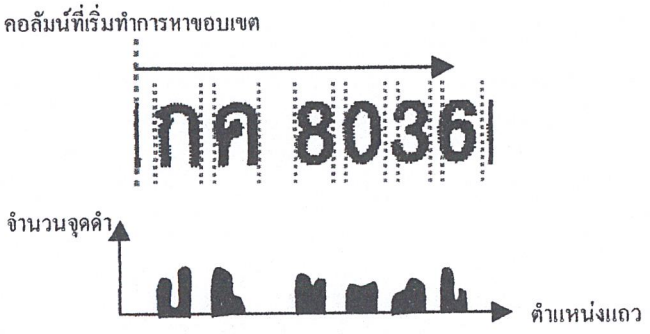
เริ่มจากทำการตัดภาพให้เหลือแต่ส่วนของป้ายทะเบียนก่อนเพื่อจะได้หาขอบเขตของตัวอักษรได้ง่ายขึ้นแล้วนำภาพส่วนป้ายทะเบียนแปลงให้เป็นภาพขาวดำ จากนั้นจึงทำการหาขอบเขตด้านบนและด้านล่างของหมายเลขทะเบียน โดยนับจำนวนของจุดสีดำในแต่ละแถวของภาพป้ายทะเบียนจนครบทุกแถว ทำการตรวจทีละแถว เริ่มจากแถวที่ 1/3 ของความสูงของป้ายทะเบียนตรวจไปด้านบน เมื่อพบแถวที่ทำให้จำนวนจุดดำต่ำลงจนเกือบจะเป็นศูนย์ ให้แถวนั้นเป็นขอบเขตบนของตัวอักขระ ทำเช่นเดียวกันกับขอบเขตล่าง เสร็จแล้วจะได้ขอบเขตบนและขอบเขตล่างของป้ายทะเบียน



รูปที่4.9 แสดงการหาขอบบน/ล่างของตัวอักขระจากป้ายทะเบียนที่เป็นภาพขาวดำ

เมื่อได้ขอบเขตด้านบนและด้านล่างแล้ว จึงนำมาหาขอบซ้ายและขวาของตัวอักขระแต่ละตัวต่อไป โดยตรวจทีละคอลัมน์เริ่มจากคอลัมน์ซ้ายสุดของป้ายทะเบียนเพื่อนับจำนวนจุดดำของแต่ละ

คอลัมน์ตั้งแต่ขอบบนและขอบล่างของหมายเลขทะเบียนที่หาได้ก่อนหน้านี้ แล้วหาขอบด้านซ้ายของตัวอักขระ โดยใช้วิธีพิจารณาคอลัมน์ที่มีการเปลี่ยนของจำนวนจุดดำจากศูนย์เป็นค่าใดๆ จะได้ขอบซ้ายของตัวอักขระ และคอลัมน์ที่มีการเปลี่ยนของจำนวนจุดดำจากค่าใดๆเป็นศูนย์ให้เป็นขอบขวาของตัวอักขระไปเรื่อยๆจนครบทุกตัว



รูปที่ 4.10 แสดงการหาขอบซ้าย/ขวาของตัวอักขระแต่ละตัวจากป้ายทะเบียน

เมื่อได้ขอบเขตครบทุกตัวแล้วจึงทำการตัดภาพให้เหลือเพียงตัวอักษร(ภาพละ 1 ตัวอักษร) จนได้จำนวนภาพเท่ากับจำนวนของตัวอักขระที่หาพบเพื่อนำไปทำการรู้จำตัวอักขระต่อไป

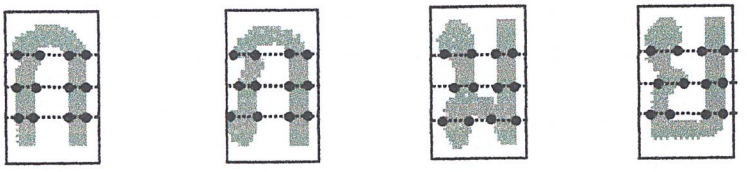
รูปที่ 4.11 ภาพตัวอักขระหลังจากตัดแยกออกเป็นตัวๆ

4.3 การจดจำทะเบียนรถยนต์

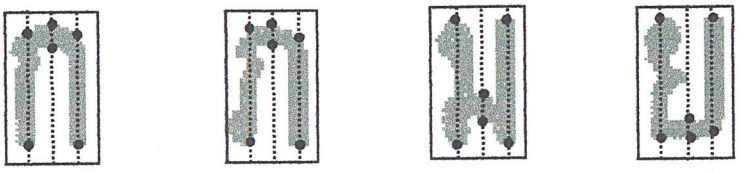
ในส่วนจดจำทะเบียนรถยนต์จะใช้ข้อมูลภาพตัวอักขระที่ได้มาจากขั้นตอนที่แล้วมาทำการวิเคราะห์ว่าเป็นตัวอักขระใด โดยในตอนแรกจะแบ่งตัวอักขระออกเป็น 2 กลุ่มคือ กลุ่มของตัวอักขระสองตัวแรก และอีกกลุ่มคือกลุ่มของตัวอักขระ 4 ตัวหลัง โดยทำการแยกพิจารณาคนละแบบ กลุ่มแรกจะใช้การจำแนกโดยการตัดและวิเคราะห์ร่วมกับการเทียบจุดภาพ ส่วนกลุ่มที่สองจะใช้วิธีการเทียบจุดภาพ

ในการตัดและวิเคราะห์ จะใช้การตัด 6 แถว คือ แนวนอน 3 แถว และแนวตั้ง 3 แถว ทำการแยกลักษณะของการตัดของแต่ละตัวอักขระเพื่อแยกแยะ โดยนับจำนวนจุดตัดของแต่ละแถว หลังจากแยกแยะแล้วจะมีตัวอักขระบางกลุ่มที่ยังแยกออกจากกันไม่ได้ เนื่องจากจุดตัดในแต่ละแถวเท่ากันหมด ซึ่งอาจแก้ปัญหาได้โดยเพิ่มเส้นที่ตัดให้มากขึ้นแต่ก็ทำให้โปรแกรมมีความซับซ้อนมากขึ้น

ซึ่งในที่นี้จะใช้นำเอาการเทียบจุดภาพเข้ามาช่วย โดยทำการเทียบจุดภาพกับภาพต้นแบบเฉพาะแค่ในกลุ่มตัวอักษรที่เหลืออยู่เท่านั้น ตัวอย่างเช่น ก,ภ,ม และ ย มีจำนวนจุดตัดเท่ากัน เมื่อหมายเลขทะเบียนมีตัวอักษร ก เข้ามาก็จะทำการเทียบจุดภาพแค่เฉพาะ ภาพต้นแบบที่เป็น ก,ภ,ม และ ย เท่านั้น ทำให้ช่วยลดจำนวนภาพต้นแบบที่นำมาเทียบให้น้อยลง ทำให้เวลาประมวลผลเร็วขึ้นด้วย



ก) แถวตัดแนวนอน



ข) แถวตัดแนวตั้ง

รูปที่ 4.12 แสดงตัวอย่างกลุ่มตัวอักษรที่มีจำนวนจุดบ่งชี้เท่ากันในทุกแถวตัด

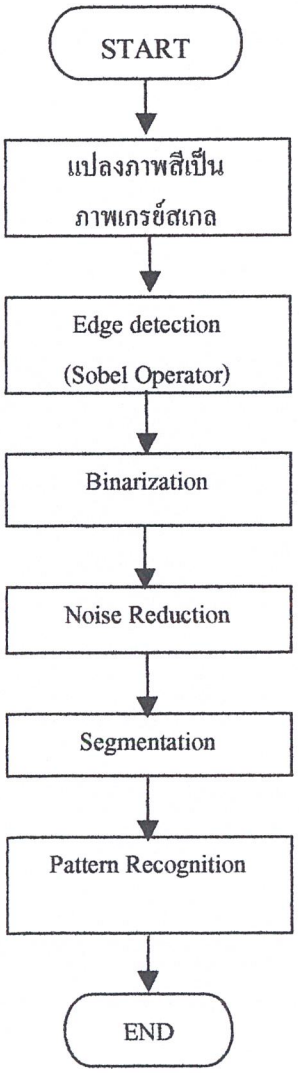
ตัวอักขระส่วนที่สองนั้น ทำการเทียบจุดภาพ โดยนำภาพหมายเลขทะเบียนมาเทียบกับภาพต้นแบบ แล้วนับจำนวนจุดภาพที่เหมือนกัน โดยเทียบตั้งแต่ภาพเลข 0 ถึงเลข 9 แล้วเพื่อดูว่าภาพหมายเลขทะเบียนนั้นเหมือนภาพใดมากที่สุด

บทที่ 5
โครงสร้างของโปรแกรม

ในส่วนของโปรแกรมที่ใช้ประมวลผลภาพในระบบจดจำทะเบียนรถยนต์นี้เขียนขึ้นด้วย Delphi 5 ซึ่งอยู่บนพื้นฐานของภาษาปาสคาล โดยในการเขียนโปรแกรมเพื่อทำการประมวลผลภาพนี้ ได้แบ่งออกเป็น ส่วนของโปรแกรมหลัก และส่วนของโปรแกรมย่อยเพื่อแยกการประมวลผลออกเป็นส่วนๆ

5.1 โปรแกรมหลัก

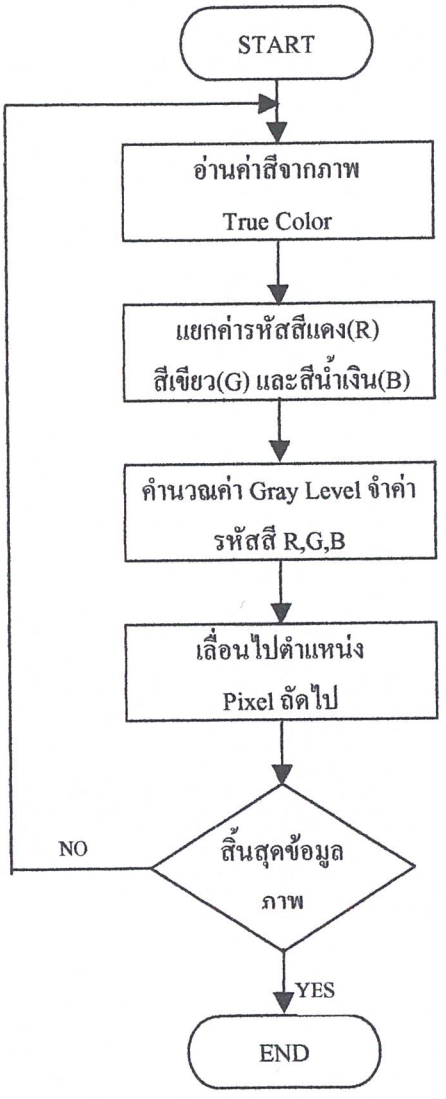
ส่วนของโปรแกรมหลัก จะมีขั้นตอนในการเรียกใช้โปรแกรมย่อยดังแสดงในรูปที่ 5.1



รูปที่ 5.1 แผนผังการทำงานของโปรแกรมหลัก

5.2 โปรแกรมแปลง ภาพสีเป็นภาพเกรย์สเกล

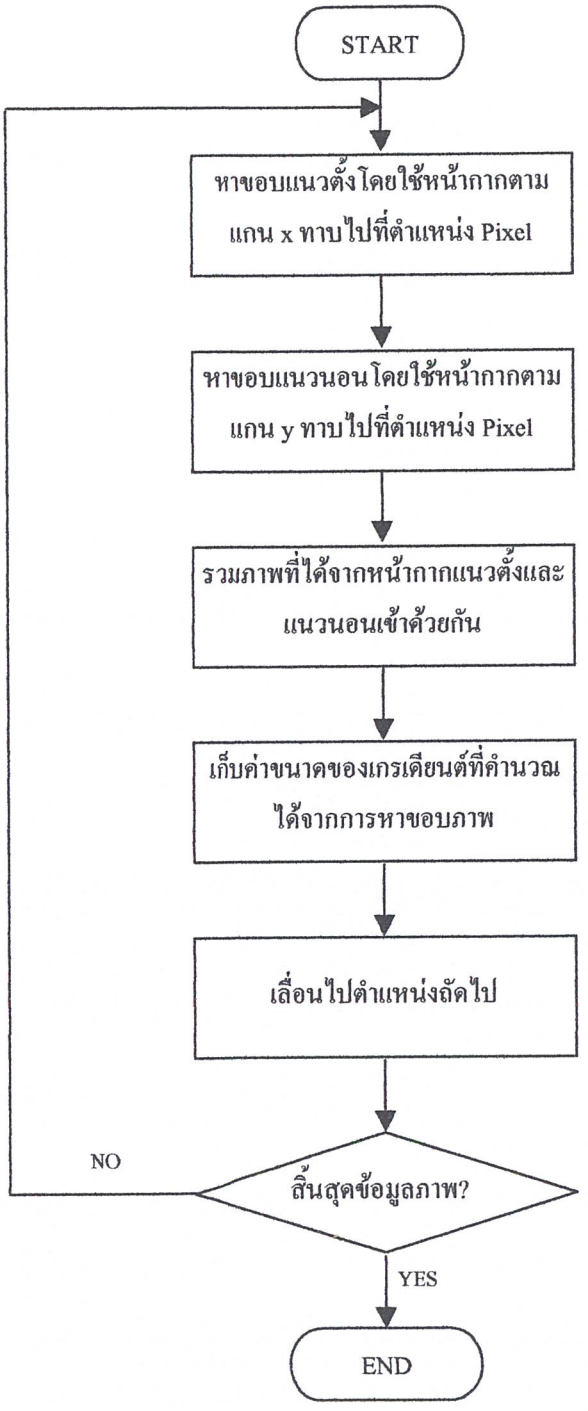
ก่อนที่จะนำภาพไปประมวลผล ต้องทำการแปลงภาพสีแบบ True Colour เป็นภาพแบบเกรย์สเกล 256 ระดับก่อน โดยใช้ค่าเฉลี่ยของค่าสีแดง สีเขียว และสีน้ำเงิน โดยมีขั้นตอนดังรูปที่ 5.2



รูปที่ 5.2 แสดงแผนผังโปรแกรมแปลงภาพสีเป็นภาพเกรย์สเกล

5.3 โปรแกรมตรวจหาขอบของภาพ

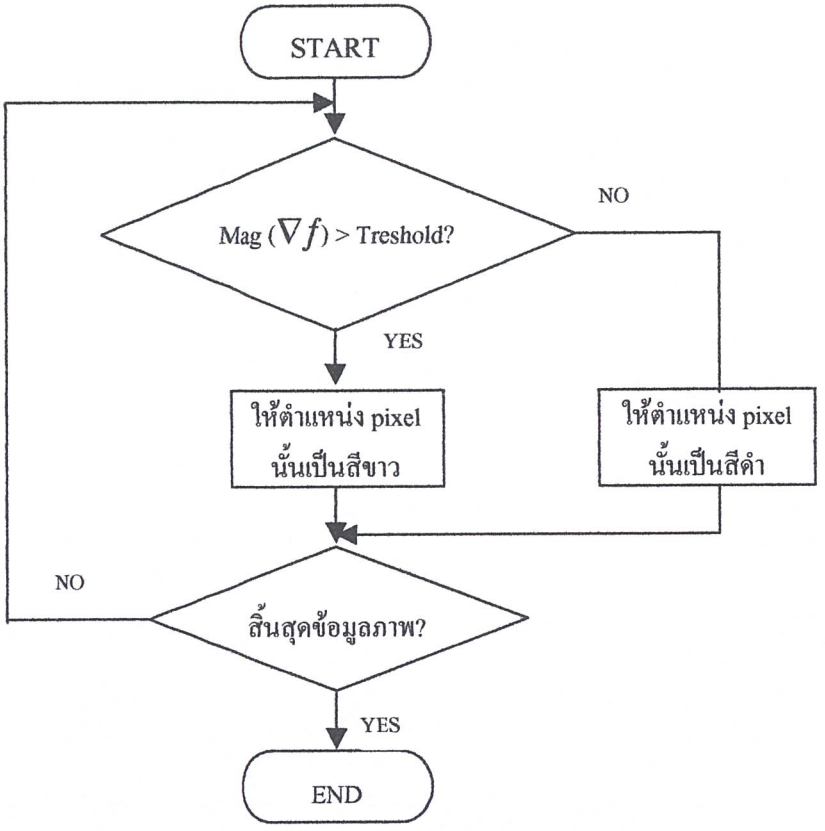
เพื่อให้ง่ายต่อการหาขอบของป้ายทะเบียน จึงต้องมีการหาขอบของภาพโดยรวมทั้งหมดก่อน ซึ่งจะใช้วิธีเกรเดียนต์ มีหน้ากากแบบโซเบล ขนาด 5x5



รูปที่5.3 แสดงแผนผังการทำงานของโปรแกรมหาขอบภาพ

5.4 โปรแกรมการทำ Binarization

เมื่อผ่านการทำ Edge detection แล้ว จะได้ภาพแบบเกรย์สเกล จากนั้นจึงนำภาพมาแปลงเป็นภาพขาวดำ เพื่อให้ง่ายต่อการตรวจสอบขอบเขตของป้ายทะเบียน โดยกำหนดค่า Threshold ที่เหมาะสม ในส่วนโปรแกรมแสดงได้ดังรูปที่ 5.4

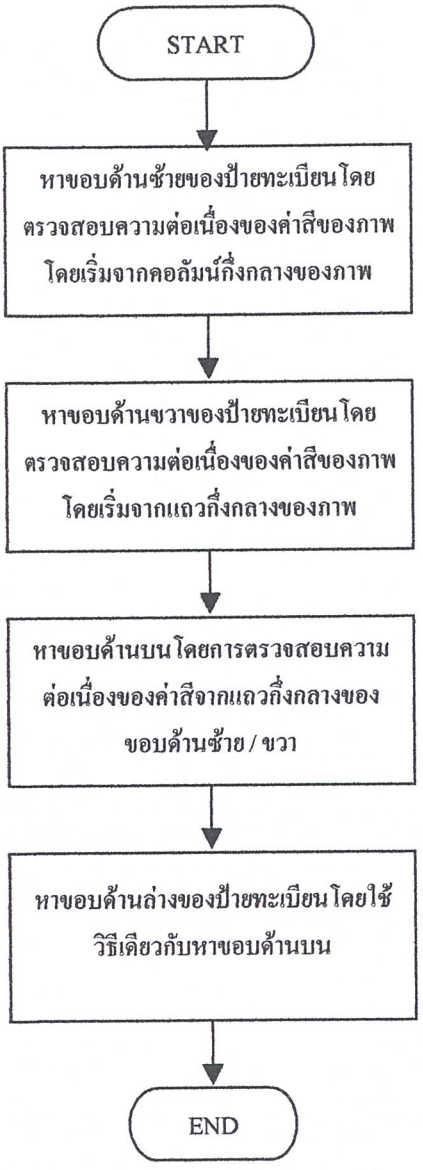


รูปที่ 5.4 แสดงแผนผังการทำงานของโปรแกรม Binarization

เมื่อได้ภาพที่เป็นขาวดำแล้ว ก็จะนำภาพที่ได้มาหาขอบเขตของป้ายทะเบียนต่อไป

5.5 โปรแกรมการหาตำแหน่งป้ายทะเบียน

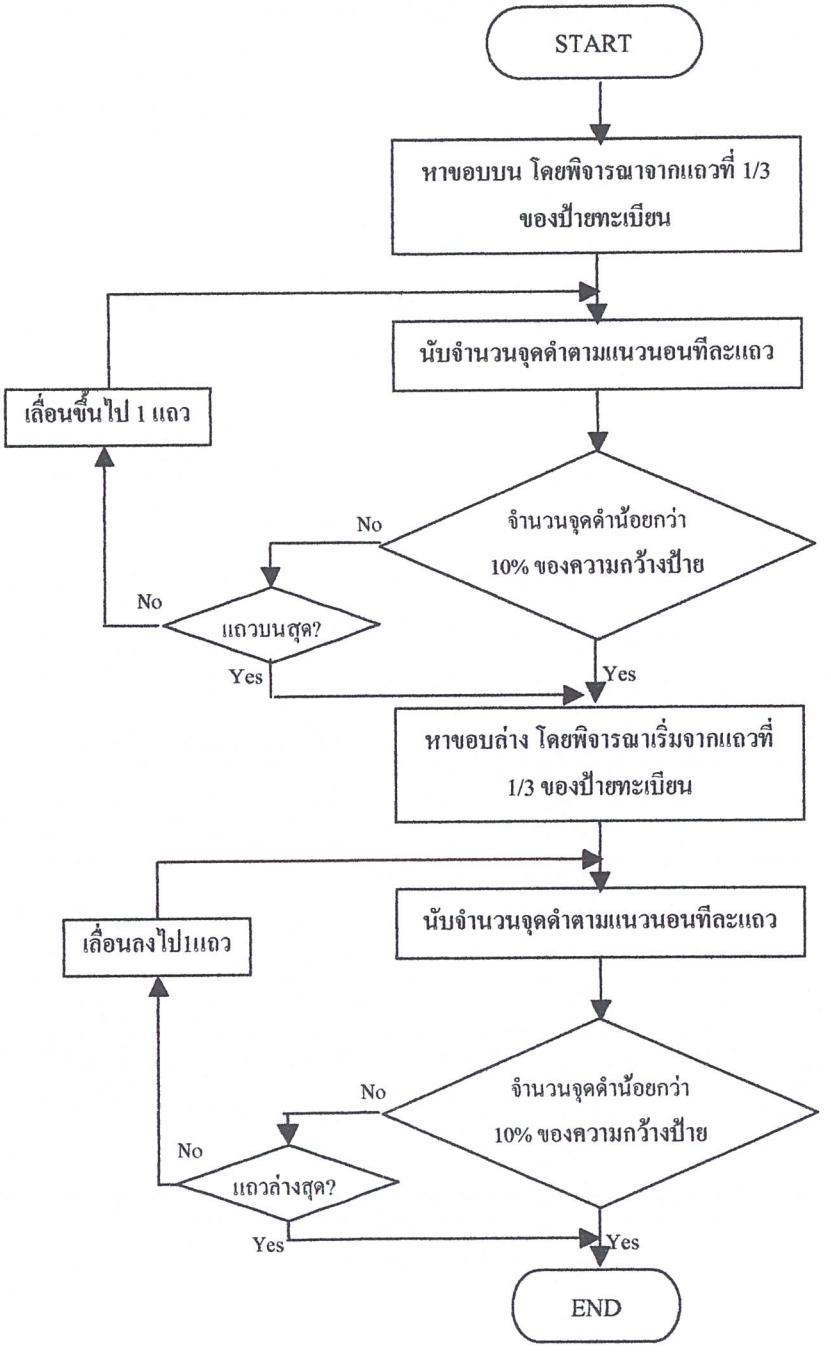
ในการหาตำแหน่งของป้ายทะเบียน จะนำภาพขาวดำที่ได้มาหาขอบเขตของป้ายทะเบียน โดยการหาตำแหน่งสีขาวที่มีลักษณะต่อเนื่องกันอยู่ในช่วงที่น่าจะเป็นทะเบียนรถยนต์ได้ ดังแสดงในโปรแกรมดังรูปที่ 5.5



รูปที่ 5.5 แสดงแผนผังการทำงานของโปรแกรมแยกป้ายทะเบียนจากรูป เมื่อได้ภาพที่หลือแต่ส่วนของทะเบียนรถยนต์ ก็จะนำภาพนี้มาตรวจหาตัวอักษรต่อไป

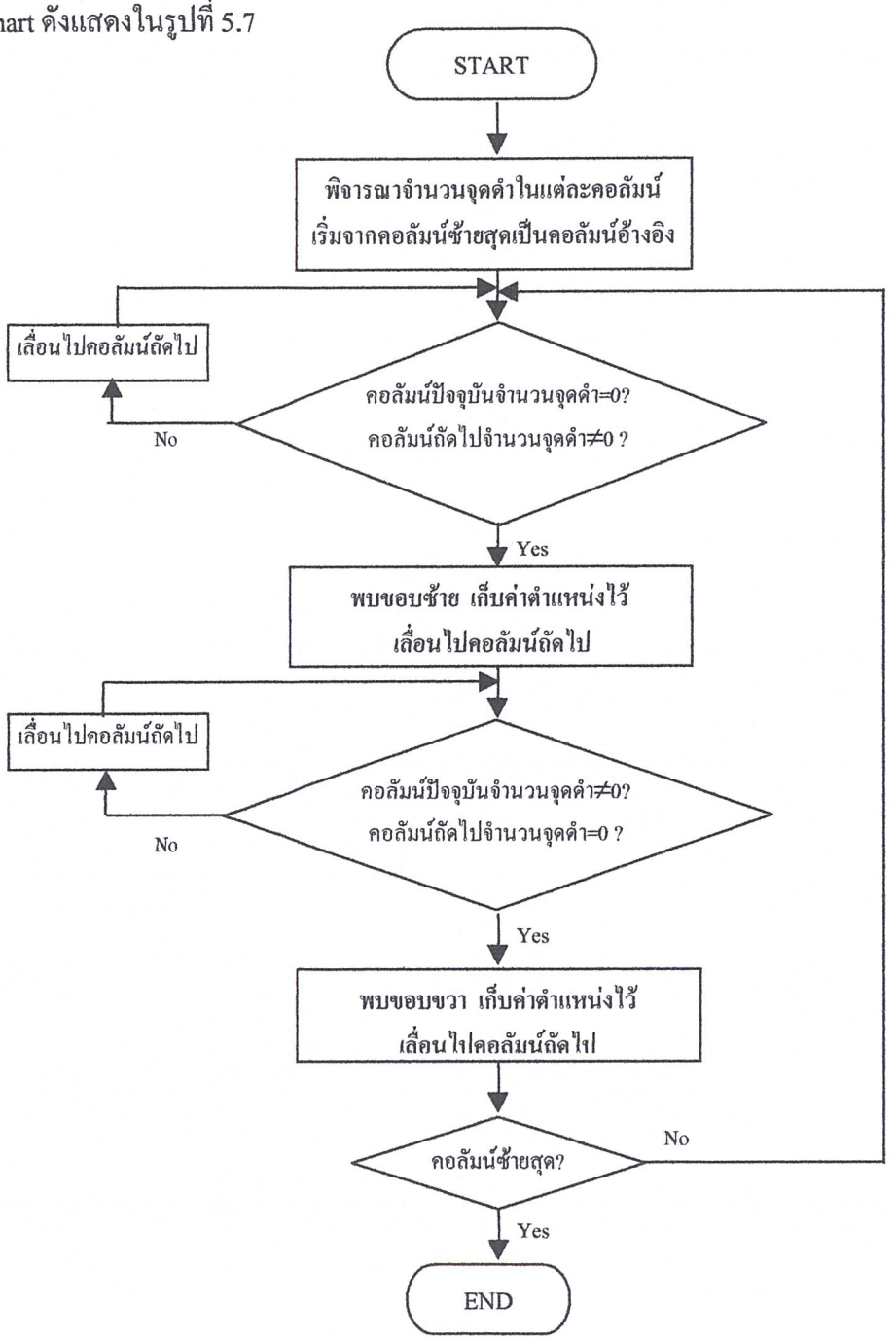
5.6 โปรแกรมหาตำแหน่งหมายเลขทะเบียน

เมื่อได้ภาพป้ายทะเบียนแล้ว ต้องแปลงภาพให้เป็นภาพขาวดำก่อน แล้วจึงมาแยกขอบเขตของหมายเลขทะเบียน ในที่นี้จะใช้การนับจำนวนจุดดำในแต่ละแถวของป้ายทะเบียนแล้วพิจารณาจำนวนจุดดำในภาพเพื่อหาขอบเขตบนและล่างของหมายเลขทะเบียน (ให้ตัวอักขระทุกตัวมีขอบเขตบนและล่างเป็นค่าเดียวกัน ดังแสดงใน Flow Chart ในรูปที่ 5.6



รูปที่ 5.6 แสดงแผนผังการทำงานของส่วนหาขอบเขตบนและล่างของหมายเลขทะเบียน

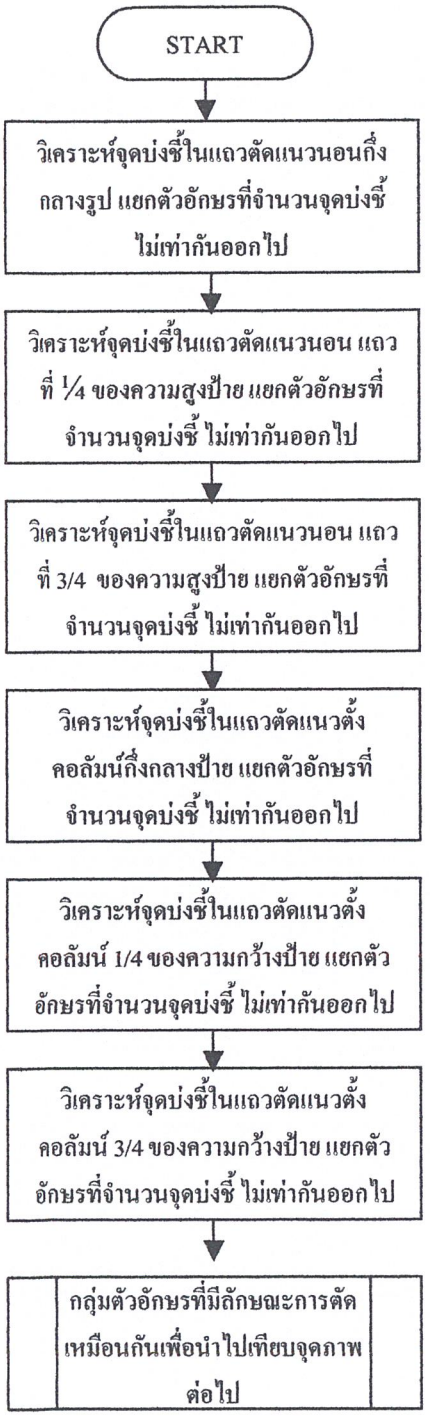
จากนั้นทำการนับจุดดำทีละหลัก ทำการนับตั้งแต่ขอบเขตบนถึงขอบเขตล่างของหมายเลขทะเบียนที่หาได้ก่อนหน้านี้ แล้วจึงแยกแยะขอบเขตซ้ายและขวาของตัวอักขระแต่ละตัว โดยพิจารณาการเปลี่ยนของจำนวนจุดดำ คือ จำนวนจุดดำเปลี่ยนจากศูนย์เป็นค่าใดๆ คือขอบซ้ายของตัวอักขระ แต่ถ้าจำนวนจุดดำเปลี่ยนจากค่าใดๆเป็นศูนย์ คือขอบขวาของตัวอักขระ จนครบทุกตัว แล้วจึงตัดป้ายทะเบียนเป็นส่วนๆตามตัวอักขระเพื่อเข้าสู่การจดจำหมายเลยทะเบียนต่อไป โดยมี Flow Chart ดังแสดงในรูปที่ 5.7



รูปที่ 5.7 แสดงแผนผังการทำงานของส่วนหาขอบเขตซ้ายและขวาของหมายเลขทะเบียน

5.7 โปรแกรมจดจำหมายเลขทะเบียน

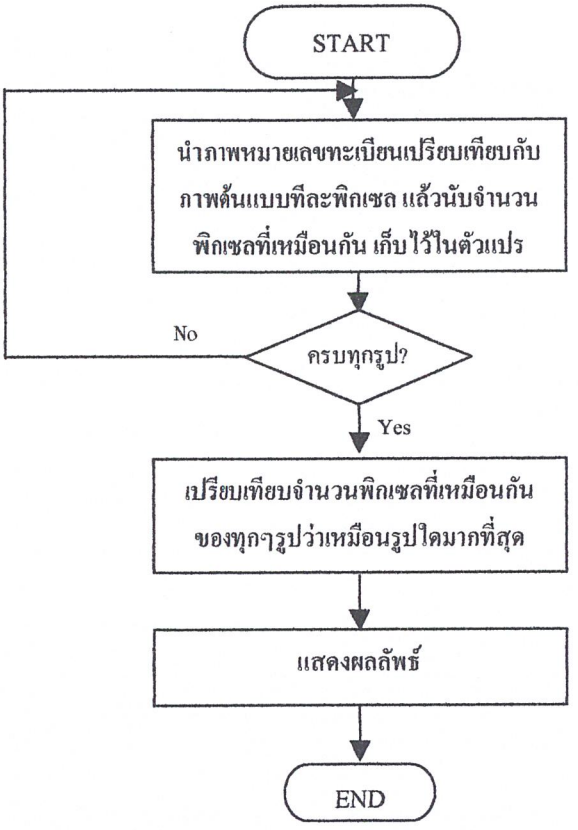
ในที่นี้จะแบ่งตัวอักขระเป็น 2 ส่วน ส่วนแรกคือตัวอักขระ 2 ตัวแรก และส่วนที่สองคือ ตัวอักขระ 4 ตัวหลัง โดยในส่วนแรกจะใช้การจะจำแบบตัดและจำแนกร่วมกับการเทียบจุดภาพ ดัง Flow Chart ดังรูปที่ 5.8



รูปที่ 5.8 แสดงแผนผังการจดจำหมายเลยทะเบียนแบบตัดและจำแนก

เมื่อได้กลุ่มของตัวอักขระที่น่าจะเป็นไปได้แล้ว ก็นำภาพหมายเลขทะเบียนมาทำการเทียบจุดภาพเฉพาะในกลุ่มภาพตัวอักขระต้นแบบที่น่าจะเป็นไปได้เท่านั้น

ตัวอักขระส่วนที่สองจะนำมาเข้าสู่การจดจำทะเบียนแบบเทียบจุดภาพซึ่งมีแผนผังการทำงานดังต่อไปนี้



รูปที่ 5.9 แสดงแผนผังการทำงานของส่วนจดจำหมายเลขทะเบียนแบบเทียบจุดภาพ

บทที่ 6
การทดลองและผลการทดลอง

6.1 การทดลองส่วนของการหาขอบเขตป้ายทะเบียนและหมายเลขทะเบียน

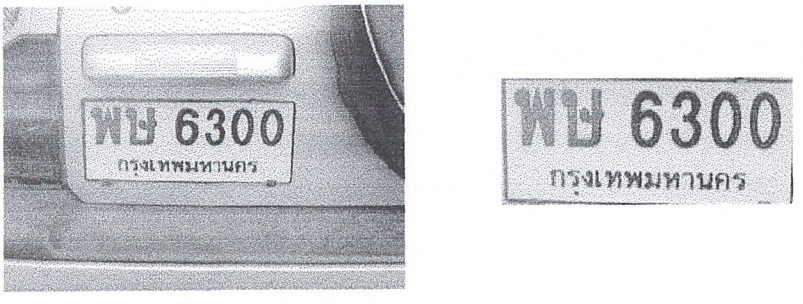
สำหรับการทดลอง จะทำการศึกษาปัจจัยต่างๆ ที่มีผลต่อการทำงานของโปรแกรมเพื่อเป็นแนวทางในการออกแบบและปรับปรุงแก้ไขต่อไป โดยในการทดลองจะใช้ภาพตัวอย่างที่ถ่ายทางด้านหน้าและด้านหลังของรถยนต์ และจัดเก็บในรูปแบบของภาพบิตแมปแบบ True Color

ในการทดลองโปรแกรมส่วนแยกป้ายทะเบียนออกจากภาพ ผลการทำงานของโปรแกรมพบว่า มีทั้งภาพที่สามารถแยกป้ายทะเบียนออกจากภาพได้ครบทุกตัว และภาพที่ไม่สามารถแยกป้ายทะเบียนออกมาได้เลย ดังแสดงได้ในตารางที่ 6.1

ตารางที่ 6.1 แสดงผลการทดลองส่วนแยกป้ายทะเบียนและแยกตัวอักษรจากภาพ

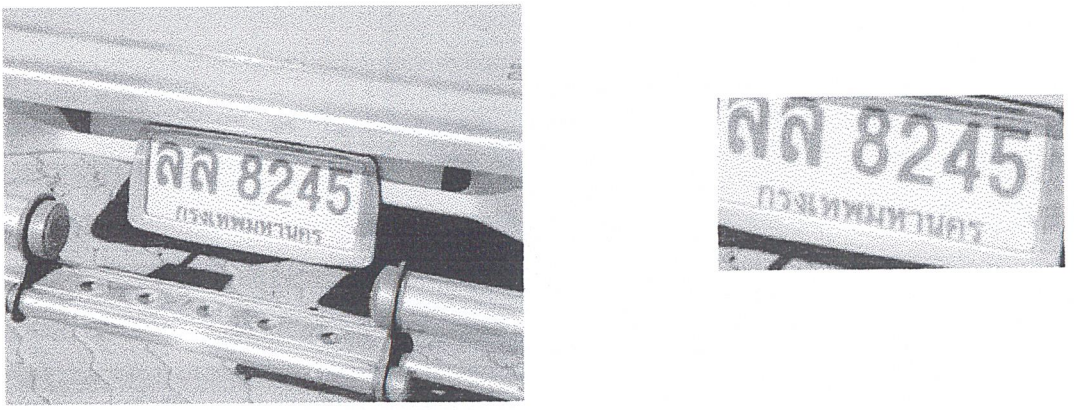
ผลการทดลอง	จำนวนภาพ (จากทั้งหมด 102 ภาพ)	เปอร์เซ็นต์
ตัดป้ายทะเบียนได้	98	96.07
ตัดตัวอักษรได้	79	77.45

จากผลการทดลองจะแยกผลออกเป็นสองส่วน คือ ส่วนที่ตัดป้ายทะเบียนได้ คิดเป็น 96.07% และส่วนที่ตัดตัวอักษรได้ คิดเป็น 77.45% ในส่วนที่สามารถตัดป้ายทะเบียนได้นั้นจะต้องเป็นภาพที่มีป้ายทะเบียนอยู่ประมาณกึ่งกลางของภาพ และต้องไม่เอียงมากนัก แสงและเงาของรูปก็จะต้องชัด ไม่มีเงามาบังบริเวณป้ายทะเบียน ตัวอย่างของภาพที่สามารถแยกป้ายทะเบียนออกมาได้ชัดเจน แสดงได้ดังรูปที่ 6.1



รูปที่ 6.1 แสดงภาพที่แยกป้ายทะเบียนได้ชัดเจน

ในส่วนของภาพที่เอียง ถ้าเป็นภาพที่เอียงไม่มากจะสามารถหาขอบเขตของป้ายทะเบียนได้ ดังรูปที่ 6.2



รูปที่ 6.2 ตัวอย่างภาพที่มีป้ายทะเบียนเอียง

ส่วนภาพที่ไม่สามารถตรวจหาขอบเขตของป้ายทะเบียนได้ มีสาเหตุดังต่อไปนี้

- ป้ายทะเบียนมีแสงและเงาที่ไม่สม่ำเสมอกัน ดังรูปที่ 6.3 จะมีเงาพาดผ่านที่ป้ายทะเบียน

รูปที่ 6.3 แสดงตัวอย่างภาพที่มีเงาพาดผ่านป้ายทะเบียน

- ภาพที่มีขนาดไม่เหมาะสม ดังตัวอย่างรูปที่ 6.4

รูปที่ 6.4 ตัวอย่างภาพที่มีขนาดป้ายทะเบียนเล็กเกินไป

- สภาพแวดล้อมไม่เหมาะสม ดังแสดงตัวอย่างในรูปที่ 6.5

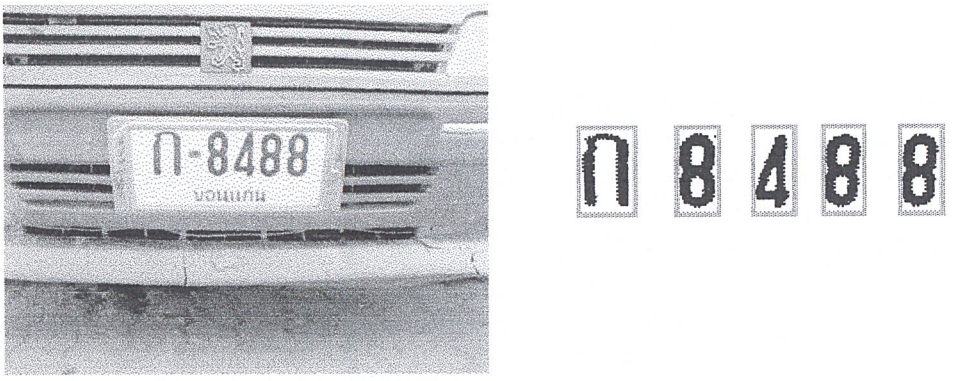
ก) ภาพสีของด้านหน้ารถ ข) ภาพที่ผ่านการทำ Edge Detection

รูปที่ 6.5 แสดงตัวอย่างภาพที่มีสภาพแวดล้อมไม่เหมาะสม

จากรูปที่ 6.5 บริเวณป้ายทะเบียนของรถมีสีเดียวกับรถ ทำให้หาบริเวณขอบของป้ายทะเบียนไม่ได้ เพราะการทำ Edge Detection จะหาขอบได้เฉพาะบริเวณที่มีสีแตกต่างกันเท่านั้น ดังจะเห็นได้จากรูปที่ 6.5ก บริเวณขอบของป้ายทะเบียนจะไม่มีเส้นที่เป็นสีขาวเลย จึงทำให้หาป้ายทะเบียนไม่ได้

จากภาพที่ทำการแยกป้ายทะเบียนได้แล้ว เมื่อนำมาแยกแยะขอบเขตของหมายเลขทะเบียน ในการทดลองส่วนนี้ ผลการทำงานของโปรแกรมพบว่า มีทั้งภาพที่สามารถแยกหมายเลข

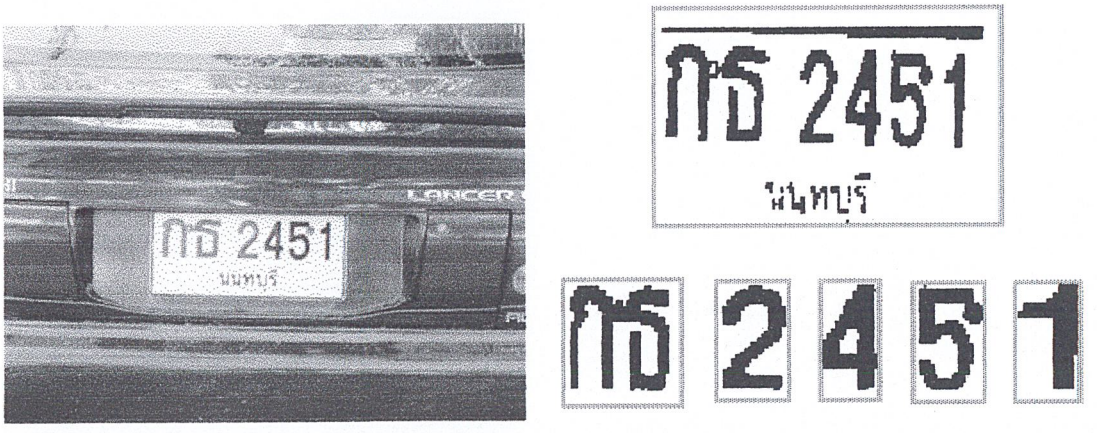
ทะเบียนได้ครบทุกตัว และภาพที่ไม่สามารถแยกหมายเลขทะเบียนได้เลย หรืออาจได้แต่ไม่ครบทุกตัว ซึ่งในส่วนนี้จะได้ผลการหาขอบเขตหมายเลขทะเบียนที่ชัดเจนก็ต่อเมื่อป้ายที่ทำการตัดได้ชัดเจน รูปที่ 6.6 แสดงให้เห็นถึงการหาขอบเขตของหมายเลขทะเบียนได้ชัดเจน



รูปที่ 6.6 แสดงตัวอย่างของการหาขอบเขตหมายเลขทะเบียนได้ชัดเจน

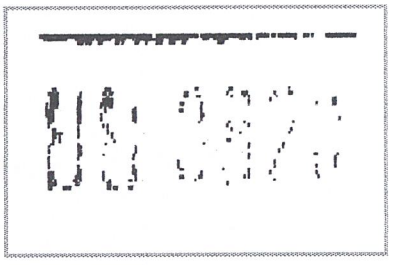
ป้ายทะเบียนบางอัน ไม่สามารถหาขอบเขตของหมายเลขทะเบียนได้ เนื่องมาจากสาเหตุหลายประการ เช่น

- มีน็อตอยู่กึ่งกลางระหว่างตัวอักษร 2 ตัว ดังรูปที่ 6.7



รูปที่ 6.7 แสดงตัวอย่างการหาขอบเขตหมายเลขทะเบียนไม่ได้เพราะมีน็อต

- สีของตัวอักษรอ่อนเกินไป ทำให้ค่า Threshold ที่คำนวณจากค่า Mean จึงต่ำ ทำให้หมายเลขทะเบียนไม่ชัดเจน หาขอบเขตหมายเลขทะเบียนไม่ได้



รูปที่ 6.8 ตัวอย่างภาพป้ายทะเบียนที่มีตัวอักษรสีอ่อน หาขอบเขตหมายเลขทะเบียนไม่ได้

6.2 การทดลองส่วนของการจดจำหมายเลขทะเบียน

การทดลองเพื่อศึกษาถึงผลการรู้จำตัวอักขระแบบตัดและวิเคราะห์ และการเทียบจุดภาพ เพื่อหาว่าโปรแกรมนี้เหมาะสมกับสภาพแวดล้อมแบบใด ค่าเอาท์พุตที่ได้มีความผิดพลาดมากน้อยเพียงใด โดยจะพิจารณาทั้งค่าของเอาท์พุตจริงแต่ละตัวเมื่อเทียบกับค่าอินพุตที่ตรงกับเงื่อนไขของโปรแกรม และค่าของร้อยละความถูกต้องสมบูรณ์ของแผ่นป้ายทะเบียนเพื่อพิจารณาความเป็นไปได้ในการใช้งานและเป็นแนวทางในการออกแบบพัฒนาและแก้ไขต่อไป

ตารางที่ 6.2 แสดงผลการทดลองส่วนรู้จำตัวอักษร โดยวิธีตัดและจำแนกและวิธีเทียบจุดภาพ

ผลการรู้จำ	วิธีที่ 1		วิธีที่ 2	
	จำนวนภาพ (จากจำนวนภาพที่ตัดอักษรได้ 79 ภาพ)	เปอร์เซ็นต์	จำนวนภาพ (จากจำนวนภาพที่ตัดอักษรได้ 79 ภาพ)	เปอร์เซ็นต์
ถูกต้องทุกตัว	58	73.41	67	84.81
รู้จำผิด 1 ตัว	16	20.25	9	11.39
รู้จำผิด 2 ตัว	4	5.06	2	2.53
รู้จำผิด 3 ตัว	1	1.26	1	1.26

หมายเหตุ : **วิธีที่ 1** คือ ใช้วิธีตัดและจำแนกกับตัวอักษรสองตัวแรก และใช้การเทียบจุดภาพกับตัวเลข 4 ตัวหลัง

: **วิธีที่ 2** คือ ใช้วิธีเทียบจุดภาพกับทุกตัวอักษรในป้ายทะเบียน

จากข้อมูลภาพที่ตัดป้ายทะเบียนได้ทั้งหมด 79 ภาพ ผลการทดลองโดยการรู้จำวิธีที่ 1 สามารถจดจำตัวอักษรได้ถูกต้องเป็นจำนวน 73.41 % ของจำนวนตัวอักษรที่แยกได้ทั้งหมด ส่วนวิธีที่ 2 ให้ผลการจดจำตัวอักษรได้ถูกต้องเป็นจำนวน 84.81 % ของจำนวนตัวอักษรที่แยกได้ทั้งหมด

6.3 เวลาในการทำงานของโปรแกรม

เวลาที่ใช้ในการประมวลผลของโปรแกรมแสดงได้ดังตารางที่ 6.3 จะเห็นว่าวิธีที่ 1 จะใช้เวลาน้อยกว่าวิธีที่ 2 ถึง 34.82%

ตารางที่ 6.3 เวลาที่ใช้ในการประมวลผลของโปรแกรม

เวลาที่ใช้ในการประมวลผล (วินาที)	
วิธีที่ 1	วิธีที่ 2
1.211	1.858

บทที่ 7
สรุปและวิจารณ์ผลการทดลอง

จากโครงงานโดยรวมทั้งหมด ซึ่งประกอบไปด้วย 2 ส่วนคือ

- การเตรียมข้อมูลภาพ
- การจดจำทะเบียนรถยนต์

7.1 ส่วนเตรียมข้อมูลภาพ

ในส่วนของการเตรียมข้อมูลภาพนี้จะนำภาพด้านหน้าหรือด้านหลังรถยนต์มาหาขอบเขตของป้ายทะเบียน ซึ่งป้ายทะเบียนที่ได้อาจไม่สมบูรณ์หรือหาป้ายทะเบียนไม่ได้เนื่องจากสาเหตุหลายประการ ได้แก่

- ขนาดของภาพไม่เหมาะสม
- ตำแหน่งของป้ายทะเบียนในภาพ ถ้าป้ายทะเบียนไม่อยู่บริเวณกึ่งกลางของรูป ก็จะทำให้หาขอบเขตของป้ายทะเบียนไม่ได้
- สภาพของแสงภายนอก ตัวอย่างเช่น ความมืด ความสว่าง มีแสงสะท้อนที่ป้ายทะเบียนทำให้ไม่สามารถแยกส่วนของป้ายทะเบียนออกมาได้
- สิ่งแวดล้อมโดยรวมของภาพ เช่น สีของรถ สีของขอบป้ายทะเบียน เป็นต้น
- ความเอียงของรูป อาจทำให้ได้ป้ายทะเบียนที่มีขอบเขตไม่เหมาะสม

จากนั้นนำป้ายทะเบียนมาหาขอบเขตของตัวอักษรทุกตัวในป้ายทะเบียน ซึ่งการหาขอบเขตอาจไม่สมบูรณ์เนื่องมาจากสาเหตุหลายประการ ได้แก่

- การตัดป้ายทะเบียนไม่สมบูรณ์
- บนป้ายทะเบียนมีน็อตอยู่กึ่งกลางระหว่างตัวอักษรสองตัว ทำให้แบ่งขอบเขตไม่ได้
- สีของหมายเลขทะเบียนอ่อนหรือซีดไป เช่นสีเขียว หรือป้ายเก่าทำให้สีของตัวอักษรซีด
- แสงบริเวณป้ายทะเบียนจ้าเกินไป
- ป้ายทะเบียนเอียง เมื่อวิเคราะห์ขอบเขตบนและล่างทำได้ไม่ดีเนื่องจากในโปรแกรมนี้เลือกวิเคราห์ขอบเขตบนและล่างเป็นค่าเดียวกันทุกตัวอักษร

ดังนั้นในทางปฏิบัติสามารถนำข้อจำกัดดังกล่าวเป็นสิ่งสำคัญในการออกแบบระบบที่ใช้งานจริง โดยเฉพาะในส่วนการรับข้อมูลภาพและการประมวลผลขั้นต้น ซึ่งต้องออกแบบให้ครอบคลุมถึงข้อจำกัดทุกๆ ข้อ เพื่อให้ระบบสามารถทำงานได้อย่างถูกต้องและแม่นยำ ซึ่งจะต้องทำการปรับปรุงโปรแกรมที่มีอยู่ให้สมบูรณ์ยิ่งขึ้น ซึ่งจะเป็นแนวทางที่ต้องพัฒนาต่อไปในอนาคต

ผลการทดลองและข้อจำกัดต่างๆเหล่านี้ จะเป็นประโยชน์อย่างยิ่งในการออกแบบเพื่อที่จะทำให้ระบบสามารถทำงานได้อย่างถูกต้องและได้ผลลัพธ์ตามต้องการ สามารถนำผลลัพธ์นี้เข้าสู่กระบวนการจดจำรูปแบบต่อไปซึ่งเป็นขั้นตอนสำคัญที่จะทำให้ระบบใช้งานได้อย่างมีประสิทธิภาพ

7.2 ส่วนจดจำทะเบียนรถยนต์

จากผลการทดลองของโปรแกรมในส่วนของการจดจำเลขทะเบียนรถยนต์ ในส่วนการวิเคราะห์แบบตัดและจำแนก จะเห็นว่าประสิทธิภาพของกระบวนการจดจำวิธีนี้จะขึ้นอยู่กับจำนวนแถวตัดที่ใช้ ถ้ายิ่งแถวตัดมีค่ามากก็จะทำให้แยกแยะตัวอักษรได้มาก ลดขนาดของกลุ่มตัวอักษรที่มีความเป็นไปได้ลง แต่ก็จะทำให้การวิเคราะห์มีความยุ่งยากมากขึ้นไปอีก และยังทำให้ผลมีความผิดพลาดได้ง่ายในกรณีที่ตัวอักษรที่ตัดมีขอบเขตไม่เหมาะสมหรือมีสิ่งรบกวนในภาพ

ในส่วนการจดจำแบบเทียบจุดภาพนั้น ประสิทธิภาพของกระบวนการจดจำวิธีนี้จะขึ้นอยู่กับความชัดเจนของภาพต้นแบบที่ทำมาเปรียบเทียบ และจำนวนของภาพต้นแบบ ยิ่งถ้าหากภาพต้นแบบมีจำนวนมากก็จะยิ่งครอบคลุมกรณีของอินพุตได้มากขึ้น ทำให้ผลที่ได้มีความถูกต้องมากขึ้น แต่ก็จะทำให้เวลาในการประมวลผลมากตามไปด้วย

ซึ่งโดยรวมแล้วจะสามารถสรุปได้ว่า วิธีที่นำเสนอนี้ ข้อมูลที่ได้จะขึ้นอยู่กับลักษณะทางกายภาพเป็นอย่างมาก โดยผลที่ได้จะต้องผ่านกระบวนการเตรียมข้อมูลตั้งแต่การแปลงภาพเป็นไบนารี การแยกป้ายทะเบียน การแยกหมายเลขทะเบียน ซึ่งอาจทำให้สูญเสียลักษณะสำคัญของหมายเลขทะเบียนไปได้ เช่นหัวของตัวอักษรในตัว ภ ถ้าผ่านการแปลงภาพหรือการตัดตัวอักษรที่ไม่ดี ก็จะทำให้ผลการรู้จำกลายเป็น ก ไปได้ ทำให้โปรแกรมนี้มีข้อจำกัดในด้านต่างๆมาก

บรรณานุกรม

Gonzalez , and R.E. Woods , **Digital Image Processing** , Addison-Wesley , 1992

Wayne Niblack , **An Introduction to Digital Image Processing** , Prentice Hall , 1986

ชัยสิทธิ์ ศิริเจริญไชย และวุฒิพงศ์ อารีกุล , **การหาตำแหน่งป้ายทะเบียนรถยนต์โดยการตรวจสอบขอบชัน** , การประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 24 , กรุงเทพมหานคร, หน้า 1176-1181 , 2544

เอกรัฐ เลากุลรัตน์ และ วุฒิพงศ์ อารีกุล , **การรู้จำป้ายทะเบียนรถยนต์ไทยโดยใช้วิธีการตัดและจำแนก** , การประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 24 , กรุงเทพมหานคร , หน้า 1260-1265 , 2544