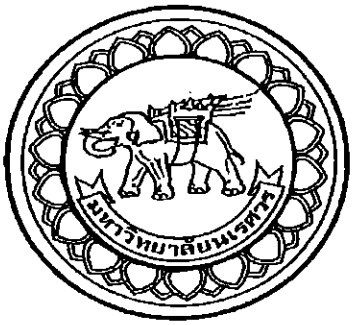
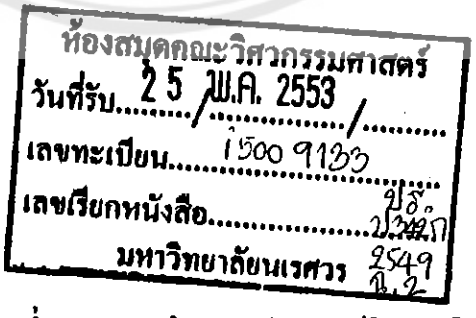


การหาขอบภาพสีแบบเรียลไทม์

The Real-time Color Image Edge Detection

นางสาวประภาพรรณ ชูพินิจ รหัส 46360038

นายกีรดิษย์ บุญญลาภาเลิศ รหัส 46361853



ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2549

ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน	การหาขอบภาพสีแบบเรียลไทม์		
ผู้ดำเนินโครงงาน	นางสาวประภาพรรณ	ชูพินิจ	รหัส 46360038
	นายกีรดิษย์	บุญญลาภาเลิศ	รหัส 46361853
อาจารย์ที่ปรึกษา	ดร.สุรเดช	จิตประไพกุลศาล	
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2549		

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะกรรมการการสอบโครงงานวิศวกรรม

.....ประธานกรรมการ
(ดร.สุรเดช จิตประไพกุลศาล)

.....กรรมการ
(ดร.พนมขวัญ ริยะมงคล)

.....กรรมการ
(ดร.อัครพันธ์ วงศ์กังแห)

หัวข้อโครงงาน	การหาขอบภาพสีแบบเรียลไทม์		
ผู้ดำเนินโครงงาน	นางสาวประภาพรรณ	ชูพินิจ	รหัส 46360038
	นายกีรติษย์	บุญญลาภาเลิศ	รหัส 46361853
อาจารย์ที่ปรึกษา	ดร.สุรเดช	จิตประไพกุลศาล	
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2549		

.....

บทคัดย่อ

โครงงานนี้เป็นการพัฒนาโปรแกรมเพื่อใช้หาขอบภาพสีแบบเรียลไทม์โดยใช้หลักการของ Color fusion ร่วมกับทฤษฎีในการหาขอบภาพ 4 algorithm ด้วยกันคือ Sobel, Robinson, Laplace first order derivative, Laplace second order derivative โดยจากโปรแกรมสามารถหาขอบภาพในบริเวณที่ต้องการได้โดยใช้พิกัดของเมาส์เป็นตัวกำหนด

จุดเด่นของโปรแกรมนี้อยู่ที่กระบวนการทำงานของโปรแกรมคือ แทนที่จะทำการประมวลผลของทั้งภาพ โปรแกรมจะทำการประมวลผลหาขอบภาพเฉพาะบริเวณที่เลือกเท่านั้น เป็นการเพิ่มประสิทธิภาพในเรื่องของเวลาประมวลผลของโปรแกรมที่น้อยลง ซึ่งในการประมวลผลแต่ละครั้งใช้เวลาเพียง 1-2 วินาทีเท่านั้น

Project title	The Realtime Color Image Edge Detection		
Name	Miss Prabhaphan	Chuphinit	ID. 46360038
	Mr. Keeradit	Boonyalapalerd	ID. 46361853
Project advisor	Suradet	Jitprapaikulsarn, Ph.D.	
Major	Computer Engineering		
Department	Electrical and Computer Engineering		
Academic year	2006		

Abstract

This project develops a program to detect edges of color images in real-time. Our program first dissects the images into three single-color image—Red-only image, Green-only image, and Blue-only image; then applies the gray-scale edge detection (Sobel, Robinson, Laplace 1st derivative, and Laplace 2nd derivative) to all single-color images; then uses the color fusion theory to combine three edge images into a single image. User then can perform a post-processing image enhancement to further improve the clarity of the images.

Instead of completely image edge-detection, this software limits the are of image to a square around the mouse pointer which the advantage on software's processing time only 1-2 second.

กิตติกรรมประกาศ

โครงงานวิศวกรรมคอมพิวเตอร์ การหาขอบภาพสีแบบเรียลไทม์นี้ สำเร็จได้ด้วยดี เนื่องจากความอนุเคราะห์จากอาจารย์ที่ปรึกษา อาจารย์สุรเดช จิตประไพกุลศาล ที่กรุณาให้คำปรึกษา แนะนำทั้งในด้านทฤษฎี การหาตัวอย่างและแหล่งข้อมูล และวิธีการในการทำงาน ตลอดถึงตรวจสอบ ชี้ให้เห็นข้อบกพร่องของงาน พร้อมทั้งเสนอแนวทางการแก้ไข สุดท้ายต้องขอขอบพระคุณอาจารย์ที่ปรึกษาท่านอื่นๆ ที่ไม่ได้เอ่ยนาม ที่กรุณาให้คำแนะนำและสนับสนุนในการทำโครงงานในครั้งนี้

สารบัญ

หน้า

หน้า

สารบัญรูป

รูปที่ หน้า

บทที่ 2 หลักการและทฤษฎีเบื้องต้น

บทที่ 3 วิธีการดำเนินการ

รูปที่ หน้า

บทที่ 4 ผลการทดลอง

ภาคผนวก

สารบัญตาราง

ตารางที่ หน้า

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของโครงงาน

ประสาทรับรู้ในด้านการมองเห็นของมนุษย์เป็นสิ่งที่มีมาตั้งแต่กำเนิด และเป็นสิ่งที่มีความสำคัญในการรับรู้และตัดสินสิ่งต่างๆที่ได้พบเห็น เนื่องจากการสื่อด้วยรูปภาพนั้นมีข้อได้เปรียบมากกว่าการสื่อสารด้วยตัวอักษรธรรมดา เพราะรูปภาพสามารถแทนได้ทั้งคำพูด ความรู้สึก และอารมณ์ จึงมีการนำรูปภาพมาใช้ในการสื่อสารแทนตัวอักษร ไม่ว่าจะเป็น สัญลักษณ์ทางจราจร ภาพยนตร์ สื่อสิ่งพิมพ์ต่างๆ งานกราฟิกและอื่นๆอีกมากมาย ทฤษฎีการประมวลผลรูปภาพจึงถูกนำมาใช้กันอย่างแพร่หลาย

ทฤษฎีการประมวลผลรูปภาพระดับดิจิตอล (Digital Image Processing) นั้นเป็นทฤษฎีว่าด้วยเรื่องการประมวลผลข้อมูลรูปภาพที่อยู่ในฟอร์แมทดิจิตอล ซึ่งมีอยู่หลายทฤษฎีด้วยกัน ทฤษฎีหนึ่งที่เรานำมาใช้ก็คือทฤษฎีในกลุ่มของ Image Enhancement ซึ่งเป็นหลักในการปรับปรุงคุณภาพของรูปภาพ เช่น การปรับระดับความคมชัดของรูปภาพ การปรับระดับความสว่างของรูปภาพ การลด noise ในรูปภาพ การเน้นขอบในรูปภาพและการหาขอบในรูปภาพ เป็นต้น

เราสามารถนำเทคนิคของการหาขอบในรูปภาพมาใช้ในงานด้านต่างๆได้อย่างกว้างขวาง เช่น การพิจารณารูปทรงของวัตถุในการก่อสร้าง การวิเคราะห์พื้นผิวต่างๆงานด้านธรณีวิทยา ใช้ด้านการรักษาความปลอดภัย ด้านการทหาร หรือแม้กระทั่งในด้านการแพทย์ก็ตาม

โครงงานนี้ได้นำเทคนิคในการหาขอบในรูปภาพแบบต่างๆ (Edge Detection) มาใช้ในการวิเคราะห์รูปภาพ โดยนำทฤษฎีการหาขอบในรูปภาพ Gray Scale มาใช้หา ซึ่งจะเป็นการทำงานแบบ Real-time

1.2 วัตถุประสงค์ของโครงงาน

1. เพื่อพัฒนา Software ตรวจหาขอบในรูปภาพ (Edge Detection) โดยใช้ algorithm และทฤษฎีทาง digital image processing แบบต่างๆ
2. เพื่อศึกษาการพัฒนาโปรแกรมโดยมีหลักการทำงานแบบ Real-time

1.3 ขอบข่ายของโครงงาน

1. ศึกษาการตรวจหาขอบในรูปภาพและทฤษฎีเบื้องต้นของ Image processing

2. สร้างและพัฒนา Software ให้สามารถตรวจหาขอบในรูปภาพสีชนิด bitmap จากทฤษฎีการตรวจหาขอบในรูปภาพของ Sobel Canny และ Laplaciant

1.4 ขั้นตอนของการดำเนินงาน

1. ศึกษาข้อมูล

1.1 Fundamental of Digital image processing

- Representing of digital image
- Grey scale
- Color representation
- Histogram representation

1.2 Edge detection and Enhancement

- Enhancement using arithmetic/Logic Operations

1.3 Algorithm

- Sobel's image edge detection algorithm
- Robinson's image edge detection algorithm
- Laplaciant's image edge detection algorithm

2. ออกแบบและพัฒนา Software

3. ทดสอบและแก้ไขข้อผิดพลาดของ Software

4. วิเคราะห์ การทดสอบ Software และบันทึกผล

5. ทำปริญญานิพนธ์และเตรียมการนำเสนอ

1.5 แผนการดำเนินงาน

กิจกรรม	ปี 2547		ปี 2548			
	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.
1. ศึกษาความรู้พื้นฐานของเรื่อง image processing,ทฤษฎีของ edge detection,หลักการทำงานและการเขียนโปรแกรมให้ทำงานแบบ real-time	←	—	→			
2. ออกแบบและพัฒนา Software			←	—	→	
3. ทดสอบและแก้ไขข้อผิดพลาดของ software			←	—	→	
4. วิเคราะห์การทดสอบ Software และบันทึกผล				←	—	→
5. ทำปริญญานิพนธ์และเตรียมการนำเสนอ					←	— →

1.6 ผลที่คาดว่าจะได้รับ

1. ได้ Software สำหรับตรวจหาขอบในรูปภาพ (Edge Detection) โดยใช้ algorithm และทฤษฎีทาง digital image processing แบบต่างๆ
2. เข้าใจหลักการทำงานของการตรวจจับขอบในรูปภาพสี (Color image edge detection)
3. เข้าใจหลักการพัฒนาโปรแกรมโดยมีหลักการทำงานแบบ Real-time

1.7 งบประมาณของโครงงาน

1. ค่าวัสดุสำนักงาน	เป็นเงิน	750	บาท
2. ค่าวัสดุคอมพิวเตอร์	เป็นเงิน	750	บาท
3. ค่าจ้างถ่ายเอกสาร	เป็นเงิน	500	บาท
	รวมเป็นเงินทั้งสิ้น	2,000	บาท

(สองพันบาทบาทถ้วน)

บทที่ 2
หลักการและทฤษฎีเบื้องต้น

ในบทนี้จะศึกษาเกี่ยวกับทฤษฎีพื้นฐานเกี่ยวกับรูปภาพในระบบดิจิตอล ซึ่งมีความแตกต่างจากรูปภาพที่เราเห็นโดยทั่วไป เนื้อหาในบทนี้จะกล่าวถึงวิธีการนำเสนอรูปภาพในระบบดิจิตอล ระบบสีที่ใช้เป็นมาตรฐานของรูปภาพในระบบดิจิตอล วิธีการประมวลผลรูปภาพเบื้องต้นและอัลกอริทึมสำหรับหาขอบในรูปภาพ

2.1 ความรู้เบื้องต้นเกี่ยวกับรูปภาพดิจิตอล (Fundamental of Digital Image Processing)

2.1.1 ลักษณะทั่วไปของรูปภาพดิจิตอล (Representing of Digital Image)

คอมพิวเตอร์จะเก็บภาพต่างๆ ในรูปแบบของเมตริกซ์ ซึ่งมีจำนวนแถวเท่ากับ M และจำนวนหลักเท่ากับ N จะทำการเก็บค่าของแต่ละจุดพิกัด f (x, y) ซึ่งจะเริ่มจาก f(0, 0) เป็นจุดเริ่มต้นของรูปภาพและจากนั้นจะเริ่มจากจุด f(0, 1) เป็นการเก็บค่าของหลักแรกของรูปภาพ จะได้ตัวเลขที่อยู่ในรูปแบบของเมตริกซ์ได้ดังนี้

$$f(x,y)=\begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ f(M-1,0) & f(M-1,1) & \dots & f(M-1,N-1) \end{bmatrix} \quad (2.1)$$

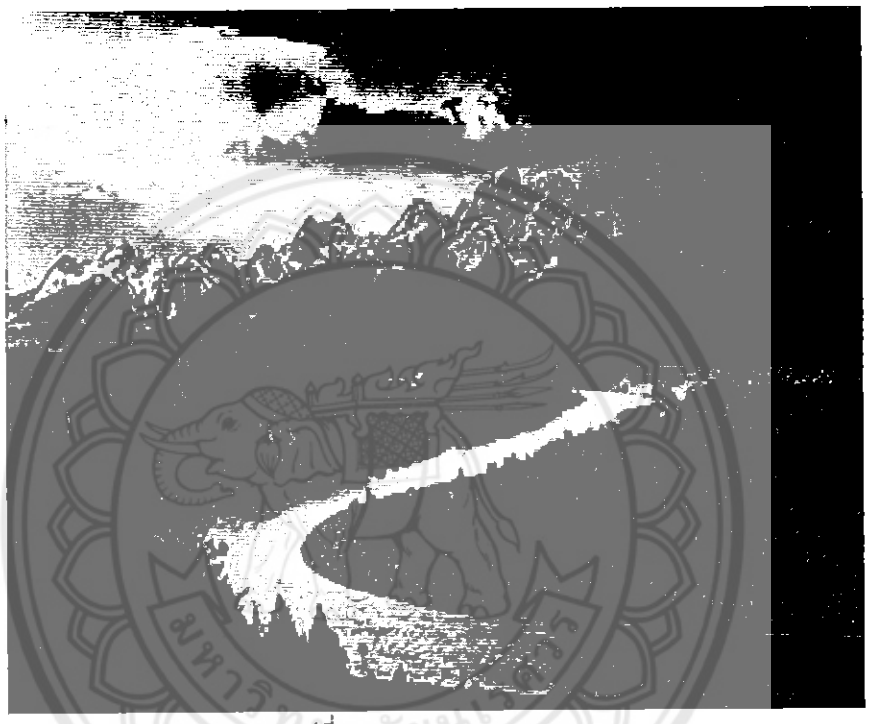
ในแต่ละค่าของเมตริกซ์แทนแต่ละส่วนของภาพหรือที่เรียกว่า พิกเซล (pixel) และถ้าให้ A แทนเมตริกซ์แทนรูปภาพในทางดิจิตอลจะแสดงได้ดังนี้

$$A=\begin{bmatrix} a_{0,0} & a_{0,1} & \dots & a_{0,N-1} \\ a_{1,0} & a_{1,1} & \dots & a_{1,N-1} \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ a_{M-1,0} & a_{M-1,1} & \dots & a_{M-1,N-1} \end{bmatrix} \quad (2.2)$$

โดยในแต่ละค่าของสมาชิกภายในเมตริกซ์ A นั้นแทนค่าระดับสีของรูปภาพ ณ จุดนั้นๆ

1. รูปภาพระดับสีเทา (Gray Scale Digital Image)

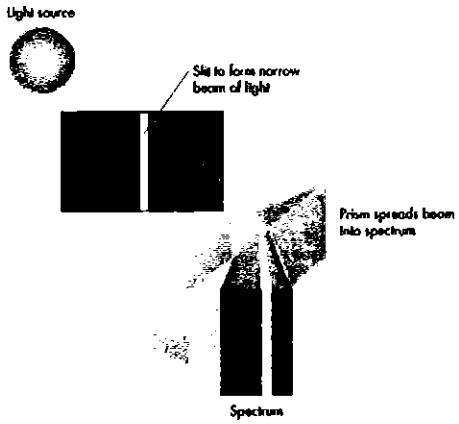
รูปภาพระดับสีเทา หรือ Gray Scale Digital Image คือ รูปภาพที่มีค่าสีเพียงค่าเดียว ซึ่งแทนความสว่างของภาพ ความละเอียดของภาพจะขึ้นอยู่กับจำนวนบิตที่ใช้ในการเก็บต่อ 1 พิกเซล (pixel)



รูปที่ 2.1 ภาพแบบ Gray Scale

2. รูปภาพสี (Color Representation)

สีของแสงที่เกิดขึ้นที่มนุษย์สามารถมองเห็น และแบ่งแยกได้ชัดเจนนั้น คือ สีรุ้ง 7 สี ซึ่งช่วงความยาวคลื่นของแต่ละสีกระบวนการในการมองเห็นสีของสิ่งของต่างๆ ของมนุษย์เกิดจากการที่แสงจากแหล่งกำเนิด (ในที่นี้ขอยกตัวอย่างพระอาทิตย์) ซึ่งเราเห็นว่าเป็นแสงที่ไม่มีสี หรือแสงสีขาว เดินทางมากระทบกับวัตถุ เข้าสู่ระบบประสาทการมองเห็นของมนุษย์ (ตา และสมอง) โดยที่การตกกระทบที่วัตถุของแสงนั้น ที่ผิววัตถุจะมีการดูดกลืนช่วงความถี่ของแสงบางส่วนไว้ และสะท้อนออกมาเฉพาะสีที่ตรงกับผิววัตถุ ดังนั้นเราจึงสามารถมองเห็นวัตถุต่างๆ มีสีได้ ซึ่งขั้นตอนการเดินทางของแสงมาสู่ตาของมนุษย์ได้แสดงให้เห็นดังภาพ



รูปที่ 2.2 แสดงการหักเหของแสงขาวเป็นแสงสี

ในการทำงานเกี่ยวกับภาพ มีการใช้ระบบสีที่แตกต่างกันที่เป็นมาตรฐานในการทำงานแต่ละประเภท เพื่อให้การแสดงผลของรูปภาพออกมาใกล้เคียงกับสีที่เรามองเห็นจากธรรมชาติมากที่สุด

2.2 ระบบสี (Color Model)

สำหรับรูปภาพแบบ Gray-scale นั้น ถ้าให้ L แทนค่าระดับของสีเทาของในแต่ละพิกเซลรูปขนาด k บิต ดังนั้น รูปภาพจะมีค่าระดับสีเทาอยู่ในช่วง [0, L-1] จะเขียนเป็นสมการแสดงจำนวนของระดับสีเทา ดังนี้

$$L = 2^k \tag{2.3}$$

และให้ b แทนจำนวนบิตที่ใช้เก็บภาพดิจิตอล

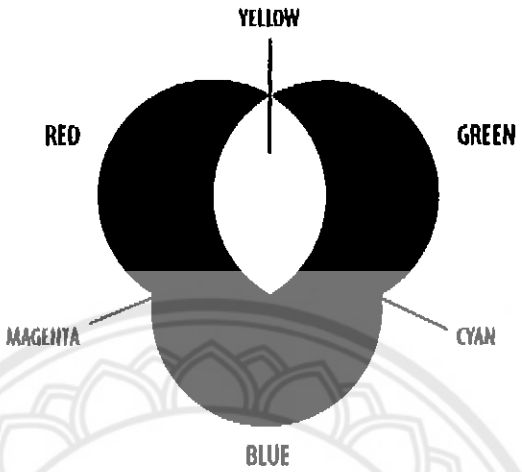
$$b = M \times N \times k \tag{2.4}$$

เช่น ภาพ Grayscale ขนาด 8 บิตนั้นจะมีระดับสีเทาทั้งหมด $2^8 = 256$ ระดับที่อยู่ในช่วง [0-255] โดยที่ค่าของ 0 จะใช้แทนสีดำไปจนถึง 255 ซึ่งจะแทนค่าของสีขาว

2.2.1 RGB (Red, Green, Blue)

เป็นระบบสีของแสงที่เกิดจากการหักเหของแสงผ่านแท่งแก้วปริซึม ซึ่งจะเกิดแถบสีที่เรียกว่าสีรุ้งทั้ง 7 สี ได้แก่ แดง แสด เหลือง เขียว น้ำเงิน คราม ม่วง เป็นพลังงานที่อยู่ในรูปของรังสีที่มีความยาวคลื่นต่างๆ กัน ซึ่งเกิดจากแสงสี 3 สี คือ แดง เขียว และน้ำเงิน เมื่อนำมาฉายรวมกันจะ

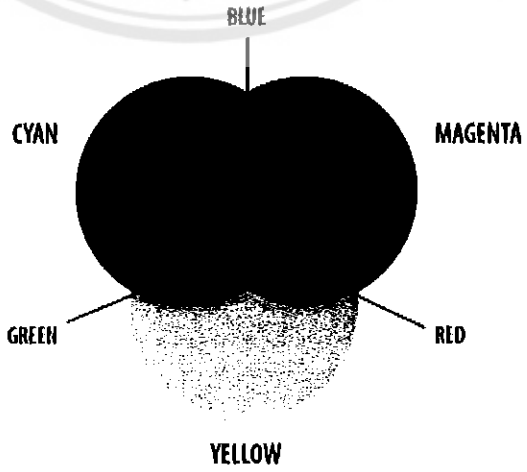
ได้เป็นสีใหม่อีกสามสีคือ Magenta Cyan Yellow นำมาใช้ประโยชน์อยู่ทั่วไป เช่น การถ่ายภาพยนตร์ การบันทึกภาพวิดีโอ ภาพโทรทัศน์ การจัดแสงสีในการแสดง การสร้างภาพเพื่อการนำเสนอทางจอคอมพิวเตอร์



รูปที่ 2.3 RGB Color Model

โดยที่สีแดง สีเขียวและสีน้ำเงินจะอยู่ที่มุมต่างๆ ของลูกบาศก์และ ที่จุด (0, 0,0) เป็นสีดำ ส่วนจุดสีขาวนั้นอยู่อีกมุมหนึ่ง (1, 1, 1) โดยค่าสีตามแกนดังกล่าวจะเป็นระดับของสีเทาหรือ gray scale จะแทนด้วยเวคเตอร์ที่มีทิศทางจากค่าสีดำไปยังสีขาว สีต่างๆเหล่านี้จะผสมกันโดยมีช่วงภายใน [0, 1]

ค่าตัวเลขที่ใช้ความเข้มของระดับสีในแต่ละพิกเซลของRGB เรียกว่า Pixel depth จะเป็นค่า (R, G, B) เช่น ในระบบภาพ 24 บิต 000000 จะแทนสีดำ และ FFFFFF แทนค่าสีขาว



รูปที่ 2.4 CMY Color Model

2.2.2 CMY (Cyan, Magenta, Yellow)

เป็นแสงสีที่เป็นวัตถุที่เกิดจากการผสมแสงสีของ RGB จะนำมาใช้ในระบบการพิมพ์ และมีการเพิ่มเติมสีดำเข้าไปจึงเป็นสี่สี เป็นระบบพิมพ์ภาพที่ทันสมัยที่สุด ได้ภาพใกล้เคียงกับภาพถ่ายมากที่สุดเกิดจากการผสมของเม็ดสีในปริมาณต่างๆ

$$\begin{aligned} C &= 1 - R \\ M &= 1 - G \\ K &= 1 - B \end{aligned} \tag{2.5}$$

2.2.3 CMYK

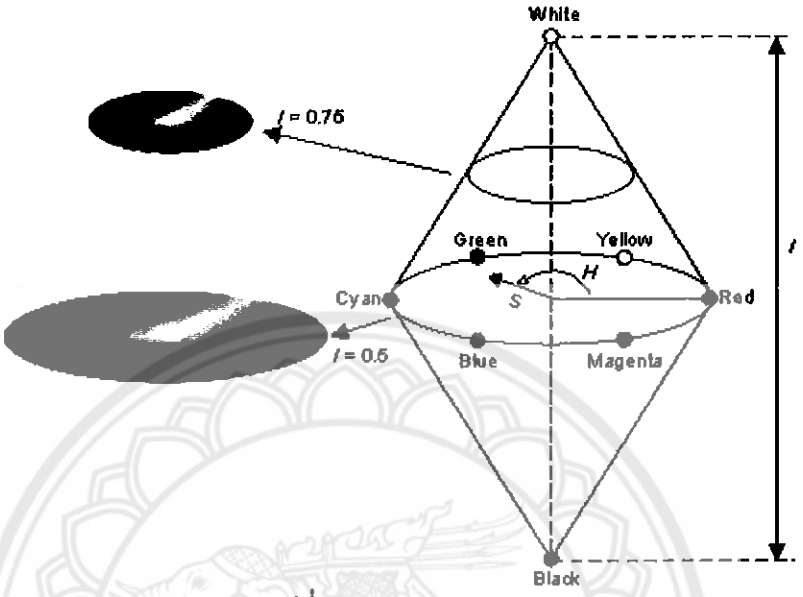
CMYK เป็นระบบ Subtractive color ที่ใช้ในการพิมพ์โดยที่สี CMY ก็คือ Cyan Magenta และ Yellow ที่เป็นแม่สีในกระบวนการพิมพ์ Color Gamut ของหมึกพิมพ์นั้นจะน้อยกว่าจอภาพและเหมือนล้ำกันเล็กน้อย ดังนั้น สีบางสีที่สามารถแสดงได้บนจอภาพอาจไม่สามารถแสดงโดยใช้หมึกพิมพ์ได้ และมีบางส่วนที่สามารถแสดงด้วยหมึกพิมพ์แต่ไม่สามารถแสดงบนจอภาพได้เช่นกัน

ค่าของ CMY จะมีค่าเป็นเปอร์เซ็นต์ตั้งแต่ 0%-100% ในแต่ละสีโดยในการพิมพ์จะทำการพิมพ์ลงบนกระดาษสีขาว ต่อมาได้มีการเพิ่มสีดำเข้าไปในกลายเป็น CMYK ขึ้นโดยสีดำในระบบ CMY นั้นสามารถหาได้จากการนำค่าต่ำสุดในทั้ง 3 Channel และ CMY ค่าใหม่ (ระบบ CMYK) จะเท่ากับค่า CMY ค่าเดิมลบกับค่าของสีดำ ดังสมการ

$$\begin{aligned} K &= \min(C, M, Y) \\ C_{CMYK} &= C - K \\ M_{CMYK} &= M - K \\ K_{CMYK} &= Y - K \end{aligned} \tag{2.6}$$

2.2.4 HSI (Hue, Saturation, Intensity)

เป็นระบบที่สร้างมาจากสัญชาตญาณการนึกคิดของมนุษย์ในการกำหนดค่าสี โดยที่ค่า Hue (สี) จะเป็นค่าตัวแทนสีตั้งแต่ 0 (แดง) ถึง 120 (เขียว) ถึง 240 (น้ำเงิน) และถึง 360 (แดง) มีลักษณะการแทนด้วยวงล้อ ค่า Saturation (ค่าความเข้มของสี) ในความเข้มระดับต่ำตั้งแต่ 0-20% จะให้ผลลัพธ์สีออกมาเป็นเทา, ความเข้มระดับกลาง 40-60% ผลลัพธ์จะเป็นสีอ่อน และ ความเข้มสูง 80-100% ผลลัพธ์จะได้สีที่จัด หรือแจ่มใส ค่าสุดท้ายที่ใช้คือ Intensity (ค่าความสว่างของสี) มีช่วงตั้งแต่ 0% (มืด หรือดำ) จนถึง 100% (สว่าง หรือขาว)

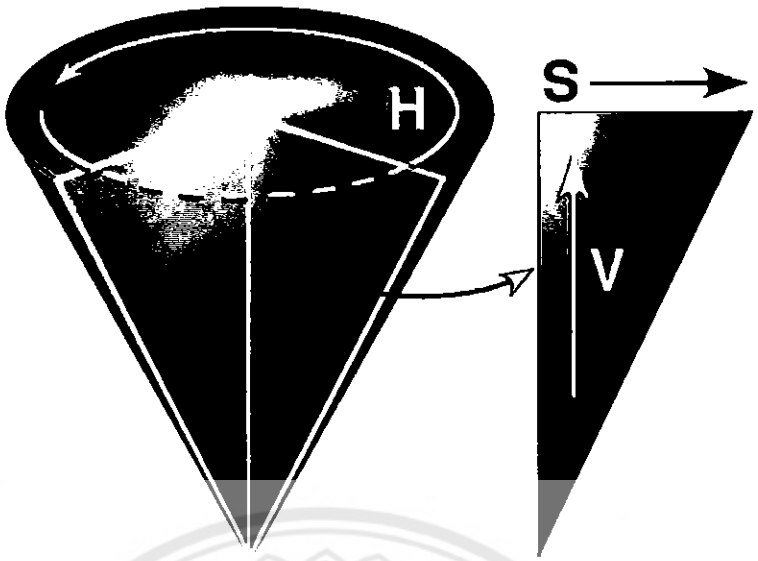


รูปที่ 2.5 HSI Color Model

2.2.5 HSV (Hue, Saturation, Value)

HSV หรือ HSB (Hue, Saturation, Brightness) ประกอบด้วย

- Hue หรือ ค่าสี มีค่าอยู่ระหว่าง 0 ถึง 360 (หรือ 0% ถึง 100% ในบางโปรแกรม)
- Saturation หรือ ค่าความอิ่มสี มีค่าตั้งแต่ 0% ถึง 100% หากลดค่า Saturation ลงภาพจะค่อยๆกลายเป็นสีเทา ในทางกลับกันหากค่อยๆเพิ่มค่า Saturation สีจะปรากฏชัดขึ้น ค่า Saturation บางครั้งอาจเรียกว่า ค่าความบริสุทธิ์ของสี
- Value หรือ ค่าความสว่างของสี มีค่าตั้งแต่ 0% ถึง 100%

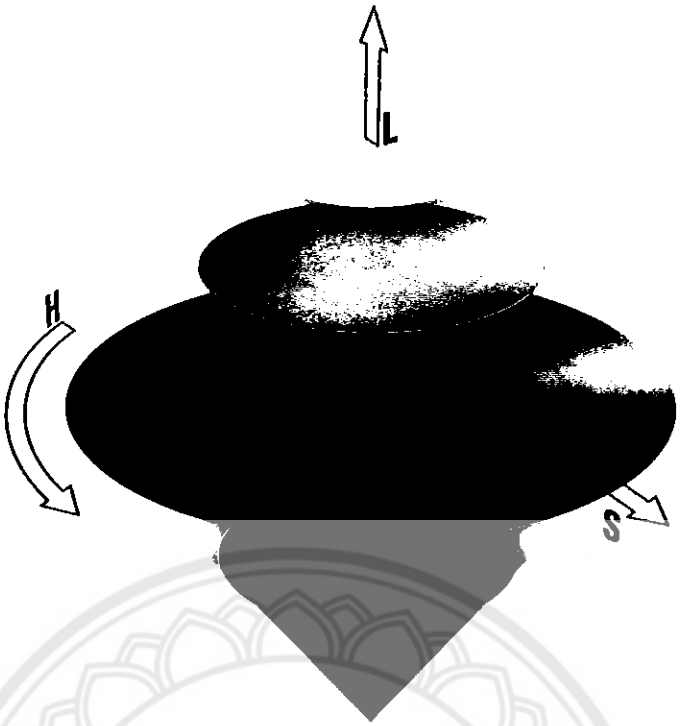


รูปที่ 2.6 HSV color space as a conical object

2.2.6 HSL (Hue, Saturation, Lightness)

ระบบสีแบบ HSL นี้อาจเรียกว่า HLS หรือ HIS สำหรับค่าสี ความอิ่มสีและความสว่างของสีหรือความเข้มของสี

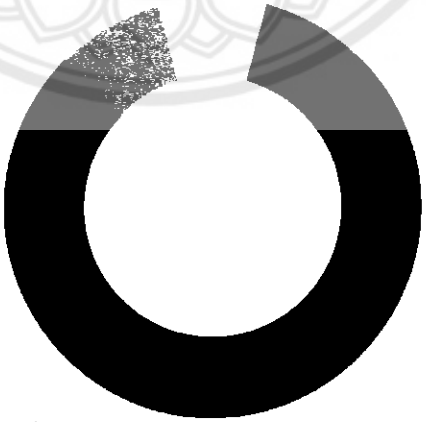
ระบบสี HSV (ค่าสี ค่าความอิ่มสีและค่าความสว่างของสี) นั้นมีปริภูมิของสี (Color Space) เป็นแบบกรวยกลมในขณะที่ระบบสี HSL จะมีเป็น 2 กรวยประกบกัน อย่างไรก็ตามทั้งสองระบบเป็นรูปแบบ RGB แบบไม่เชิงเส้น (Non-linear) โดยจุดยอดของกรวยทั้งสองด้านจะแทนสีดำไล่ไปจนถึงสีขาว ค่ามุมในแนวระนาบ (วัดจาก 0 องศาคือสีแดง) แทนค่าสี (hue) ความสูงของกรวยคือค่าความสว่าง (Lightness) และระยะห่างจากแกนแนวดิ่งจะแทนค่าความอิ่มสี (Lightness)



รูปที่ 2.7 HSV color space as a conical object

2.2.7 RYB (Red Yellow Blue)

RYB หรือ Red Yellow Blue คือ ระบบของสีที่ใช้ในงานศิลปะ ซึ่งเป็นระบบสีแบบลบ (subtractive primary colors) สีทั้งสามเรียกว่า "แม่สี" หรือ "สีอันดับที่หนึ่ง (Primary Color)" ซึ่งเมื่อนำแม่สีมาผสมกันเป็นคู่ๆ จะได้สีใหม่สามสี เรียกว่า "สีดันดับที่สอง (Secondary Color)" และเมื่อนำสีเหล่านี้มาผสมกันอีกก็จะได้ "สีในอันดับที่สาม" อีก 6 สี ดังรูป

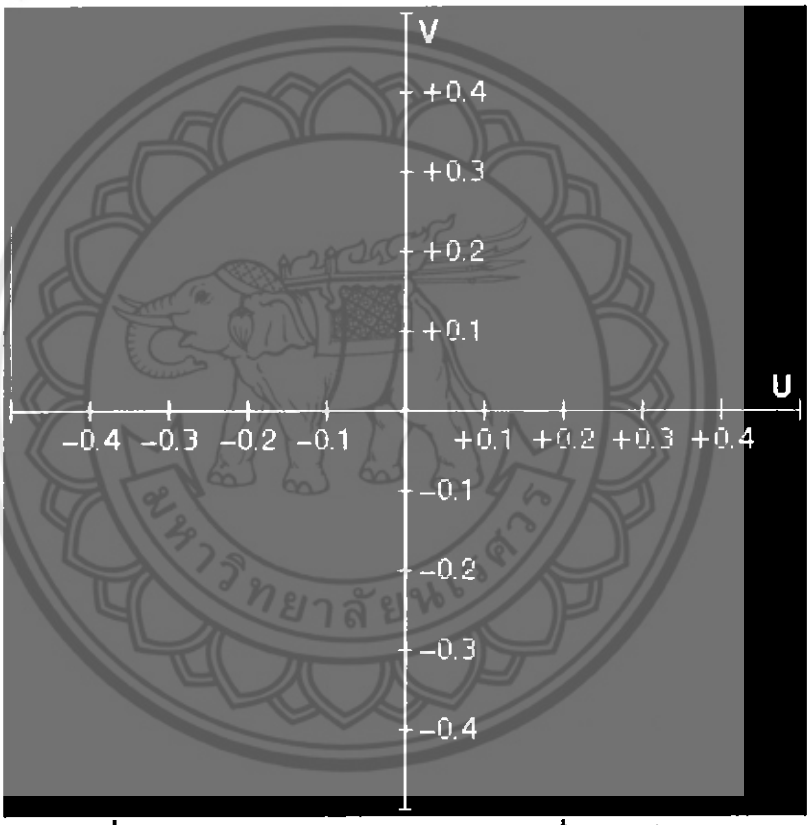


รูปที่ 2.8 HSV color space as a conical object

2.2.8 YUV (luminance information, information of blue component, information of red component)

YUV ใช้เป็นมาตรฐานในการแพร่ภาพทางโทรทัศน์ทั้งระบบ PAL (phase-alternating line, phase alternation by line or phase alternation line)

- Y คือ ค่าความสว่าง (Luminance)
- U หรือ Cb คือ ค่าแชนแนลของสีน้ำเงิน (Blue component)
- V หรือ Cr คือ ค่าแชนแนลของสีแดง (Red component)



รูปที่ 2.9 ตัวอย่างระนาบ U-V (U-V color plane) เมื่อค่า Y เท่ากับ 0.5

2.2.9 YIQ (luminance information, in-phase, quadrature)

ใช้สำหรับโทรทัศน์ระบบ NTSC (National Television System(s) Committee) ซึ่งเป็นระบบโทรทัศน์ที่ใช้ในหลายประเทศ เช่น เกาหลี ญี่ปุ่น สหรัฐอเมริกา แคนาดา และกลุ่มประเทศในทวีปอเมริกา

2.3 Color Conversion Algorithms

2.3.1 การแปลงภาพสี RGB เป็น Gray Scale

$$g = (0.299*R) + (0.587*G) + (0.114*B);\ (0 \leq g \leq 1) \tag{2.7}$$

2.3.2 การแปลงภาพสี RGB เป็น CMY

$$\begin{aligned} C &= 1 - R \\ M &= 1 - G \\ Y &= 1 - B \end{aligned} \tag{2.8}$$

2.3.3 การแปลงภาพสี CMY เป็น CMYK

$$\begin{aligned} K &= \min(C, M, Y) \\ C_{CMYK} &= C - K \\ M_{CMYK} &= M - K \\ Y_{CMYK} &= Y - K \end{aligned} \tag{2.9}$$

2.3.4 การแปลงภาพสี RGB เป็น HSI

กำหนดค่า MAX และค่า MIN คือ ค่าสูงสุดและค่าต่ำสุดของค่าสีแดง (Red) สีเขียว (Green) หรือสีน้ำเงิน (Blue) ตามลำดับ จะสามารถหาค่า H ค่า S และ ค่า I ได้จากสมการ

$$H = \begin{cases} 40 * \dfrac{G-B}{MAX-MIN} & if\ R = MAX \\ 80 + 40 * \dfrac{B-R}{MAX-MIN} & if\ G = MAX \\ 160 + 40 * \dfrac{R-G}{MAX-MIN} & if\ B = MAX \end{cases} \quad (0 \leq H \leq 239) \tag{2.10}$$

$$S = 1 - \frac{[\min(R, G, B)]}{I} \quad (0 \leq S \leq 1) \tag{2.11}$$

$$I = \frac{(R+G+B)}{3} \quad (0 \leq I \leq 1) \tag{2.12}$$

2.3.5 การแปลงภาพสี RGB เป็น HSL

กำหนดค่า MAX และค่า MIN คือ ค่าสูงสุดและค่าต่ำสุดของค่าสีแดง (Red) สีเขียว (Green) หรือสีน้ำเงิน (Blue) ตามลำดับ จะสามารถหาค่า H ค่า S และ ค่า L ได้จากสมการ

$$H=\begin{cases} 60*\dfrac{G-B}{MAX-MIN} & if\ MAX=R\ \ and\ G\geq B \\ 360+60*\dfrac{G-B}{MAX-MIN} & if\ MAX=R\ \ and\ G<B \\ 120+60*\dfrac{B-R}{MAX-MIN} & if\ MAX=G \\ 240+60*\dfrac{R-G}{MAX-MIN} & if\ MAX=B \end{cases} \tag{2.13}$$

$$S=\begin{cases} \dfrac{MAX-MIN}{MAX+MIN}=\dfrac{MAX-MIN}{2L} & if\ L\leq\dfrac{1}{2} \\ \dfrac{MAX-MIN}{2-(MAX+MIN)}=\dfrac{MAX-MIN}{2-2L} & if\ L\geq\dfrac{1}{2} \end{cases} \tag{2.14}$$

$$L=\frac{1}{2}(MAX+MIN) \tag{2.15}$$

2.3.6 การแปลงภาพสี RGB เป็น YIQ

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix}=\begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix}\begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.16}$$

สำหรับการแปลงจาก YIQ เป็น RGB

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix}=\begin{bmatrix} 1 & 0.956 & 0.621 \\ 1 & -0.272 & -0.647 \\ 1 & -1.105 & 1.702 \end{bmatrix}\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} \tag{2.17}$$

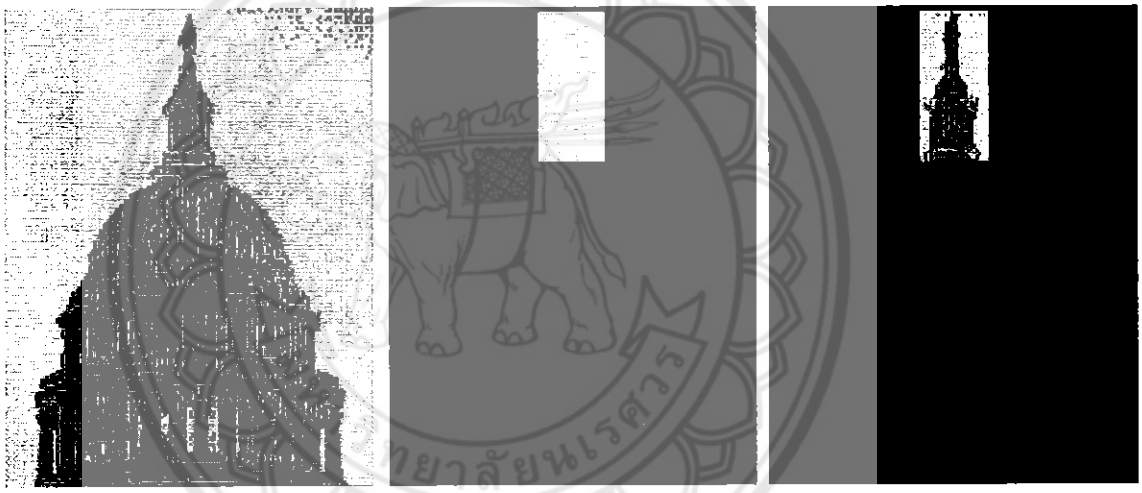
2.4 การปรับปรุงคุณภาพและคำสั่งที่ใช้กับรูปภาพดิจิตอล (Image Enhancement using Arithmetic/Logic Operations)

Arithmetic/Logic Operations เป็นพื้นฐานในการสร้างรูปภาพขึ้นมาใหม่โดยใช้การคำนวณทางคณิตศาสตร์หรือทางตรรกะระหว่างรูปภาพสองรูปขึ้นไป

2.4.1 คำสั่งทางตรรกศาสตร์ (Logic Operations)

จะใช้กับรูปภาพตั้งแต่สองภาพขึ้นไป โดยใช้กระบวนการทางตรรกะ เช่น AND, OR, และ COMPLEMENT (COMPLEMENT Operation ที่จะทำอยู่ในรูปเดียวกัน)

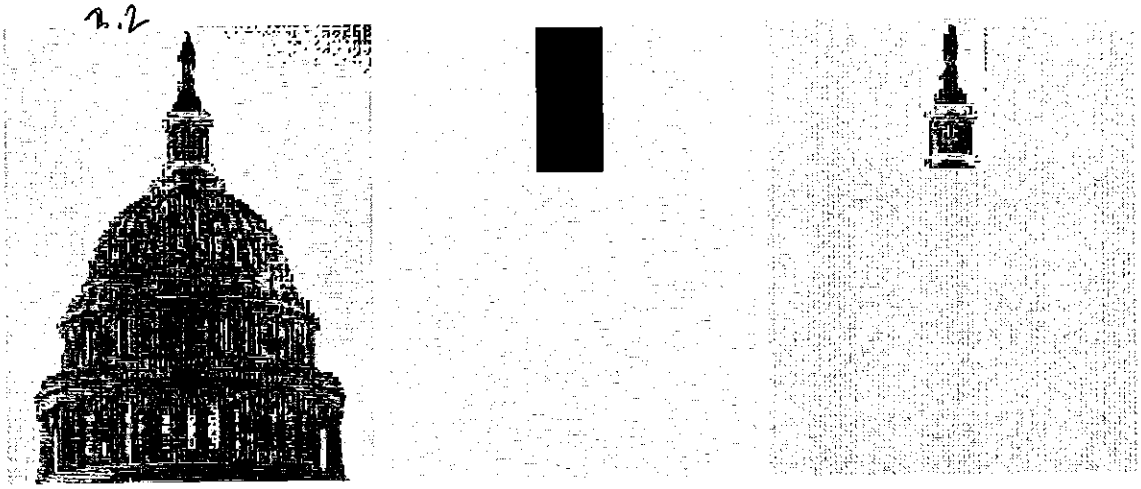
- **AND Operation** จะได้ภาพใหม่ที่เกิดจากการนำจุดแต่ละจุดในภาพต้นฉบับแต่ละภาพมา AND กัน



Original Image AND Image Mask Result of AND Operation

รูปที่ 2.10 And Operation

- **OR Operation** จะได้ภาพใหม่ที่เกิดจากการนำจุดแต่ละจุดในภาพต้นฉบับแต่ละภาพมา OR กัน



Original Image OR Image Result of OR Operation

Mask

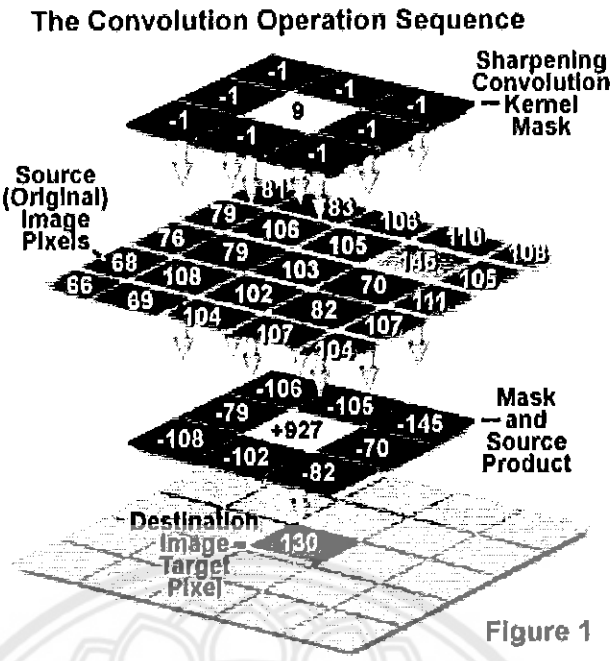
รูปที่ 2.11 Or Operation

2.5 Masking Operation

เทคนิคในการหาขอบภาพของโครงงานนี้ คือวิธี Mask Operation ซึ่งเป็นการคำนวณโดยผลลัพธ์ของแต่ละ pixel มาจากการคำนวณค่าจากรอบจุดดังกล่าว กระบวนการนี้เรียกว่า Convolution ในการคำนวณหาขอบภาพจะใช้เทคนิค Convolution แบบสองมิติซึ่งเรียกว่า Mask operation หรือ Convolution kernel

Mask operation หรือ Convolution kernel จะเป็นการคำนวณค่าจากค่า grey scale จากพิกัดรอบ input pixel ซึ่ง Mask เป็นเมตริกซ์จตุรัสที่มีจำนวนแถวและคอลัมน์เป็นเลขคี่ เช่น 3x3 ,5x5 เป็นต้น โดยมีลำดับการทำงานดังนี้

1. Mask จะวางบน Input Pixel โดยจะนำ mask มาวางทับโดยตำแหน่งกลางของ matrix ดังกล่าวจะทับบน input pixel โดยค่าตรงกลางดังกล่าวจะเรียกว่า target pixel
2. นำค่าของ mask ดังกล่าวมาคูณกับค่าความเข้มของ pixel ที่ตำแหน่งตรงกัน
3. นำค่าที่ได้จากข้อ 2 มาบวกกันเป็นค่าของ target pixel



รูปที่ 2.12 แสดงภาพแสดงการคำนวณของการใช้ Mask Operation

การ Convolution นี้จะทำงานแบบ Pixel-by-Pixel และนำค่าที่ได้มาแทนในแต่ละ target pixel จนกระทั่งครบทุก Pixel บนภาพต้นฉบับ จะได้ผลลัพธ์ออกมาเป็นขอบภาพ โดยแต่ละอัลกอริทึมจะมีค่า Convolution Mask ไม่เท่ากัน จึงทำให้ขอบภาพที่ออกมาแตกต่างกันออกไปตามค่าที่กำหนดไว้

2.6 ความรู้เบื้องต้นเกี่ยวกับการตรวจจับขอบภาพ (Fundamental of Gray Scale Edge Detection Algorithms)

เราสามารถทราบได้ว่าส่วนใดเป็นขอบในรูปภาพโดยดูจากอัตราการเปลี่ยนแปลงของระดับสีในรูปภาพซึ่งโดยปกติจะเป็นส่วนที่ตัดสองพื้นที่ออกจากกัน และการตรวจหาขอบ (Edge Detection) ก็คือรูปแบบการนำรูปภาพๆหนึ่งมาทำให้เห็นขอบ

อัลกอริทึมที่ใช้ในการหาขอบในรูปภาพนั้นมีอยู่หลายชนิดด้วยกัน ซึ่งแต่ละชนิดจะมีข้อดีและข้อเสียที่แตกต่างกัน ในหัวข้อนี้จะกล่าวถึงในส่วนของการตรวจจับขอบภาพ Gray Scale ดังนี้

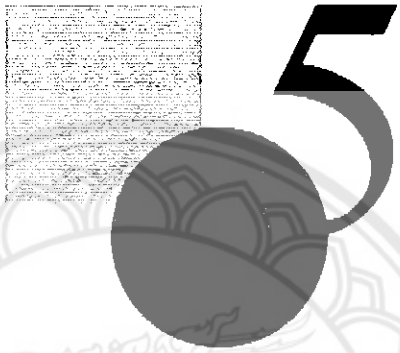
2.6.1 Sobel's Image Edge Detection Algorithm [1]

เป็นเทคนิคการหาขอบภาพที่ค่อนข้างจะง่ายแต่สามารถทำความเข้าใจได้ง่าย โดยใช้สมการ

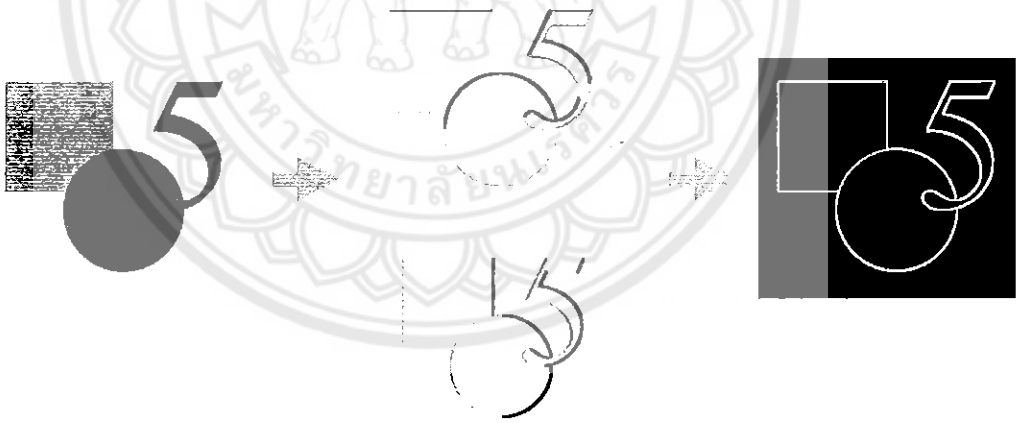
$$N(x,y) = \sum_{k=-1}^{1}\sum_{j=-1}^{1} K(j,k)\,p(x-j,y-k) \qquad (2.18)$$

สมการดังกล่าวสามารถแทนด้วย Mask 2 แบบคือ hx และ hy โดยที่ hx เป็นการหาขอบในแนวนอนและ hy เป็นการหาขอบในแนวตั้ง

$$H(x) = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad H(y) = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{2.19}$$

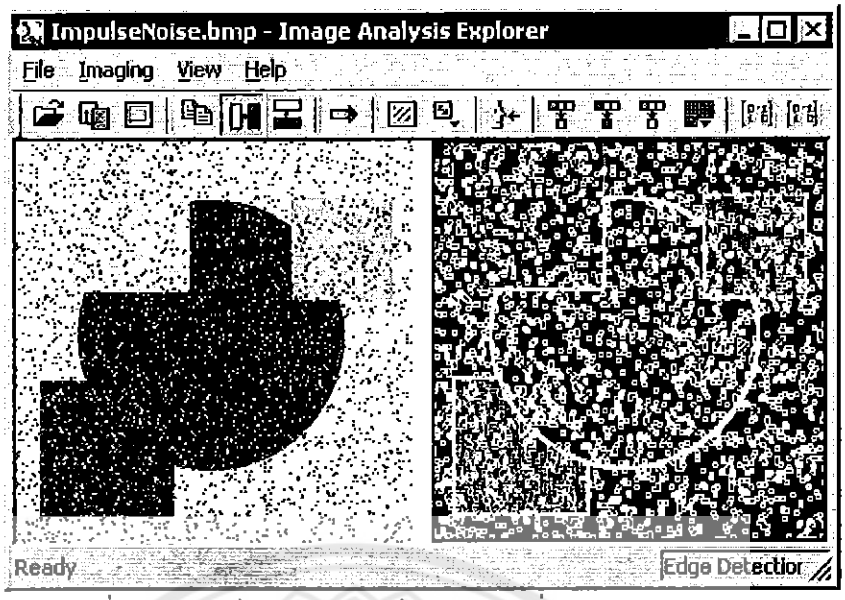


รูปที่ 2.13 รูปภาพต้นฉบับ



รูปที่ 2.14 รูปภาพหลังจากที่ใช้อัลกอริทึมของ Sobel

- ข้อดี คือทำความเข้าใจง่าย
- ข้อเสีย คือ หากรูปภาพต้นฉบับมีสัญญาณรบกวน (Noise) มากรูปที่ได้ก็จะมี Noise มากตามไปด้วย ดังรูป



รูปที่ 2.15 แสดงข้อเสียในอัลกอริทึมของ Sobel ที่เกิดจากสัญญาณรบกวน

2.6.2 Robinson's Image Edge Detection Algorithm [5]

เป็นทฤษฎีที่มีพื้นฐานมาจากเทคนิคการหาขอบภาพของ Sobel กล่าวคือ ไม่มีความซับซ้อน แต่ทำงานได้อย่างมีประสิทธิภาพ

$$H(x) = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad H(d) = \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 0 \\ -1 & -1 & 0 \end{bmatrix} \tag{2.20}$$

2.6.2 Laplacian first derivative's Image Edge Detection Algorithm [4]

เป็นการหาขอบภาพจากการหาความต่างของระดับของค่าสีที่เปลี่ยนแปลง

$$\nabla I = \frac{\partial I}{\partial x} + \frac{\partial I}{\partial y} \tag{2.21}$$

Mask ของวิธี Laplacian's edge detection คือ

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \tag{2.22}$$

2.6.3 Laplacian second derivative's Image Edge Detection Algorithm [4]

ใช้หลักการของ Second Order Derivative Operators ดังสมการ

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \tag{2.23}$$

Mask ของวิธี Laplacian second derivative's edge detection คือ

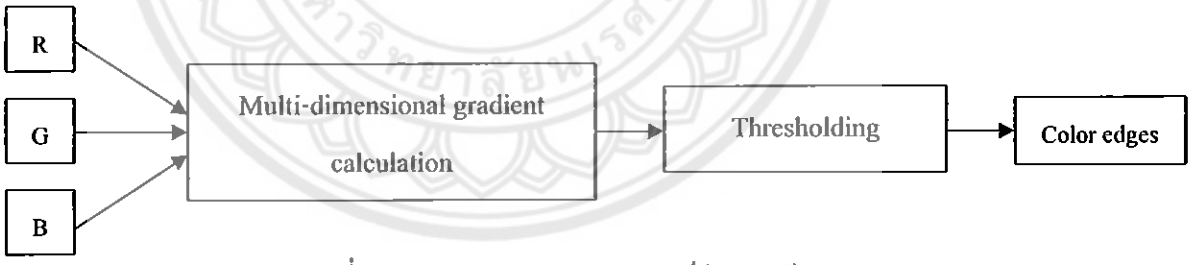
$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{2.24}$$

2.6 การตรวจจับขอบรูปภาพสี (Color Image Edge Detection Algorithms)

วิธีการหาขอบภาพสีสามารถแบ่งตามลักษณะได้ 2 กลุ่มใหญ่ๆ ดังนี้

2.6.1 การหาขอบภาพแบบรวมองค์ประกอบสี (Multi-dimensional gradient methods)

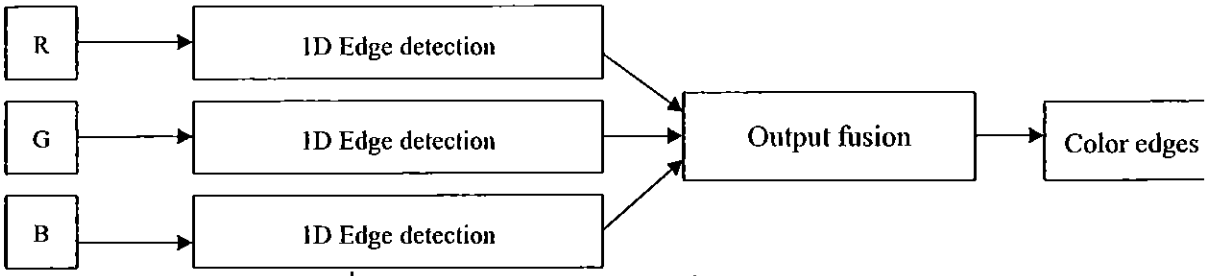
การหาขอบภาพแบบรวมองค์ประกอบสี (Multi-dimensional gradient methods) คือ การนำทุกสีที่เป็นองค์ประกอบของภาพนั้นๆ มาผ่านกระบวนการหาขอบภาพในครั้งเดียว



รูปที่ 2.16 การหาขอบภาพแบบแยกองค์ประกอบสี

2.6.1 การหาขอบภาพแบบแยกองค์ประกอบสี (Output fusion methods)

การหาขอบภาพแบบแยกองค์ประกอบสี (Output fusion methods) คือ การนำแต่ละสีที่เป็นองค์ประกอบของภาพมาแยกหาขอบโดยใช้วิธีพื้นฐานแล้วรวมกันในภายหลัง



รูปที่ 2.17 การหาขอบภาพแบบแยกองค์ประกอบสี

2.7 การปรับปรุงคุณภาพ (Image Enhancement)

ในการปรับปรุงคุณภาพของผลลัพธ์เพื่อให้แสดงขอบภาพได้ชัดเจนขึ้นจะนำเทคนิคการ Threshoding ซึ่งเป็นวิธีการแยกองค์ประกอบของรูปภาพโดยดูจากความเหมือนกันของคุณสมบัติของพิกเซลภายในพื้นที่ มาใช้ซึ่งมีหลักการคิดเบื้องต้นดังนี้

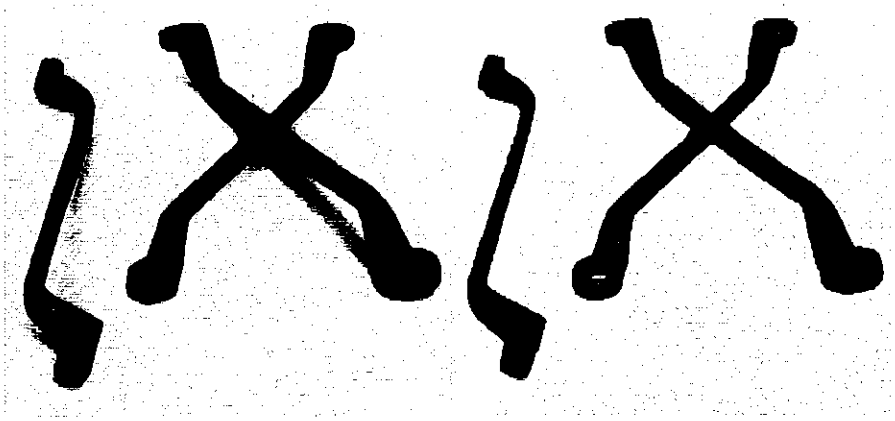
กำหนด θ เป็น brightness threshold ที่นำมาใช้กับภาพ a[m,n]

$$\begin{aligned} &If \quad a[m,n] \geq \theta \qquad a[m,n] = object = 1 \\ &\quad Else \qquad\quad a[m,n] = background = 0 \end{aligned} \tag{2.25}$$

หรือในกรณีที่ วัตถุสีเข้มและพื้นหลังสีอ่อน จะใช้เงื่อนไขดังนี้

$$\begin{aligned} &If \quad a[m,n] < \theta \qquad a[m,n] = object = 1 \\ &\quad Else \qquad\quad a[m,n] = background = 0 \end{aligned} \tag{2.26}$$

ผลลัพธ์ที่ได้ออกมาจะแบ่งเป็นสองส่วนคือ วัตถุ และ พื้นหลัง จากวิธีการข้างต้น จะใช้สัญลักษณ์ทาง Boolean คือ 1 และ 0 แทนผลลัพธ์จากการแยกวัตถุออกจากพื้นหลังดังกล่าว



รูปที่ 2.21 ตัวอย่างการใช้ Threshold

บทที่ 3
วิธีการดำเนินการ

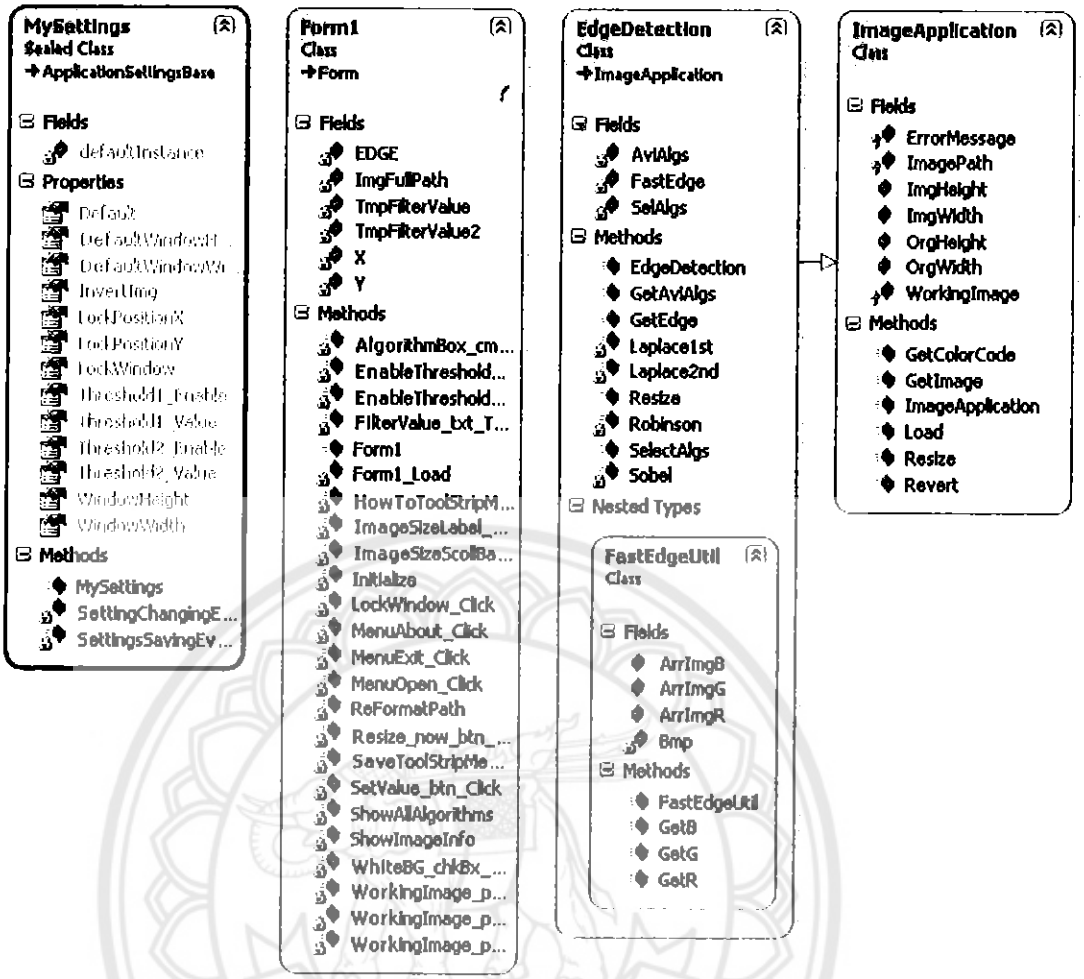
จากการศึกษาข้อมูลในบทที่ 2 องค์ประกอบของภาพสีนั้นมีทฤษฎีและกระบวนการคำนวณที่ยุ่งยากและซับซ้อนไม่เหมือนกับรูปภาพแบบขาว-ดำทั่วไป ในการประมวลผลจึงต้องสามารถเลือกใช้ Algorithm สำหรับการหาขอบภาพ เพื่อเปรียบเทียบหา Algorithm ที่เหมาะสมที่จะนำมาใช้ เพื่อให้ได้ผลลัพธ์ตรงกับความตามต้องการมากที่สุด

3.1 การเลือกเครื่องมือในการพัฒนาโปรแกรม

โครงงานนี้เลือกใช้ Microsoft Visual Studio 2005 เนื่องจาก Microsoft Visual Studio 2005 มีเครื่องมือในการพัฒนาอยู่หลายตัวด้วยกัน หนึ่งในนั้นคือ Microsoft Visual C# 2005 ซึ่งมีเครื่องมือสำหรับจัดการรูปภาพและรองรับการเขียนโปรแกรมแบบเชิงวัตถุ (OOP) ทำให้มีความสะดวกในการพัฒนา และทำให้โปรแกรมใช้เวลาในการทำงานน้อยลง

3.2 โครงสร้างภายในของโปรแกรม

โปรแกรมหาขอบภาพแบบเรียลไทม์นี้มีองค์ประกอบหลัก 3 ส่วนดังนี้



รูปที่ 3.1 แสดง Class Diagram ของโปรแกรม

- **Form1.cs** : ทำหน้าที่เป็นส่วนควบคุมการแสดงผลและติดต่อกับผู้ใช้
- **ImageApplication.cs** : ฟังก์ชันการจัดการรูปภาพ
- **EdgeDetection.cs** : ฟังก์ชันการทำงานในส่วนของการหาขอบภาพสี**โดยใช้อัลกอริธึม**

3.3 การพัฒนาโปรแกรม

เมื่อทำการออกแบบโครงสร้างการทำงานและฟังก์ชันต่างๆ ของโปรแกรมเรียบร้อยแล้ว กระบวนการพัฒนาโปรแกรมโดยใช้เครื่องมือที่ได้เลือกไว้แล้วคือ Microsoft Visual Studio C# 2005 เป็นการพัฒนาโปรแกรมโดยอาศัยหลักการ OOP (Object-Oriented Programming)

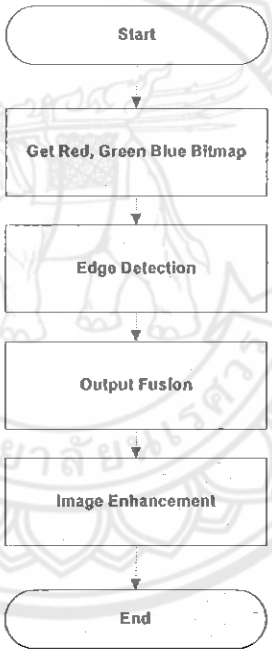
3.4 การออกแบบและหลักการทำงานของโปรแกรมหาขอบภาพ

โปรแกรมจะประมวลผลโดยการนำเข้ารูปภาพจากไฟล์รูปภาพ แสดงผล เพื่อให้ผู้ใช้กำหนดบริเวณที่ต้องการหาขอบภาพ จากนั้นโปรแกรมจะนำรูปภาพที่อยู่ในขอบเขตที่กำหนด ไปประมวลผล และแสดงผลลัพธ์ยังบริเวณที่จัดเตรียมไว้

ส่วนประกอบของโปรแกรมหาขอบภาพแบบเรียลไทม์มีดังนี้

- ส่วนนำเข้ารูปภาพ
- ส่วนประมวลผลและหาขอบภาพ
- ส่วนแสดงผลลัพธ์และปรับแต่งคุณภาพ
- ส่วนติดต่อผู้ใช้

3.4.1 โครงสร้างของโปรแกรมหาขอบภาพแบบเรียลไทม์มีดังนี้

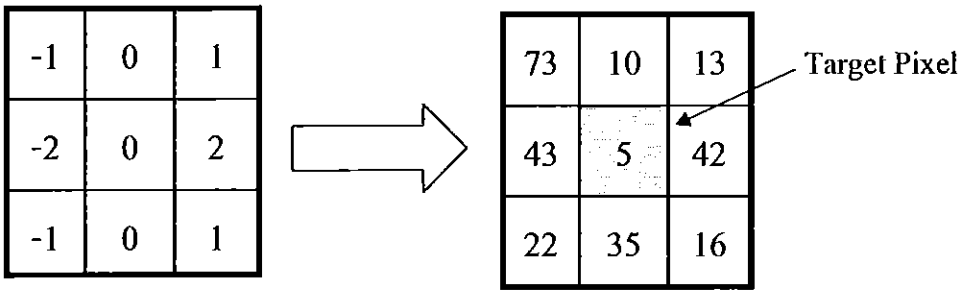


รูปที่ 3.2 แสดงผังแสดงการทำงานของโปรแกรม

3.5 การคำนวณหาขอบภาพโดยใช้วิธี Mask Operation

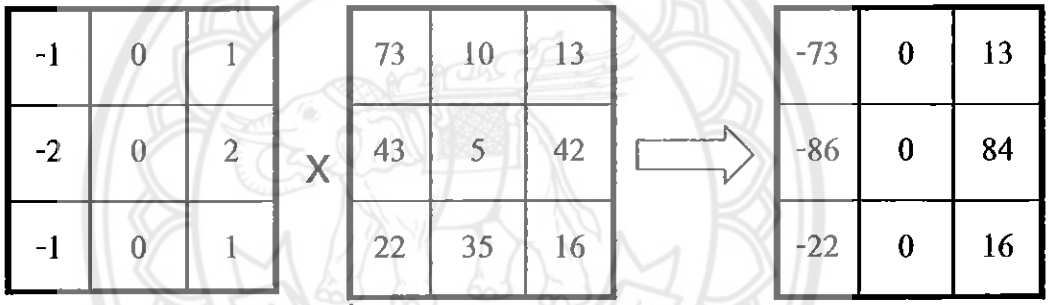
ตัวอย่าง การคำนวณด้วย Mask Operation ของ Sobel คือ $\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ หรือ {-1, 0, 1, -2, 0, 2, -1, 0, 1} ในรูปแบบ Array

1. วางmask ลงบนรูปที่คำนวณโดยตำแหน่งกลางของ Mask จะวางทับบน pixel ที่เราพิจารณา



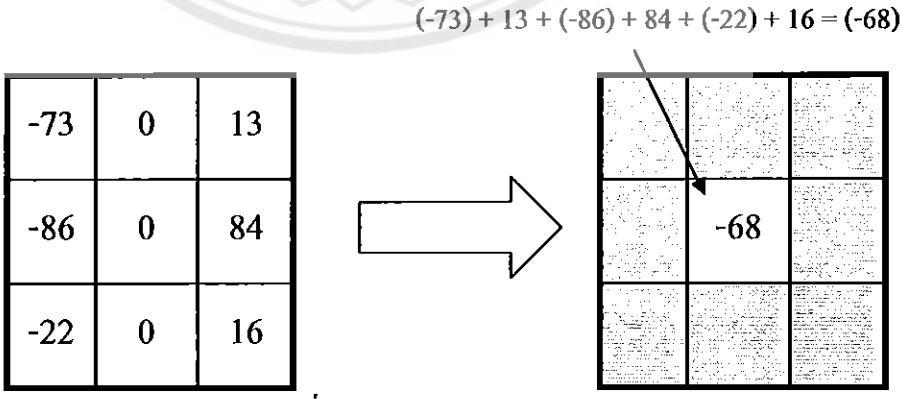
รูปที่ 3.3 ตำแหน่งการนำ mask มาวางทับภาพต้นฉบับ

2. นำค่าของ mask ดังกล่าวมาคูณกับค่าความเข้มของ pixel ที่ตำแหน่งตรงกัน



รูปที่ 3.4 การนำ mask มาคูณกับภาพต้นฉบับ

3. นำผลคูณมารวมกันเป็นผลลัพธ์ของ Target Pixel



รูปที่ 3.5 การหาค่า Target Pixel

เมื่อได้ Target pixel มาแล้ว นำมาพิจารณาด้วยเงื่อนไข

- Target pixel ใดๆ ที่มีค่ามากกว่า 255 ให้มีค่าเป็น 255 หรือ
- Target pixel ใดๆ ที่มีค่าน้อยกว่า 0 ให้มีค่าเป็น 0

จากตัวอย่าง Target pixel จะถูกกำหนดให้มีค่าเป็น 0

3.6 การรวมผลลัพธ์ (Output Fusion)

เมื่อได้ Target pixel ของแต่ละ map แล้ว จึงนำ Target pixel มารวมกัน ดังสมการ

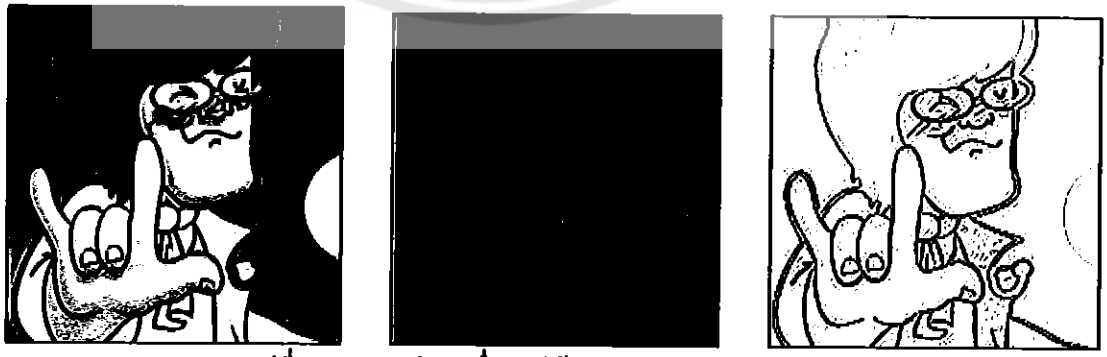
$$Output = \sqrt{R^2 + G^2 + B^2} \tag{3.1}$$

3.7 การปรุงคุณภาพของผลลัพธ์

เมื่อเราได้ผลลัพธ์ออกมาแล้วนั้น ผลลัพธ์ที่ได้อาจจะยังไม่ให้ความคมชัดพอที่จะนำไปใช้งานหรืออาจจะขนาดเล็กเกินไป ในการพัฒนาโปรแกรมจึงได้ทำฟังก์ชันเพื่อปรับปรุงคุณภาพของผลลัพธ์เพื่อสามารถปรับ ขนาด ความคมชัด และคุณสมบัติอื่นๆ ของผลลัพธ์เพื่อให้ได้ภาพที่เหมาะสมที่สุดนำไปใช้ในงานต่อไป

3.7.1 Inverted method

การประมวลผลหาขอบภาพของพื้นที่ที่ได้ทำการเลือกจากพิกัดของเมาส์ (Mouse Over) แล้ว ผลลัพธ์ที่ถูกแสดงผลออกมาจะเป็นพื้นหลังสีดำเป็นสีเขียว จะใช้วิธีการ Invert ให้พื้นหลังเป็นสีขาวและเส้นขอบภาพเป็นสีดำที่ดูได้ง่ายขึ้น เพื่อสะดวกในการนำไปใช้ประโยชน์ในขั้นต่อไป



รูปที่ 3.6 แสดงรูปภาพเมื่อเปิดใช้งานโหมด White Background

3.7.2 Thresholding

การใช้ Threshold เพื่อปรับความคมชัดของขอบภาพ โดยพิจารณาจากค่าสี ณ จุดใดๆ ดังนี้

- **Threshold 1 Value** ใช้สำหรับกำหนดค่าของจุดพิกเซล หากค่าของพิกเซลใดๆ มีค่ามากกว่าค่า Threshold 1 จะกำหนดค่าให้เป็น 255

- **Threshold 2 Value** ใช้สำหรับกำหนดค่าของจุดพิกเซล หากค่าของพิกเซลใดๆ มีค่าน้อยกว่าค่า Threshold 1 แต่มากกว่าค่า Threshold 2 จะกำหนดค่าให้เท่ากับค่า Threshold 2 หากค่าพิกเซลใดๆ มีค่าน้อยกว่าค่า Threshold 2 จะกำหนดค่าให้เป็นศูนย์

บทที่ 4

ผลการทดลอง

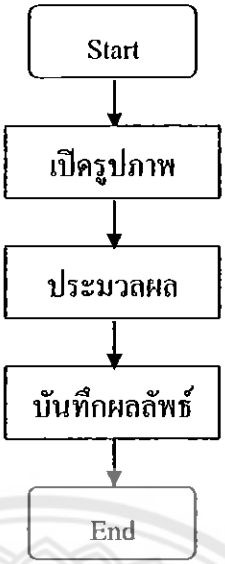
ในการทดลองใช้โปรแกรมหาขอบภาพแบบเรียลไทม์หาขอบภาพของรูภาพประเภทต่าง ๆ ไม่ว่าจะเป็น ภาพการ์ตูน ภาพคน ภาพที่มีความเข้มสีน้อย หรือภาพที่มี noise เป็นต้น เนื่องจากภาพแต่ละประเภทนั้นมีรายละเอียดแตกต่างกันไป ดังนั้นอัลกอริทึมที่เหมาะสมกับภาพแต่ละประเภทนั้นก็จะไม่เหมือนกันขึ้นอยู่กับรายละเอียดของภาพ ไม่มีอัลกอริทึมใดที่ดีที่สุด ดังนี้

4.1 การทดลองการหาขอบรูปภาพประเภทต่างๆ ด้วยโปรแกรม Real-time Color Image Edge Detection

เนื่องจากรูปภาพมีอยู่หลายประเภท การทดสอบผลลัพธ์ของอัลกอริทึมต่างๆจึงนำรูปภาพที่มีลักษณะต่างๆ มาทดสอบ ดังนี้

1. รูปภาพขาว-ดำ คือ รูปภาพที่มีเฉพาะสีขาวและสีดำ
2. รูปภาพ Gray Scale คือ รูปภาพที่มีเฉพาะสีขาวและสีเทา แต่จะมีการไล่สีเกิดเป็นสีเทาหลายระดับตั้งแต่สีขาวไปจนถึงสีดำ
3. รูปภาพสี คือ รูปภาพสีที่มีองค์ประกอบทั้งสีสีแดง สีเขียวและน้ำเงิน
4. รูปภาพการ์ตูน คือ รูปภาพสีที่มีขอบเขตของสีและแต่ละองค์ประกอบในรูปภาพอย่างเด่นชัด และไม่มีการไล่สี (Gradient)
5. รูปภาพใบหน้าคน คือ รูปภาพใบหน้าคน ซึ่งใช้รูปภาพใบหน้าของลีน่ามาทดลอง
6. รูปภาพที่มี Noise คือ รูปภาพที่มีจุดรบกวน (Noise) อยู่ในรูปภาพ
7. รูปภาพที่มีความเข้มสีน้อย คือ รูปภาพที่มีความสว่างมาก
8. รูปภาพจากฟิล์มเอ็กซเรย์ คือ ตัวอย่างรูปภาพที่นำมาจากตัวอย่างฟิล์มเอ็กซ์เรย์ ซึ่งเป็นรูปภาพที่ไม่มีขอบเขตที่ชัดเจน

สำหรับการทดลองนั้นจะทำการเปิดรูปภาพทีละภาพ สั่งให้โปรแกรประมวลผล จากนั้นบันทึกผลลัพธ์จากการประมวลผลนั้นไว้เพื่อเปรียบเทียบต่อไป ดังกราฟ



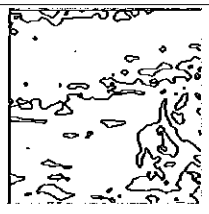
รูปภาพ 4.1 แสดงลำดับและวิธีการทดสอบ

4.2 ผลการทดลองการหาขอบรูปภาพแบบต่างๆ

4.2.1 ผลการทดลองการหาขอบรูปภาพขาว-ดำ

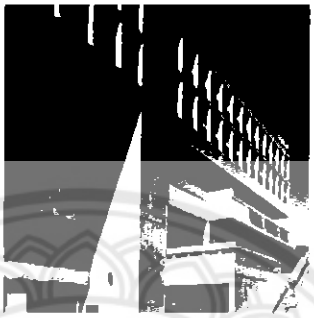
รูปภาพ 4.2 แสดงตัวอย่างรูปภาพขาว-ดำ

ตารางที่ 4.1 แสดงผลการทดลองหาขอบรูปภาพขาว-ดำ

Laplaciant 1st Derivative	Laplaciant 2nd Derivative	Robinson algorithm	Sobel algorithm
			

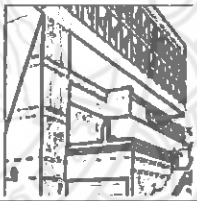
รูปภาพขาว-ดำที่นำมาทดสอบ มีลักษณะเป็นสีขาวตัดกับสีดำอย่างชัดเจน ในการหาขอบภาพ ทุก Algorithm ได้ผลลัพธ์เป็นที่หน้าพอใจ กล่าวคือ เห็นเป็นเส้นคมชัดสามารถแยกขอบรูปภาพได้อย่างชัดเจนทุก Algorithm

4.2.2 ผลการทดลองการหาขอบรูปภาพ Gray Scale



รูปภาพ 4.3 แสดงตัวอย่างรูปภาพ Gray Scale

ตารางที่ 4.2 แสดงผลการทดลองหาขอบรูปภาพ Gray Scale

Laplaciant 1st Derivative	Laplaciant 2nd Derivative	Robinson algorithm	Sobel algorithm
			

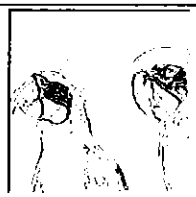
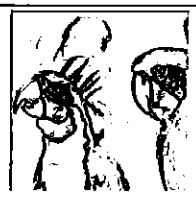
รูปภาพ Gray Scale ที่นำมาใช้ทดสอบ เป็นรูปภาพที่มีการไล่สี โดยประกอบไปด้วยสีขาวไปจนถึงสีดำ

ผลลัพธ์ของแต่ละ Algorithm ให้ผลลัพธ์ที่ใกล้เคียงกัน แต่ Algorithm ของ Laplaciant First Derivative นั้นให้ผลลัพธ์ที่ชัดเจนมากกว่าอัลกอริทึมอื่นๆ

4.2.3 ผลการทดลองการหาขอบรูปภาพสี

รูปภาพ 4.4 แสดงตัวอย่างรูปภาพสี

ตารางที่ 4.3 แสดงผลการทดลองหาขอบรูปภาพสี

Laplaciant 1st Derivative	Laplaciant 2nd Derivative	Robinson algorithm	Sobel algorithm
			

รูปภาพที่นำมาทดสอบ เป็นรูปภาพสีที่มีองค์ประกอบสีครบถ้วน เพื่อใช้สำหรับการหาขอบภาพในกรณีที่รูปภาพมีการตัดกันของสีอย่างเด่นชัด จะเห็นว่า Algorithm ของ Robinson ให้ผลลัพธ์ของการหาขอบภาพได้เป็นอย่างดี

4.2.4 ผลการทดลองการหาขอบรูปภาพการ์ตูน

รูปภาพ 4.5 แสดงตัวอย่างรูปภาพการ์ตูน

ตารางที่ 4.4 แสดงผลการทดลองหาขอบรูปภาพการ์ตูน

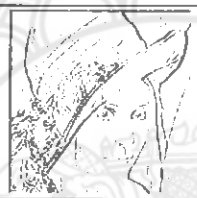
Laplaciant 1st Derivative	Laplaciant 2nd Derivative	Robinson algorithm	Sobel algorithm
			

รูปภาพการ์ตูนที่นำมาทดสอบนี้ ให้ผลลัพธ์เป็นที่น่าพอใจ แต่ Algorithm ของ Laplacint First Derivative นั้นกลับให้ผลลัพธ์ที่ไม่น่าพอใจ กล่าวคือ ทำให้เกิดเส้นแสดงขอบขึ้นมาถึง 2 เส้น

4.2.5 ผลการทดลองการหาขอบรูปภาพคน

รูปภาพ 4.6 แสดงตัวอย่างรูปภาพใบหน้าคน

ตารางที่ 4.5 แสดงผลการทดลองหาขอบรูปภาพคน

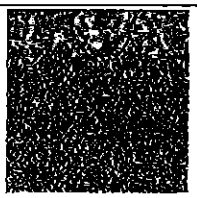
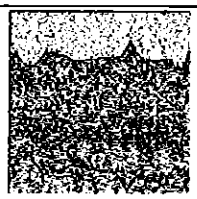
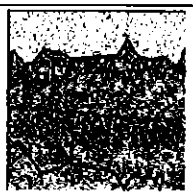
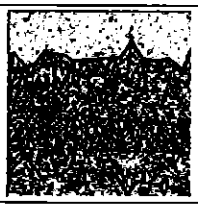
Laplaciant 1st Derivative	Laplaciant 2nd Derivative	Robinson algorithm	Sobel algorithm
			

จากตาราง จะเห็นว่า Algorithm ที่สามารถหาขอบภาพได้ดีที่สุดคือ Algorithm ของ Laplaciant Second Derivative ส่วน Algorithm อื่นๆให้ผลลัพธ์ที่ไม่ชัดเจน

4.2.6 ผลการทดลองการหาขอบรูปภาพที่มี Noise

รูปภาพ 4.7 แสดงตัวอย่างรูปภาพที่มี Noise

ตารางที่ 4.6 แสดงผลการทดลองหาขอบรูปภาพที่มี Noise

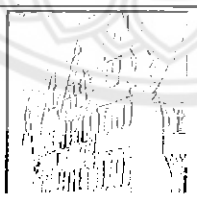
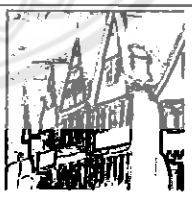
Laplaciant 1st Derivative	Laplaciant 2nd Derivative	Robinson algorithm	Sobel algorithm
			

รูปภาพที่มีจุดรบกวน หรือ Noise นั้น ไม่มี Algorithm ใดที่สามารถหาขอบภาพได้ เนื่องจากขอบภาพที่เกิดจากจุดรบกวน ทำให้เกิดเส้นมากมายขึ้นบนภาพ จึงไม่สามารถหาขอบภาพที่แท้จริงได้

4.2.7 ผลการทดลองการหาขอบรูปภาพที่มีความเข้มสีน้อย

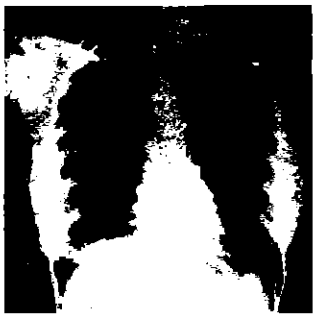
รูปภาพ 4.8 แสดงตัวอย่างรูปภาพที่มีความเข้มสีน้อย

ตารางที่ 4.7 แสดงผลการทดลองหาขอบรูปภาพที่มีความเข้มสีน้อย

Laplaciant 1st Derivative	Laplaciant 2nd Derivative	Robinson algorithm	Sobel algorithm
			

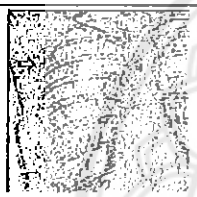
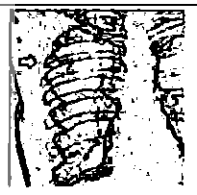
เช่นเดียวกับผลลัพธ์ของการหาขอบภาพสีทั่วไป การหาขอบภาพด้วยรูปภาพประเภทนี้ ให้ผลลัพธ์ที่ชัดเจน แต่ไม่สามารถสรุปได้ว่า Algorithm ใดให้ผลลัพธ์ที่ที่สุด ขึ้นอยู่กับปัจจัยของรูปภาพที่นำมาทดสอบว่า เป็นรูปภาพประเภทใด ทั้งนี้ การหาขอบรูปภาพชนิดนี้อาจต้องใช้ Filter value และ Boundary value ที่มีในโปรแกรมหาขอบภาพสีแบบ Real-time นี้ เพื่อปรับแต่งให้ผลลัพธ์นั้นชัดเจนมากขึ้น

4.2.8 ผลการทดลองการหาขอบรูปภาพจากฟิล์มเอ็กซเรย์



รูปภาพ 4.9 แสดงตัวอย่างรูปภาพจากฟิล์มเอ็กซเรย์

ตารางที่ 4.8 แสดงผลการทดลองหาขอบรูปภาพจากฟิล์มเอ็กซเรย์

Laplaciant 1st Derivative	Laplaciant 2nd Derivative	Robinson algorithm	Sobel algorithm
			

ตัวอย่างรูปภาพฟิลม์เอ็กซ์เรย์ที่นำมาใช้ทดสอบ เป็นรูปภาพซี่โครงมนุษย์ ซึ่งสามารถใช้โปรแกรมหาขอบภาพแบบ Real-time เพื่อนำไปใช้ในการวินิจฉัยโรคได้

เมื่อนำมาทดสอบ จะเห็นว่าบางอัลกอริทึมจะได้ผลลัพธ์ที่แทบมองไม่เห็นเลย แต่อัลกอริทึมของ Sobel และ Robinson นั้นให้ผลลัพธ์ที่ชัดเจนมากกว่า อย่างไรก็ตาม ผลลัพธ์ของการหาขอบภาพชนิดนี้ สามารถปรับปรุงได้ โดยใช้ Filter value และ Boundary value เพื่อปรับปรุงผลลัพธ์ให้ดีขึ้นได้

บทที่ 5
สรุปผล

โครงงานนี้เป็นการศึกษาและพัฒนาโปรแกรมว่าด้วยเรื่อง Color Image Edge Detection ซึ่งเป็นการนำทฤษฎีทาง Image Processing เข้ามาใช้ในการหาขอบภาพสี โดยการทำงานของโปรแกรมเป็นแบบเรียลไทม์ ภาพที่นำมาประมวลผลเป็นภาพแบบ Digital Format เมื่อโปรแกรมได้นำเข้าภาพต้นฉบับมาแล้วจะทำการแยกภาพดังกล่าวออกเป็น 3 map (RGB) แล้วหาขอบภาพด้วยอัลกอริทึมต่างๆ แล้วสามารถนำภาพที่ได้มาทำการปรับปรุงคุณภาพโดยการใช้ Treshold value เพื่อเป็นการหาผลลัพธ์ที่เหมาะสมเพื่อนำไปใช้งานต่อไป

ผลการทดลองจากการใช้โปรแกรมหาขอบภาพกับภาพสีประเภทต่างๆ ที่มีความสว่าง ความเข้มสี เฉดสี ที่แตกต่างกันออกไปแล้วนั้นจะพบว่าอัลกอริทึมที่สามารถทำงานได้ดีในการหาขอบภาพแต่ละประเภทนั้นแตกต่างกันไปตามรายละเอียดของภาพ

5.1 สรุปผลการทดลอง

จากการทดลองนำรูปภาพแบบประเภทต่างๆ มาทดสอบหาขอบภาพแบบ Real-time โดยใช้ภาษา C# ในเบื้องต้นโปรแกรมสามารถประมวลผลหาขอบภาพได้ จากการทดลองหาขอบภาพจากโปรแกรม ประกอบไปด้วยรูปภาพ 8 รูป และ 4 Edge Detection Algorithms ได้สรุปดังตาราง

ตารางที่ 5.1 ผลลัพธ์การหาขอบภาพด้วยโปรแกรม แยกตามประเภทของรูปภาพ

ประเภทของรูปภาพ	ตัวอย่างรูปภาพ	ผลของการหาขอบภาพ			
		Laplaciant First Derivative	Laplaciant Second Derivative	Robinson	Sobel
รูปภาพขาว-ดำ		ดี	ดี	ดี	ดี
รูปภาพ Gray-Scale		พอใช้	ไม่ดี	ดี	ไม่ดี

รูปภาพสี		ไม่ดี	พอใช้	ดี	พอใช้
รูปภาพการ์ตูน		ไม่ดี	ดี	ดี	พอใช้
รูปภาพใบหน้าคน		พอใช้	ดี	ไม่ดี	ไม่ดี
รูปภาพที่มี Noise		ไม่ดี	ไม่ดี	ไม่ดี	ไม่ดี
รูปภาพที่มีความเข้มสีน้อย		พอใช้	ดี	พอใช้	พอใช้
รูปภาพจากฟิล์มเอ็กซ์เรย์		พอใช้	ไม่ดี	ดี	ดี

5.2 ปัญหาและอุปสรรค

- การหาขอบรูปภาพ บางครั้งผลลัพธ์ที่ได้ไม่สามารถวิเคราะห์ได้ว่า ส่วนใดเป็นขอบภาพ เนื่องจากรูปภาพนั้น มีส่วนที่มีการไล่สี จึงทำให้รูปภาพที่ได้

- โปรแกรมไม่สามารถหาขอบของรูปภาพที่มีจุดรบกวน Noise ได้

- การเขียนโปรแกรมที่อาศัยความเร็วหรือการทำงานแบบ Real-time นั้นทำได้ยาก เนื่องจากการใช้ภาษา C# ซึ่งเป็นภาษาที่อาศัยการเขียนโปรแกรมเชิง OOP จึงอาจทำให้โปรแกรมทำงานช้า

5.3 ข้อเสนอแนะ

- การทำให้ผลลัพธ์มีความชัดเจนมากขึ้น สามารถใช้ Threshold value ในการปรับความชัดเจนได้

- สามารถย่อรูปภาพก่อนทำการประมวลผลได้ หากรูปภาพที่ต้องการหาขอบภาพมีขนาดใหญ่เกินไป

5.4 สรุป

- ศึกษาเกี่ยวกับทฤษฎีของรูปภาพขาว-ดำ และองค์ประกอบของรูปภาพสีแบบต่างๆ
- ศึกษาประโยชน์ของรูปแบบการจัดเก็บรูปภาพสีแบบต่างๆ
- ศึกษาการประมวลผลรูปภาพดิจิตอล
- ศึกษาการใช้ Convolution mask สำหรับการประมวลรูปภาพ (Image Processing)
- ศึกษาทฤษฎีการหาขอบภาพสำหรับรูปภาพขาวดำและนำมาทดลองใช้
- ศึกษาหลักการประมวลผลสำหรับรูปภาพสี
- ศึกษาการแยกประมวลผลรูปภาพทีละ Channel และการรวมผลลัพธ์ Channel ต่างๆ
- ศึกษาหลักการเขียนโปรแกรมเชิงวัตถุ
- ศึกษาการใช้งานและพัฒนาโปรแกรมหาขอบภาพแบบ Real-time

บรรณานุกรม

[1] Rafael C. Gonzalez, Richard E. Woods, " Digital image Processing " second edition , Upper Saddle River, New Jersey 07458, Prentice-Hall, Inc., 2001.

[2] Rafael C. Gonzalez, Richard E. Woods, " Digital image Processing " Reprinted with corrections September, Addison-Wesley Publishing Company,Inc., 1992.

[3] Gregory A Baxes, "Digital Image Processing, principle and applications", United States of America, John Wiley & Son, Inc., 1994.

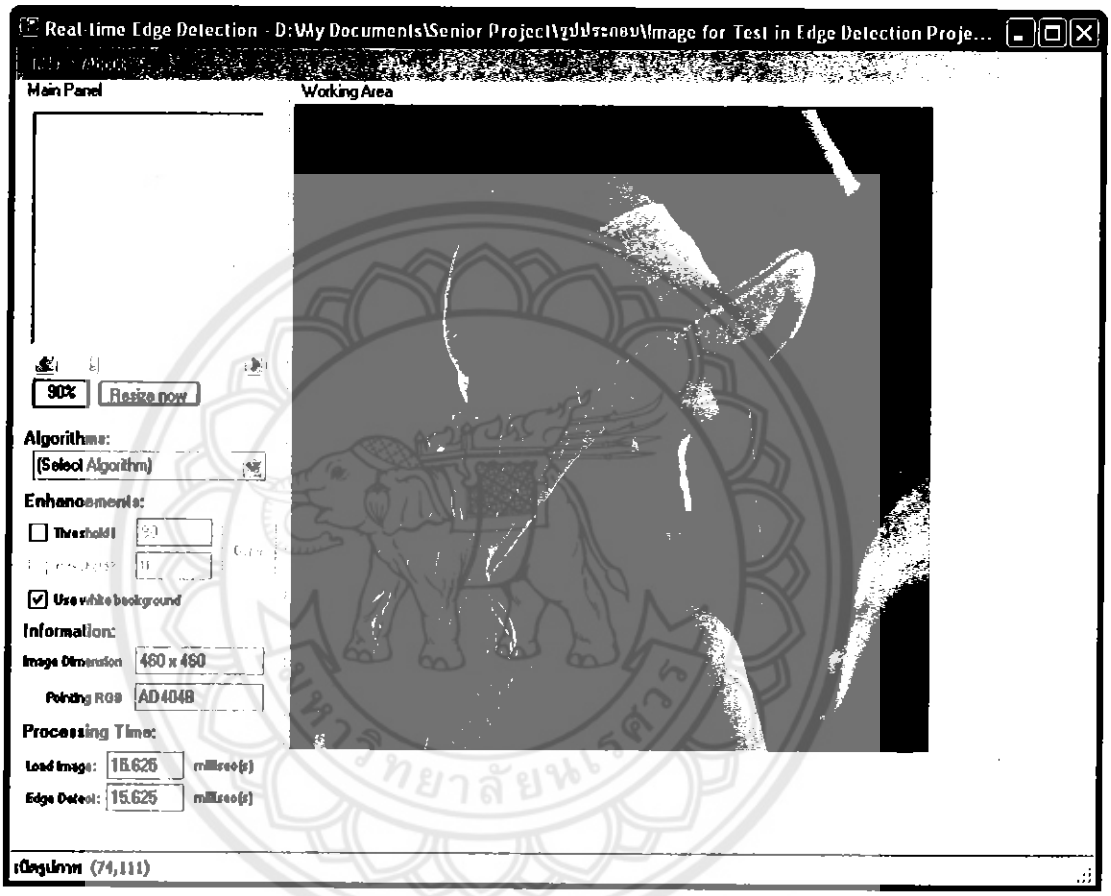
[4] http://www.cs.haifa.ac.il/~dkeren/ip/short-lecture9.pdf

[5] http://www.cee.hw.ac.uk/hipr/html/prewitt.html

ภาคผนวก

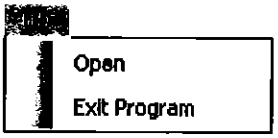
ภาคผนวก ก

ส่วนประกอบและวิธีการใช้โปรแกรม



รูปภาคผนวก 1 รูปแสดง Interface ของโปรแกรม

1. Main Menu

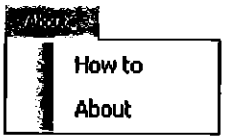


รูปภาคผนวก 2 รูปแสดงเมนู File ของโปรแกรม

- **Open**ใช้สำหรับเปิดรูปภาพเพื่อนำเข้ามาใช้ประมวลผล

- **Exit** ปิดโปรแกรม

2. เมนู About



รูปภาคผนวก 3 รูปแสดงเมนู About ของโปรแกรม

- **How to** อธิบายวิธีใช้โปรแกรม
- **About** แสดงข้อมูลผู้พัฒนาโปรแกรม

3. Main Panel

กลุ่มเครื่องมือสำหรับแสดงผลลัพธ์ของการประมวลผลการปรับแต่งผล**ลัพธ์ ดังนี้**

รูปภาคผนวก 4 รูปแสดงเมนู About ของโปรแกรม

4. Working Area

หน้าต่างสำหรับแสดงรูปภาพต้นฉบับ เพื่อใช้แสดงและสามารถเลือกขอบเขตของรูปภาพที่จะถูกประมวลผลโดยการเลื่อนเมาส์ไปยังตำแหน่งใดๆ ภายใน Working Area แล้วโปรแกรมจะทำการแสดงขอบเขตของรูปภาพที่จะถูกนำมาประมวลผลยังบริเวณนี้

รูปภาคผนวก 5 รูปแสดง Work Area

5. Algorithm

ใช้สำหรับเลือกอัลกอริทึมสำหรับประมวลผลหาขอบภาพใน Working Area

รูปภาคผนวก 6 รูปแสดง Algorithm

6. Enhancement

6.1 Use White Background

เลือก Checkbox ปรับภาพพื้นหลังในพื้นที่การแสดงผลให้เป็นสีขาว และขอบภาพเป็นสีดำ โดยการนำหลักการ Invert รูปภาพมาประยุกต์ใช้กับการแสดงผลลัพธ์ หลังจากการหาขอบภาพ

6.2 Thresholding

เลือก Checkbox เพื่อ Enable การใช้ Thresholding กับขอบภาพ สามารถกำหนดค่าและปรับเปลี่ยนค่า Threshold value โดยการเปลี่ยนตัวเลขที่ textbox และเลือกที่ปุ่ม Done

6.3 Information

- Size แสดงขนาดของรูปภาพต้นฉบับ
- Color แสดงค่าสีของรูปภาพ ณ พิกัดที่เมาส์กำลังชี้อยู่
- Load image แสดงเวลาที่ใช้ไปในการโหลดรูปภาพขึ้นมาบน Working Area
- Edge detect แสดงเวลาที่ใช้ในการหาขอบภาพจากรูปภาพต้นฉบับไปยังพื้นที่แสดงผลลัพธ์

ภาคผนวก ข

ประวัติผู้เขียนโครงงาน

ชื่อ นางสาวประภาพรรณ ชูพินิจ

ภูมิลำเนา 586 หมู่ 1 ตำบลต่อม อำเภอเมือง จังหวัดพะเยา 56000

ประวัติการศึกษา

- จบการศึกษาชั้นมัธยมศึกษาโรงเรียนพะเยาพิทยาคม จังหวัดพะเยา
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E-mail: mamichan14@gmail.com

ชื่อ นายกีรดิษย์ บุญญลาภาเลิศ

ภูมิลำเนา 43/6 หมู่ 8 ตำบลหัวรอ อำเภอเมือง จังหวัดพิษณุโลก 65000

ประวัติการศึกษา

- จบการศึกษาชั้นมัธยมศึกษาจากโรงเรียนสวนกุหลาบวิทยาลัย กรุงเทพมหานคร
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4 สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E-mail: oskee121@gmail.com