ใบรับรองวิทยานิพนธ์

บัณฑิตวิทยาลัย สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ

เรื่อง การวิเคราะห์และตรวจสอบความบกพร่องของผ้าย้อมด้วยโครงข่ายประสาทเทียม
โดย นายณัฐวัชร มาลัย

ได้รับอนุมัติให้นับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมเครื่องกล

_____คณบดีบัณฑิตวิทยาลัย
(อาจารย์ ดร.มงคล หวังสถิตย์วงษ์)
7 มีนาคม 2550

คณะกรรมการสอบวิทยานิพนธ์

_____ประธานกรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.สินชัย ชินวรรัตน์)

_____กรรมการ
(รองศาสตราจารย์ ดร.สุวัฒน์ กุลธนปรีดา)

_____กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.ศิริพรรณ ธงชัย)

_____กรรมการ
(อาจารย์ ดร.พิชัย อัษฎมงคล)

การวิเคราะห์และตรวจสอบความบกพร่องของผ้าย้อมด้วยโครงข่ายประสาทเทียม

นายณัฐวัชร มาลัย

วิทยานิพนธ์เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมเครื่องกล ภาควิชาวิศวกรรมเครื่องกล
บัณฑิตวิทยาลัย สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ปีการศึกษา 2549

ชื่อ : นายณัฐวัชร มาลัย
ชื่อวิทยานิพนธ์ : การวิเคราะห์และตรวจสอบความบกพร่องของผ้าย้อมด้วยโครงข่ายประสาทเทียม
สาขาวิชา : วิศวกรรมเครื่องกล
สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ที่ปรึกษาวิทยานิพนธ์ : ผู้ช่วยศาสตราจารย์ ดร.สินชัย ชินวรรัตน์
รองศาสตราจารย์ ดร.สุวัฒน์ กุลธนปรีดา
ปีการศึกษา : 2006

บทคัดย่อ

วิทยานิพนธ์ฉบับนี้กล่าวถึงการศึกษา ออกแบบและสร้างโปรแกรมตรวจสอบความเพี้ยนของสีของผ้าย้อมโทนเดียว ซึ่งมีจุดประสงค์ที่จะปรับปรุงการตรวจสอบคุณภาพของผ้าย้อม ซึ่งในปัจจุบันเป็นการตรวจสอบและตัดสินใจคุณภาพโดยการมองและตัดสินใจโดยคน ซึ่งให้ผลที่ไม่แม่นยำมากนัก สิ้นเปลืองบุคลากรและเวลาในการตรวจสอบ ดังนั้นงานวิจัยนี้ได้ศึกษา ออกแบบและสร้างระบบวิเคราะห์และตรวจสอบความบกพร่องของผ้าย้อมขึ้น โดยการสร้างโปรแกรมตรวจสอบความเพี้ยนของสีของผ้าย้อมโทนเดียว ซึ่งใช้หลักการของโครงข่ายประสาทเทียม (Artificial Neural Network) ซึ่งโครงข่ายประสาทเทียมที่เลือกใช้นั้น จะเป็นโครงข่ายที่ไม่มีการป้อนกลับ (Feed Forward Network) ที่มี 3 ชั้น และใช้ระเบียบวิธีแบบแพร่กลับ (Back Propagation Training Algorithm) ซึ่งจัดเป็นโครงสร้างที่มีการใช้งานในระบบจำแนก (Classification System) ทั่วไป นอกจากนี้โปรแกรมนี้จะทำการเปรียบเทียบกับทฤษฎีทางสถิติที่ออกแบบขึ้น ซึ่งใช้หลักการในเชิงปริมาณ เพื่อเปรียบเทียบความสามารถและประโยชน์ของแต่ละหลักการ เพื่อการวิเคราะห์ผลการทดลองโปรแกรมอย่างแท้จริง

(วิทยานิพนธ์มีจำนวนทั้งสิ้น 165 หน้า)

คำสำคัญ : โครงข่ายประสาทเทียม, โครงข่ายที่ไม่มีการป้อนกลับ, ระเบียบวิธีแบบแพร่กลับ, ระบบจำแนก

_____อาจารย์ที่ปรึกษาวิทยานิพนธ์

Name : Mr.Nattavat Malai
Thesis Title : Analysis and Defect Detection of Dyed Fabric by Artificial Neural Network
Major Field : Mechanical Engineering
King Mongkut's Institute of Technology North Bangkok
Thesis Advisors : Assistant Professor Dr.Sinchai Chinvorarat
Associate Professor Dr.Suwat Kuntanapreeda
Academic Year : 2006

Abstract

The objective of this research is to study, design and create the colour defect detection program aiming to improve quality inspection of dyed fabric. Nowadays quality inspection of dyed fabric is inspected by human that is not so much accuracy, required manpower for inspection and long time consumption. Therefore this research has purposed the new technique for analysis and inspection system using the colour defect detection program. Artificial Neural Network along with Feed Forward Network in 3-layer type and Back Propagation Training Algorithm for structure of general classification system has been a core technique in this program. Moreover this program will show the comparison with a statistical theory in term of performance and advantage between two methods.

(Total 165 pages)

Keywords : Artificial Neural Network, Feed Forward Network, Back Propagation Training Algorithm, Classification System

_____Advisor

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปด้วยความช่วยเหลืออย่างดียิ่งของ ผู้ช่วยศาสตราจารย์ ดร.สินชัย ชินวรรัตน์ และ รองศาสตราจารย์ ดร.สุวัฒน์ กุลธนปรีดา อาจารย์ที่ปรึกษาวิทยานิพนธ์ที่ได้ให้คำแนะนำและข้อคิดเห็นต่างๆ ของการวิจัยมาโดยตลอด และทุนวิจัยบางส่วนได้รับจากทุนอุดหนุนการวิจัยของบัณฑิตวิทยาลัย จึงขอขอบพระคุณบัณฑิตวิทยาลัยที่ได้ให้ทุนอุดหนุนการวิจัยครั้งนี้มา ณ ที่นี้ด้วย

ท้ายนี้ผู้วิจัยใคร่ขอกราบขอบพระคุณบิดา มารดา และคุณป้า ซึ่งสนับสนุนในด้านการเงินและให้กำลังใจแก่ผู้วิจัยเสมอมาจนสำเร็จการศึกษา

ณัฐวัชร มาลัย

สารบัญ

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญภาพ

สารบัญภาพ (ต่อ)

สารบัญภาพ (ต่อ)

สารบัญภาพ (ต่อ)

คำอธิบายสัญลักษณ์

A	จำนวนพิกเซลที่ประกอบกันขึ้นมาเป็นวัตถุ
a	ข้อมูลออกของนิวรอน (Network Output)
B	จำนวนบิตของระบบภาพ (Bit Number)
B	ค่าความสว่างของสีฟ้า
b	ค่าขีดเริ่มเปลี่ยน (Bias)
BBH	ด้านสูงของกรอบสี่เหลี่ยม
BBW	ด้านกว้างของกรอบสี่เหลี่ยม
BLOB	วัตถุที่เป็นภาพ 2 ระดับ (Binary Large Object)
C_{max}	ตำแหน่งพิกเซลที่หลักมากสุด
C_{min}	ตำแหน่งพิกเซลที่หลักน้อยสุด
e	ค่าความผิดพลาด (Error)
F	เซทของพิกเซลที่มีค่าความเข้มแสงเท่ากับ 255 ซึ่งจัดเป็นวัตถุหรือพื้นหน้า (Foreground)
f	ฟังค์ชั่นกระตุ้น (Activation Function)
G	ค่าความสว่างของสีเขียว
H	ค่าเฉดสีของสี (Hue)
I	ค่าความเข้มแสงของพิกเซล (Intensity)
m	ความชันของสมการเส้นตรง
N_F	เซทของพิกเซลมีความเข้มแสงเท่ากับ 255
n_B (T)	จำนวนพิกเซลทั้งหมดของด้านสว่าง
n_D (T)	จำนวนพิกเซลทั้งหมดของด้านมืด
O	ค่าความเข้มแสงของภาพขาออกที่ตำแหน่งเดียวกัน
p	ข้อมูลเข้า (Input Data)
R	ค่าความสว่างของสีแดง
r_{max}	ตำแหน่งพิกเซลที่แถวมากสุด
r_{min}	ตำแหน่งพิกเซลที่แถวน้อยสุด
S	ค่าความอิ่มตัวของสี (Saturation)
s	ค่าความไหว (Sensitivity)
T	ค่าเทรดโชว์ (Threshold)

คำอธิบายสัญลักษณ์ (ต่อ)

t	ข้อมูลออกที่ต้องการ (Target Output)
μ	ค่าเฉลี่ยรวมของทั้งฮิสโตแกรม
w	ตัวถ่วงน้ำหนักเชื่อมโยง (Connection Weight)
$\sigma_{D(T)}$	ความแปรปรวนของบริเวณด้านมืด
$\sigma_{B(T)}$	ความแปรปรวนของบริเวณด้านสว่าง
$\sigma^2_{Between}$	ความแปรปรวนรวมของทั้งฮิสโตแกรม
α	อัตราการเรียนรู้ (Learning Rate)
γ	โมเมนตัม (Momentum Coefficient)
θ	มุมของเส้นตรงที่มีความชันเท่ากับ m

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

อุตสาหกรรมการผลิตผ้าย้อมเป็นอุตสาหกรรมหนึ่งที่มีการตรวจสอบคุณภาพของผลิตภัณฑ์ผ้าย้อมโดยวิธีการฉายแสงลอดผ่านจากด้านหลังของผ้าย้อมและใช้วิธีการจำแนกและตัดสินคุณภาพโดยผู้ตรวจสอบหรือวิธีการมองของมนุษย์นั่นเอง ซึ่งยังเป็นระบบที่ไม่มีความแน่นอนและแม่นยำ ก่อให้เกิดข้อผิดพลาดอยู่บ้างและใช้เวลาในการตรวจสอบพอสมควร รวมถึงความสิ้นเปลืองของบุคลากรด้วย ดังนั้นผลงานวิจัยนี้มีเป้าหมายที่จะสร้างระบบตรวจสอบที่จะเพิ่มความแม่นยำ เพิ่มความสะดวก ลดเวลา และลดค่าใช้จ่ายของบุคลากรในการตรวจสอบ

ดังนั้น จึงได้มีการออกแบบและสร้างโปรแกรมเพื่อนำมาใช้ในการวิเคราะห์และตรวจสอบความบกพร่องของสีผ้าย้อมแบบโทนสีเดียวโดยใช้หลักการของโครงข่ายประสาทเทียม (Artificial Neural Network) โดยโครงข่ายประสาทเทียมที่เลือกใช้นั้น จะเป็นโครงข่ายที่ไม่มีการป้อนกลับ (Feed Forward Network) ที่มี 3 ชั้น และใช้ระเบียบวิธีแบบแพร่กลับ (Back Propagation Training Algorithm) ซึ่งจัดเป็นโครงสร้างที่มีการใช้งานในระบบจำแนก (Classification System) ทั่วไป นอกจากนี้ยังมีการสร้างโปรแกรมสำหรับตรวจสอบความบกพร่องของสีผ้าย้อมแบบโทนสีเดียวด้วย โดยใช้หลักการในเชิงปริมาณเพื่อเปรียบเทียบความแตกต่างและข้อดีข้อเสียของระบบจำแนกทั้งสอง เพื่อการวิเคราะห์ผลการทดลองโปรแกรมอย่างแท้จริง

นอกจากนี้โปรแกรมวิเคราะห์และตรวจสอบความบกพร่องของสีผ้าย้อมจะใช้ควบคู่กับระบบควบคุมที่มีกล้องรับภาพเป็นตัวรับภาพอินพุทและเข้าสู่โปรแกรมในคอมพิวเตอร์ที่เป็นตัวควบคุมและประมวลผล จากนั้นก็ให้เอ๊าพุทออกมาที่แอคชัวเอเตอร์ของระบบ อนึ่ง โปรแกรมวิเคราะห์และตรวจสอบคุณภาพของผ้าย้อมนี้ มุ่งเน้นไปที่การตรวจสอบความผิดพลาดของสีผ้าย้อม ซึ่งผ้าย้อมนั้นต้องมีเฉดสีเดียวกันหรือสีโทนเดียวเท่านั้น

1.2 วัตถุประสงค์ของการวิจัย

1.2.1 ออกแบบและสร้างระบบการวิเคราะห์และตรวจสอบเพื่อตรวจสอบความบกพร่องของสีผ้าย้อมแบบสีโทนเดียว

1.2.2 ประยุกต์ทฤษฎีการประมวลผลภาพดิจิตอล (Digital Image Processing) และนิวรอนเน็ตเวิร์ก (Neural Networks) เพื่อนำมาใช้ในการวิเคราะห์และตรวจสอบ

1.2.3 ปรับปรุงกระบวนการตรวจสอบคุณภาพของผ้าย้อม

1.3 ขอบเขตของการวิจัย

1.3.1 ศึกษาทฤษฎีการประมวลผลภาพดิจิตอล (Digital Image Processing) และนิวรอลเน็ตเวิร์ก (Neural Networks) เพื่อนำมาใช้ในการสร้างโปรแกรม

1.3.2 สร้างโปรแกรมวิเคราะห์ความบกพร่องของสีผ้าที่เกิดขึ้น

1.3.3 ออกแบบระบบควบคุม

1.4 วิธีวิจัย

ประกอบด้วย 2 ส่วนหลัก

1.4.1 สร้างโปรแกรมวิเคราะห์และตรวจสอบ ศึกษาวิจัยโปรแกรมวิเคราะห์และตรวจสอบความบกพร่องของผ้าย้อมแบบสีโทนเดียว ซึ่งในงานวิจัยนี้แบ่งออกเป็น 2 วิธีจำแนกคือ

1.4.1.1 วิธีทางสถิติ (Statistical) ซึ่งใช้วิธีจำแนกโดยการเปรียบเทียบเชิงปริมาณของค่าคุณสมบัติต่างๆของกลุ่มพิกเซลในภาพ

1.4.1.2 วิธีโครงข่ายประสาทเทียม (Artificial Neural Network) ซึ่งใช้โครงข่ายที่ไม่มีการป้อนกลับ (Feed Forward Network) ที่มี 3 ชั้น และใช้ระเบียบวิธีแบบแพร่กลับ (Back Propagation Training Algorithm) โดยนำค่าคุณสมบัติต่างๆของกลุ่มพิกเซลในภาพเป็นอินพุทของนิวรอลโดยระบุเป็น Training Data Set ให้นิวรอลจดจำกลุ่มพิกเซลที่เป็น Good and Bad (Defect) โดยผ่านกระบวนการเทรนนิ่ง ซึ่งกระบวนการเรียนรู้เป็นการเรียนรู้แบบมีผู้สอน (Supervised Learning)

1.4.2 ออกแบบระบบควบคุม

1.4.3 ทดลองและสรุปผล

1.5 ประโยชน์ที่ได้รับจากการวิจัย

1.5.1 ปรับปรุงกระบวนการตรวจสอบคุณภาพของผ้าย้อมเพื่อช่วยลดต้นทุน ประหยัดเวลาและให้ความแม่นยำสูง

1.5.2 สามารถนำไปใช้กับงานที่วิเคราะห์ทางภาพได้

1.5.3 สามารถนำไปปรับหรือดัดแปลงเพื่อใช้กับงานบางประเภทได้เช่น การตรวจนับชิ้นส่วนที่มีลักษณะเฉพาะ การตรวจจับรอยเปื้อนปนบนพื้นผิวงานอื่นๆ เป็นต้น

1.5.4 สามารถนำไปพัฒนาต่อในระดับที่ซับซ้อนมากขึ้นในอนาคตเช่น การตรวจสอบผ้าย้อมสำหรับผ้าที่มีลายซับซ้อนมากขึ้น

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

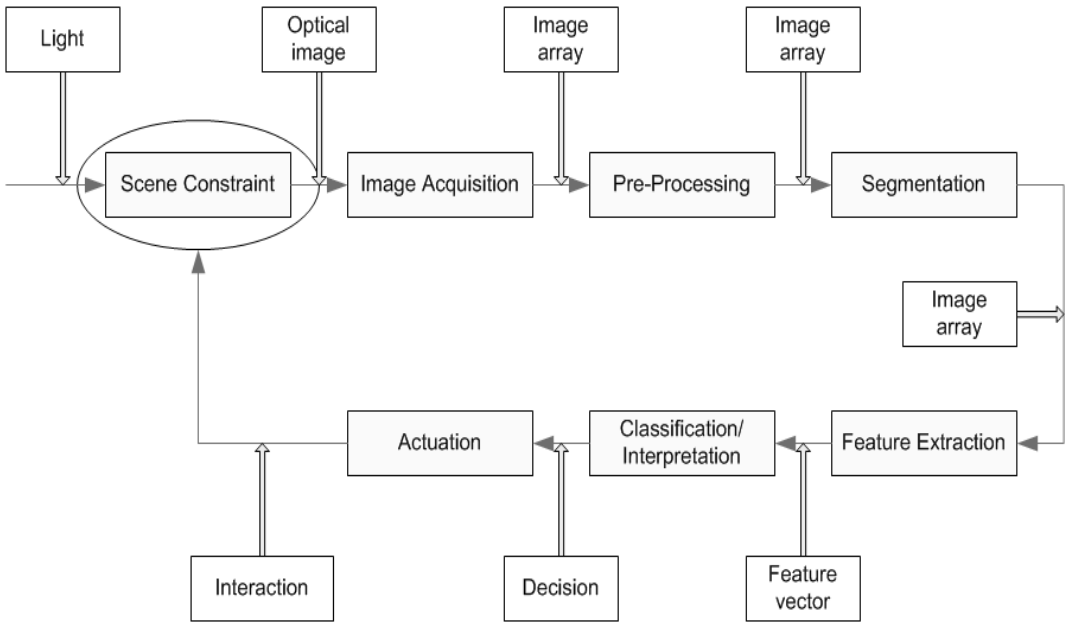
2.1 Machine Vision

Machine Vision[1] เป็นวิธีการที่ทำให้อุปกรณ์ประมวลผลต่างๆ เช่น คอมพิวเตอร์ หรืออุปกรณ์ประมวลสัญญาณดิจิตอล (Digital Signal Processor, DSP) มีความสามารถในการ “รับรู้” ภาพ ซึ่งรวมทั้งการทำให้อุปกรณ์ประมวลผลสามารถตัดสินใจและสั่งงานต่างๆ ได้ จากข้อมูลที่ได้จากภาพหรือกลุ่มของภาพนั้นๆ จุดมุ่งหมายสูงสุดของ Machine Vision คือ ทำให้เครื่องจักรหรืออุปกรณ์ประมวลผลต่างๆ มีความสามารถให้ได้เทียบเท่ากับระบบการมองเห็นของมนุษย์ที่มีวิวัฒนาการต่อเนื่องกันมาหลายสิบล้านปี อย่างไรก็ตาม เทคโนโลยีปัจจุบันยังคงไม่สามารถทำให้เครื่องจักรสามารถ “มองเห็นและรับรู้” ได้เทียบเท่ากับความสามารถของมนุษย์ ยกตัวอย่างเช่น สมมุติว่า เราที่เป็นมนุษย์ที่มีสภาพร่างกายปกติ จะสามารถแยกแยะสิ่งของที่ต้องการจากกองสิ่งของหลายๆ อย่าง หรือสามารถแยกแยะหน้าคนที่คุ้นเคยได้จากกลุ่มคนที่อยู่ภายใต้สภาพแวดล้อมที่มีฝนตกหรือหมอกลงจัด การทำกิจกรรมดังกล่าวนั้น จะสามารถทำได้โดยแทบจะไม่ต้องใช้ความพยายามมากเท่าไรนัก ซึ่งหากต้องการให้เครื่องจักรหรืออุปกรณ์ประมวลผลมีความสามารถที่จะทำกิจกรรมดังกล่าวได้นั้น นอกจากจะต้องใช้ความพยายามเป็นอย่างมากแล้ว ยังต้องใช้กระบวนการทางคณิตศาสตร์ที่ซับซ้อนอีกด้วย ทั้งนี้เนื่องจากความแตกต่างกันระหว่างการทำงานของอุปกรณ์ประมวลผลและสมองมนุษย์ ที่แม้อุปกรณ์ประมวลผลจะมีความสามารถและความเร็วในการประมวลผลทางคณิตศาสตร์สูงกว่าสมองของมนุษย์มาก ดังจะเห็นได้ง่ายๆ จากการบวกเลข 20 หลักเข้าด้วยกัน จะพบว่าคอมพิวเตอร์ที่มีอยู่ในปัจจุบัน หรือแม้กระทั่งเครื่องคิดเลขธรรมดาๆ ทั่วไป จะสามารถทำงานดังกล่าวได้โดยใช้เวลาเพียงเศษเสี้ยววินาทีเท่านั้น ซึ่งต่างกับสมองของมนุษย์ซึ่งเป็นหน่วยประมวลอย่างง่ายๆ อย่างไรก็ตาม เนื่องจากหน่วยย่อยๆ เหล่านี้มีจำนวนมากมายมหาศาล และทำงานไปพร้อมๆ กัน (Parallel Processing) แทนที่จะทำงานทีละขั้นตอน (Serial Processing) ซึ่งเป็นวิธีการทำงานของอุปกรณ์ประมวลผลที่มีอยู่ในปัจจุบัน จึงทำให้ประสิทธิภาพการทำงานของสมองมนุษย์สูงกว่าอุปกรณ์ประมวลผลที่อยู่ในปัจจุบันเป็นอย่างมาก

แม้ว่า ด้วยเทคโนโลยีที่มีอยู่ในปัจจุบันจะไม่สามารถพัฒนาความสามารถ “มองเห็นและรับรู้” ของเครื่องจักรให้ได้เทียบเท่ากับของสมองมนุษย์ แต่ผลที่ได้จากการพัฒนาความรู้เรื่อง Machine Vision ก็สามารถนำไปใช้กับงานได้หลากหลายรูปแบบด้วยกัน เช่น การนำไปใช้กับระบบการมองเห็นหุ่นยนต์ ใช้กับระบบรักษาความปลอดภัย หรือแม้กระทั่งการประยุกต์ใช้ในเชิงอุตสาหกรรม เป็นต้น

2.2 ระบบตรวจสอบชิ้นงานด้วยภาพแบบอัตโนมัติ (Automated Visual Inspection System)

ระบบตรวจสอบชิ้นงานด้วยภาพแบบอัตโนมัติ (Automated Visual Inspection System) จัดเป็นการนำเอาความรู้เรื่อง Machine Vision ไปประยุกต์ใช้ในงานอุตสาหกรรมเพื่อใช้ตรวจสอบคุณภาพของผลิตภัณฑ์ ซึ่งมีหัวข้อในการตรวจสอบอยู่หลายๆ หัวข้อด้วยกัน เช่น การตรวจสอบการปนเปื้อนบนพื้นผิวของผลิตภัณฑ์ การผิดเพี้ยนของสีของผลิตภัณฑ์ การนับจำนวนชิ้นส่วนต่างๆ ที่อยู่บนผลิตภัณฑ์ เป็นต้น ซึ่งผลของการตรวจสอบดังกล่าวอาจจะใช้เพื่อคัดแยกงานดีออกจากงานเสีย หรือใช้เพื่อคัดเลือกเกรดของชิ้นงานก็ได้เช่นกัน และจากความต้องการจากด้านของอุตสาหกรรมที่ต้องการระบบที่สามารถทำงานได้รวดเร็วมากที่สุด ทำให้การทำงานของระบบอัตโนมัติ ซึ่งโดยหลักแล้วก็คือ การทำงานของโปรแกรมที่อยู่ในตัวอุปกรณ์ประมวลผล จะต้องใช้เวลาที่สั้นที่สุด ดังนั้นวิธีการคำนวณต่างๆ จะต้องใช้เวลาให้น้อยที่สุด นอกจากนั้น ความรู้ต่างๆ ที่มีอยู่ในที่ปฏิบัติงานกับผลิตภัณฑ์นั้นๆ มาก่อน จะต้องถูกนำมาใช้เพื่อ “ช่วย” ให้ระบบสามารถตัดการคำนวณที่ไม่จำเป็นออกไปให้ได้มากที่สุด และก่อนที่เราจะกล่าวล่วงไปถึงเนื้อหาในส่วนดังกล่าว ผู้วิจัยจะขอกล่าวถึง ส่วนประกอบของระบบตรวจสอบชิ้นงานด้วยภาพแบบอัตโนมัติ ซึ่งเป็นส่วนประกอบของโดยทั่วไปของระบบ Machine Vision และนำมาใช้และพัฒนากับโปรแกรมที่ผู้วิจัยได้ทำขึ้นมา ดังในภาพที่ 2-1



ภาพที่ 2-1 แสดงส่วนประกอบของระบบตรวจสอบชิ้นงานจากภาพอัตโนมัติ

2.3 การจัดสภาพแวดล้อม (Scene Constraint)

จุดมุ่งหมายหลักของการจัดสภาพแวดล้อมของระบบตรวจสอบชิ้นงานด้วยภาพแบบอัตโนมัติ คือ เพื่อลดความซับซ้อนในการประมวลผลให้มากที่สุด ทั้งนี้ก็เนื่องจากการที่ความสามารถในการ “มองเห็นและรับรู้” ของอุปกรณ์ประมวลผลที่มีอยู่ในปัจจุบันมีอยู่อย่างจำกัด และไม่เทียบเท่ากับความสามารถของมนุษย์ เราจึงต้อง “ช่วย” ลดความยุ่งยากของประมวลผล ทั้งนี้ก็เพื่อให้อุปกรณ์ประมวลผลใช้เวลาส่วนใหญ่ไปกับงานที่ไม่ซับซ้อนและเท่าที่มีความจำเป็นเท่านั้น ซึ่งเราจะสามารถทำได้หลายๆ วิธีร่วมกัน ยกตัวอย่างเช่น

2.3.1 การจัดการกับชิ้นงาน ในสภาพแวดล้อมจริงในโรงงานอุตสาหกรรม ชิ้นงานแต่ละชิ้นที่จะถูกป้อนให้กับระบบตรวจสอบจะต้องถูกจัดให้วางตัวในทิศทางเดียวกัน ซึ่งหากไม่มีการจัดการเกี่ยวกับการวางตัวของชิ้นงานเหล่านี้แล้วอุปกรณ์ประมวลผลจะต้องหาทิศทางของชิ้นงานแต่ละชิ้นเอง ก่อนที่จะเริ่มทำการการตรวจสอบชิ้นงานจริงๆ นอกจากนั้น ประเภทของผลิตภัณฑ์ที่ให้ระบบอัตโนมัติทำการตรวจสอบนั้นจะต้องถูกจำกัดด้วยเช่นกัน ยกตัวอย่าง เช่น ในสายการผลิตหนึ่ง ผลิตภัณฑ์ที่จะสามารถใช้ระบบตรวจสอบด้วยภาพแบบอัตโนมัติจะถูกกำหนดรุ่นไว้อย่างแน่นอน ไม่สามารถนำผลิตภัณฑ์รุ่นอื่นมาตรวจสอบกับระบบตรวจสอบชิ้นส่วนที่ออกแบบไว้ได้

2.3.2 ระยะระหว่างกล้องหรือเลนส์ถึงวัตถุ และทิศทางของกล้อง ตัวแปรเหล่านี้จะเป็นตัวกำหนดขนาดของชิ้นงานที่ระบบอัตโนมัติ “มองเห็น” เช่น หากระยะดังกล่าวสั้นลงแล้ว ชิ้นงานที่ระบบอัตโนมัติมองเห็นก็จะมีขนาดใหญ่ ดังนั้นสำหรับระบบตรวจสอบชิ้นงานด้วยภาพแบบอัตโนมัติโดยทั่วไปแล้ว ตัวแปรเหล่านี้จะต้องถูกกำหนดไว้ตายตัวมิฉะนั้นแล้ว การวัดขนาดของชิ้นส่วนต่างๆ ซึ่งจัดเป็นการตรวจสอบพื้นฐานของการตรวจสอบชิ้นงานก่อนจะทำการตรวจสอบในหัวข้ออื่นๆ ก็จะผิดเพี้ยนไป

2.3.3 การจัดการเรื่องแสง แสงจัดเป็นองค์ประกอบสำคัญมาก เนื่องจากการมองเห็นภาพของระบบอัตโนมัตินั้น เกิดจากการที่มีแสงมาตกกระทบวัตถุ แล้วสะท้อนผ่านเลนส์มาเข้าตัวเซนเซอร์รับภาพของกล้องที่ใช้กับระบบอัตโนมัติ ซึ่งการจัดการเกี่ยวกับแสงนั้น จำเป็นจะต้องพิจารณาทั้งเรื่องการเลือกใช้แหล่งกำเนิดแสง การกระเจิงของแสง และคุณสมบัติอื่นๆ สำหรับการตรวจสอบชิ้นงานโดยทั่วไปแล้ว จะทำการติดตั้งแหล่งกำเนิดแสงไว้ที่ด้านเดียวกับตัวกล้องแล้วส่องไปที่วัตถุที่ต้องการจับภาพ ซึ่งเรียกกันว่า Front Lighting อย่างไรก็ตามจะพบว่าสำหรับงานตรวจสอบชิ้นงานบางอย่าง อาจจำเป็นจะต้องใช้การส่องแสงมาจากด้านหลังของวัตถุที่ใช้พิจารณาที่เรียกกันว่า Back Lighting แล้วใช้ภาพที่ได้ทั้ง 2 ภาพมาประมวลผลเพื่อตรวจสอบวัตถุ ยกตัวอย่างเช่น งานวิจัยเพื่อหาอัตราการปลอมปนข้าวราคาต่ำในข้าวหอมมะลิที่ชื่อว่า โครงการการประเมินคุณภาพของเมล็ดข้าวโดยใช้ระบบวิเคราะห์ทางภาพ (Image Processing for Rice Kernel Quality Evaluation)[2]

นอกไปจากการจัดสภาพแวดล้อมในการทำงานให้กับระบบตรวจสอบชิ้นส่วนจากภาพแบบอัตโนมัติแล้ว งานบางประเภทอาจจะต้องมีการใช้ภาพจากกล้องหลายๆ ตัวเพื่อใช้ตรวจสอบชิ้นงานจากหลายๆ มุมมอง บางกรณีอาจจะเป็นการใช้กล้องเพียงตัวเดียว แต่ตัวกล้องสามารถเคลื่อนที่ไปตามส่วนต่างๆ ของชิ้นงานได้ และสำหรับบางกรณีอาจจะมีเก็บภาพของวัตถุเดียวกัน ที่ได้จากแหล่งกำเนิดแสงหลายๆ แหล่ง หลายๆ ประเภทก็เป็นได้

2.4 การดึงข้อมูลภาพ (Image Acquisition)

หากจะกล่าวอย่างง่ายๆ แล้วกระบวนการดึงข้อมูลภาพ คือ กระบวนการที่เริ่มตั้งแต่การถ่ายภาพโดยกล้อง ตลอดจนถึงการดึงภาพซึ่งเป็นข้อมูลที่อยู่ในกล้องมาสู่คอมพิวเตอร์ หรืออุปกรณ์ประมวลผล เพื่อที่จะได้ประมวลผลและตัดสินใจสั่งงานจากผลที่ได้ต่อไป กระบวนการดังกล่าวมีรายละเอียดปลีกย่อยที่สำคัญดังนี้

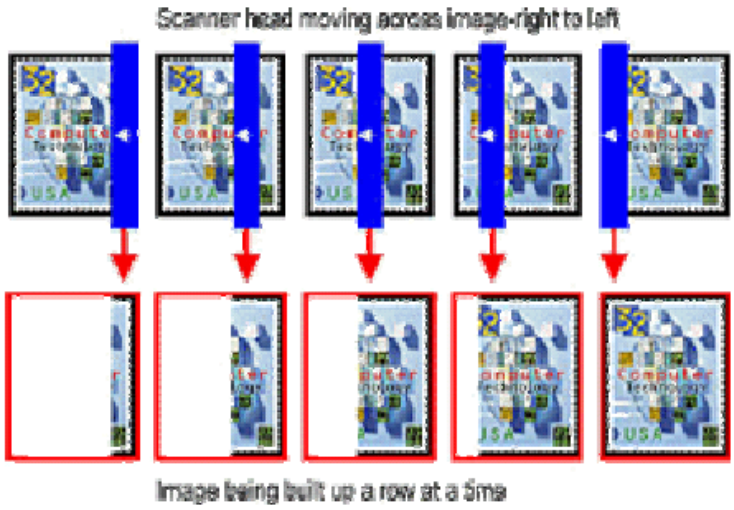
2.4.1 ประเภทของกล้องที่ใช้ในงานตรวจสอบชิ้นส่วนในเชิงอุตสาหกรรม กล้องที่ใช้กับงานตรวจสอบชิ้นงานในปัจจุบันนั้น จะเป็นกล้องดิจิตอล ซึ่งใช้อุปกรณ์สารกึ่งตัวนำที่เรียกกันว่า เซ็นเซอร์รับภาพ (Image Sensor) เพื่อใช้ในการรับภาพ เซ็นเซอร์ดังกล่าวมีขนาดเล็กมากเท่าเล็บมือคนเท่านั้น ซึ่งจะประกอบด้วยไดโอดที่มีความไวต่อแสงเรียงตัวกันอยู่เป็นจำนวนมาก และในทันทีทันใดที่แสงมีการตกกระทบไดโอดเหล่านี้ ไดโอดแต่ละตัวจะทำการจดจำความเข้มแสงหรือความสว่างของแสงที่ตกกระทบไดโอดแต่ละตัวไว้ โดยปริมาณประจุไฟฟ้าที่สะสมอยู่ในตัวไดโอด ซึ่งแปรผันกับแรงดันตกคร่อมตัวไดโอดนั้น จะเพิ่มขึ้นตามความเข้มของแสงที่ตกกระทบ ซึ่งความเข้มแสงที่ได้จดจำไว้ในไดโอดแต่ละตัว จะถูกแปลงให้อยู่ในรูปข้อมูลที่เป็นดิจิตอลและเก็บไว้ในหน่วยความจำที่อยู่ในตัวกล้อง เพื่อรอส่งต่อให้อุปกรณ์ที่อยู่ภายนอกกล้องต่อไป

สำหรับไดโอดนี้เรียกกันว่า เซลล์รับภาพ (พิกเซล Pixel) ซึ่งหนึ่งเซลล์รับภาพจะให้ค่าความเข้มแสงที่ตกกระทบเพียงค่าหนึ่งเท่านั้น โดยทั่วไปค่าที่ได้จากเซลล์รับภาพจะมีค่าระหว่าง 0-255 เท่านั้น (ช่วงข้อมูลดังกล่าว สามารถแทนด้วยข้อมูลขนาด 1 Byte หรือ 8 บิต ที่จะให้ความละเอียด 2^8 หรือ 256 ระดับ ซึ่งเป็นความละเอียดของกล้องที่สามารถพบเห็นได้ทั่วไปในท้องตลาดอย่างไรก็ตาม จะมีกล้องบางประเภทที่ให้ค่าความเข้มแสงที่มีความละเอียดสูงถึง 16 บิตเลยทีเดียว) โดยหากค่าที่ได้มีค่าเท่ากับ 0 แสดงว่าที่เซลล์รับภาพนั้น มีความเข้มแสงต่ำสุดหรือเป็นด้านมืด และหากมีค่าเท่ากับ 255 ก็แสดงว่าที่เซลล์รับภาพที่ตำแหน่งนั้น มีความเข้มแสงสูงสุดหรือเป็นด้านสว่าง ดังแสดงไว้ในภาพที่ 2-2

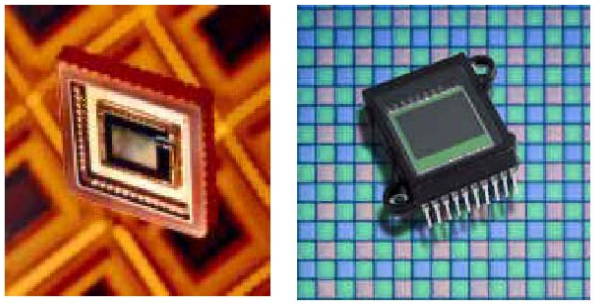
ภาพที่ 2-2 แสดงความเข้มแสงเทียบกับค่าที่ได้จากเซลล์รับภาพ

เมื่อพิจารณากระบวนการทั้งหมด จะพบว่าเซลล์รับภาพแต่ละเซลล์นั้นจะให้ค่าความเข้มแสงออกมาเป็นตัวเลขที่อยู่ระหว่าง 0 ถึง 255 เท่านั้น ซึ่งทำให้ได้ภาพที่เป็นโทนขาวดำหรือเรียกว่า Gray Scale Image เท่านั้น สำหรับกล้องถ่ายภาพสีนั้น ยังใช้เซลล์รับภาพเหล่านี้เช่นกัน โดยทำการแยกสีหลัก 3 สี ได้แก่ สีแดง เขียว และน้ำเงินออกจากกัน โดยการติดตั้งตัวกรองแสงสี (Filter) แต่ละสีไว้หน้าเซลล์รับภาพแล้วทำการวัดความเข้มของแต่ละสีนั่นเอง

สำหรับกล้องสำหรับงานตรวจสอบชิ้นส่วนนั้น แบ่งออกเป็น 2 ประเภท คือ กล้องประเภท Area Scan และกล้องประเภท Line Scan ข้อแตกต่างของกล้องทั้ง 2 ประเภทคือ กล้องประเภท Line Scan นั้นเซลล์รับภาพสำหรับรับความเข้มแสง (สำหรับกล้องที่ให้ภาพออกมาเป็น Gray Scale) หรือความเข้มสี (ในกรณีที่เป็นกล้องที่ใช้ถ่ายภาพสี) จะเรียงตัวเป็นแถวยาวที่อาจจะมีจำนวนมากถึง 12000 เซลล์ ทำให้การที่จะสามารถจับภาพของทั้งวัตถุได้ กล้องจะต้องมีการเคลื่อนที่สัมพันธ์กับวัตถุ ดังแสดงไว้ในภาพที่ 2-3 ซึ่งโดยทั่วไปแล้ว จะออกแบบให้กล้องตรึงอยู่กับที่ และตัววัตถุถูกเลื่อนไปโดยการใช้ระบบสายพาน (Conveyer) ข้อดีของกล้องประเภทนี้คือ จะให้ความละเอียดของภาพสูงมาก อย่างไรก็ตามสำหรับงานตรวจสอบชิ้นส่วนด้วยภาพแบบอัตโนมัติโดยทั่วไป จะใช้กล้องประเภท Area Scan ที่เซลล์รับภาพมีการเรียงตัวกันอยู่ในพื้นที่ ซึ่งโดยมากเป็นรูปสี่เหลี่ยมผืนผ้า ซึ่งเซลล์แต่ละเซลล์จะทำการแปลงค่าความเข้มแสงหรือความเข้มสีให้ออกมาเป็นค่าตัวเลขในเวลาพร้อมๆ กัน และถึงแม้ว่า กล้องชนิดนี้จะให้ภาพที่มีความละเอียดน้อยกว่าของกล้องประเภท Line Scan เป็นอย่างมาก แต่กล้องประเภทนี้สามารถนำไปใช้ได้อย่างสะดวกง่ายดาย โดยไม่จำเป็นต้องออกแบบการเคลื่อนไหวสัมพัทธ์ระหว่างตัวกล้องกับชิ้นงาน จึงทำให้กล้องชนิดนี้เป็นที่นิยมใช้กันอย่างกว้างขวางในปัจจุบัน ตัวอย่างของเซ็นเซอร์รับภาพของกล้องแบบ Area Scan นั้นแสดงไว้ในภาพที่ 2-4



ภาพที่ 2-3 แสดงการทำงานของกล้องประเภท Line Scan ซึ่งตัวกล้องและวัตถุจะต้องมีการเคลื่อนที่สัมพัทธ์กัน



ภาพที่ 2-4 แสดงเซ็นเซอร์รับภาพของกล้องประเภท Area Scan ของกล้องถ่ายภาพ Gray Scale และกล้องถ่ายภาพสี ซึ่งเซลล์รับภาพมีการเรียงตัวกันเต็มพื้นที่ของเซ็นเซอร์

จากภาพที่ 2-4 ซึ่งแสดงเซ็นเซอร์ที่ใช้กับกล้องถ่ายภาพสี สำหรับพื้นหลังของรูปดังกล่าวแสดงรูปแบบการเรียงตัวของเซลล์รับภาพที่วัดความเข้มสี จะพบว่า ตัวกรองแสงสีแดง เขียว และน้ำเงิน ได้ถูกติดเข้ากับเซลล์รับภาพแต่ละเซลล์ และมีการกระจายตัวอย่างเป็นระเบียบ ซึ่งเป็นวิธีการที่ทำให้เซลล์รับภาพซึ่งสามารถวัดได้เพียงความเข้มแสง สามารถนำมาใช้วัดความเข้มสีได้สำหรับเซ็นเซอร์ภาพบางรุ่นอาจจะมีกลไกการเปลี่ยนตัวกรองแสงสีที่อยู่ด้านหน้าเซลล์รับภาพ แทนที่จะติดตัวกรองแสงสีแบบตายตัวในลักษณะดังกล่าว

2.4.2 ภาพที่อุปกรณ์ประมวลผล “มองเห็น” หลักการทำงานของกล้องก็เป็นเช่นเดียวกับระบบการมองเห็นของมนุษย์ นั่นคือ ภาพเกิดจากการที่มีแสงตกกระบุวัตถุแล้วมีแสงสะท้อนจากวัตถุ ผ่านเลนส์เข้ามาตกกระทบเซ็นเซอร์รับภาพ (Image Sensor) ของกล้อง ซึ่งประกอบด้วยเซลล์รับภาพ (Pixel) จำนวนมาก เซลล์รับภาพแต่ละเซลล์จะทำหน้าที่แปลงความเข้มแสงสำหรับกรณีที่เป็นกล้องขาวดำหรือแปลงความเข้มสีของแสงสีแดง เขียวและน้ำเงินสำหรับกรณีของกล้องที่ใช้ถ่ายภาพสี ให้อยู่ในรูปของค่าสัญญาณแรงดันไฟฟ้า ซึ่งจะถูกแปลงไปเป็นสัญญาณดิจิตอลด้วยตัวแปลงสัญญาณอนาล็อกเป็นดิจิตอลอีกทีหนึ่ง อย่างไรก็ตามการทำงานของเซลล์รับภาพของกล้องจะแตกต่างจากเซลล์รับภาพของมนุษย์อยู่ 2 ประการด้วยกัน คือ

2.4.2.1 จำนวนเซลล์รับภาพที่ประกอบขึ้นมาเป็นเซ็นเซอร์รับภาพของกล้องนั้น มีจำนวนน้อยกว่าของมนุษย์เป็นอย่างมาก ทำให้ภาพที่ได้จากกล้องนั้นมีความละเอียดน้อยกว่าของมนุษย์เป็นอย่างมาก ภาพที่ได้จากกล้องจึงเกิดการสุ่มจับภาพจริงด้วยจำนวนที่จำกัดของเซลล์รับภาพ (Spatial Sampling) นั่นเอง

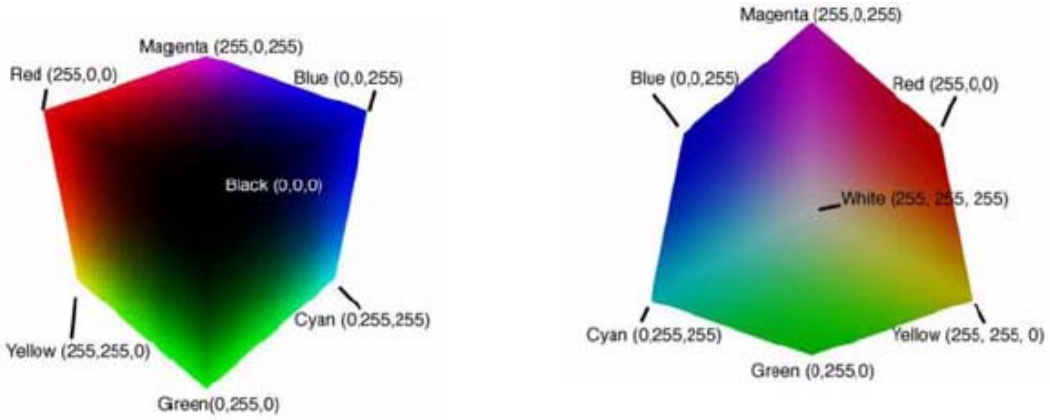
2.4.2.2 ค่าความเข้มแสงที่ได้จากเซลล์รับภาพ (หรือความเข้มสีในกรณีของกล้องถ่ายภาพสี) ของเซ็นเซอร์รับภาพที่อยู่ในกล้องนั้น จะเป็นค่าไม่ต่อเนื่อง (Discrete Value) เนื่องจากเป็นการทำงานของอุปกรณ์ดิจิตอล ซึ่งจะเป็นการสุ่มขนาดของความเข้มแสงที่ตกกระทบ(Amplitude Sampling) ไม่เหมือนกับของมนุษย์ที่มีความต่อเนื่อง เนื่องจากเป็นการทำงานของสารเคมีที่อยู่ในเซลล์รับภาพ

จากข้อจำกัดดังกล่าว จึงทำให้ข้อมูลของภาพ Gray Scale ที่อุปกรณ์ประมวลผลมองเห็นมีลักษณะเป็นอาร์เรย์ 2 มิติ โดยที่ค่าแต่ละช่องของอาร์เรย์จะแทนความเข้มแสงหรือความเข้มสีที่ตกกระทบเซลล์รับภาพที่ตำแหน่งนั้น ซึ่งค่าความเข้มแสงดังกล่าวจะเป็นค่าที่ไม่ต่อเนื่องและโดยทั่วไปมีค่าระหว่าง 0 ถึง 255 เท่านั้น ดังสมการที่ 2-1 ซึ่งเป็นตัวอย่างภาพที่มีจำนวนแถวหรือความสูงของภาพเท่ากับ m แถว และมีจำนวนหลักหรือความกว้างของภาพเท่ากับ n หลัก

$$\text{Image} = \begin{bmatrix} I(1,1) & I(1,2) & . & . & I(1,n) \\ I(2,1) & . & & & . \\ . & & . & & . \\ . & & & . & . \\ I(m,1) & I(m,2) & . & . & I(m,n) \end{bmatrix} \qquad (2\text{-}1)$$

เมื่อ (m,n) คือ ค่าความเข้มแสง ณ แถว m และหลัก n ของเซ็นเซอร์รับภาพซึ่งเป็นค่าไม่ต่อเนื่อง

สำหรับกรณีที่เป็นกล่องถ่ายภาพสีนั้น ข้อมูลของภาพจะเป็นอาร์เรย์ 2 มิติจำนวน 3 อาร์เรย์ และโดยทั่วไปแล้วอาร์เรย์เหล่านี้จะเก็บค่าความเข้มสีของสีแดง เขียว น้ำเงิน ตามลำดับ ซึ่งแต่ละช่องของอาร์เรย์เหล่านี้ก็จะมีค่าอยู่ระหว่าง 0 ถึง 255 เช่นกัน ดังนั้น การแทนสีที่เกิดขึ้นจริงตามธรรมชาติของอุปกรณ์ประมวลผล จะแทนด้วยค่าความเข้มสี ณ ตำแหน่งเดียวกันของอาร์เรย์ทั้งสามมาผสมกัน สำหรับตัวอย่างสีที่เกิดจากการผสมกันของสีทั้งสามนั้น แสดงไว้ในภาพที่ 2-5 จะพบว่า การผสมแสงสีดังกล่าวจะคล้ายคลึงกับวิชาศิลปะที่เป็นการผสมสีของแม่สีเข้าด้วยกันนั่นเอง อย่างไรก็ตาม การรวมกันของแสงสีที่อยู่ในอุปกรณ์ประมวลผลนั้นเรียกว่า Additive Color System ที่หากมีการรวมกันของแสงสีต่างๆ ด้วยค่าสูงสุดแล้ว จะให้แสงสีขาวออกมาในขณะที่การผสมสีในวิชาศิลปะนั้น เมื่อผสมแม่สีทั้งสามเข้าด้วยกันด้วยปริมาณที่เท่าๆ กัน จะได้สีดำออกมา ซึ่งเรียกระบบสีนี้ว่า Subtractive Color System



ภาพที่ 2-5 แสดงการแทนสีจริงด้วยค่าสีแดง สีเขียวและน้ำเงิน

เมื่อพิจารณาจากด้านสีดำ จะพบว่าค่าของสีทั้งสามเท่ากับ 0 และเมื่อพิจารณาจากด้านสีขาว จะพบว่าค่าของสีทั้งสามเท่ากับ 255 ในรูปแรกนั้น หากสีที่เซลล์รับภาพนั้นได้รับมาเป็นสีแดงล้วน ค่าความเข้มสีของอาร์เรย์สีแดง ณ ตำแหน่งดังกล่าว จะมีค่าสูงสุด คือ 255 ในขณะที่ ณ ที่ตำแหน่งเดียวกันของอาร์เรย์อื่นๆ ที่เหลือจะมีค่าเป็น 0 ซึ่งจะพบว่า แบบจำลองการแทนสีดังกล่าว มีลักษณะเป็นกล่องสี่เหลี่ยมด้านเท่า โดยที่แต่ละด้านจะมีความยาว 255 หน่วย และเรียกการแทนสีที่เกิดขึ้นจริงด้วยการผสมระหว่างสีแดง เขียว และน้ำเงินนี้ว่า RGB Color Space ซึ่งเป็นวิธีแทนสีที่เกิดขึ้นจริงของคอมพิวเตอร์หรืออุปกรณ์ประมวลผลโดยทั่วไป อย่างไรก็ตาม ในปัจจุบันมีวิธีการระบุสีที่เกิดขึ้นจริงในหลายรูปแบบ ซึ่งแต่ละรูปแบบก็เหมาะสมกับการใช้งานเฉพาะอย่าง

สำหรับในทางปฏิบัตินั้น หากพิจารณาจากแง่ของโปรแกรม การส่งข้อมูลภาพจากกล้องมาสู่คอมพิวเตอร์ จะไม่ได้อยู่ในรูปของอาร์เรย์ 2 มิติ แต่จะอยู่ในรูป Byte Stream ที่เป็นข้อมูลที่มีความต่อเนื่องเรียงต่อกัน โดยทั่วไปแล้ว ข้อมูลค่าแรกจะเป็นค่าความเข้มแสงของเซลล์รับภาพที่อยู่มุมล่างขวาของเซ็นเซอร์รับภาพในกรณีที่เป็นกล้องถ่ายภาพแบบ Gray Scale และสำหรับกรณีที่เป็นกล้องที่ถ่ายภาพสีนั้น ข้อมูลค่าแรกจะเป็นสีน้ำเงินของเซลล์รับภาพที่อยู่มุมล่างขวาของเซ็นเซอร์รับภาพ ซึ่งจะตามด้วยสีเขียวและสีแดงของจุดเดียวกันเป็นลำดับต่อเนื่องกันไป ดังนั้น โปรแกรมจะต้องทำการจัดเรียงข้อมูลที่มีความต่อเนื่องเหล่านี้ให้อยู่ในรูปอาร์เรย์ 2 มิติเสียก่อน ทั้งนี้เพื่อความสะดวกในการอ้างถึงตำแหน่งของข้อมูลของกระบวนการต่อไปนั่นเอง

2.5 การประมวลภาพเบื้องต้น (Pre – processing)

การประมวลผลภาพมีด้วยกันหลากหลายกระบวนการด้วยกัน กระบวนการเหล่านี้เป็นความรู้ที่สามารถพบได้ทั่วไปในสาขาเรื่อง การประมวลภาพดิจิตอล (Digital Image Processing)[3,4,5] ซึ่งจะพบว่าในสาขาวิชานี้มีความรู้เกี่ยวกับการประมวลผลภาพอยู่มากมายที่นำประยุกต์ใช้กับงานตรวจสอบชิ้นส่วนด้วยภาพแบบอัตโนมัติ เช่น

2.5.1 การลดทอนสัญญาณรบกวนที่ปรากฏขึ้นในภาพ

2.5.2 การตรวจจับขอบของวัตถุที่อยู่ในภาพ

2.5.3 การแปลงคุณสมบัติทางกายภาพของภาพ เช่น การหมุน การเลื่อน การย่อและขยายภาพ เป็นต้น

2.5.4 การแปลงสี (Color Space Conversion)

2.5.5 การวิเคราะห์ภาพในเชิงความถี่

2.5.6 การบีบอัดข้อมูลภาพ

2.5.7 และความรู้อื่นๆ อีกมากมาย

ซึ่งจะพบว่า ระเบียบวิธี (Algorithm) ของวิธีประมวลภาพบางอย่างก็ไม่เหมาะสมที่จะนำมาใช้กับงานตรวจสอบชิ้นส่วนแบบอัตโนมัติ เนื่องจากเป็นการนำไปใช้ในทางอุตสาหกรรมที่

ต้องการการทำงานที่รวดเร็วที่สุด ดังนั้น จึงมีความจำเป็นอย่างยิ่งที่จะต้องเลือกใช้ระเบียบวิธีเฉพาะที่ง่ายและใช้เวลาในการทำงานน้อยที่สุด ซึ่งเงื่อนไขเหล่านี้จะสามารถเป็นจริงได้ ก็ด้วยการจัดสภาพแวดล้อมในการจับภาพที่ดี นอกจากนั้นแล้ว ความรู้เกี่ยวกับผลิตภัณฑ์นั้นซึ่งมีอยู่แล้วในผู้ปฏิบัติงานก็ควรนำมาใช้เพื่อช่วยให้ระบบสามารถทำงานให้ได้เร็วที่สุดด้วยเช่นกัน ยกตัวอย่างเช่น

2.5.8 การระบุบริเวณที่ต้องการตรวจสอบ (Region of Interest, ROI) โดยผู้ปฏิบัติงาน รวมทั้งประเภทการตรวจสอบที่ใช้กับบริเวณนั้นๆ จะพบว่า การระบุเฉพาะบริเวณที่สนใจรวมทั้งรูปแบบการวัดหรือการตรวจสอบที่ใช้เฉพาะกับบริเวณนั้นๆ จะทำให้โปรแกรมสามารถตัดการคำนวณของบริเวณที่ไม่เกี่ยวข้องออกไป ทำให้โปรแกรมสามารถทำงานได้เร็วขึ้น ยกตัวอย่างเช่น การวิเคราะห์สีที่ผิดเพี้ยน ซึ่งจะต้องมีการแปลงระบบสี ซึ่งเป็นกระบวนการที่ใช้เวลาในการคำนวณค่อนข้างมาก หากทำการแปลงระบบสีของภาพทั้งภาพนั้นจะต้องใช้เวลาค่อนข้างมาก หากผู้ปฏิบัติงานช่วยบอกกับระบบว่า เฉพาะบริเวณใดบ้างที่ต้องการตรวจจับสีผิดเพี้ยน โปรแกรมก็จะสามารถทำการแปลงหรือคำนวณเฉพาะบริเวณที่ระบุไว้ โดยละบริเวณที่ไม่ได้ระบุไว้ ซึ่งจะทำการทำงานของระบบมีความรวดเร็วขึ้นมาก

2.5.9 การนำความรู้เรื่องรูปร่างของผลิตภัณฑ์มาใช้ เช่น การตรวจสอบบริเวณขอบของผลิตภัณฑ์ที่เป็นส่วนของเส้นตรงหรือการหารัศมีของชิ้นส่วนหรือบริเวณที่เป็นวงกลม จะพบว่าการตรวจสอบประเภทนี้จะต้องทำการตรวจจับส่วนของเส้นตรงหรือการตรวจจับวงกลมทั้งวงให้ได้เสียก่อนๆ ที่จะทำการพิจารณาคุณสมบัติอื่นๆ เช่น มุมของเส้นตรงหรือจุดศูนย์กลางของวงกลม เป็นต้น กระบวนการตรวจจับดังกล่าวสามารถทำได้หลายวิธีด้วยกัน ซึ่งวิธีที่เป็นที่นิยมกันเป็นอย่างมากในการตรวจจับเส้นตรงหรือวงกลม เรียกว่า Hough Transformation ซึ่งเป็นการหาสมการของเส้นตรงหรือสมการของวงกลมโดยไม่ต้องมีข้อมูลใดๆ มาช่วย วิธีการดังกล่าวเป็นวิธีการที่เป็นที่นิยมกันมากในการทำงานภายใต้สภาวะแวดล้อมแบบเปิด ที่มีสิ่งของที่ไม่ทราบรูปร่างมาก่อนเข้ามาในระบบ อย่างไรก็ตามสำหรับระบบตรวจสอบชิ้นส่วนด้วยภาพแบบอัตโนมัติที่เป็นการนำไปใช้ในสภาพแวดล้อมที่ถูกควบคุมไว้ อีกทั้งการที่รูปร่างของผลิตภัณฑ์และการวางตัวของผลิตภัณฑ์ที่ถูกกำหนดไว้ตายตัวอยู่แล้ว ทำให้เราสามารถใช้วิธีการธรรมดาๆ เช่น Linear Regression ซึ่งเป็นวิธีการที่ใช้เวลาในการคำนวณน้อยมาก เพื่อใช้หาสมการของเส้นตรงก็ได้เช่นกัน

2.6 การแยกบริเวณ (Segmentation)

หลังจากที่เราได้ทำการประมวลภาพเบื้องต้นเพื่อการทำการปรับปรุงคุณภาพของภาพไม่ว่าจะเป็นการลดสัญญาณรบกวน การเพิ่มความคมชัดของภาพ รวมทั้งการเน้นรายละเอียดเฉพาะส่วนที่มืดหรือเฉพาะส่วนที่สว่างแล้ว ในกระบวนการต่อไป เราจะทำการแยกบริเวณที่เป็นวัตถุที่เราสนใจออกจากพื้นหลัง ที่จะทำให้ทราบว่า ในภาพมีวัตถุอยู่กี่ชิ้น และพิกเซลใดเป็นของ

วัตถุชิ้นใด ซึ่งกระบวนการดังกล่าว ถือเป็นพื้นฐานของการประมวลขั้นสูง ที่จะนำไปสู่การตัดสินเกี่ยวกับคุณภาพของผลิตภัณฑ์ต่อไปและจะพบว่า วิธีการแยกบริเวณนั้นจะแบ่งออกเป็น 2 ประเภทหลักๆ ด้วยกัน คือ การแยกบริเวณด้วยการใช้ค่า Threshold หรือที่เรียกว่า Area Based Segmentation หรือ Region Based Segmentation และอีกวิธีหนึ่งคือ การแยกบริเวณด้วยขอบวัตถุที่ตรวจจับได้ด้วยตัวตรวจจับขอบ ซึ่งเรียกกันว่า Edge Based Segmentation และเนื่องจากในการตรวจสอบชิ้นงานด้วยภาพแบบอัตโนมัตินั้น จะเป็นการทำงานที่มีแสงกระจายตัวอยู่อย่างสม่ำเสมอ ทำให้ภาพที่ได้จะมีบริเวณที่เป็นวัตถุและพื้นหลัง ที่มีความเข้มแสงแตกต่างกันอย่างเห็นได้ชัดเจน ด้วยสาเหตุดังกล่าว จึงทำให้การแยกบริเวณด้วยวิธีแรกสามารถทำงานได้อย่างมีประสิทธิภาพสูงสุด นอกจากนั้นจะพบว่า ด้วยวิธีการแยกบริเวณวิธีแรกจะทำให้เรารู้บริเวณทั้งหมดของวัตถุแต่ละชิ้น (แทนที่จะรู้แค่เพียงขอบวัตถุเหมือนในกรณีที่แยกบริเวณโดยการใช้วิธี Edge Based Segmentation) ซึ่งข้อมูลที่ได้สามารถนำไปคำนวณลักษณะ (Feature) ต่างๆ ของวัตถุได้ง่ายกว่า ดังนั้น ในหัวข้อนี้จะมุ่งเน้นไปที่การแยกบริเวณด้วยวิธี Area Based Segmentation หรือการแยกด้วยค่า Threshold ซึ่งมีอยู่ 3 ขั้นตอนด้วยกัน คือ

1. การเลือกและการใช้ค่า Threshold กับภาพ Gray Scale ตั้งต้น
2. กระบวนการ Connected Component Labeling เพื่อจำแนกว่า พิกเซลใดเป็นของวัตถุชิ้นใด
3. การจัดเก็บพิกัดของพิกเซลที่เป็นของวัตถุชิ้นเดียวกัน

2.6.1 การเลือกและการใช้ค่า Threshold การใช้ค่า Threshold เพื่อแปลงสภาพ Gray Scale ให้เป็นภาพที่มีเพียง 2 ระดับ (Binary Image) นั้น ได้กล่าวถึงไว้บ้างแล้ว ในตอนที่กล่าวถึงองค์ประกอบต่างๆ ของระบบตรวจสอบชิ้นงาน กระบวนการดังกล่าวจัดเป็นขั้นตอนที่มีความสำคัญสำหรับเนื่องจากเป็นกระบวนการที่ง่าย และทำได้อย่างรวดเร็ว ซึ่งสามารถทำได้โดยการเปรียบเทียบค่าความเข้มแสงของภาพ Gray Scale ตั้งต้น ณ พิกเซลที่กำลังพิจารณากับค่าคงที่ค่าหนึ่ง ที่เรียกกันว่าค่า Threshold หากความเข้มแสงของภาพตั้งต้นน้อยกว่าค่า Threshold แล้ว ก็ให้ภาพขาออกที่ตำแหน่งเดียวกันเป็นจุดมืด และในทางกลับกันถ้าความเข้มแสงมากกว่าหรือเท่ากับค่า Threshold ก็ให้ภาพขาออกที่ตำแหน่งนั้นเป็นจุดสว่างหรือค่า 255 ซึ่งสามารถเขียนได้ดังสมการที่ (2-2)

$$O = \begin{cases} 0, & \text{if } I < T \\ 2^B - 1, & \text{otherwise} \end{cases} \qquad (2\text{-}2)$$

เมื่อ B คือ จำนวนบิตของระบบภาพ

I คือ ค่าความเข้มแสงของพิกเซล ณ ตำแหน่งที่กำลังพิจารณา

O คือ ค่าความเข้มแสงของภาพขาออกที่ตำแน่งเดียวกัน

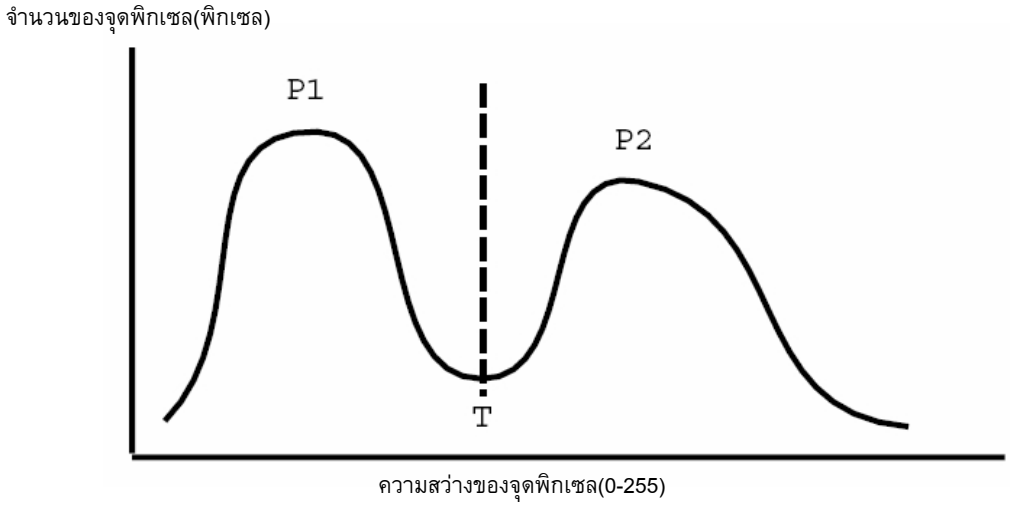
T คือ ค่า Threshold

จากสมการข้างต้นจะพบว่า กระบวนการดังกล่าวสามารถทำได้อย่างง่ายมาก โดยการใช้เพียงคำสั่ง for เพียง 2 คำสั่งและคำสั่ง if เพียง 1 คำสั่งเพื่อทำการเป็นเปรียบเทียบความเข้มแสงของภาพตั้งต้นกับค่า Threshold เท่านั้น คำถามที่ตามมาก็คือ แล้วค่า Threshold ที่เหมาะสมนั้น ควรมีค่าเท่าใดกันแน่ ก่อนที่จะไปถึงคำตอบของคำถามดังกล่าว เราควรจะพิจารณาวิธีการใช้ค่า Threshold ที่มีการให้คำจำกัดความไว้ในงาน Machine Vision ทั่วไป ซึ่งแบ่งออกเป็น 2 รูปแบบด้วยคือ

2.6.1.1 Global Thresholding คือ การใช้ค่า Threshold ค่าเดียวกับทั้งภาพ

2.6.1.2 Local Thresholding คือ การแบ่งภาพหลักออกเป็นภาพย่อยๆ ที่แต่ละภาพย่อยเหล่านั้นจะมีค่า Threshold เป็นของตัวเอง สำหรับการทำงานของระบบตรวจสอบชิ้นส่วนด้วยภาพแบบอัตโนมัติ ซึ่งแต่ละบริเวณจะถูกกำหนดค่าโดยผู้ใช้งาน

ไม่ว่าจะเป็นการใช้ค่า Threshold ในลักษณะใดก็ตาม ก็จะมีคำถามสำคัญคือ แล้วค่า Threshold ที่เหมาะสมกับภาพหรือภาพย่อยๆ เหล่านั้นควรมีค่าเท่าใด และเราจะสามารถเขียนโปรแกรมเพื่อคำนวณหาค่า Threshold อย่างอัตโนมัติได้อย่างไร เพื่ออย่างน้อยจะได้เป็นแนวทางให้ผู้ใช้ไว้กำหนดค่า Threshold จริงๆ ในแต่ละบริเวณได้หรือไม่ การเลือกค่า Threshold ที่เหมาะสมสำหรับแต่ละบริเวณย่อยๆ แบบอัตโนมัตินั้น โดยทั่วไปแล้ว ตั้งอยู่บนสมมุติฐานที่ว่าความเข้มแสงของบริเวณที่เป็นวัตถุที่สนใจ และบริเวณที่เป็นฉากหลังมีความแตกต่างกันพอประมาณ ซึ่งค่า Threshold ที่เลือกใช้นั้น จะต้องสามารถแบ่งฉาดหลังและวัตถุออกจากกันได้เป็นอย่างดี ดังแสดงไว้ในภาพที่ 2-6



ภาพที่ 2-6 แสดงฮิสโตแกรมที่วัตถุและพื้นหลังมีค่าความเข้มแสงแยกออกจากกัน

จากภาพที่ 2-6 ซึ่งแสดงฮิสโตแกรมที่มี 2 ยอด (Bimodal Histogram) ทั้งนี้ขึ้นอยู่กับลักษณะงานแต่ละประเภทว่า ส่วนที่เป็นวัตถุนั้นจะเป็นด้านมืด (P_1) หรือด้านสว่าง (P_2) ซึ่งค่า Threshold ที่เหมาะสมนั้น จะต้องสามารถแบ่งแยกบริเวณที่เป็นวัตถุและบริเวณที่เป็นพื้นหลังได้อย่างถูกต้อง ในปัจจุบันมีวิธีการเลือกค่า Threshold ที่มีสมมุติฐานว่าฮิสโตแกรมมี 2 ยอดอยู่หลากหลายวิธี แต่วิธีที่เป็นที่นิยมมากที่สุด อีกทั้งยังนำมาใช้ใน MATLAB ด้วย คือวิธีการของ Otsu (Otsu's Thresholding Method)[6]

หลักการเลือกค่า Threshold ของ Otsu นั้นคือ จะต้องเป็นค่าที่สามารถทำให้ฮิสโตแกรมทั้งสองกลุ่มมีการ "กระจายตัว" น้อยที่สุด ซึ่งในทางปฏิบัติเราไม่สามารถทำการเปลี่ยนรูปร่างของฮิสโตแกรมทั้งสองยอดได้ แต่เราสามารถเปลี่ยนลักษณะการกระจายตัวของทั้งสองยอดได้ด้วยการใช้ค่า Threshold เป็นตัวแบ่ง นั่นคือ ถ้าเราเพิ่มค่าดังกล่าว เรากำลังทำให้การกระจายตัวของยอดหนึ่งลดลง และการกระจายตัวของอีกยอดหนึ่งเพิ่มขึ้น ซึ่งเป้าหมายของ Otsu คือการเลือกค่า Threshold ที่ทำให้ "การกระจายตัวรวม" ของทั้งสองยอดมีค่าต่ำที่สุด

การกระจายตัวรวมของทั้งสองยอดนั้น สามารถวัดได้โดยความแปรปรวนภายในกลุ่มรวมกัน (Within – Class Variance) ซึ่งมีค่าเท่ากับผลรวมของความแปรปรวน (Variance) คูณกับจำนวนพิกเซลของแต่ละกลุ่ม และสมการทางคณิตศาสตร์ที่ใช้วัดการกระจายตัวรวมของทั้งสองกลุ่มนั้น แสดงไว้ในสมการที่ (2-3)

$$\sigma^2_{within}(T) = n_D(T)\sigma^2_D(T) + n_B(T)\sigma^2_B(T) \qquad (2\text{-}3)$$

เมื่อ T คือ ค่า Threshold ที่ใช้แบ่งทั้งสองบริเวณออกจากกัน

n_D (T) คือ จำนวนพิกเซลทั้งหมดของบริเวณด้านมืด (Dark Area) ที่มีค่าความเข้มแสงตั้งแต่ 0 จนถึงค่าความเข้มแสงเท่ากับ T-1 ซึ่งสามารถคำนวณได้จากสมการที่ (2-4)

$$n_D(T) = \sum_{i=0}^{T-1} p(i) \qquad (2\text{-}4)$$

เมื่อ n_B (T) คือ จำนวนพิกเซลทั้งหมดของด้านสว่าง (Blight Area) ที่มีค่าความเข้มแสงตั้งแต่ T จนถึงค่าความเข้มแสงเท่ากับค่าสูงสุดคือ 2^B-1 เมื่อ B คือ จำนวนบิตของระบบภาพ ซึ่งถ้าเป็นระบบทั่วไปที่เป็นระบบภาพ 8 บิต พจน์ 2^B-1 จะมีค่าเท่ากับ 255 และจำนวนพิกเซลทั้งหมดของด้านสว่างสามารถคำนวณได้จากสมการที่ (2-5)

$$n_B(T) = \sum_{i=T}^{2^B-1} p(i) \qquad (2\text{-}5)$$

$\sigma_D(T)$ คือ ความแปรปรวน (Variance) ของบริเวณด้านมืด

$\sigma_B(T)$ คือ ความแปรปรวน (Variance) ของบริเวณด้านสว่าง

เราสามารถใช้สมการที่ (2-3) เพื่อหาค่า Threshold ที่เหมาะสมได้ โดยการเลือกค่า Threshold ที่ทำให้พจน์ดังกล่าวมีค่าน้อยที่สุดอย่างไรก็ตาม การคำนวณสมการที่ (2-3) กับทุกค่า Threshold ที่เป็นไปได้นั้นมีความยุ่งยากมาก เนื่องจากจะต้องคำนวณความแปรปรวนของแต่ละบริเวณ ทั้งบริเวณที่มืดและสว่างของ Threshold ทุกค่า ซึ่งเราสามารถเลือกค่า Threshold ที่เหมาะสมได้ด้วยวิธีการที่ง่ายกว่านี้ นั่นคือ ถ้าเรานำค่าความแปรปรวนภายในกลุ่มรวมกันมาลบออกจากค่าความแปรปรวนรวม เราจะได้พจน์ที่ Otsu เรียกว่า ความแปรปรวนระหว่างกลุ่ม (Between – Class Variance, $\sigma^2_{Between}$) ซึ่งสามารถคำนวณได้จากสมการที่ (2-6)

$$\sigma^2_{Between} = \sigma^2 - \sigma^2_{within} \quad (2\text{-}6)$$

$$\sigma^2_{Between} = n_D(T)[\, u_B(T) - \mu]^2 + n_B(T)[\, u_B(T) - \mu]^2 \quad (2\text{-}7)$$

เมื่อ $\sigma^2_{Between}$ คือ ความแปรปรวนรวมของทั้งฮิสโตแกรม

μ คือ ค่าเฉลี่ยรวมของทั้งฮิสโตแกรม

จากสมการข้างต้นจะสังเกตเห็นว่า ความแปรปรวนระหว่างกลุ่ม (Between – Class Variance, $\sigma^2_{Between}$) คือ ผลบวกถ่วงน้ำหนักของผลต่างระหว่างค่าเฉลี่ยของแต่ละบริเวณกับค่าเฉลี่ยรวมของทั้งฮิสโตแกรม ซึ่งค่าเฉลี่ยของทั้งฮิสโตแกรมก็คือ ผลบวกถ่วงน้ำหนักของค่าเฉลี่ยของแต่ละบริเวณ และสามารถเขียนได้ดังสมการที่ (2-8)

$$\mu = n_D(T)\, u_B(T) + n_B(T) u_B(T) \quad (2\text{-}8)$$

เมื่อแทนค่าของสมการที่ (2-8) ลงไปในสมการที่ (2-7) แล้วทำการจัดพจน์ใหม่จะได้ว่าเราสามารถคำนวณความแปรปรวนระหว่างกลุ่ม (Between – Class Variance, $\sigma^2_{Between}$) ได้ง่ายขึ้น ดังสมการที่ (2-9)

$$\sigma^2_{Between} = n_D(T)\, n_B(T)[\, u_D(T) - u_B(T)]^2 \quad (2\text{-}9)$$

เมื่อเปรียบเทียบอย่างง่ายๆ ระหว่างสมการที่ (2-9) และสมการที่ (2-3) ซึ่งเป็นการคำนวณค่าความแปรปรวนระหว่างกลุ่มและการคำนวณความแปรปรวนภายในกลุ่มรวมกัน ตามลำดับ จะพบว่า สมการทั้งสองสามารถนำไปใช้หาค่า Threshold ได้อย่างอัตโนมัติ และให้ค่า

Threshold ที่เท่ากัน จะพบว่า สมการที่ (2-3) เป็นการคำนวณที่เปรียบเสมือนดัชนีวัดการกระจายตัวของแต่ละบริเวณรวมกัน ซึ่งถ้ามีค่าสูงแสดงว่า สมาชิกที่อยู่ในแต่ละบริเวณจะมีการกระจายตัวออกห่างไปจากค่าเฉลี่ยมาก ซึ่งไม่เป็นการดี เนื่องจากความเข้มแสงของบริเวณเดียวกันควรจะใกล้เคียงกันให้มากที่สุด ดังนั้นในการใช้สมการที่ (2-3) เพื่อหาค่า Threshold แบบอัตโนมัตินั้น จะต้องเลือกค่า Threshold ที่ทำให้ผลการคำนวณตามสมการที่ (2-3) มีค่าน้อยที่สุด ในขณะที่ สมการที่ (2-9) ซึ่งเป็นการคำนวณที่เปรียบเทียบเสมือนการวัดระยะห่างในฮิสโตแกรมระหว่าง 2 บริเวณหรือ 2 ยอด ซึ่งหากค่าที่ได้มีค่าสูงเท่าใดก็ยิ่งดีเท่านั้น ดังนั้นในการใช้สมการที่ (2-9) เพื่อหาค่า Threshold แบบอัตโนมัตินั้น จะต้องเลือกค่า Threshold ที่ทำให้ผลการคำนวณตามสมการที่ (2-9) มีค่ามากที่สุด

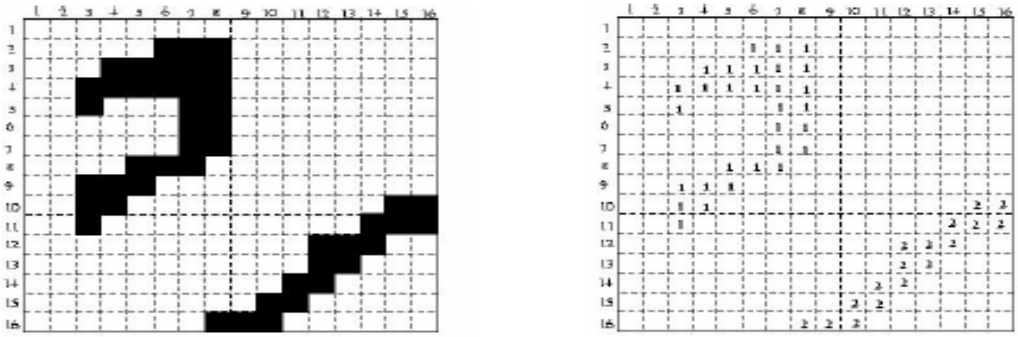
อย่างไรก็ตาม เมื่อเปรียบเทียบความซับซ้อนในการคำนวณนั้น จะพบว่า เราสามารถคำนวณค่าตามสมการที่ (2-9) ได้ง่ายกว่า ดังนั้น สมการที่ (2-9) จึงเป็นสมการสุดท้าย ที่เราจะนำไปเขียนโปรแกรมเพื่อหาค่า Threshold แบบอัตโนมัติตามวิธีของ Otsu ซึ่งมีขั้นตอนการคำนวณ ดังนี้

1. คำนวณฮิสโตแกรม และแยกกลุ่มของพิกเซลออกเป็น 2 กลุ่มโดยใช้ค่า Threshold, T
2. หาค่าความเข้มแสงเฉลี่ยของพิกเซลทั้งสองกลุ่ม
3. หาค่ากำลังสองของผลต่างของค่าเฉลี่ยของทั้งสองบริเวณ
4. คูณผลที่ได้จากขั้นตอนที่ 3 ด้วยผลคูณระหว่างจำนวนพิกเซลของทั้งสองบริเวณ
5. เลือกค่า Threshold ที่ทำให้ผลการคำนวณในข้อ 4 มีค่ามากที่สุด

2.6.2 Connected Component Labeling กระบวนการนี้ จัดเป็นกระบวนการที่สำคัญที่สุดกระบวนการหนึ่งของการตรวจสอบชิ้นส่วนด้วยภาพแบบอัตโนมัติ หรือแม้แต่งาน Machine Vision ทั่วไป ซึ่งเป็นกระบวนการที่ใช้เพื่อหาว่า พิกเซลใดเป็นของวัตถุชิ้นใด อนึ่ง กระบวนการ Connected Component Labeling นั้นไม่ได้ถูกจำกัดที่การนำไปใช้กับภาพ Binary ที่ได้จากการแยกบริเวณโดยใช้ค่า Threshold หรือที่เรียกว่า Region – Based Segmentation เท่านั้น กระบวนการดังกล่าวยังสามารถนำไปใช้กับภาพ Binary ซึ่งเป็นผลที่ได้จากการแยกบริเวณโดยการใช้ขอบวัตถุหรือ Edge – Based Segmentation ได้ด้วยเช่นกัน ซึ่งถ้าเป็นกรณีแรก ผลของกระบวนการ Connected Component Labeling จะทำให้เราทราบว่าพิกเซลใดเป็นบริเวณของวัตถุใด และถ้าเป็นกรณีที่สอง กระบวนการนี้จะทำให้เราทราบว่า พิกเซลใดเป็นขอบวัตถุใด แต่เพื่อความสอดคล้องกับเนื้อหา ที่ได้มุ่งเน้นการแยกบริเวณโดยใช้ค่า Threshold มาตั้งแต่ต้น เนื่องด้วยสภาพแวดล้อมการทำงานที่เหมาะสมดังที่ได้แจ้งไปแล้ว โปรแกรมที่นำเสนอจะเป็นกระบวนการที่ทำกับภาพ Binary ที่ได้จากการการแยกบริเวณโดยการใช้ค่า Threshold เท่านั้น อย่างไรก็ตามขอให้ผู้สนใจพึงระลึกเสมอว่า กระบวนการนี้สามารถนำไปใช้ได้กับผลที่ได้จากการแยกบริเวณทั้งสองแบบ

การทำงานของกระบวนการนี้จัดเป็น “คอขวด (Bottleneck)” ของงานตรวจสอบชิ้นส่วน เนื่องจากใช้เวลานาน และต้องทำให้เสร็จก่อนที่จะทำกระบวนการอื่นๆ ต่อไป ซึ่งหากในภาพประกอบด้วยวัตถุเพียงชิ้นเดียวและพื้นหลังแล้ว ก็ไม่จำเป็นต้องทำกระบวนการนี้ อย่างไรก็ตาม สำหรับบางหัวข้อการตรวจสอบของผลิตภัณฑ์ เช่น การตรวจสอบการปนเปื้อนของพื้นผิวของผลิตภัณฑ์หรือการนับจำนวนชิ้นงานที่มีหลายๆ ชนิดคละกันไป ซึ่งมีรอยเปื้อนหรือจำนวนชิ้นของวัตถุที่ไม่ทราบจำนวน และแต่ละชิ้นมีขนาด ตำแหน่ง และรูปร่าง ที่ไม่ทราบมาก่อน ทำให้เราจำเป็นต้องใช้กระบวนการนี้เพื่อใช้วิเคราะห์วัตถุที่อยู่ในบริเวณที่สนใจนั้นว่า มีจำนวนเท่าใด และพิกเซลใดเป็นของวัตถุใด แต่ละชิ้นมีขนาดและคุณสมบัติต่างๆ เท่าใดบ้าง (ซึ่งจะรู้ได้จากกระบวนการ Features Extraction) และมีชิ้นใดบ้างที่ถือว่าเป็นรอยเปื้อนบนพื้นผิว หรือถ้าเป็นการนับชิ้นงานก็จะรู้ว่า แต่ละชนิดมีจำนวนเท่าใดบ้าง (ซึ่งจะรู้ได้จากกระบวนการจำแนก)

Connected Component Labeling นั้น จะทำการจัดให้พิกเซลที่ “เชื่อมต่อกัน” และมีค่าความเข้มแสงเดียวกัน เป็นพิกเซลของวัตถุชิ้นเดียวกัน กระบวนการดังกล่าว จะให้หมายเลขวัตถุที่พิกเซลแต่ละตำแหน่งจัดเป็นสมาชิกอยู่ ออกมาในรูปของเมตริกซ์ที่เรียกกันว่า Label Matrix ดังแสดงไว้ในภาพที่ 2-7 ซึ่งเป็นกรณีที่วัตถุที่สนใจเป็นสีดำ



ภาพที่ 2-7 แสดงการทำงานของกระบวนการย่อยที่ชื่อว่า Connected Components Labeling หรือ Connected Components Extraction

ซึ่งภาพแรกคือ Binary Image ที่ได้จากวิธี Global Thresholding ซึ่งในที่นี้ เป็นการพิจารณาวัตถุสีดำที่มีพื้นหลังเป็นสีขาว และภาพที่สองคือ ผลที่ได้จากการทำงานของ Connected Components Labeling ที่ทำให้เราทราบว่า พิกเซลในแต่ละตำแหน่งนั้นเป็นของวัตถุชิ้นใด ซึ่งในรูปนั้นจะมีวัตถุอยู่ 2 ชิ้นด้วยกัน คือวัตถุหมายเลข 1 และ 2

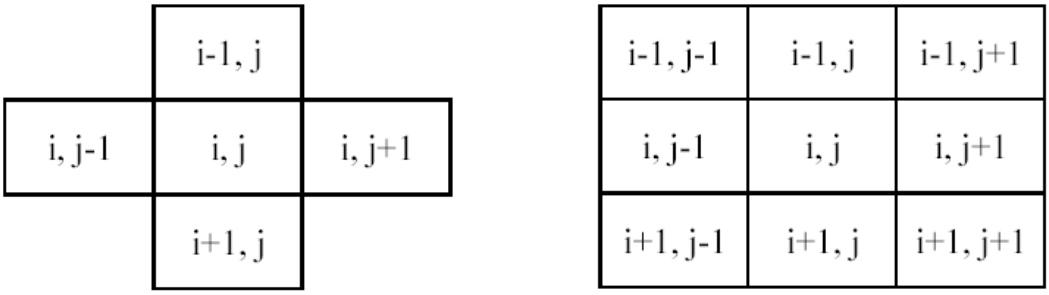
ภาพ Binary ซึ่งเป็นข้อมูลขาเข้าและ Label Matrix ที่เป็นผลลัพธ์หรือข้อมูลขาออกของกระบวนการ Connected Component Labeling นั้นจะมีขนาดเท่ากัน ทั้งจำนวนแถวและจำนวนหลัก ซึ่งสมาชิกหรือข้อมูลที่อยู่ในภาพ Binary นั้น จะมีได้แค่ 0 และ 255 เท่านั้น แต่

ข้อมูลที่อยู่ใน Label Matrix นั้น จะเป็นจำนวนเต็มหรือจำนวนนับตั้งแต่ 1 เรื่อยไป ซึ่งค่าที่อยู่ในแต่ละตำแหน่งของ Label Matrix นั้น จะเท่ากับหมายเลขวัตถุที่พิกเซลของภาพ Binary ในตำแหน่งเดียวกันจัดเป็นสมาชิกอยู่ และจากผลที่ได้จะสามารถนำไปคำนวณหาคุณสมบัติของวัตถุแต่ละชิ้นในขั้นตอน Feature Extraction ต่อไป และก่อนที่เราจะไปถึงระเบียบวิธีการคำนวณของกระบวนการ Connected Components Labeling นี้ เราจะมาพิจารณาลักษณะการเชื่อมต่อของพิกเซล ที่โดยทั่วไปแบ่งออกเป็น 2 ลักษณะ ดังนี้

2.6.2.1 การเชื่อมต่อแบบ 4 ทิศทาง (4 Connectivity) หากพิกเซลที่กำลังพิจารณานั้นอยู่ที่ตำแหน่ง i , j พิกเซลที่มีการเชื่อมต่อแบบ 4 ทิศทางกับพิกเซลดังกล่าว คือ พิกเซลที่มีการต่อเชื่อมอยู่ทั้ง 4 ทิศ คือ เหนือ ใต้ ตะวันออก ตะวันตก

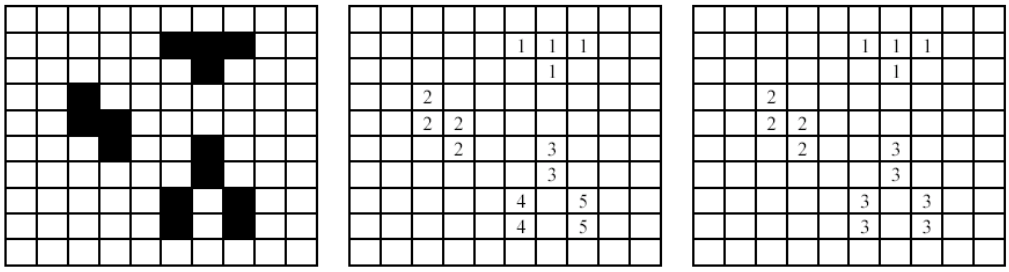
2.6.2.2 การเชื่อมต่อแบบ 8 ทิศทาง (8 Connectivity) หากพิกเซลที่กำลังพิจารณานั้นอยู่ที่ตำแหน่ง i , j พิกเซลที่มีการเชื่อมต่อแบบ 8 ทิศทาง คือ พิกเซลทั้งหมดที่ล้อมรอบพิกเซลที่กำลังพิจารณานั้นๆ อยู่

สำหรับรูปแบบการเชื่อมต่อทั้ง 2 รูปแบบ แสดงไว้ในภาพที่ 2-8



ภาพที่ 2-8 แสดงการเชื่อมต่อของพิกเซล i,j ที่มีการเชื่อมต่อแบบ 4 ทิศทาง (4 Connectivity) และที่มีการเชื่อมต่อแบบ 8 ทิศทาง (8 Connectivity)

กระบวนการ Connected Components Labeling นั้น เป็นการพิจารณาว่า พิกเซลที่เชื่อมต่ออยู่นั้นมีค่าความเข้มแสงเท่ากันหรือไม่ หากเท่ากันก็จะจัดว่าพิกเซลนั้นเป็นของวัตถุชิ้นเดียวกัน และกำหนดหมายเลขวัตถุลงไป Label Matrix ซึ่งการเลือกรูปแบบการเชื่อมต่อแบบ 4 ทิศทางหรือ 8 ทิศทางนั้น จะส่งผลต่อจำนวนวัตถุที่อยู่ในภาพ ดังในภาพที่ 2-9



ภาพที่ 2-9 แสดงจำนวนวัตถุที่นับได้ ซึ่งขึ้นอยู่กับรูปแบบการเชื่อมต่อที่ผู้ใช้เลือกใช้

รูปแรกคือ ภาพ Binary ซึ่งในกรณีนี้เป็นการพิจารณาบริเวณสีดำเป็นวัตถุ รูปที่สองคือ Label Matrix ที่ได้เมื่อเลือกใช้การเชื่อมต่อแบบ 4 ทิศทาง และรูปที่สามคือ Label Matrix ที่ได้เมื่อเลือกใช้การเชื่อมต่อแบบ 8 ทิศทาง

จากรูปที่สองของภาพที่ 2-9 ซึ่งแสดง Label Matrix ซึ่งเป็นผลที่ได้จากกระบวนการ Connected Components Labeling ที่เลือกใช้การเชื่อมต่อแบบ 4 ทิศทางนั้น จะพบว่า เราจะมีวัตถุอยู่ในภาพถึง 5 ชิ้นด้วยกัน เนื่องจากชิ้นที่ 3 4 และ 5 นั้น กระบวนจะไม่ถือว่าเป็นพิกเซลที่มีการเชื่อมต่อกันแบบ 4 ทิศทาง ซึ่งตรงกันข้ามกับรูปที่สามของภาพที่ 2-9 ที่เป็นการเลือกใช้การเชื่อมต่อแบบ 8 ทิศทาง ที่จำนวนวัตถุที่อยู่ในภาพนั้นมีเพียง 3 ชิ้นเท่านั้น อย่างไรก็ตาม เวลาที่ใช้สำหรับพิกเซลที่มีการเชื่อมต่อแบบ 8 ทิศทางนั้น จะยาวนานกว่ากรณีของพิกเซลที่มีการเชื่อมต่อแบบ 4 ทิศทาง ซึ่งการเลือกใช้การเชื่อมต่อแบบใดนั้น จะขึ้นอยู่กับความต้องการของงานแต่ละประเภท

2.6.3 ระเบียบวิธีการคำนวณของกระบวนการ Connected Components Labeling แบ่งออกเป็น 2 วิธี คือ วิธีวนทำซ้ำ (Recursive Algorithm) และวิธีทำทีละขั้นตอน (Sequential Algorithm) ซึ่งและวิธีมีรายละเอียดการทำงานดังนี้

2.6.3.1 วิธีวนทำซ้ำ (Recursive Algorithm) วิธีการวนทำซ้ำที่จะนำเสนอต่อไปนี้โดยทั่วไปมักจะไม่นำมาใช้ในคอมพิวเตอร์ทั่วๆ ไป เนื่องจากเป็นวิธีการที่ฟังค์ชั่นมีการเรียกตัวเองซ้ำๆ กัน (Recursive Function) ซึ่งเป็นวิธีการที่ไม่มีประสิทธิภาพในทางปฏิบัติ เนื่องจากการเรียกฟังค์ชั่นแต่ละครั้งนั้น จะต้องมีการเก็บค่าตัวแปรต่างๆ รวมทั้งตำแหน่งการทำงานปัจจุบันของโปรแกรมหลักเสียก่อน ก่อนที่จะกระโดดไปทำงานในฟังค์ชั่นที่เรียกใช้ อย่างไรก็ตาม วิธีวนทำซ้ำจะนำไปใช้กันมากกับหน่วยประมวลผลแบบขนาน (Parallel Processor) ซึ่งมีระเบียบวิธีคำนวณ (Algorithm) ดังนี้

ก) กำหนดให้ค่า Number ซึ่งเป็นจำนวนเต็มและเป็นจำนวนวัตถุที่อยู่ในภาพ มีค่าเริ่มต้นเท่ากับ 0 และให้ทุกตำแหน่งใน Label Matrix มีค่าเท่ากับ 0 หมดทุกตำแหน่ง

ข) ตรวจสอบภาพ Binary จากบนลงล่าง จากซ้ายไปขวาว่า มีพิกเซลใดที่มีค่าความเข้มแสงเท่ากับ 255 ซึ่งจัดเป็นวัตถุในภาพหรือไม่ ถ้ามีให้กำหนดค่า Label Matrix ที่ตำแหน่งพิกเซลเดียวกันให้เท่ากับ Label + 1

ค) ทำการตรวจสอบพิกเซลที่ “เชื่อมต่อ” กับพิกเซลที่กำลังพิจารณาว่า มีค่าเท่ากับ 255 หรือไม่ ถ้ามีก็ให้กำหนดค่า Label Matrix ที่ตำแหน่งนั้นให้เท่ากับ Label + 1

ง) วนทำซ้ำขั้นตอนในข้อที่ 3 ไปเรื่อยๆ จนกว่าพิกเซลที่เชื่อมต่อกับพิกเซลที่กำลังพิจารณา จะไม่มีพิกเซลใดเลยที่มีค่าเท่ากับ 255 แล้วไม่ถูกกำหนดหมายเลขวัตถุ

จ) เพิ่มค่า Label อีก 1 แล้ววนกลับไปที่ขั้นตอนที่ 2 จนกว่าจะทำหมดทุกแถวและทุกหลักของภาพ Binary ตั้งต้น

ฟังค์ชั่นของ MATLAB ที่ทำหน้าที่นี้คือ ฟังค์ชั่นที่ชื่อว่า bwlabel ซึ่งจะรับภาพ Binary ไปประมวลผลและจะคืนค่า Label Matrix ให้ ซึ่งสามารถนำไปปรับเปลี่ยนเป็นภาษา C++ ที่สามารถนำไปใช้ในทางปฏิบัติจริงได้

2.6.3.2 วิธีทำทีละขั้นตอน (Sequential Algorithm) เป็นกระบวนที่นำมาใช้ในการเขียนโปรแกรมสำหรับงานวิจัยนี้จริง กระบวนการ Connected Components Labeling โดยวิธีทำทีละขั้นตอนนั้น[7,8] โดยทั่วไปจะใช้ระเบียบวิธีคำนวณที่เรียกว่า Two Pass Algorithm ที่มีการตรวจสอบภาพ Binary จากบนลงล่าง จากซ้ายไปขวาถึง 2 รอบด้วยกัน โดยในรอบแรกจะทำการกำหนดหมายเลขวัตถุชั่วคราวให้กับแต่ละพิกเซลลงไปใน Label Matrix โดยพิจารณาจากหมายเลขวัตถุของพิกเซลที่เชื่อมติดกันอยู่ ยกตัวอย่าง เช่น ในภาพที่ 2-10 ที่มี x เป็นพิกเซลที่กำลังพิจารณาและยังไม่มีหมายเลขวัตถุ หากเลือกใช้การเชื่อมต่อแบบ 8 ทิศทางนั้น จะได้ว่า เซทของพิกเซลที่เชื่อมต่ออยู่กับพิกเซล x (เซท N) จะประกอบด้วยพิกเซล p, q, r และ s หรือจะได้ว่า N = {p, q, r และ s} หากเลือกใช้การเชื่อมต่อแบบ 4 ทิศทางนั้น เซทของพิกเซลที่เชื่อมต่ออยู่กับพิกเซล x (N) จะประกอบด้วยพิกเซล p และ q เท่านั้นหรือ N = {p, q}

ภาพที่ 2-10 พิกเซลที่ต้องพิจารณาหมายเลขวัตถุ (สมาชิกของเซท N) เมื่อต้องการกำหนดหมายเลขวัตถุให้กับพิกเซก x ในกรณีที่เลือกใช้การเชื่อมต่อแบบ 8 ทิศทาง และ 4 ทิศทาง

กำหนดให้ F คือ เซทของพิกเซลที่มีค่าความเข้มแสงเท่ากับ 255 ซึ่งจัดเป็นวัตถุหรือพื้นหน้า (Foregound)

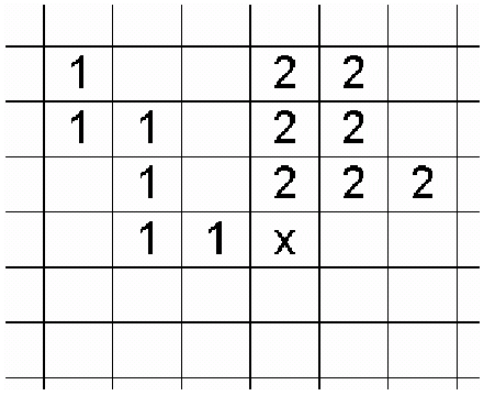
N_F คือ เซทของพิกเซลมีความเข้มแสงเท่ากับ 255 (เซท F) และอยู่ในกลุ่มของพิกเซลที่เชื่อมต่ออยู่กับพิกเซล x (เซท N) หรือจะได้ว่า $N_F = N \cap F$

การตรวจสอบภาพ Binary จากบนลงล่าง และจากซ้ายไปขวาในรอบแรก จะมีเงื่อนไขการตรวจสอบ และจะกำหนดหมายเลขชั่วคราวให้กับวัตถุดังนี้

ก) ถ้าพิกเซล x มีค่าความเข้มแสงเท่ากับ 255 ซึ่งจัดเป็นวัตถุ และ N_F คือ เซทว่าง นั่นคือในกลุ่มของพิกเซลที่เชื่อมต่อกับพิกเซล x นั้น ไม่มีพิกเซลใดเลยที่มีความเข้มแสงเท่ากับ 255 ให้ทำการกำหนดหมายเลขวัตถุใหม่ลงไปในตำแหน่งเดียวกันกับ x ใน Label Matrix ซึ่งในกรณีนี้ x จะเป็นพิกเซลแรกของวัตถุหมายเลขใหม่นั้นๆ

ข) ถ้าพิกเซล x มีค่าความเข้มแสงเท่ากับ 255 และหมายเลขวัตถุของทุกพิกเซลที่อยู่ในเซท N_F เท่ากับ L ให้กำหนดหมายเลขวัตถุของพิกเซล x เท่ากับ L ลงไปใน Label Matrix ด้วยเช่นกัน ในกรณีนี้หมายถึง พิกเซล x ก็จัดเป็นพิกเซลหนึ่งของวัตถุหมายเลข L นั่นเอง

ค) ถ้าพิกเซล x มีค่าความเข้มแสงเท่ากับ 255 และมี 2 พิกเซลใดในเซท N_F มีหมายเลขวัตถุต่างกัน ในกรณีเช่นนี้ หมายถึง มีพิกเซลที่เชื่อมต่อกับ x อยู่ 2 พิกเซล ซึ่งมีหมายเลขวัตถุชั่วคราวไม่เท่ากัน ดังแสดงไว้ในภาพที่ 2-11 ให้กำหนดหมายเลขวัตถุใน Label Matrix ของพิกเซล x ให้เท่ากับค่าใดค่าหนึ่งจากทั้งสองค่า และบันทึกหมายเลขวัตถุของทั้งสองบริเวณไว้ว่า เป็นบริเวณที่มีการเชื่อมต่อกัน โดยมีพิกเซล x เป็นจุดเชื่อม



ภาพที่ 2-11 แสดงกรณีที่มี 2 พิกเซลในเซท N_F (ซึ่งเป็นเซทที่ต่อเชื่อมกับพิกเซล x และมีความเข้มแสงเท่ากับ 255) มีหมายเลขวัตถุต่างกัน

จากการตรวจสอบภาพ Binary จากบนลงล่างจากซ้ายไปขวาในรอบแรกนั้น จะพบว่าด้วยเงื่อนไขการตรวจสอบข้อที่ 3 นั้น จะทำให้วัตถุชิ้นเดียวกันมีหมายเลขชั่วคราวของวัตถุต่างกันเกิดขึ้น ดังภาพที่ 2-10 ดังนั้น หลังจากที่จบการตรวจสอบในรอบแรกไปแล้วนั้น จะต้องทำการตรวจสอบครั้งที่ 2 เพื่อทำการแทนหมายเลขวัตถุชั่วคราวเหล่านั้นให้เป็นหมายเลขเดียวกัน ซึ่งจะพบว่าขั้นตอนการตรวจสอบในรอบที่สอง เพื่อเปลี่ยนหมายเลขชั่วคราวของวัตถุชิ้นเดียวกันให้เป็นหมายเลขเดียวกันนั้น ในปัจจุบันมีอยู่หลายวิธีด้วยกัน

2.6.4 การจัดเก็บพิกัดของพิกเซลที่เป็นของวัตถุชิ้นเดียวกัน ผลที่ได้จากกระบวนการก่อนหน้าคือ Label Matrix ที่ทำให้เราทราบว่า ในภาพมีวัตถุกี่ชิ้นและมีพิกเซลใดเป็นของวัตถุใด ซึ่งสามารถส่งผลต่อไปยังกระบวนการต่อไป เพื่อคำนวณลักษณะหรือคุณสมบัติของวัตถุแต่ละชิ้นที่ปรากฏอยู่ในภาพได้เลย เช่น พื้นที่ซึ่งเป็นจำนวนพิกเซลทั้งหมดของวัตถุนั้น หรือ เส้นรอบรูป (Perimeter) ซึ่งเป็นจำนวนพิกเซลของวัตถุเฉพาะที่มีการเชื่อมต่ออยู่กับพื้นหลัง (Background) เป็นต้น อย่างไรก็ตาม เมื่อมองให้แง่ของหน่วยความจำที่ใช้เก็บข้อมูลของ Label Matrix ซึ่งมีจำนวนแถวและจำนวนหลักเท่ากันกับภาพตั้งต้นนั้น จะพบว่า การเก็บพิกัดของพิกเซลทั้งหมดที่เป็นของวัตถุแต่ละชิ้นในรูปแบบของ Label Matrix เป็นการไม่ประหยัดพื้นที่หน่วยความจำของอุปกรณ์ประมวลผลเท่าใดนัก ดังนั้น โดยทั่วไปในทางปฏิบัติแล้ว กระบวนการ Connected Components Labeling นั้น จะทำควบคู่ไปพร้อมๆกันกับการเข้ารหัส เพื่อลดขนาดของข้อมูลที่เป็นพิกัดของพิกเซลที่เป็นของวัตถุชิ้นเดียวกัน นอกจากนั้นจะพบว่า การคำนวณคุณสมบัติพื้นฐานบางประการของวัตถุก็สามารถทำควบคู่ไปได้พร้อมๆกัน เช่นกัน

ดังนั้น อาจจะกล่าวได้ว่า เส้นแบ่งระหว่าง 3 กระบวนการ คือ กระบวนการ Connected Components Labeling การเข้ารหัสพิกัดของพิกเซลที่เป็นของวัตถุชิ้นเดียว และการคำนวณคุณสมบัติพื้นฐานของวัตถุแต่ละชิ้นนั้น เป็นเส้นแบ่งที่ไม่ชัดเจน เนื่องจากทั้ง 3 กระบวนการสามารถทำควบคู่ไปได้พร้อมๆกัน อย่างไรก็ตาม สาเหตุที่นำเสนอกระบวนทั้ง 3 แยกกันนั้น เนื่องจากผู้วิจัยต้องการลดความซับซ้อนของโปรแกรมตัวอย่าง เพื่อให้สามารถเข้าใจการทำงานของแต่ละกระบวนการย่อยได้ง่ายๆ และสามารถมองภาพการทำงานของแต่ละกระบวนการได้อย่างชัดเจน

การแยกบริเวณด้วยค่า Threshold หรือ Region Based Segmentation ที่ทำให้เราทราบบริเวณของวัตถุแต่ละชิ้นในภาพนั้น โดยทั่วไปจะใช้การเข้ารหัสแบบ Run – Length Encoding และหากเป็นแยกบริเวณด้วยขอบของวัตถุหรือ Edge – Based Segmentation ที่ให้เราทราบขอบของวัตถุนั้นจะใช้การเข้ารหัสของ Chain Code ซึ่งทั้ง 2 วิธีมีรายละเอียดดังนี้

2.6.4.1 การเข้ารหัสแบบ Run – Length Encoding (RLE) การเข้ารหัสลักษณะนี้จะเป็นการแยกบริเวณของวัตถุที่อยู่ในแต่ละแถวเป็นแท่งๆ ซึ่งเรียกกันว่า Run ซึ่งการระบุคุณสมบัติของแต่ละ Run ที่แยกออกมานั้นมี 2 วิธี วิธีแรกคือ ระบุหมายเลขวัตถุ แถว หลักที่เป็นจุดเริ่มและความยาวของแท่ง และวิธีที่สองคือ ระบุหมายเลขวัตถุ แถว หลักที่เป็นจุดเริ่มและหลักที่เป็นจุดสิ้นสุด ซึ่งเมื่อพิจารณาถึงความสะดวกในการโปรแกรมที่ใช้คำสั่ง for แล้ว โดยทั่วไปวิธีที่สองจะเป็นตัวเลือกที่ดีกว่า สำหรับตัวอย่างการเข้ารหัสแบบ Run – Length Encoding แบบที่ 2 นั้นแสดงไว้ในภาพที่ 2-12

R\C	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	0	0	0	0
2	0	1	1	1	0	0	0	2	2	0
3	0	0	0	0	0	0	0	0	0	0
4	0	3	3	3	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	4	4	4	0	0	0
7	0	5	0	0	4	4	0	0	6	0
8	0	5	0	0	0	0	0	6	0	0
9	5	5	0	0	0	0	0	6	6	0
10	0	0	0	0	0	0	0	0	0	0

Label	R	StartC	StopC
1	2	2	4
2	2	8	9
3	4	2	4
4	6	5	7
4	7	5	6
5	7	2	2
5	8	2	2
5	9	1	2
6	7	9	9
6	8	8	8
6	9	8	9

ภาพที่ 2-12 การเข้ารหัสแบบ Run – Length Encoding ซึ่งแสดงตัวอย่าง Label Matrix และผลการเข้ารหัส

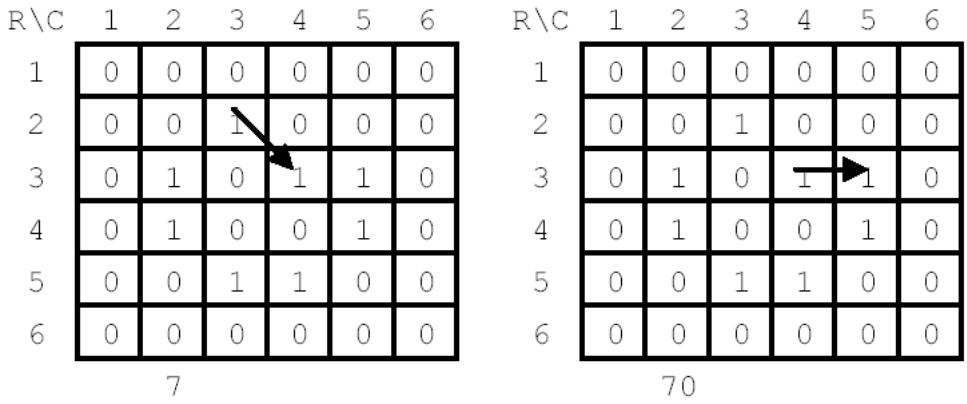
2.6.4.2 การเข้ารหัสแบบ Chain Code เป็นการเข้ารหัสแบบที่นำไปใช้กับการแบ่งแยกบริเวณโดยใช้ขอบวัตถุ หรือ Edge Based Segmentation ซึ่งสามารถทำควบคู่ไปได้กับกระบวนการ Connected Components Labeling ได้เช่นกัน สำหรับกระบวนการเข้ารหัสแบบ Chain Code นี้จะทำการตรวจสอบ Label Matrix จากบนลงล่างจากซ้ายไปขวา จนกระทั่งเจอกับขอบวัตถุจุดแรก หลักจากนั้นจะทำการเดินตามขอบวัตถุและลบขอบที่ได้เดินผ่านมาแล้วไปเรื่อยๆ ในทิศทางตามเข็มนาฬิกาจนกระทั่งกลับมาที่เดิม ซึ่งในการเดินตามขอบวัตถุแต่ละช่องนั้น จะมีการเก็บทิศทางในการเดินจากพิกเซลหนึ่งไปสู่อีกพิกเซลหนึ่งไปเรื่อยๆ ซึ่งจะมีทิศทางที่เป็นไปได้ทั้งหมด 8 ทิศทาง และในแต่ละทิศทางจะมีรหัสประจำ ดังแสดงตัวอย่างไว้ในภาพที่ 2-13

3	2	1
4	x	0
5	6	7

ภาพที่ 2-13 ตัวอย่างรหัส Chain code

จากภาพที่ 2-13 ตัวอย่างรหัส Chain Code ในแต่ละทิศทาง โดยตำแหน่งปัจจุบันอยู่ที่พิกเซล x สำหรับตัวอย่างขั้นตอนการเข้ารหัสแบบ Chain Code นั้นแสดงไว้ในภาพที่ 2-14 ซึ่งค่าที่อยู่ใน Label Matrix ที่เป็นหมายเลข 1 นั้น ได้มาจากกระบวนการ Connected

Components Labeling ซึ่งเป็นการระบุว่าเป็นขอบวัตถุชิ้นที่ 1 และการเข้ารหัสแบบ Chain Code นั้นจะทำ การวนไปตามขอบวัตถุ จนกว่าจะกลับมาที่จุดแรกที่พบขอบวัตถุชิ้นนั้น



ภาพที่ 2-14 การเข้ารหัสแบบ Chain Code ซึ่งในกรณีนี้พิกเซลพิกัด 2,3 เป็นจุดเริ่มและหลังจากครบรอบแล้วจะได้รหัสเป็น 7065432

การเข้ารหัสเพื่อจัดเก็บพิกัดของพิกเซลที่เป็นของวัตถุแต่ละชิ้นที่อยู่ในภาพนั้น เป็นกระบวนการบีบอัดมูลแบบไม่สูญเสียเนื้อหาของข้อมูลเก่า (Lossless Data Compression) แบบหนึ่ง ซึ่งมีจุดประสงค์เพื่อลดขนาดของข้อมูลลง เพื่อให้สามารถส่งต่อไปยังกระบวนการต่อไปได้อย่างสะดวกและมีประสิทธิภาพ อย่างไรก็ตาม เพื่อง่ายต่อการทำความเข้าใจ กระบวนการต่อๆ ไปที่จะนำเสนอจะทำกับ Label Matrix เท่านั้น ซึ่งหลักการรวมทั้งโปรแกรมที่จะนำเสนอนั้นสามารถนำไปใช้ได้กับข้อมูลที่ถูกบีบอัดเหล่านี้ได้เช่นกัน

2.7 การคำนวณหาคุณสมบัติของวัตถุ (Feature Extraction)

เนื่องจากเราได้มุ่งเน้นการแยกบริเวณโดยใช้ค่า Threshold หรือ Region Based Segmentation มาตั้งแต่ต้น ซึ่งเป็นกระบวนการที่ทำให้เราทราบว่า บริเวณของวัตถุที่สนใจของแต่ละชิ้นครอบคลุมพิกัดของพิกเซลใดบ้าง นอกจากนั้น ยังเป็นกระบวนการที่สามารถทำงานได้อย่างมีประสิทธิภาพสูงสุด เมื่อการกระจายตัวของแสงในภาพมีลักษณะสม่ำเสมอ ซึ่งเป็นสภาพแวดล้อมการทำงานตามปกติของระบบตรวจสอบชิ้นส่วนด้วยภาพแบบอัตโนมัติ ดังนั้นเนื้อหาโดยส่วนใหญ่ของหัวข้อนี้คือการศึกษา Label Matrix ที่ได้จากกระบวนการแยกบริเวณวิธีว่าเราจะสามารถคำนวณหาค่าคุณสมบัติของวัตถุแต่ละชิ้นที่ปรากฏอยู่ในภาพได้อย่างไร

สำหรับงานตรวจสอบชิ้นส่วนด้วยภาพแบบอัตโนมัติ คุณสมบัติที่จะทำการวัดออกมาจากวัตถุแต่ละชิ้น จะไม่ใช่การคำนวณที่ซับซ้อนมากนัก เพราะคุณสมบัติเหล่านี้จะถูกนำไปใช้เป็นเงื่อนไขการตรวจสอบผลิตภัณฑ์ชิ้นนั้นๆ ซึ่งจะต้องมีความหมายทางกายภาพ ใช้เวลาคำนวณไม่นานและสามารถเข้าใจได้อย่างไม่ซับซ้อนนัก นอกจากนั้น คุณสมบัติที่วัดได้จากวัตถุนั้น จะสามารถนำไปใช้ในทางตัดสินใจขั้นสูงได้อีกด้วย ซึ่งคุณสมบัติที่สำคัญของวัตถุมีดังนี้

2.7.1 พื้นที่ (Area, A) พื้นที่สำหรับวัตถุที่ปรากฏในภาพดิจิตอลนั้น คือ จำนวนพิกเซลที่ประกอบกันขึ้นมาเป็นวัตถุชิ้นเล็กๆ นอกจากนั้น จะพบว่าหน่วยของปริมาณทางกายภาพต่างๆ ของวัตถุที่อยู่ภาพดิจิตอลจะมีหน่วยเป็นพิกเซลทั้งสิ้น ซึ่งหากต้องการให้ปริมาณต่างๆ มีหน่วยเป็นปริมาณเชิงวิศวกรรมแล้ว จำเป็นจะต้องเทียบจำนวนพิกเซลที่ปรากฏในภาพกับหน่วยความยาวมาตรฐาน เช่น เซนติเมตรหรือมิลลิเมตรเสียก่อน แล้วจึงเทียบผลการวัดแต่ละชนิดที่ได้กับอัตราส่วนดังกล่าว

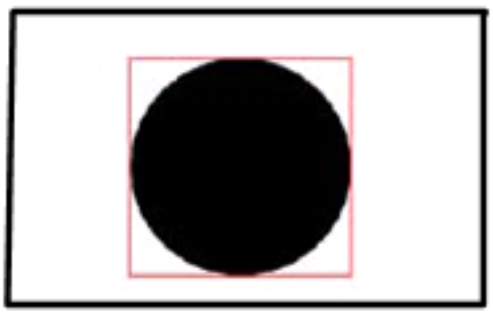
2.7.2 จุดศูนย์ถ่วงของวัตถุ (Centroid-(r,c)) จุดศูนย์ถ่วง คือ ตำแหน่งแถวเฉลี่ยและหลักเฉลี่ยของจากทุกพิกเซล และถึงแม้ว่า หมายเลขแถวและหมายเลขหลักของระบบภาพจะเป็นจำนวนเต็ม แต่ตำแหน่งแถวเฉลี่ยและหลักเฉลี่ยที่ติดจุดทศนิยม จะถูกนำไปใช้พิจารณาคุณภาพของผลิตภัณฑ์มากกว่า เมื่อกำหนดให้ R คือเซทของพิกเซลที่เป็นของวัตถุแต่ละชิ้น จะได้ว่าตำแหน่งแถวและหลักที่เป็นจุดศูนย์ถ่วงของวัตถุจะสามารถคำนวณได้จากสมการที่ (2-10) และที่ (2-11) ตามลำดับ

$$\bar{r} = \left(\frac{1}{A} \sum_{i \in R} r_i \right) \tag{2-10}$$

$$\bar{c} = \left(\frac{1}{A} \sum_{i \in R} c_i \right) \tag{2-11}$$

โดยทั่วไป การหาตำแหน่งแถวเฉลี่ยและหลักเฉลี่ยนั้น ควรจะหารผลบวกของแถวหรือผลบวกของหลักด้วยจำนวนพิกเซลทั้งหมด แต่เราได้นิยามไว้ก่อนหน้าแล้วว่า จำนวนพิกเซลของวัตถุเป็นพื้นที่ของวัตถุนั้น ดังนั้นสมการที่ (2-10) และสมการที่ (2-11) เราจึงสามารถนำค่าพื้นที่ของวัตถุมาเป็นตัวหาผลบวกของแถวหรือหลัก เพื่อหาค่าเฉลี่ยได้เลย

2.7.3 Bounding Box บางครั้งเราต้องการทราบแต่ขนาดคร่าวๆ ของวัตถุเท่านั้น และสามารถทำได้โดยใช้การวัดที่เรียกว่าการหาขนาดของ Bounding Box ซึ่งก็คือ สี่เหลี่ยมที่เล็กที่สุดที่สามารถคลุมทั้งวัตถุได้ ดังแสดงในภาพที่ 2-15



ภาพที่ 2-15 แสดง Bounding Box ล้อมรอบวัตถุที่มีรูปร่างเป็นวงกลม

การหาขนาดของ Bounding Box นั้นจำเป็นจะต้องรู้ 2 พิกัดด้วยกัน คือ จุดบนซ้าย (r_{min} , C_{min}) และจุดล่างขวาของ (r_{max} , c_{max}) ของกรอบสี่เหลี่ยม ซึ่งได้จากพิกัดของพิกเซลที่ประกอบกันขึ้นมาเป็นวัตถุนั้นๆ หลังจากนั้น เราจะสามารถคำนวณหาด้านกว้าง (Bounding Box Width, BBW) และด้านสูงของกรอบสี่เหลี่ยมดังกล่าว (Bounding Box Height, BBH) ได้จากสมการที่ (2-12) และสมการที่ (2-13) ตามลำดับ

$$BBW = (C_{max} - C_{min}) + 1 \quad (2\text{-}12)$$

$$BBH = (r_{max} - r_{min}) + 1 \quad (2\text{-}13)$$

2.7.4 แนวการวางตัวของวัตถุ (Orientation) การคำนวณหาแนวการวางตัวของวัตถุใดๆ ที่ปรากฏขึ้นในภาพนั้น แท้ที่จริงแล้ว คือ การคำนวณมุมของสมการเส้นตรงที่ลากผ่านวัตถุที่ปรากฏในภาพ ซึ่งสมการของเส้นตรงดังกล่าว สามารถคำนวณได้โดยการใช้กระบวนการคณิตศาสตร์ที่ชื่อว่า Linear Regression

สมมุติให้เรามีคู่อันดับ (x,y) อยู่ n จุด หากเราต้องการหาสมการ y=mx+b ที่ “เหมาะสมที่สุด” ที่ทำให้การกระจายตัวไปจากเส้นตรงที่จุดเหล่านี้รวมกันแล้วมีค่าน้อยที่สุด สามารถคำนวณหาสมการเส้นตรงนี้ได้ด้วยกระบวนการทางคณิตศาสตร์ที่ชื่อว่า Linear Regression ซึ่งกระบวนการนี้ถูกนำมาใช้อย่างมากสำหรับงานตรวจสอบชิ้นส่วนด้วยภาพ รวมทั้งงานทางด้าน Machine Vision ทั่วไป ซึ่งตัวอย่างหนึ่งของการนำ Linear Regression มาใช้คือ การนำมาหาทิศทางหรือมุมของวัตถุในภาพและอีกตัวอย่างหนึ่ง คือ การหาทิศทางการวางตัวของผลิตภัณฑ์ สำหรับกระบวนการ Linear Regression นั้น มีรายละเอียดการคำนวณดังนี้

สมมุติให้เรามีคู่อันดับ (x,y) อยู่ n จุด (x_1, y_1) (x_n, y_n) เราสามารถหาความชันและจุดตัดที่ทำกับแกน x ของสมการ y = mx + b ที่จัดว่า “เหมาะสมที่สุด” กับคู่อันดับเหล่านี้ได้ด้วยสมการที่ (2-14) และสมการที่ (2-15) ตามลำดับ

$$\text{slop} = m = \frac{n\left(\sum xy\right) - \left(\sum x\right)\left(\sum y\right)}{n\left(\sum x^2\right) - \left(\sum x\right)^2} \quad (2\text{-}14)$$

$$\text{intercept} = b = \frac{\sum y - m\sum x}{n} \quad (2\text{-}15)$$

เมื่อ $\sum xy$ คือ ผลรวมของผลคูณระหว่าง xy ซึ่งมีค่าเท่ากับ $x_1y_1 + x_2y_2 + \ldots\ldots + x_ny_n$

$\sum x$ คือ ผลรวมของค่า x ซึ่งมีค่าเท่ากับ $x_1 + x_2 + \ldots + x_n$

$\sum y$ คือ ผลรวมของค่า y ซึ่งมีค่าเท่ากับ $y_1 + y_2 + \ldots + y_n$

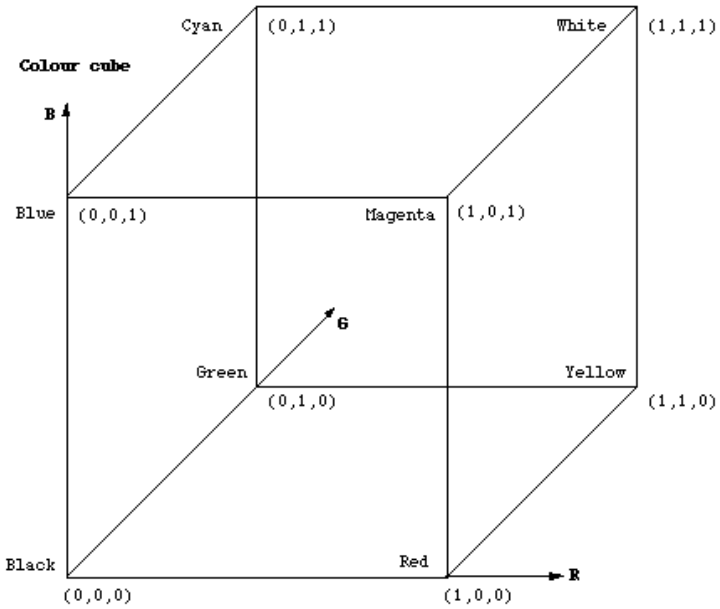
$\sum x^2$ คือ ผลรวมของค่า x^2 ซึ่งมีค่าเท่ากับ $x_1^2 + x_2^2 + \ldots + x_n^2$

เมื่อนำสมการที่ (2-14) ซึ่งใช้เพื่อคำนวณค่าความชันมาใช้กับระบบพิกัดภาพนั้น ค่า x จะถูกแทนที่ค่าหมายเลขหลัก (Column) และ y จะถูกแทนที่ค่าติดลบของหมายเลขแถว (-Row) ทั้งนี้ เนื่องจากสมการที่ (2-14) นั้นเป็นการหาความชันของระบบแกน xy ที่มีแกน y ตั้งขึ้น เมื่อนำสมการนี้มาใช้กับระบบพิกัดของภาพที่มีแกนแถวหรือแกน y ของภาพพุ่งลงนั้น จำเป็นจะต้องติดเครื่องหมายลบไปกับค่าหมายเลขแถวทุกครั้ง เพื่อให้ผลมุมที่คำนวณได้เป็นค่ามุมที่ทำกับแกน xy ปกตินั่นเอง และการหาทิศทางการวางตัวของวัตถุใดๆ ที่ปรากฏในภาพนั้น แท้ที่จริงแล้วก็คือ การคำนวณมุม θ ของเส้นตรงที่มีความชันเท่ากับ m ซึ่งมุมที่มีหน่วยเป็นองศาและความชันของเส้นตรงที่คำนวณได้จากสมการที่ (2-14) มีความสัมพันธ์ดังสมการที่ (2-16)

$$\theta = \frac{180}{\pi}\tan^{-1}(m) \qquad (2\text{-}16)$$

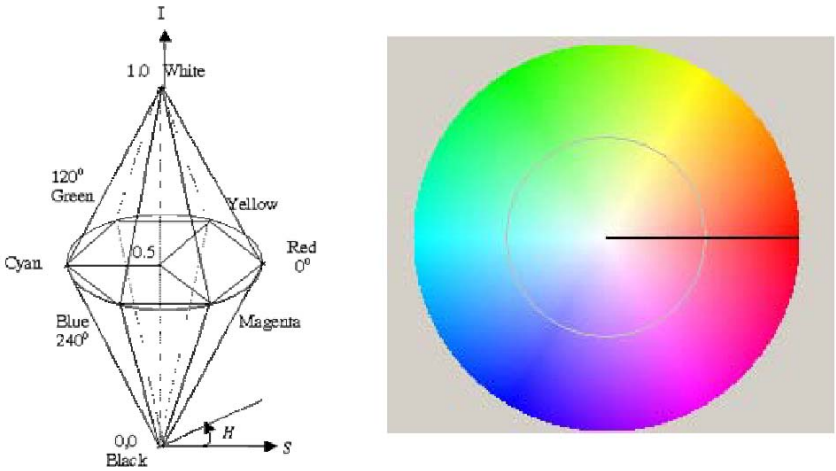
2.7.5 เฉดสีเฉลี่ย (Average Hue) ความสว่างสีเฉลี่ย (Average Intensity) และความสดสีเฉลี่ย(Average Saturation) จากคุณสมบัติที่ผ่านมานั้น จะพบว่า เป็นการวัดคุณสมบัติที่เกี่ยวข้องกับขนาดของวัตถุเป็นส่วนมาก อย่างไรก็ตาม สำหรับการตรวจสอบบางประเภทนั้น จำเป็นอย่างยิ่งที่จะต้องรู้คุณสมบัติพื้นฐานของพื้นผิววัตถุ เช่น เฉดสีเฉลี่ย (Average Hue) และความเข้มแสงเฉลี่ย (Average Intensity) ซึ่งจาก Label Matrix นั้น จะทราบว่า วัตถุแต่ละชิ้นครอบคลุมพิกัดของพิกเซลใดบ้าง และเมื่อนำข้อมูลดังกล่าวไปพิจารณากับภาพ Gray Scale ที่บรรจุข้อมูลความเข้มแสงของแต่ละจุดอยู่ ก็จะสามารถหาค่าความเข้มแสงเฉลี่ยของวัตถุแต่ละชิ้นได้ นอกจากนั้น เมื่อนำข้อมูลใน Label Matrix นี้ไปพิจารณา ควบคู่กับเมตริกซ์สีแดง เขียว น้ำเงิน ก็จะสามารถหาค่าสีแดงเฉลี่ย สีเขียวเฉลี่ย และสีน้ำเงินเฉลี่ย ได้เช่นกัน อย่างไรก็ตาม การหาค่าเฉลี่ยดังกล่าวเพื่อใช้ในการจำแนกสีของวัตถุที่อยู่ในภาพ ยังมี “จุดด้อย” อยู่ดังนี้

โดยปกตินั้น คอมพิวเตอร์จะมีการแทนสีจริงที่เกิดขึ้นตามธรรมชาติ ด้วยการผสมกับของแสงสีแดง เขียว และน้ำเงิน ตามลำดับซึ่งเรียกระบบสีแบบนี้ว่า RGB Color Space ที่เมื่อพิจารณาแล้ว ระบบสีดังกล่าวจะมีลักษณะกล่องสี่เหลี่ยมลูกบาศก์ โดยที่มีความเข้มของแต่ละแสงสีค่าสูงสุดอยู่ในแต่ละมุมดังแสดงไว้ในภาพที่ 2-16 ซึ่งจะพบว่า การระบุสีจริงที่ใกล้เคียงกันทำได้ยากมากเพราะต้องระบุ 3 ค่าของแต่ละแสงสีไปพร้อมๆกัน นอกจากนั้นจะพบว่า สีจริงที่เกิดขึ้นในธรรมชาติที่คล้ายคลึงกันจะมีค่า R G และ B ที่แตกต่างกันอย่างมาก



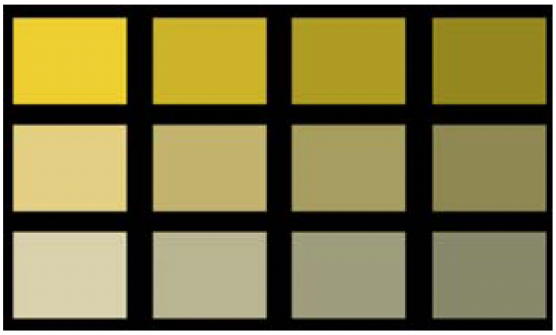
ภาพที่ 2-16 แสดงระบบสี RGB Color Space

การแทนสีจริงที่เกิดขึ้นในธรรมชาติด้วยผลรวมของแสงสีทั้งสามที่มีความเข้มต่างๆ กันนั้น ไม่ใช่วิธีรับรู้สีของมนุษย์ปกติ แต่เราซึ่งเป็นมนุษย์จะรับรู้ได้ว่า สีของวัตถุในธรรมชาตินั้นเป็นสีอะไร (Hue) และเป็นสีที่สดหรือซีดมากร้อยเท่าใด (Saturation) นอกจากนั้น เราจะสามารถระบุได้ว่า สีที่เห็นอยู่มีความสว่าง (Intensity) มากน้อยเพียงใด ดังนั้น เพื่อเลียนแบบการรับรู้สีของมนุษย์ จึงมีผู้คิดค้นระบบสีที่เรียกกันว่า HSI (Hue, Saturation and Intensity) Color Space ขึ้นมา ซึ่งมีแบบจำลองดังในภาพที่ 2-17



ภาพที่ 2-17 แสดงแบบจำลองระบบสี HSI Color Space และ ภาพตัดขวางของแบบจำลอง

ซึ่งมีค่า H เป็นมุมกวาด และมีค่า S คือ รัศมี ของวงกลมจากภาพที่ 2-17 ซึ่งแสดงระบบสี HSI Color Space นั้น จะพบว่า ค่า Hue นั้นใช้สำหรับกำหนดสีของวัตถุ ซึ่งเป็นค่ามุมกวาดตั้งแต่สีแดงที่มีค่ามุมเท่ากับ 0 องศา สีเขียวอยู่ที่มุมเท่ากับ 120 องศา และสีน้ำเงินอยู่ที่มุมเท่ากับ 240 องศาเรื่อยไปจนถึง 360 องศา ซึ่งสีที่อยู่ระหว่างมุมทั้งสามนั้นจะค่อยๆ เปลี่ยนแปลงไปทีละน้อยตามมุมกวาด สำหรับค่าความสว่าง (Intensity) นั้น จะเป็นแกนตั้งของแบบจำลองของระบบสีนี้ ซึ่งจะมีค่าตั้งแต่มืดสุดเท่ากับ 0 จนถึงสว่างสุดเท่ากับ 1 และท้ายที่สุดสำหรับค่า Saturation ที่จะเป็นตัวกำหนดความสดหรือความซีดของสี และจะเป็นตัวกำหนดรัศมีของวงกลมที่มีค่าตั้งแต่ 0 ถึง 1 และจะเห็นว่า ถ้าค่า Saturation มีค่าสูง สีจะสดมากขึ้น ดังแสดงไว้ในภาพที่ 2-18



ภาพที่ 2-18 แสดงสีที่เกิดขึ้นจริงตามธรรมชาติที่มีค่าเฉดสี (Hue) และค่าความสว่าง (Intensity) คงที่และเปลี่ยนเพียงค่า Saturation ไปเท่านั้น

ซึ่งจะพบว่าค่า Saturation นี้จะเป็นตัวกำหนด สีที่เกิดขึ้นนั้นสดหรือซีดมากน้อยเพียงใด จากภาพที่ 2-17 และภาพที่ 2-18 นั้น จะพบว่า สำหรับระบบสีแบบ HSI Color Space นั้น สีที่มีความคล้ายคลึงกันนั้น จะมีค่า Hue ที่ใกล้เคียงกันอย่างมาก โดยไม่ขึ้นอยู่กับความสดของสีหรือความสว่างของสี ดังจะได้เห็นจากตัวอย่างการทำงานของโปรแกรมที่ยกมา ซึ่งการทำให้คอมพิวเตอร์จดจำสีใดๆ ที่มีความคล้ายคลึงกัน เพื่อใช้สำหรับตรวจสอบสีที่ผิดเพี้ยนของผลิตภัณฑ์นั้น โดยทั่วไปจะเป็นการให้คอมพิวเตอร์จดจำค่าพร้อมทั้งส่วนเบี่ยงเบนของค่า Hue ของสีนั้นๆ เพียงเท่านั้น เมื่อนำไปตรวจสอบผลิตภัณฑ์ ก็ทำการพิจารณาค่า Hue ของพื้นผิวผลิตภัณฑ์ว่าเบี่ยงเบนไปมากกว่าที่กำหนดหรือไม่ แล้วจึงตัดสินคุณภาพของผลิตภัณฑ์จากค่าเบี่ยงเบนนั้นๆ ซึ่งการตรวจสอบดังกล่าวทำให้เราต้องแปลงระบบสีจากระบบ RGB Color Space ไปเป็นระบบสี HSI Color Space เสียก่อน ก่อนที่จะทำการวิเคราะห์และเปรียบเทียบต่อไป ซึ่งถ้าค่าความเข้มสีแดง เขียว และน้ำเงิน ณ ที่ตำแหน่งพิกเซลใดๆ มีค่าเท่ากับ R G และ B ตามลำดับ กระบวนการแปลงดังกล่าว มีขั้นตอนการคำนวณดังต่อไปนี้

ทำการปรับขนาดของความเข้มสีทั้งสาม โดยการหาแต่ละค่าความเข้มสีด้วยผลรวมของความเข้มสีทั้งสาม ตามสมการดังต่อไปนี้

$$r = \frac{R}{R+G+B} \tag{2-17}$$

$$g = \frac{G}{R+G+B} \tag{2-18}$$

$$b = \frac{B}{R+G+B} \tag{2-19}$$

ทำการคำนวณค่าความเข้มแสง ค่า Saturation และค่า Hue ด้วยสมการดังต่อไปนี้

$$I = \frac{1}{3}(r+g+b) \tag{2-20}$$

$$S = 1 - \frac{3}{r+g+b}[\min(r,g,b)] \tag{2-21}$$

$$H = \cos^{-1}\left[\frac{\frac{1}{2}[(r-g)+(r-b)]}{\sqrt{(r-g)^2+(r-b)(g-b)}}\right] \tag{2-22}$$

จากขั้นตอน การแปลงระบบสีดังกล่าว มีข้อควรระวังเรื่องช่วงข้อมูล นั่นคือ ระบบภาพ 8 บิต โดยทั่วไป จะมีค่าตั้งแต่ 0 ถึง 255 ซึ่งเป็นจำนวนเต็ม ในขณะที่ผลที่ได้จากการแปลงเป็นระบบสีแบบ HSI Color Space ตามสมการที่ 30,31 และสมการที่ 32 นั้น ค่า I และค่า S ที่ได้จะมีค่าอยู่ในช่วง 0 ถึง 1 ที่เป็นจำนวนจริง ในขณะที่ค่า H ที่เป็นค่ามุมกวาดในแบบจำลองนั้น มีค่าอยู่ในหน่วยเรเดียนส์ คือ ตั้งแต่ 0 ถึง 2π ที่สามารถคำนวณกลับไปเป็นค่ามุมตั้งแต่ 0 ถึง 360 ในหน่วยองศาได้ ซึ่งหากต้องการแสดงผลภาพเมตริกซ์ H ที่ได้จากการแปลงระบบสีนั้น จะต้องทำการปรับช่วงของข้อมูลให้อยู่ในช่วง 0 ถึง 255 และเป็นจำนวนเต็มเสียก่อน

ตารางที่ 2-1 แสดงการเปรียบเทียบค่า RGB และค่า HIS ของสีเขียว 7 เฉด

	1	2	3	4	5	6	7	Average	Std.
Red	27	21	16	92	0	0	145	43	55
Green	204	161	122	161	122	224	255	178	51
Blue	56	44	34	103	10	19	178	63	59
Hue	129	129	129	129	124	124	137	129	4
Saturation	183	184	184	57	255	255	63	169	81
Intensity	96	75	57	119	44	81	193	95	50

จากตารางที่ 2-1 จะพบว่าค่าแสงสีแดง เขียว และน้ำเงิน ในระบบ RGB Color Space ของสีเขียวเหล่านี้มีความแตกต่างกันมาก หากสีของพื้นผิวของผลิตภัณฑ์เป็นสีเขียวหมายเลข 2, 3 และ 5 และเราต้องการตรวจสอบว่า พื้นผิวเหล่านี้มีสีอื่นมาเจือปนหรือไม่ หากใช้ค่า R G B ที่อยู่ในระบบสี RGB Color Space นั้น เราจะต้องเขียนเงื่อนไขให้โปรแกรมยอมรับสีเขียวเหล่านี้ทั้งหมด แล้วตรวจจับสีอื่น แต่ด้วยความที่ค่าเบี่ยงเบนมาตรฐานของแต่ละแสงสีกว้างมากนี่เอง ทำให้โปรแกรมจะต้องยอมรับสีอื่นๆ ไปด้วย ซึ่งบ่งชี้ว่า การเขียนโปรแกรมเพื่อให้จดจำและยอมรับเฉพาะสีเขียวหมายเลข 2, 3 และ 5 แล้วตรวจจับสีอื่น โดยใช้ค่าแสงสีแดง เขียว และ น้ำเงิน ในระบบ RGB Color Space นั้น ทำได้ยากมาก ซึ่งจะกลับกันกับกรณีที่แปลงระบบสีมาเป็น HSI Color Space แล้วทำการระบุสีโดยใช้ค่า Hue เพียงค่าเดียว ที่มีค่าเบี่ยงเบนมาตรฐานแคบกว่ามาก ดังนั้น ถ้าเป็นการระบุลักษณะพื้นผิวที่เป็นสีนั้น โดยมากแล้วจะต้องมีการเปลี่ยนแปลงระบบสีเป็น HSI เสียก่อน ก่อนที่จะทำการตัดสินใจขั้นสูงต่อไป

2.8 การจำแนก (Classification)

ถ้าวัตถุแต่ละกลุ่มมีคุณสมบัติที่สามารถแยกกันได้อย่างชัดเจนแล้ว ก็สามารถใช้ Decision Tree ในการจำแนกวัตถุที่กำลังพิจารณานั้นๆ ว่าเป็นกลุ่มใดหรือชนิดใดได้โดยง่าย อย่างไรก็ตาม หากคุณสมบัติของวัตถุแต่ละกลุ่มมีความซับซ้อนไม่สามารถแยกออกจากกันอย่างชัดเจนแล้ว จำเป็นจะต้องใช้วิธีที่มีประสิทธิภาพมากกว่านี้ และดังที่ได้กล่าวไปแล้วว่า การทำงานของตัว Classifier นั้น คือรับเอา Feature Vector ซึ่งเป็นการเขียนคุณสมบัติต่างๆ ของวัตถุที่สามารถวัดได้หรือคำนวณได้ให้อยู่ในรูปของเวคเตอร์เข้าไป แล้วทำการตัดสินใจว่าวัตถุนั้นๆ อยู่ในกลุ่มหมายเลขใดหรือชื่อใด และให้ชื่อหรือหมายเลขกลุ่มที่วัตถุเป็นสมาชิกอยู่ออกมา ซึ่งการที่ Classifier จะสามารถตัดสินใจได้ จะต้องมีวัตถุตัวอย่างของแต่ละกลุ่มเสียก่อน Classifier มีรูปแบบการทำงานอยู่หลายวิธีด้วยกัน แต่ที่นิยมใช้กันมากในปัจจุบันคือ

2.8.1 Classifier ที่ทำงานโดยการใช้ระเบียบวิธีของ k-Nearest Neighborhood ซึ่งเป็นการพิจารณาวัตถุตัวอย่างที่มี Feature Vector ที่ “ใกล้” กับ Feature Vector ของวัตถุที่กำลังพิจารณามากที่สุด k ตัวอย่างว่า ภายใน k ตัวอย่างนั้น มีสมาชิกของกลุ่มวัตถุใดมากที่สุด แล้วจึงจัดให้วัตถุที่กำลังพิจารณาอยู่ในกลุ่มนั้นๆ

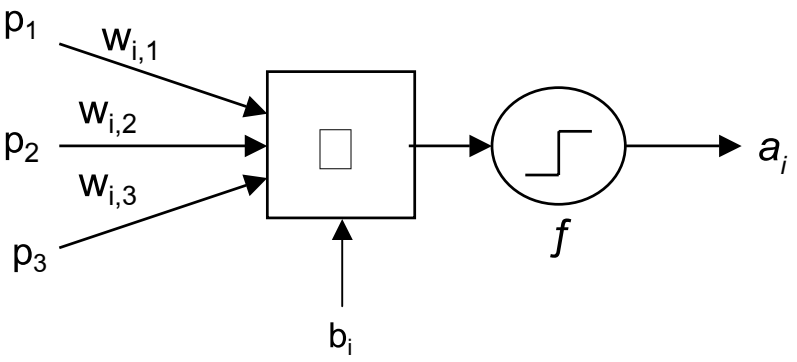
2.8.2 Classifier ที่ทำงานโดยการใช้โครงข่ายระบบประสาทเทียม (Artificial Neural Network Classifier) ซึ่งเป็นแบบจำลองทางคณิตศาสตร์ที่สามารถลอกเลียนการทำงานของสมองมนุษย์มาไว้ในคอมพิวเตอร์

2.9 โครงข่ายประสาทเทียม (Artificial Neural Network)

ในระยะที่สองของการวิจัยนี้จะเป็นการนำระบบโครงข่ายประสาทเทียม (Artificial Neural Network)[9] มาใช้เพื่อทำให้การใช้งานระบบสามารถทำได้อย่างง่ายดายมากขึ้น โดยหากพิจารณาจากมุมมองของผู้ใช้งานนั้น จะพบว่า จากภาพตัวอย่าง ผู้ใช้เพียงทำการระบุว่า รอยด่างที่ตรวจจับได้จากภาพของผลิตภัณฑ์ในขณะนั้น มีรอยด่างใดบ้างที่จัดเป็นของเสียหรือของดี ซึ่งหลังจากนั้น ระบบจะสามารถทำการตรวจสอบภาพของผลิตภัณฑ์ โดยใช้เกณฑ์การตัดสินใจที่ได้จากตัวอย่างที่ระบุโดยผู้ใช้นั้นเอง

อนึ่ง สำหรับโครงสร้างของโครงข่ายประสาทเทียมที่เลือกใช้นั้น จะเป็นโครงข่ายที่ไม่มีการป้อนกลับ (Feed Forward Network) ที่มี 3 ชั้น และใช้ระเบียบวิธีแบบแพร่กลับ (Back Propagation Training Algorithm) ซึ่งจัดเป็นโครงสร้างที่มีการใช้งานในระบบจำแนก (Classification System) ทั่วไป

2.9.1 Single Layer Perceptron Networks



$$
\begin{aligned}
a_i &= f(w_{i,1}p_1 + w_{i,2}p_2 + w_{i,3}p_3 + b_i) \\
&= f(\sum_j w_{ij}x_j + b_i) \\
&= f(\mathbf{W}^T\mathbf{p} + b)
\end{aligned}
\qquad \text{(2-23)}
$$

เมื่อ p_j = ข้อมูลเข้า (Input Data from Node j in the Input Layer)

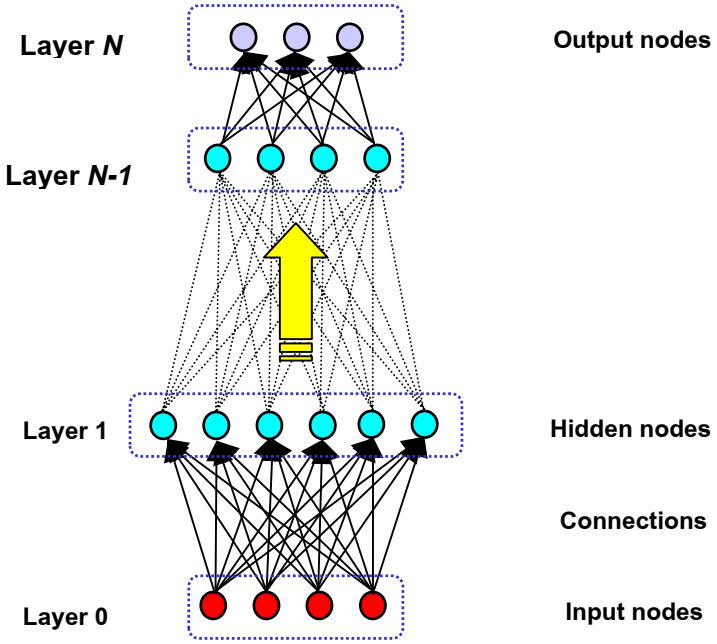
w_{ij} = ตัวถ่วงน้ำหนักเชื่อมโยง (Connection Weight of Branch (i,j))

b_i = ค่าขีดเริ่มเปลี่ยน (Bias)

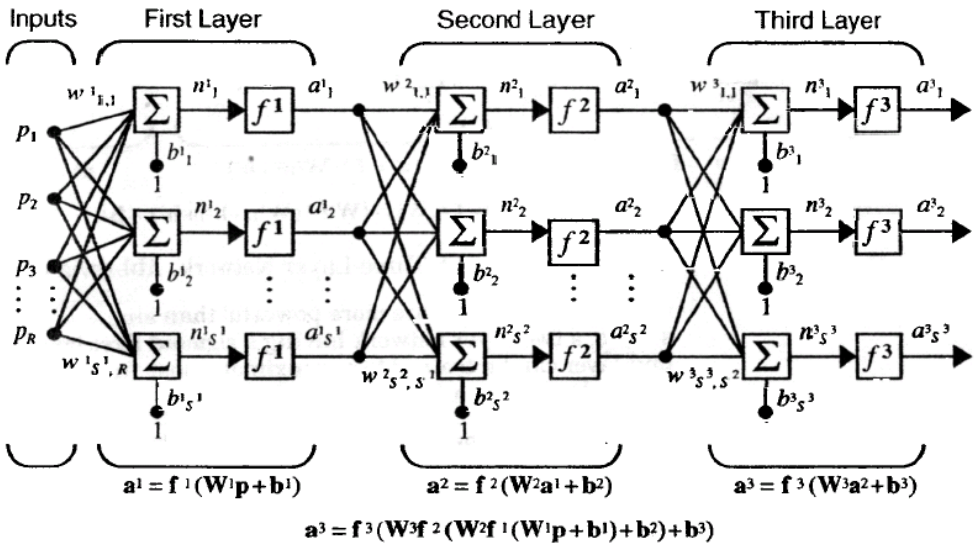
f = ฟังค์ชั่นกระตุ้น (Activation Function)

a_i = ข้อมูลออกของนิวรอน (Network Output)

2.9.2 Multilayer Feedforward Network Structure



ภาพที่ 2-19 แสดงโครงสร้างของโครงข่ายที่ไม่มีการป้อนกลับ (Feed Forward Network) แบบหลายชั้น



ภาพที่ 2-20 แสดงโครงสร้างโครงข่ายประสาทเทียมแบบ 3 ชั้น ที่ไม่มีการป้อนกลับ (3-layer Feed Forward Artificial Neural Network)

2.9.3 Backward Propagation Equation

$$s^M = -2\dot{F}^M(n^M)(t-a) \tag{2-24}$$

$$s^m = \dot{F}^m(n^m)(W^{m+1})^T s^{m+1}, \text{for m} = M-1,....,2,1, \tag{2-25}$$

เมื่อ

$$\dot{F}^m(n^m) = \begin{bmatrix} \dot{f}^m(n_1^m) & 0 & .... & 0 \\ 0 & \dot{f}^m(n_2^m) & .... & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & & \dot{f}^m(n_{S^m}^m) \end{bmatrix} \tag{2-26}$$

$$\dot{f}^m(n_j^m) = \frac{\partial f^m(n_j^m)}{\partial n_j^m} \tag{2-27}$$

การคำนวณ $f'(n)$ สำหรับกรณี Nonlinear unit

Log-Sigmoid function

$$a = f(n) = \frac{1}{1+e^{-n}} \tag{2-28}$$

$$f'(n) = \frac{e^{-n}}{(1+e^{-n})^2} = \left(1 - \frac{1}{(1+e^{-n})}\right)\left(\frac{1}{1+e^{-n}}\right) = (1-a)(a) \tag{2-29}$$

Weight Update (Approximate Steepest Descent)

$$\mathrm{W}^m(k+1) = \mathrm{W}^m(k) + \Delta\mathrm{w}^m(k) \tag{2-30}$$

$$b^m(k+1) = b^m(k) + \Delta b^m(k) \tag{2-31}$$

$$\Delta\mathrm{w}^m(k) = \gamma\Delta\mathrm{w}^m(k-1) - (1-\gamma)\alpha s^m\left(a^{m-1}\right)^T \tag{2-32}$$

$$\Delta b^m(k) = \gamma\Delta b^m(k-1) - (1-\gamma)\alpha s^m \tag{2-33}$$

เมื่อ p = ข้อมูลเข้า (Input Data)

w = ตัวถ่วงน้ำหนักเชื่อมโยง (Connection Weight)

b = ค่าขีดเริ่มเปลี่ยน (Bias)

f = ฟังค์ชั่นกระตุ้น (Activation Function)

a = ข้อมูลออกของนิวรอน (Network Output)

s = ค่าความไหว (Sensitivity)

γ = โมเมนตัม, การใช้โมเมนตัมเทอมจะช่วยทำให้ Algorithm มีความเสถียร (Stable) มากขึ้น โดยการใช้โมเมนตัม เราสามารถใช้ค่า Learning Rate ที่สูง ขณะที่ระบบยังคงรักษาความเสถียรอยู่ นอกจากนี้ยังช่วยเร่งการลู่เข้าหาจุดต่ำสุดของ Error โดยวิถีการเคลื่อนอยู่ในทิศทางที่ไม่เปลี่ยนแปลง

α = อัตราการเรียนรู้ เป็นพารามิเตอร์ที่มีผลกับค่าถ่วงน้ำหนัก และมีผลต่อเวลาในการลู่เข้าหาจุดต่ำสุดของ Error

บทที่ 3
วิธีการดำเนินการวิจัย

3.1 ขั้นตอนการวิจัย

งานวิจัยชิ้นนี้ ได้แบ่งขั้นตอนการวิจัยออกเป็น 2 ช่วงด้วยกัน คือ

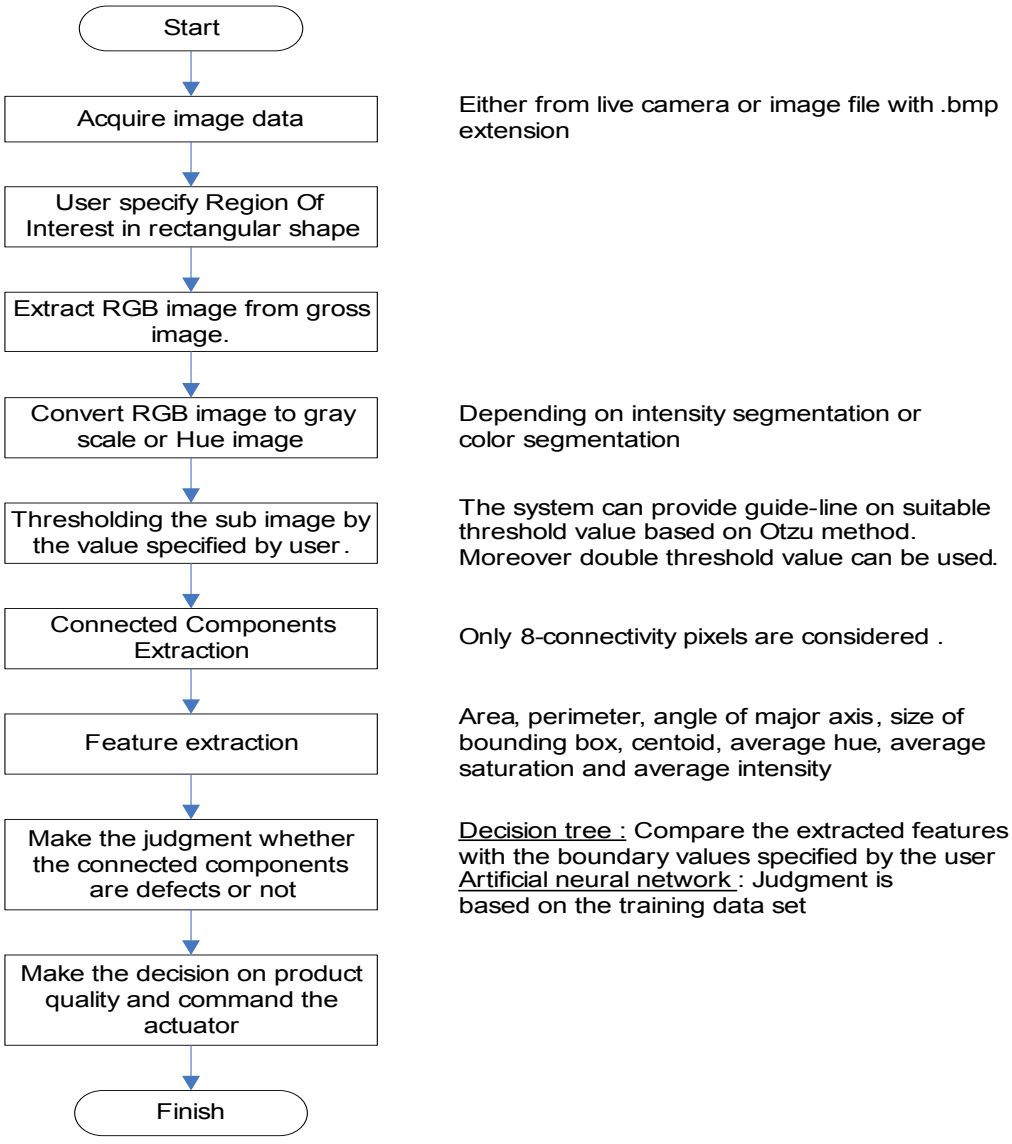
3.1.1 ช่วงแรกจะเป็นสร้างระบบตรวจสอบชิ้นงานด้วยภาพอย่างง่ายสำหรับการตัดสินคุณภาพของผลิตภัณฑ์ซึ่งในที่นี้คือ ผ้าสีพื้นเดียวแบบไม่มีลวดลาย สามารถทำได้โดยใช้ค่าคุณสมบัติต่าง ๆของรอยด่างที่จัดเป็นของเสีย ที่ผู้ใช้งานระบบหรือผู้ควบคุมคุณภาพของผลิตภัณฑ์เป็นผู้กำหนดให้เท่านั้น ซึ่งผลที่ได้จากการวิจัยในขั้นตอนนี้ คือ ระบบที่สามารถทำการวัดคุณสมบัติต่าง ๆของรอยด่างที่ปรากฏอยู่บนผืนผ้าสี เช่น พื้นที่, ขนาดของ Bounding Box, เฉดสีเฉลี่ย เป็นต้น และทำการเปรียบเทียบคุณเสมบัติของรอยด่างเหล่านี้กับค่าคุณสมบัติของรอยด่างที่จัดเป็นของเสียที่กำหนดโดยผู้ใช้งานระบบ แล้วตัดสินว่า รอยด่างที่ตรวจจับได้จากภาพของผลิตภัณฑ์ในขณะนั้นเป็นของเสียหรือไม่ ซึ่งจากข้อมูลดังกล่าว ระบบจะสามารถทำการตัดสินคุณภาพของผลิตภัณฑ์ได้ว่าเป็นชิ้นงานเสียหรือไม่ นอกจากนั้น ผลการตัดสินคุณภาพของผลิตภัณฑ์ที่ได้จะสามารถนำไปสั่งงานกลไกที่เกี่ยวข้องต่อไปได้

3.1.2 ช่วงที่สองของงานวิจัยนี้ จะเป็นการเพิ่มความสามารถในการตัดสินใจให้กับระบบตรวจสอบที่ได้จากการวิจัยในระยะแรก ทั้งนี้เพื่อลดความยุ่งยากให้กับผู้ใช้งานระบบตรวจสอบ ซึ่งจะพบว่า การทำงานของระบบที่ได้จากการวิจัยในระยะแรกนั้น ผู้ใช้จำเป็นจะต้องระบุคุณสมบัติของรอยด่างที่จัดว่าเป็นของเสียในเชิงปริมาณ หรือกล่าวง่าย ๆอีกนัยหนึ่งก็คือ ค่าคุณสมบัติต่าง ๆของรอยด่างที่เป็นตัวเสียที่ใช้เปรียบเทียบกับคุณสมบัติของรอยด่างที่ตรวจจับได้จากภาพในขณะนั้น ผู้ใช้งานจะต้องระบุออกมาเป็นตัวเลขให้กับระบบเท่านั้น ซึ่งในทางปฏิบัตินั้น การที่จะให้ผู้ใช้งานระบบทำการระบุค่าคุณสมบัติต่าง ๆของรอยด่างที่จัดเป็นตัวเสียในรูปของตัวเลข ค่อนข้างซับซ้อนและไม่สะดวก แต่ถ้าให้ผู้ใช้งานเพียงระบุว่า รอยด่างใดจัดเป็นตัวเสียบ้าง ซึ่งเป็นหน้าที่โดยปกติของผู้ใช้งานระบบหรือผู้ควบคุมคุณภาพของผลิตภัณฑ์อยู่แล้วนั้น จะทำให้ผู้ใช้สามารถใช้งานระบบได้ง่ายขึ้น ดังนั้น ในระยะที่สองของการวิจัยนี้จะเป็นการนำระบบโครงข่ายประสาทเทียม (Artificial Neural Network) มาใช้เพื่อทำให้การใช้งานระบบสามารถทำได้อย่างง่ายดายมากขึ้น โดยหากพิจารณาจากมุมมองของผู้ใช้งานนั้น จะพบว่า จากภาพตัวอย่าง ผู้ใช้เพียงทำการระบุว่า รอยด่างที่ตรวจจับได้จากภาพของผลิตภัณฑ์ในขณะนั้น มีรอยด่างใดบ้างที่จัดเป็นของเสียหรือของดี ซึ่งหลังจากนั้น ระบบจะสามารถทำการตรวจสอบภาพของผลิตภัณฑ์ โดยใช้เกณฑ์การตัดสินใจที่ได้จากตัวอย่างที่ระบุโดยผู้ใช้นั้นเอง

อนึ่ง สำหรับโครงสร้างของโครงข่ายประสาทเทียมที่เลือกใช้นั้น จะเป็นโครงข่ายที่ไม่มีการป้อนกลับ (Feedforward Network) ที่มี 3 ชั้น และใช้ระเบียบวิธีแบบแพร่กลับ (Back Propagation Training Algorithm) ซึ่งจัดเป็นโครงสร้างที่มีการใช้งานในระบบจำแนก (Classification System) ทั่วไป

3.2 ขั้นตอนการทำงานของโปรแกรม

งานส่วนใหญ่ของระบบตรวจสอบผืนผ้าด้วยภาพนี้ จะเป็นการทำงานของโปรแกรมทั้งสิ้น ซึ่งมีขั้นตอนการทำงานของโปรแกรมเป็นดังแสดงไว้ในภาพที่ 3-1 และสามารถอธิบายรายละเอียดการทำงานได้ดังต่อไปนี้



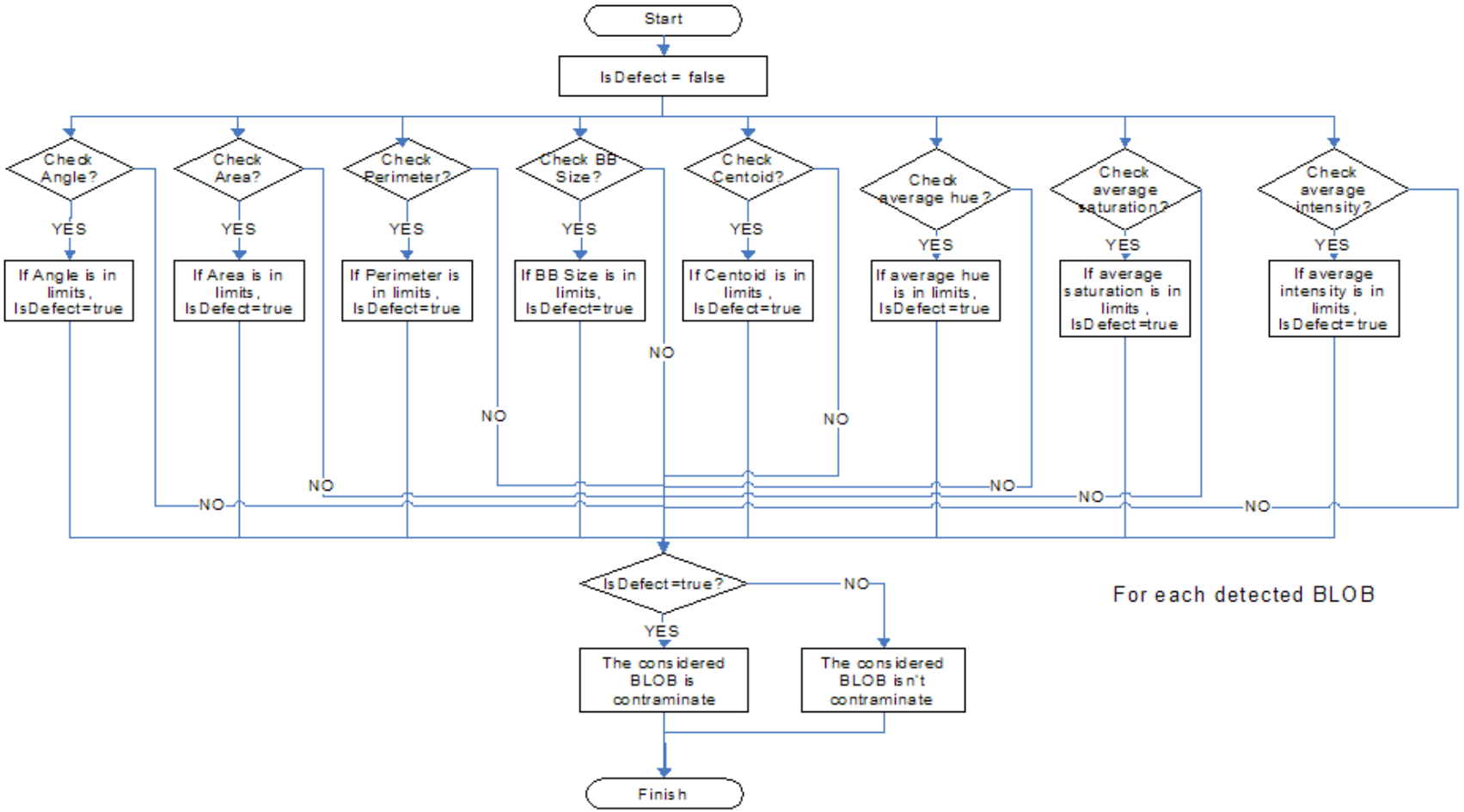
ภาพที่ 3-1 ขั้นตอนการทำงานของโปรแกรมโดยรวม

ในตอนเริ่มแรกนั้น โปรแกรมจะทำการดึงข้อมูลภาพสีไม่ว่าจะมาจากกล้องหรือจากไฟล์ภาพที่เก็บไว้ในคอมพิวเตอร์และแสดงภาพขึ้นมา เพื่อให้ผู้ใช้ทำการระบุบริเวณที่สนใจหรือทั้งผ้าก็ได้ ทั้งนี้ เนื่องจากบริเวณของผลิตภัณฑ์อาจจะไม่คลอบคลุมทั้งบริเวณภาพ หลังจากนั้นโปรแกรมจะทำการแยกข้อมูลภาพเฉพาะบริเวณที่ระบุโดยผู้ใช้ว่า เป็นบริเวณที่จะต้องทำการตรวจสอบ และทำการแปลงให้เป็นภาพที่มีสองระดับ (Binary Image) หรือ ภาพในระบบสี HSI โดยใช้ค่า Threshold ที่ระบุโดยผู้ใช้ ซึ่งโปรแกรมสามารถช่วยระบุค่า Threshold ที่เหมาะสมให้กับผู้ใช้ เพื่อให้ผู้ใช้ช่วยประกอบการตัดสินใจได้ด้วย สำหรับค่า Threshold ที่ระบุโดยโปรแกรมนั้นได้มาจากการเลือกค่า Threshold โดยวิธีของ Otsu ซึ่งเป็นวิธีการเลือกค่า Threshold แบบอัตโนมัติที่จัดว่ามีการนำไปใช้อย่างแพร่หลายมากที่สุด หลังจากที่ได้ภาพที่มีเพียงสองระดับแล้ว โปรแกรมจะทำการกระบวนการ Connected Components Extraction เพื่อทำการตรวจจับรอยด่างที่ปรากฏขึ้นในภาพ ซึ่งโปรแกรมจะทำการพิจารณาพิกเซลที่มีการเชื่อมต่อแบบ 8 ทิศทางเท่านั้น และจากขั้นตอนนี้จะทำให้โปรแกรมทราบว่า มีรอยด่างจำนวนเท่าใดอยู่ในภาพ นอกจากนั้น จะทำให้โปรแกรมทราบว่า รอยด่างที่ตรวจจับได้นั้นครอบคลุมพิกัดของพิกเซลใดบ้าง หลังจากนั้น โปรแกรมจะทำการตรวจวัดคุณสมบัติต่างๆของรอยด่างที่ตรวจจับได้ ซึ่งได้แก่ มุมของแกนหลักของรอยด่างแต่ละชิ้น (Angle) พื้นที่ (Area) เส้นรอบรูป (Perimeter) ตำแหน่งจุดศูนย์กลางของรอยด่าง (Centroid) นอกจากนั้น โปรแกรมยังทำการวัดคุณสมบัติของสีของรอยด่างแต่ละชิ้นด้วย ซึ่งคุณสมบัติของสีที่ทำการตรวจวัด คือ ค่าเฉดสีเฉลี่ย (Average Hue) ค่าความอิ่มตัวสีเฉลี่ย (Average Saturation) และ ค่าความสว่างสีเฉลี่ย (Average Intensity) ของรอยด่างแต่ละชิ้น ซึ่งถ้าเป็นการทำงานของโปรแกรมที่ได้จากการวิจัยในระยะแรกนั้น จากคุณสมบัติต่างๆ ของรอยด่างแต่ละชิ้นจะนำไปเปรียบเทียบกับค่าที่กำหนดโดยผู้ใช้ เพื่อพิจารณาว่า รอยด่างใดบ้างที่จัดเป็นของเสียตามข้อกำหนดของผู้ใช้ ซึ่งหากมีเพียงหนึ่งรอยด่างที่จัดเป็นของเสีย ผืนผ้าช่วงนั้นจะจัดว่าเป็นชิ้นงานเสียทันที

3.3 วิธีการจำแนกโดยวิธีทางสถิติ

เป็นวิธีที่ใช้ค่าคุณสมบัติต่างๆ ของรอยด่างที่จัดเป็นของเสีย ที่ผู้ใช้งานระบบหรือผู้ควบคุมคุณภาพของผลิตภัณฑ์เป็นผู้กำหนดให้เท่านั้น ซึ่งผลที่ได้จากการวิจัยในขั้นตอนนี้ คือ ระบบสามารถทำการวัดคุณสมบัติต่างๆ ของรอยด่างที่ปรากฏอยู่บนผืนผ้าสี เช่น พื้นที่, ขนาดของ Bounding Box, เฉดสีเฉลี่ย เป็นต้น และทำการเปรียบเทียบคุณสมบัติของรอยด่างเหล่านี้กับค่าคุณสมบัติของรอยด่างที่จัดเป็นของเสียที่กำหนดโดยผู้ใช้งานระบบ แล้วตัดสินว่า รอยด่างที่ตรวจจับได้จากภาพของผลิตภัณฑ์ในขณะนั้นเป็นของเสียหรือไม่ ซึ่งจากข้อมูลดังกล่าว ระบบจะสามารถทำการตัดสินคุณภาพของผลิตภัณฑ์ได้ว่าเป็นชิ้นงานเสียหรือไม่ ดังภาพที่ 3-2

หลังจากทำการแยกภาพพื้นหลังออกจากภาพวัตถุที่เป็นรอยด่างโดยการใช้ Threshold ผู้ใช้สามารถระบุคุณสมบัติซึ่งจัดว่าเป็นรอยด่างได้ เช่นผู้ใช้สามารถเลือกช่วงพื้นที่ของภาพวัตถุ (Area) ซึ่งจัดว่าเป็นรอยด่างได้ เพื่อไม่ต้องไปคำนึงถึงวัตถุที่มีขนาดเล็กมากๆ ที่ในบางกรณีนั้น

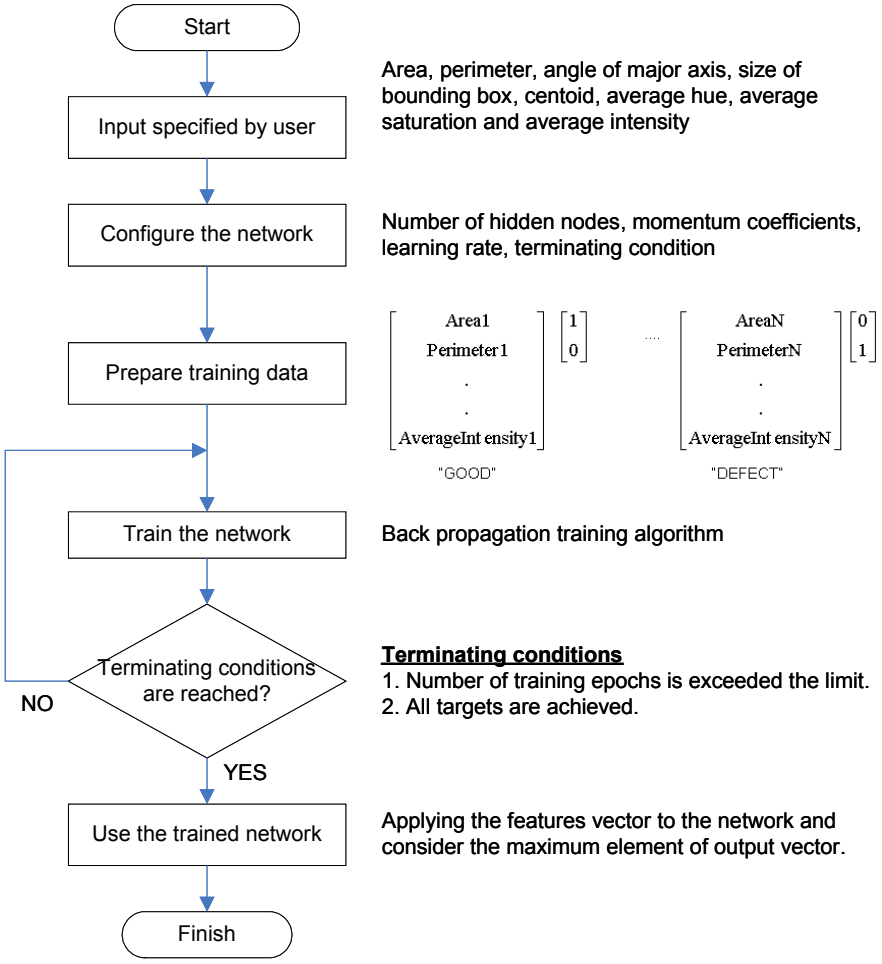


ภาพที่ 3-2 แสดงวิธีการจำแนกโดยวิธีทางสถิติ (Statistical Algorithm)

ไม่สนใจ ขึ้นอยู่กับการคัดเกรดของผ้าย้อม ซึ่งวัตถุที่อยู่นอกช่วงที่กำหนดจะไม่จัดว่าเป็นรอยด่าง หรือจะเป็นการกำหนดช่วงของค่าเฉดสีเฉลี่ย (Average Hue) เพื่อเป็นการกำหนดช่วงของเฉดสีที่จัดเป็นตัวเสียนั่นเอง สรุปก็คือภาพวัตถุใดก็ตามที่มีค่าคุณสมบัติตกอยู่ในช่วงของค่าคุณสมบัติที่เรากำหนด ให้จัดว่าวัตถุนั้นเป็นพื้นผิวเสียทันที เมื่อการกำหนดเสร็จสิ้น ผู้ใช้สามารถนำไปใช้กับภาพอื่นๆ ได้ โดยควรจะมีลักษณะพื้นหลังใกล้เคียงกัน และผู้ใช้สามารถทำการเซฟฐานข้อมูล (Data Base) เพื่อสามารถนำไปใช้ให้เหมาะกับงานที่จะใช้ได้

3.4 ขั้นตอนจำแนกของวิธีโครงข่ายประสาทเทียม

วิธีการจำแนกนี้ ผู้ใช้งานเพียงระบุว่า รอยด่างใดจัดเป็นตัวเสียบ้าง ซึ่งเป็นหน้าที่โดยปกติของผู้ใช้งานระบบหรือผู้ควบคุมคุณภาพของผลิตภัณฑ์อยู่แล้วนั้น จะทำให้ผู้ใช้สามารถใช้งานระบบได้ง่ายขึ้น ดังนั้น ในระยะที่สองของการวิจัยนี้จะเป็นการนำระบบโครงข่ายประสาทเทียม (Artificial Neural Network) มาใช้เพื่อทำให้การใช้งานระบบสามารถทำได้อย่างง่ายดายมากขึ้น ดังภาพที่ 3-3



ภาพที่ 3-3 แสดงวิธีจำแนกโดยโครงข่ายประสาทเทียม (Neural Network Algorithm)

โดยหากพิจารณาจากมุมมองของผู้ใช้งานนั้น ผู้ใช้เพียงทำการระบุว่า รอยด่างที่ตรวจจับได้จากภาพของผลิตภัณฑ์ในขณะนั้น มีรอยด่างใดบ้างที่จัดเป็นของเสียหรือของดี ซึ่งข้อมูลที่ผู้ใช้ได้ระบุไว้จัดเป็น Input Data สำหรับการเทรนนิวรอลโดยเราสามารถระบุค่าต่างๆที่เกี่ยวข้องกับการเทรนได้เช่น ค่า Learning Rate, Hidden Node, ค่าคุณสมบัติที่จะใช้เทรน และ ค่า Tolerance ที่สามารถยอมรับได้ของ Training Output เทียบกับ Target Output หลังจากนั้นโปรแกรมจะทำการเทรนนิวรอลโดย Back Propagation Training Algorithm โดยการปรับ Weight และ Bias ในทุกๆ 1 Cycle หรือเทรนครบทุก Input Data ก่อนแล้วจึงปรับ Weight และ Bias ซึ่งเรียกว่า Batch Mode กระทั่ง Training Output และ Target Output มีค่าใกล้เคียงกันและอยู่ในค่าที่ยอมรับได้ซึ่งผู้ใช้ต้องเป็นคนกำหนด หลังจากเสร็จการเทรนนิ่งแล้ว ระบบจะสามารถทำการตรวจสอบคุณภาพของผลิตภัณฑ์ โดยใช้เกณฑ์การตัดสินใจที่ได้จากตัวอย่างที่ระบุโดยผู้ใช้นั้นเอง

การทำงานของวิธีจำแนกโดยโครงข่ายประสาทเทียม (Artificial Neural Network) นั้นมีขั้นตอนดังนี้

3.4.1 ผู้ใช้ระบุวัตถุที่จัดว่าเป็นตัวเสียโดยคลิกเลือกบนภาพวัตถุนั้น โดยวัตถุที่ไม่ได้ถูกเลือกนั้นจะจัดว่าเป็นภาพดี ซึ่งแต่วัตถุบนภาพจะมีค่าคุณสมบัติของมันเองทั้งหมด 8 ค่า ดังนี้ คือ ค่า เฉดสีเฉลี่ย (Average Hue) ค่าความเข้มแสงเฉลี่ย (Average Intensity) ค่าความอิ่มตัวของสีเฉลี่ย (Average Saturation) ค่าพื้นที่ (Area) ค่าจุดศูนย์ถ่วง (Centroid) ค่า Bounding Box ค่าเส้นรอบรูป (Perimeter) และ ค่าแนวการวางตัวของวัตถุ (Angle) ซึ่งคุณสมบัติเหล่านี้ก็คือข้อมูลอินพุท (Input Data) ทั้งหมด ซึ่งผู้ใช้สามารถเลือกข้อมูลขาเข้าได้จาก 8 ค่านี้

3.4.2 ระบุข้อมูลอินพุท (Input Data) ของวัตถุที่ต้องการใช้ในการเทรน (Training) ซึ่งสามารถระบุได้มากสุด 8 ตัว

3.4.3 กำหนดค่าพารามิเตอร์ต่างๆ ที่จะใช้สำหรับการเทรน ดังนี้

3.4.3.1 ค่าปมซ่อน (Hidden Node) สามารถกำหนดได้อย่างไม่มีลิมิต ถ้ากำหนดมากก็จะใช้เวลาในการเทรนนาน ถ้ากำหนดน้อยเกินไป ก็อาจจะเทรนไม่สำเร็จ ต้องกำหนดค่าปมซ่อน (Hidden Node) ในแต่ละชั้นของโครงข่ายที่ไม่มีการป้อนกลับ (Feed Forward Network) ที่มี 3 ชั้น

3.4.3.2 ค่าอัตราการเรียนรู้ (Learning Rate) เป็นค่าที่มีผลต่อการลู่เข้าหาจุดต่ำสุดของค่าความผิดพลาด (Error) และต้องกำหนดให้ในแต่ละเลเยอร์ด้วย

3.4.3.3 ค่าสัมปะสิทธ์โมเมนตัม (Momentum Coefficient) เป็นค่าที่มีผลต่อความเสถียร (Stable) ในการลู่เข้า โดยการใช้โมเมนตัม เราสามารถใช้ค่า Learning Rate ที่สูง ขณะที่ระบบยังคงรักษาความเสถียรอยู่

3.4.3.4 โหมดการ Train หากผู้ใช้ Check ลงในช่อง Batch Mode Training การ Train โครงข่ายประสาทเทียมจะเป็นแบบ Batching โดยโปรแกรมจะทำการ Update ค่า

Weight Matrix และ Bias Vector เพียงครั้งเดียวเท่านั้น หลังจากที่ป้อนกลุ่มตัวอย่างทั้งหมดเข้าไปให้กับโครงข่ายและในทางตรงกันข้าม หากผู้ใช้ Uncheck ช่องดังกล่าว การ Train โครงข่ายประสาทเทียมจะเป็นแบบ Incremental Training ซึ่งจะทำการ Update ค่า Weight Matrix และ Bias Vector ทุกครั้งที่มีการป้อนตัวให้กับโครงข่ายประสาทเทียม

3.4.3.5 จำนวนรอบในการ Train (Maximum Epoch) ในการ Train โครงข่ายประสาทเทียมโดยการใช้ระเบียบวิธีคำนวณแบบ Back Propagation Training Algorithm นั้น การป้อนตัวอย่างครบทั้งหมดนับเป็นการ Train 1 รอบ (Epoch) ระบบจะทำการเทรนและปรับ Weight และ Bias ไปเรื่อยๆ จนกระทั่งตัวอย่างทุกตัวได้ค่าความผิดพลาดอยู่ในช่วงที่ยอมรับได้แล้วจึงหยุดเทรน ซึ่งผู้ใช้สามารถกำหนดเงื่อนไขหยุด Train ได้จากช่องนี้

3.4.3.6 ช่วงยอมรับ (Tolerance) ซึ่งระบุในรูปร้อยละ ค่าดังกล่าว คือค่าสูงสุดของผลต่างระหว่าง Target Vector และ Output Vector ของโครงข่ายประสาทเทียม ซึ่งหากตัวอย่างใดมีค่าน้อยกว่าค่าที่กำหนดไว้ จะถือว่าการ Train สำหรับตัวอย่างนั้นเป็นผลสำเร็จ ซึ่งโปรแกรมจะหยุดเทรนก็ต่อเมื่อตัวอย่างทุกตัวมีค่าดังกล่าวน้อยกว่าค่าที่กำหนดไว้ในช่องนี้

3.4.4 เตรียมข้อมูลเทรนนิ่ง ซึ่งข้อมูลอินพุทจะเป็นเวกเตอร์ 6 x 1 ในกรณีที่เลือกค่าคุณสมบัติที่จะเทรนไว้ 6 ค่า ขณะที่ข้อมูลขาออกที่ต้องการ (Target Output) จะถูกกำหนดให้เป็นเวกเตอร์ 2 x 1 โดยถ้าเป็นวัตถุดี (Good) ค่าข้อมูลขาออกที่ต้องการคือ $[\ 1\ 0\ ]^T$ และถ้าเป็นวัตถุเสีย (Defect) ค่าข้อมูลขาออกที่ต้องการคือ $[\ 0\ 1\ ]^T$

3.4.5 การเทรนนิวรอล สำหรับโครงสร้างของโครงข่ายประสาทเทียมที่เลือกใช้นั้น จะเป็นโครงข่ายที่ไม่มีการป้อนกลับ (Feed Forward Network) ที่มี 3 ชั้น และใช้ระเบียบวิธีแบบแพร่กลับ (Back Propagation Training Algorithm) โดยกำหนดให้มีค่าถ่วงน้ำหนักเชื่อมโยง (Connection Weight) เริ่มต้นและค่าขีดเริ่มเปลี่ยน (Bias) เริ่มต้น เป็นค่าสุ่ม (Random) ระหว่าง -1.5 ถึง 1.5 และมีข้อมูลเข้า (Input Data) เป็นเวกเตอร์ที่เราสามารถกำหนดจำนวนของข้อมูลเข้าได้ สำหรับข้อมูลขาออกที่ต้องการ (Target Output) จะถูกกำหนดให้เป็นเวกเตอร์ 2 x 1 โดยที่ข้อมูลขาออก (Network Output) ก็เป็นเวกเตอร์ 2 x 1 สำหรับฟังก์ชั่นการกระตุ้น (Activation Function) ที่ใช้ในแต่ละชั้นคือ Log-Sigmoid Function สำหรับการปรับ (Update) ค่าถ่วงน้ำหนักเชื่อมโยง (Connection Weight) และ ค่าขีดเริ่มเปลี่ยน (Bias) ในแต่ละครั้ง จะขึ้นอยู่กับการเลือกโหมดสำหรับการเทรนซึ่งมี 2 แบบ คือ Batch Mode และ Incremental Mode

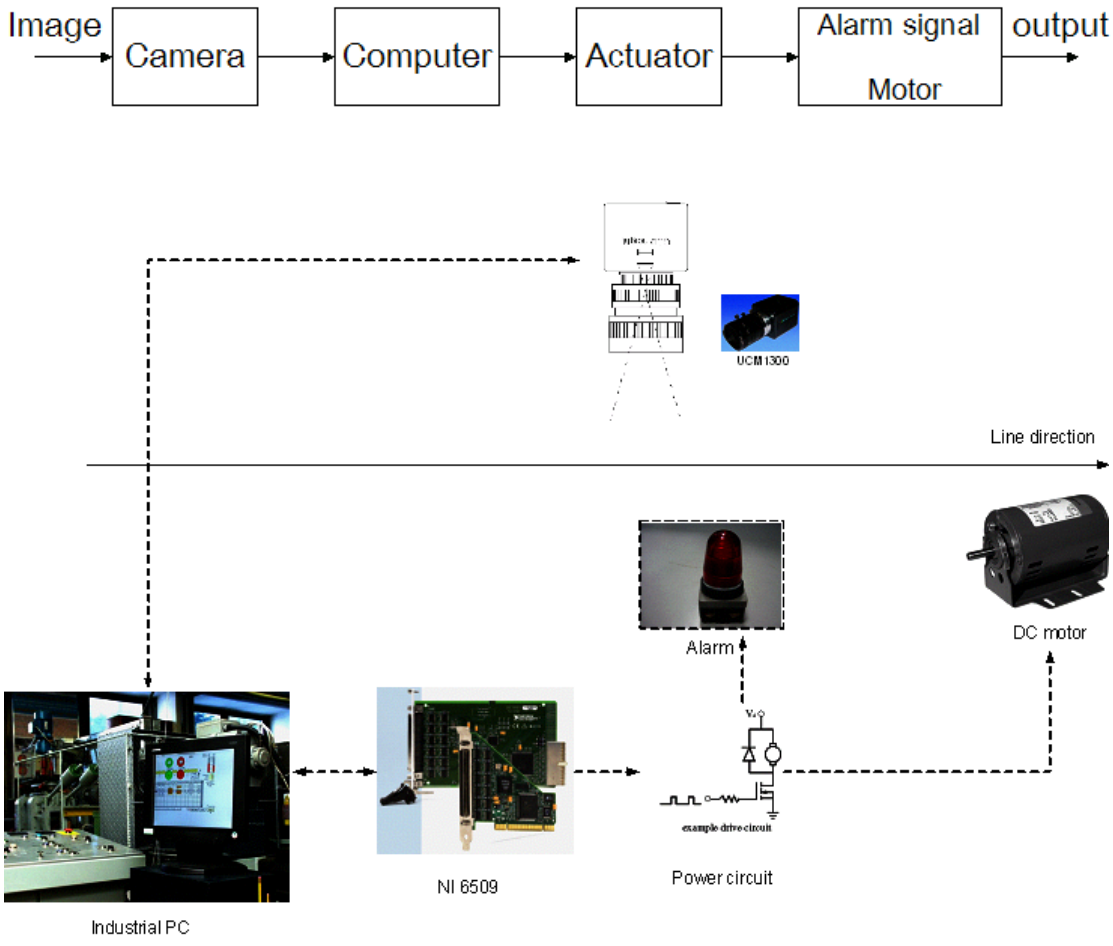
3.4.6 การเทรนนิ่งจะสำเร็จก็ต่อเมื่อตัวอย่างทุกตัวได้ค่าความผิดพลาดอยู่ในช่วงที่ยอมรับได้ และถ้าการเทรนไม่สามารถเทรนสำเร็จจนกระทั่งรอบการเทรนถึงค่าที่เรากำหนด (Maximum Epoch) การเทรนจะหยุดลงและเทรนไม่สำเร็จ ต้องกำหนดค่าต่างๆ ใหม่และต้องทำการเทรนใหม่

3.4.7 หลังจากเทรนนิวรอลสำเร็จแล้ว ผู้ใช้สามารถนำไปใช้ได้ทันที นิวรอลจะสามารถตัดสินคุณภาพผ้าเมื่อมีภาพต่าง ๆเข้ามาได้ทันที รวมทั้งยังสามารถเซฟฐานข้อมูล (Data Base) หลังจากการเทรนนั้นได้ เพื่อสามารถนำไปใช้ได้ภายหลัง

3.5 ระบบควบคุม

ระบบควบคุมแบบ OPEN LOOP ประกอบด้วย

1. Input Unit ประกอบด้วย ภาพของผ้าย้อมและกล้อง
2. Controller ประกอบด้วย คอมพิวเตอร์และโปรแกรมควบคุม
3. Output Unit ประกอบด้วย Actuator Alarm Signal และ Motor



ภาพที่ 3-4 แสดงระบบการทำงานที่ออกแบบขึ้น

ในท้องตลาดปัจจุบันนั้น หากแบ่งตามหน้าที่การประมวลผลภาพแล้ว จะสามารถแบ่งกล้องที่ใช้สำหรับงานตรวจสอบทางอุตสาหกรรมออกเป็น 2 ประเภทหลักๆ คือ

1. กล้องที่รับภาพและส่งข้อมูลภาพมาที่คอมพิวเตอร์ ซึ่งอาจจะส่งผ่านทาง USB หรือ FireWire

2. กล้องที่รับภาพและมีความสามารถในการประมวลผลอยู่ในตัว ซึ่งในท้องตลาดเรียกว่า Smart Camera

จะพบว่า เมื่อเปรียบเทียบระหว่าง 2 ประเภทนี้ จะพบว่า กล้องประเภทแรกนั้น เหมาะสำหรับงานที่มีความเร็วในการเคลื่อนที่ไม่มาก นอกจากนั้น ยังมีราคาถูกกว่าประเภทที่ 2 เป็นอย่างมาก ดังนั้นในระบบที่ออกแบบขึ้นจึงเลือกใช้งานกล้องประเภทแรกนี้ อย่างไรก็ตาม จำนวนกล้องที่เลือกมาใช้งานนั้น จะต้องนำอัตราขยายของเลนส์, ขนาดที่เล็กที่สุดของผืนผิวเสียที่ต้องการตรวจจับได้ และความกว้างของผืนผ้าทั้งหมด มาพิจารณาอีกทีหนึ่ง

จากภาพที่ 3-4 ซึ่งแสดงระบบการทำงานของทั้งระบบนั้น จะพบว่า การประมวลผลทั้งหมดจะกระทำโดยโปรแกรม Color Inspection System ที่อยู่ในคอมพิวเตอร์เกรดอุตสาหกรรม ทั้งนี้ เนื่องจากสภาพแวดล้อมการทำงานจริงนั้น มีฝุ่นและความสั่นสะเทือน ซึ่งส่งผลต่ออายุการใช้งานของคอมพิวเตอร์เป็นอย่างมาก หากนำเครื่องคอมพิวเตอร์ทั่วไปมาใช้ก็จะมีอายุการใช้งานสั้นเป็นอย่างมาก

การทำงานของโปรแกรมนั้น จะรับข้อมูลภาพจากกล้อง UCM1300 ซึ่งสามารถต่อเข้าโดยตรงกับพอร์ต USB และสามารถทำงานร่วมกับโปรแกรมที่เขียนขึ้นได้ทันที เนื่องจากกล้องดังกล่าวนั้นสนับสนุนการทำงานของ Video For Window Driver ซึ่งกล้องจะทำการจับภาพของผืนผ้าตลอดเวลา ดังนั้น บริเวณของผืนผ้าจะปรากฏอย่างต่อเนื่องในภาพ เช่น ในภาพแรกบริเวณหนึ่ง อาจจะอยู่ในส่วนบนของภาพ ในขณะที่ผืนผ้ามีการเคลื่อนที่ภาพที่ 2 ที่กล้องจับได้นั้น บริเวณนั้น อาจจะปรากฏอยู่ที่ตอนกลางหรือตอนล่างของภาพที่จับ ซึ่งสิ่งนี้เองที่จะส่งผลต่ออัตราการไหลของผืนผ้านั้นเอง

จากภาพที่รับได้นั้น โปรแกรมจะทำการตรวจสอบและตัดสินคุณภาพของผืนผ้าว่าดีหรือเสีย ซึ่งในกรณีที่เสียนั้นก็จะสั่งให้มอเตอร์หยุดหมุน และส่งสัญญาณไฟ Alarm ดังแสดงไว้ในภาพ 3-4 และหากเป็นผืนผ้าดีก็จะสั่งให้มอเตอร์ที่ใช้ดึงผ้าทำงานต่อไปเรื่อยๆ ซึ่งจะพบว่า การส่งสัญญาณจากคอมพิวเตอร์ไปทั้งส่วนที่เป็นมอเตอร์หรือไฟ Alarm นั้น คอมพิวเตอร์จะสั่งผ่านการ์ด I/O รุ่น NI6509 ซึ่งมีข้อดีคือ ในขณะที่เริ่มเปิดคอมพิวเตอร์ซึ่งโปรแกรมยังไม่ได้ทำงานนั้น ผู้ใช้สามารถกำหนดค่าที่พอร์ตขาออกของการ์ดดังกล่าวซึ่งเป็นค่าที่ถูกต้องและเก็บค่าดังกล่าวไว้ในความจำแบบไม่ลบเมื่อไม่มีไฟเลี้ยง และให้การ์ดจ่ายสัญญาณดังกล่าวออกมาเมื่อไม่มีโปรแกรมควบคุมได้ คุณสมบัติข้อนี้สำคัญกับงานควบคุมเครื่องจักรเป็นอย่างมาก มิฉะนั้น ในขณะเริ่มเปิดคอมพิวเตอร์ที่มีการต่อสัญญาณควบคุมมอเตอร์ มอเตอร์อาจจะเดินขึ้นมาอย่างกะทันหันได้

อย่างไรก็ตาม ขนาดแรงดันและกระแสของการ์ดดังกล่าวนั้น ไม่สามารถนำไปใช้ควบคุมมอเตอร์หรือไฟ Alarm ได้โดยตรง ดังนั้น จึงจำเป็นต้องใช้วงจร Transistor มาเป็นตัวควบคุมการนำกระแสหรือหยุดนำของอุปกรณ์เหล่านี้ อีกทีหนึ่ง

3.6 อุปกรณ์หลักที่สามารถนำไปใช้ในงานจริง

3.6.1 Digital Imaging Camera (UCM1300)

3.6.1.1 ทำหน้าที่เป็น Input Devices ของระบบ

3.6.1.2 เนื่องจากกล้อง UCM1300 สามารถต่อเข้าคอมพิวเตอร์ได้โดยผ่านพอร์ท USB ทำให้สะดวกในการใช้โดยไม่ต้องซื้อ Video Card เพิ่มเติม

3.6.1.3 ความละเอียดในการจับภาพแบบ Real Time ของกล้องสูงถึง 1.3 ล้านพิกเซล ซึ่งเพียงพอและใกล้เคียงกับความสามารถของมนุษย์ในการแยกแยะความละเอียดของภาพที่ 1.2 ล้านพิกเซล

3.6.1.4 ความลึกบิตสีที่ 24 บิต ซึ่งสามารถแสดงสีได้ถึง 16.7 ล้านสีและเพียงพอเมื่อเทียบกับมนุษย์ที่สามารถแยกแยะสีได้ 14.2 ล้านสี

3.6.1.5 นอกจากนี้พอร์ท USB ยังมีความเร็วในการถ่ายโอนข้อมูลสูงถึง 12 Mbits/s ซึ่งเร็วกว่าพอร์ท Serial Port หรือ RS232C ถึง 109 เท่า ทำให้การถ่ายโอนภาพแบบ Real Time มีประสิทธิภาพและรวดเร็วมากยิ่งขึ้น

3.6.2 NI6509

3.6.2.1 ทำหน้าที่เป็น Actuator ของระบบ

3.6.2.2 รับคำสั่งจากคอมพิวเตอร์ (Controller)

3.6.2.3 เป็น I/O Board ที่ทำหน้าที่ควบคุมดีซีมอเตอร์และสัญญาณไฟ Alarm

3.6.3 Drive Circuit

3.6.3.1 ทำหน้าที่เป็นวงจรทรานซิสเตอร์หรือเปิด-ปิดการทำงานของดีซีมอเตอร์และสัญญาณไฟ Alarm

3.6.3.2 รับสัญญาณจาก I/O Board “NI6509”

3.6.4 DC Motor

3.6.4.1 ทำหน้าที่ขับสายพานลำเลียงผ้าย้อม

3.6.4.2 จะถูกหยุดการทำงานเมื่อโปรแกรมตรวจพบความบกพร่องของผ้าย้อม

3.6.5 LED Lamp

3.6.5.1 ทำหน้าที่แสดงสัญญาณไฟว่าตรวจพบความบกพร่องของผ้าย้อมโดยดีซีมอเตอร์จะหยุดทำงาน

3.6.5.2 จะแสดงสัญญาณไฟก็ต่อเมื่อโปรแกรมตรวจพบความบกพร่องของผ้าย้อม

3.7 ราคาประเมินอุปกรณ์หลักที่สามารถนำไปใช้ได้จริง

3.7.1 Digital Imaging Camera (UCM1300)	24,000	บาท
3.7.2 I/O Board (NI6509)	11,500	บาท
3.7.3 Drive Circuit	2,500	บาท
3.7.4 Signal Tower & Alarm	2,000	บาท
รวม	40,000	บาท

3.8 อุปกรณ์หลักที่ใช้ในงานวิจัย

3.8.1 กล้อง USB Web Cam

3.8.1.1 ใช้สำหรับออกแบบและทดสอบโปรแกรม

3.8.1.2 ความละเอียดในการจับภาพแบบ Real Time ไม่สูงสำหรับใช้งานจริง

3.8.1.3 ความลึกบิตสีที่ 24 บิต ซึ่งสามารถแสดงสีได้ถึง 16.7 ล้านสีและเพียงพอเมื่อเทียบกับมนุษย์ที่สามารถแยกแยะสีได้ 14.2 ล้านสี

3.8.1.4 นอกจากนี้พอร์ท USB ยังมีความเร็วในการถ่ายโอนข้อมูลสูงที่ 12 Mbits/s ซึ่งเร็วกว่าพอร์ท Serial Port หรือ RS232C ถึง 109 เท่า ทำให้การถ่ายโอนภาพแบบ Real Time มีประสิทธิภาพและรวดเร็วมากยิ่งขึ้น

3.8.2 กล้องดิจิตอล

3.8.2.1 ใช้สำหรับออกแบบและทดสอบโปรแกรมที่ความละเอียดของภาพที่ใช้จริงและเทียบเท่ากับความละเอียดที่มนุษย์สามารถมองเห็นคือประมาณ 1.2 ล้านพิกเซล

3.8.2.2 ใช้จับภาพแบบภาพนิ่งเท่านั้นและโหลดเข้าสู่โปรแกรมดังไฟล์ภาพ BMP

3.8.3 ตัวอย่างผ้าย้อมจากโรงงานผ้าย้อม

3.8.3.1 เป็นผ้าจากโรงงานผ้าย้อมจริงๆ ที่มี Defect เกิดขึ้นโดยการตรวจสอบจากสายตามนุษย์

3.8.3.2 ใช้สำหรับทดสอบโปรแกรม

บทที่ 4
ผลการวิจัย

บทนี้จะกล่าวถึงการทดลองกับตัวอย่างผ้าย้อม โดยจะแบ่งเป็นการทดลองที่ทำงานโดยวิธีการทางสถิติ (Statistical Method) และการทดลองที่ทำงานโดยวิธีการทางโครงข่ายประสาทเทียม (Artificial Neural Network Method) เพื่อเปรียบเทียบข้อดีข้อเสียและประสิทธิภาพต่างๆ ของทั้งสองวิธี

4.1 ชุดทดลอง

ประกอบด้วย

4.1.1 ตัวอย่างผ้าย้อมที่มีพื้นผิวเสีย 10 ตัวอย่าง

4.1.2 ตัวอย่างผ้าย้อมที่มีพื้นผิวดี 10 ตัวอย่าง

4.1.3 หลอดไฟแสงสีขาว

4.1.4 กล้องดิจิตอล

4.1.5 โปรแกรมตรวจสอบความบกพร่องของผ้าย้อมแบบสีโทนเดียว

4.2 ขั้นตอนการทดลองที่ทำงานโดยวิธีการทางสถิติ (Statistical Method)

4.2.1 ถ่ายภาพตัวอย่างผ้าย้อมที่มีพื้นผิวเสียและตัวอย่างผ้าย้อมที่มีพื้นผิวดี อย่างละ 10 ตัวอย่าง โดยกล้องดิจิตอลด้วยความละเอียด 1280 x 960 หรือ 1.2 ล้านพิกเซล

4.2.2 นำภาพดิจิตอลเข้าสู่โปรแกรมตรวจสอบความบกพร่องของผ้าย้อมแบบสีโทนเดียว

4.2.3 ทำการวิเคราะห์กับภาพตัวอย่างที่มีพื้นผิวเสียกับ 1 ตัวอย่าง เพื่อหาลักษณะของพื้นผิวเสียโดยโปรแกรมจะวิเคราะห์หาค่าเทรดโชรว์ที่เหมาะสม

4.2.4 กำหนดช่วงค่าของค่าคุณสมบัติที่เราสนใจว่าเป็นพื้นผิวเสียลงในโปรแกรม

4.2.5 ทดลองกับภาพผ้าย้อมตัวอย่างอื่นที่มีสีพื้นหลังเดียวกันหรือใกล้เคียงกัน

4.2.6 เก็บผลการทดลองที่ได้

4.3 ขั้นตอนการทดลองที่ทำงานโดยวิธีการทางโครงข่ายประสาทเทียม (Artificial Neural Network Method)

4.3.1 ถ่ายภาพตัวอย่างผ้าย้อมที่มีพื้นผิวเสียและตัวอย่างผ้าย้อมที่มีพื้นผิวดี อย่างละ 10 ตัวอย่าง โดยกล้องดิจิตอลด้วยความละเอียด 1280 x 960 หรือ 1.2 ล้านพิกเซล

4.3.2 นำภาพดิจิตอลเข้าสู่โปรแกรมตรวจสอบความบกพร่องของผ้าย้อมแบบสีโทนเดียว

4.3.3 ทำการวิเคราะห์กับภาพตัวอย่างที่มีพื้นผิวเสียกับ 1 ตัวอย่าง เพื่อหาลักษณะของพื้นผิวเสีย

4.3.4 เลือกพื้นผิวเสียที่สนใจว่าเป็นพื้นผิวเสียลงในโปรแกรม

4.3.5 กำหนดพารามิเตอร์ต่างๆสำหรับการเทรนนิวรอลและทำการเทรนนิวรอล

4.3.6 ถ้าโปรแกรมไม่สามารถเทรนนิวรอลสำเร็จ ทำการกำหนดพารามิเตอร์สำหรับการเทรนใหม่และทำการเทรนอีกครั้ง

4.3.7 หลังจากเทรนนิวรอลสำเร็จแล้ว ทดลองกับภาพผ้าย้อมตัวอย่างอื่นที่มีลักษณะพื้นผิวเสียใกล้เคียงกัน

4.3.8 เก็บผลการทดลองที่ได้

4.4 การจัดสภาพแวดล้อม (Scene Constraint) ในการทดลอง

4.4.1 การวางตัวของชิ้นงาน ชิ้นงานถูกจัดให้วางตัวในทิศทางเดียวกัน

4.4.2 การจัดการเรื่องแสง ติดตั้งแหล่งกำเนิดแสงไว้ที่ด้านเดียวกับตัวกล้อง (Front Lighting) แล้วส่องไปที่วัตถุที่ต้องการจับภาพ

4.4.3 ระยะระหว่างกล้องหรือเลนส์ถึงชิ้นงานคือ 30-50 เซนติเมตร และทำมุม 90 องศา กับจุดศูนย์กลางของชิ้นงาน

4.4.4 ค่าความสว่างที่ชิ้นงานคือ 530 ลักซ์ โดยแหล่งกำเนิดแสงที่มีแสงสีขาวอยู่ห่างจากชิ้นงาน 1.8 เมตร และทำมุม 90 องศา กับจุดศูนย์กลางของชิ้นงาน

4.5 พารามิเตอร์ต่าง ๆ ในโปรแกรมที่ใช้ในการทดลอง

4.5.1 สำหรับวิธีทางสถิติ

ช่วงของขนาดพื้นที่ของพื้นผิวเสียที่สนใจ คือ 50 พิกเซล ถึง ขนาดสูงสุดที่ปรากฏในภาพ

4.5.2 สำหรับวิธีทางโครงข่ายประสาทเทียม

ข้อมูลเข้า 6 ค่า คือ ค่าเฉดสี (Hue)

ค่าความเข้มแสง (Intensity)

ค่าความอิ่มตัวของสี (Saturation)

ค่าพื้นที่ (Area)

ค่าจุดศูนย์ถ่วง (Centroid)

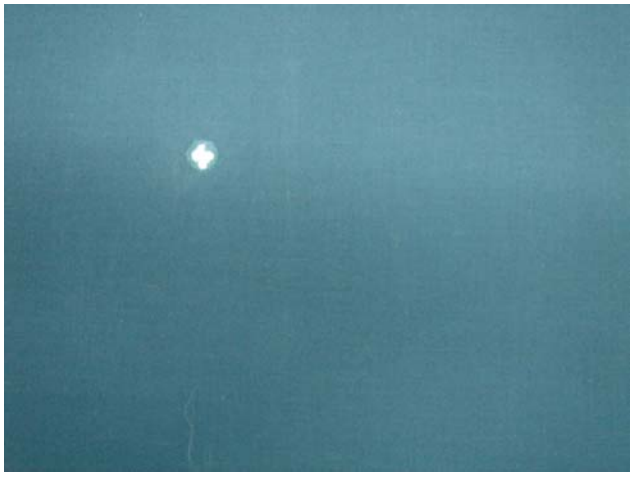
ค่า Bounding Box

ค่าเส้นรอบรูป (Perimeter)

ค่าอัตราการเรียนรู้ (Learning Rate) ในปมซ่อนของชั้นที่หนึ่งคือ 0.65
ค่าอัตราการเรียนรู้ (Learning Rate) ในปมซ่อนของชั้นที่สองคือ 0.65
ค่าอัตราการเรียนรู้ (Learning Rate) ในปมซ่อนของชั้นที่สามคือ 0.65
ค่าสัมปะสิทธ์โมเมนตัม (Momentum Coefficient) ในปมซ่อนของชั้นที่หนึ่งคือ 0.4
ค่าสัมปะสิทธ์โมเมนตัม (Momentum Coefficient) ในปมซ่อนของชั้นที่สองคือ 0.4
ค่าสัมปะสิทธ์โมเมนตัม (Momentum Coefficient) ในปมซ่อนของชั้นที่สามคือ 0.4
จำนวนปมซ่อน (Hidden node) ในชั้นที่หนึ่ง คือ 20
จำนวนปมซ่อน (Hidden node) ในชั้นที่สอง คือ 20
ค่าถ่วงน้ำหนักเชื่อมโยง (Connection weight) เริ่มต้น คือ ค่าสุ่มระหว่าง –1.5 ถึง 1.5
ค่าขีดเริ่มเปลี่ยน (Bias) เริ่มต้น คือ ค่าสุ่มระหว่าง –1.5 ถึง 1.5
โหมดการ Train คือ Batch mode
ช่วงยอมรับ (Tolerance) หรือ ผลต่างความผิดพลาด คือ 8 % หรือ 0.08
จำนวนรอบสูงสุดในการ Train (Maximum Epoch) คือ 30000 รอบ

4.6 ผลการทดลอง

เป็นการทดลองโดยเปรียบเทียบประสิทธิภาพต่างๆ ระหว่างวิธีทางสถิติ (Statistical Method) และวิธีทางโครงข่ายประสาทเทียม (Artificial Neural Network Method) กับภาพผ้าย้อมที่มีพื้นผิวดีและพื้นผิวเสียอย่างละ 10 ตัวอย่างจากโรงงานผ้าย้อมจริง โดยแสดงภาพตัวอย่างที่ใช้ในการทดลองบางภาพในภาพที่ 4-1, 4-2, 4-3, 4-4, 4-5, 4-6 และ 4-7



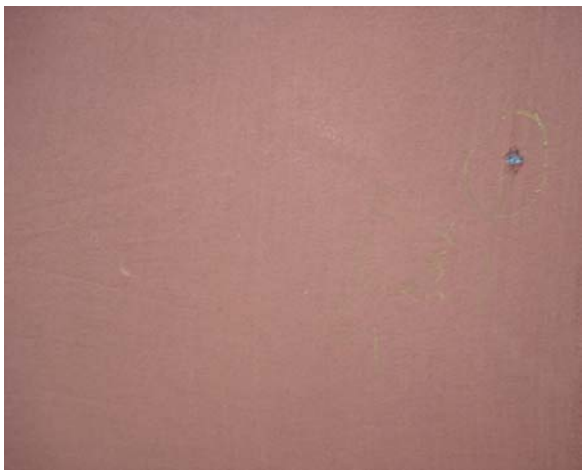
ภาพที่ 4-1 ลักษณะภาพที่มีพื้นผิวเสียเป็นรอยขาดเป็นรู

ภาพที่ 4-2 ลักษณะภาพที่มีพื้นผิวเสียเป็นรอยด่าง

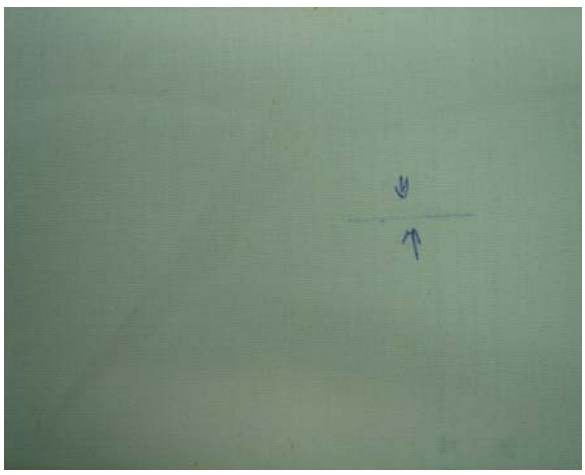
ภาพที่ 4-3 ลักษณะภาพที่มีพื้นผิวเสียเป็นรอยด่างยาว

ภาพที่ 4-4 ลักษณะภาพที่มีพื้นผิวเสียเป็นรอยขาด

ภาพที่ 4-5 ลักษณะภาพที่มีพื้นผิวเสียเป็นรอยด่าง



ภาพที่ 4-6 ลักษณะภาพที่มีพื้นผิวเสียเป็นรอยขาด



ภาพที่ 4-7 ลักษณะภาพที่มีพื้นผิวเสียเป็นรอยสี

ตารางที่ 4-1 ผลการทดลองโดยเปรียบเทียบความแม่นยำ

ผ้าตัวอย่างที่	ลักษณะทั่วไปของผิวเสีย			การตรวจสอบโดยวิธีทางสถิติ		การตรวจสอบโดยโครงข่ายประสาทเทียม		หมายเหตุ
	ลักษณะ	จำนวนผิวเสียจริง	ขนาดผิวเสีย (พิกเซล)	จำนวนผิวเสียที่พบ	ผลตรวจสอบ	จำนวนผิวเสียที่พบ	ผลตรวจสอบ	
1	รอยขาด	1	583	1	ถูกต้อง	1	ถูกต้อง	เทรนได้ 575 รอบ
2	รอยขาด	1	1302	1	ถูกต้อง	1	ถูกต้อง	
3	รอยขาด	1	507	1	ถูกต้อง	1	ถูกต้อง	
4	รอยขาด	1	667	1	ถูกต้อง	1	ถูกต้อง	
5	รอยขาด	1	839	1	ถูกต้อง	1	ถูกต้อง	
6	ผ้าดี	0	-	0	ถูกต้อง	0	ถูกต้อง	
7	ผ้าดี	0	-	0	ถูกต้อง	0	ถูกต้อง	
8	ผ้าดี	0	-	0	ถูกต้อง	0	ถูกต้อง	
9	ผ้าดี	0	-	0	ถูกต้อง	0	ถูกต้อง	
10	ผ้าดี	0	-	0	ถูกต้อง	0	ถูกต้อง	
11	รอยด่าง	1	1504	1	ถูกต้อง	1	ถูกต้อง	เทรนได้ 3672 รอบ
12	รอยด่าง	1	1388	1	ถูกต้อง	1	ถูกต้อง	
13	รอยด่าง	1	955	0	ผิดพลาด	1	ถูกต้อง	
14	รอยด่าง	1	1127	1	ถูกต้อง	1	ถูกต้อง	
15	รอยด่าง	1	1893	1	ถูกต้อง	1	ถูกต้อง	
16	ผ้าดี	0	-	0	ถูกต้อง	0	ถูกต้อง	
17	ผ้าดี	0	-	0	ถูกต้อง	0	ถูกต้อง	
18	ผ้าดี	0	-	2	ผิดพลาด	0	ถูกต้อง	
19	ผ้าดี	0	-	2	ผิดพลาด	1	ผิดพลาด	
20	ผ้าดี	0	-	0	ถูกต้อง	0	ถูกต้อง	
ตรวจสอบถูกต้อง					17		19	
ตรวจสอบผิดพลาด					3		1	
% ความผิดพลาด					15%		5%	

ตารางที่ 4-2 ผลการทดลองโดยเปรียบเทียบความเร็วในการประมวลผล

ผ้าตัวอย่างที่	ลักษณะทั่วไปของผิวเสีย			การตรวจสอบโดยวิธีทางสถิติ	การตรวจสอบโดยโครงข่ายประสาทเทียม
	ลักษณะ	จำนวนผิวเสีย	ขนาดผิวเสีย (พิกเซล)	เวลาประมวลผล (วินาที)	เวลาประมวลผล (วินาที)
1	รอยขาด	1	583	-	-
2	รอยขาด	1	1302	2.51	2.40
3	รอยขาด	1	507	2.40	2.33
4	รอยขาด	1	667	2.37	2.33
5	รอยขาด	1	839	2.36	2.25
6	ผ้าดี	0	-	2.32	2.46
7	ผ้าดี	0	-	2.35	2.40
8	ผ้าดี	0	-	2.41	2.41
9	ผ้าดี	0	-	2.33	2.48
10	ผ้าดี	0	-	2.35	2.34
11	รอยด่าง	1	1504	-	-
12	รอยด่าง	1	1388	3.25	3.20
13	รอยด่าง	1	955	3.45	3.05
14	รอยด่าง	1	1127	3.23	3.03
15	รอยด่าง	1	1893	3.33	3.11
16	ผ้าดี	0	-	3.27	3.06
17	ผ้าดี	0	-	3.22	2.93
18	ผ้าดี	0	-	3.29	3.16
19	ผ้าดี	0	-	3.18	3.19
20	ผ้าดี	0	-	3.34	3.02
เวลาเฉลี่ยในประมวลผล (วินาที)				2.83	2.73

ตารางที่ 4-3 ผลการทดลองโดยเปรียบเทียบเวลาในการกำหนดค่าพารามิเตอร์เริ่มต้น

ผ้าตัวอย่างที่	ลักษณะทั่วไปของผิวเสีย			การตรวจสอบโดยวิธีทางสถิติ	การตรวจสอบโดยโครงข่ายประสาทเทียม	
	ลักษณะ	จำนวนผิวเสีย	ขนาดผิวเสีย(พิกเซล)	เวลา (นาที)	รอบเทรน (รอบ)	เวลา (นาที)
1	รอยขาด	1	583	0.43	575	1.03
2	รอยด่าง	1	1504	2.40	3672	1.41
3	รอยขาดยาว	1	10303	2.19	95	1.08
4	รอยมาร์ค	1	483	2.26	1703	1.39
5	รอยขาดเป็นรู	1	1872	0.35	300	0.38
6	รอยปั้มสี	7	240-283	0.17	17	0.40
7	รอยสีมาร์ค	1	345	1.42	772	1.32
8	รอยขาดยาว	1	8341	1.55	221	1.13
9	รอยเปื้อนสี	3	466-920	2.13	1302	1.45
10	รอยเปื้อนดำ	1	1020	1.55	1274	1.43
11	รอยด่างสี	6	301-525	2.23	103	2.02
12	รอยขาดจุด	1	352	1.59	1245	2.01
13	รอยย้วยผ้า	2	5063, 6532	4.34	8071	4.11
14	รอยแต้มสี	1	793	2.00	308	1.53
15	รอยวงผ้า	1	3427	2.21	1609	1.55
16	รอยยับ	1	2083	3.52	4896	3.15
17	รอยลอก	1	1841	4.02	5019	4.00
18	รอยด้ายขาด	1	973	2.42	1023	1.59
19	รอยแต้มสี	2	2031, 3041	2.32	892	1.48
20	รอยจางสี	1	1806	4.40	3019	3.49
เวลาเฉลี่ยในการกำหนดพารามิเตอร์เริ่มต้น (นาที)				2.21.70		1.59.75

4.7 เปรียบเทียบผลของวิธีทางสถิติ (Statistical Method) และวิธีทางโครงข่ายประสาทเทียม (Artificial Neural Network Method)

ตารางที่ 4-4 เปรียบเทียบผลของ Statistical และ Neural Methods

คุณสมบัติ	Statistical Method	Neural Method
การคลาสซิฟาย	ต้องกำหนดค่าสำหรับ defect ในเชิงปริมาณ	เลือก defect ตัวอย่างจากภาพได้เลย
ความรวดเร็วในการประมวลผลและ Classify	2.83 วินาที	2.73 วินาที
เวลาในการกำหนดพารามิเตอร์	2.21 นาที	1.59 นาที
ความยุ่งยากในการกำหนดพารามิเตอร์	ค่อนข้างยุ่งยากกว่า	กำหนดค่าที่จะใช้ในการเทรนได้ง่ายกว่า
ความสะดวก	ต้องกำหนดหาค่าช่วงที่เหมาะสมเอง	แค่คลิกบนวัตถุ defect แล้วเทรนตัวอย่าง
ความแม่นยำ	ใช้ได้ แต่ขึ้นอยู่กับภาพและการกำหนดค่า	สูง แต่ต้องเป็น defect ที่มีลักษณะใกล้เคียงกับ defect ที่เคยเทรนมาแล้ว
จุดด้อย	ถ้ามีกลุ่มสีมากกว่า 2 กลุ่มหรือความสว่างของ defect กับพื้นหลังใกล้เคียงกัน โปรแกรมอาจจะไม่สามารถตรวจพบdefect	อาจใช้เวลาเทรนนานและอาจจะไม่สามารถเทรนสำเร็จครบทุก data set และโปรแกรมอาจจะไม่สามารถตรวจพบ defect ตัวใหม่ๆได้

4.8 สรุปผลการทดลอง

การทำงานของเครื่องมือวัดทั้งสองชนิดนั้น มีประสิทธิภาพในการทำงานไม่แตกต่างกันมากนัก เริ่มจากความเร็วในการประมวลผลนั้น แทบจะไม่แตกต่างกันเลย โดยวิธีโครงข่ายประสาทเทียมจะเร็วกว่าการทำงานโดยวิธีทางสถิติเล็กน้อย สำหรับเวลาในการกำหนดพารามิเตอร์เริ่มต้นนั้นก็ไม่แตกต่างกันมากนัก โดยการทำงานโดยโครงข่ายประสาทเทียมนั้นจะใช้เวลาเฉลี่ยน้อยกว่าการทำงานโดยวิธีทางสถิติไม่ถึงนาทีเท่านั้น แต่ทั้งนี้ทั้งนั้น เวลาการตั้งค่าโดยโครงข่ายประสาทเทียมอาจมากกว่าวิธีทางสถิติในบางครั้ง ซึ่งขึ้นอยู่กับลักษณะของพื้นผิวเสีย การกำหนดค่าการเทรนนิ่งและจำนวนรอบที่เทรนสำเร็จ ส่วนความแม่นยำของเครื่องมือวัดทั้งสองชนิดนั้น จากการทดลองจะพบว่าการทำงานโดยโครงข่ายประสาทเทียมให้ความแม่นยำที่สูงกว่าการทำงานโดยวิธีทางสถิติพอสมควร

จากการทดลอง จะพบว่าประสิทธิภาพในการตรวจสอบของเครื่องมือวัดทั้งสองชนิดนั้นจะมีประสิทธิภาพบางอย่างที่ไม่แตกต่างกันมากนักคือ เวลาในการกำหนดพารามิเตอร์เริ่มต้นและความเร็วในการประมวลผล แต่ความแม่นยำของเครื่องมือที่ทำงานโดยโครงข่ายประสาทเทียมนั้น จะมีความถูกต้องมากกว่า และสามารถนำไปใช้กับภาพที่มีพื้นหลังแบบอื่นๆที่มีลักษณะของพื้นผิวเสียใกล้เคียงกันได้ดี เมื่อเปรียบเทียบกับเครื่องมือวัดที่ทำงานโดยวิธีทางสถิตินั้น จะพบว่า เครื่องมือชนิดหลังนี้จะมีการทำงานที่ผิดพลาดอยู่มาก โดยเฉพาะอย่างยิ่งเมื่อนำไปตรวจสอบกับภาพที่มีพื้นหลังแตกต่างหรือคนละเฉดสี

บทที่ 5
บทสรุปและข้อเสนอแนะ

5.1 บทสรุป

การนำระบบโครงข่ายประสาทเทียม (Artificial Neural Network) มาใช้เพื่อจำแนกว่า BLOB ที่ตรวจจับได้นั้น มีชิ้นใดบ้างที่จัดเป็นพื้นผิวงานเสีย ซึ่งเมื่อพิจารณาจากมุมมองของผู้ใช้ ระบบจะสามารถทำงานได้อย่างอัตโนมัติมากขึ้น เพราะเมื่อเปรียบเทียบกับวิธีการทำงานแบบสถิติ ซึ่งเป็นการจำแนก BLOB ที่เป็นพื้นผิวงานเสีย โดยการเปรียบเทียบค่าคุณสมบัติต่างๆ ของ BLOB ที่ตรวจจับได้ กับค่าคุณสมบัติของพื้นผิวงานเสีย ที่ผู้ใช้งานระบบจะต้องทำการกำหนดด้วยตนเอง ซึ่งจะต้องทำการกำหนดค่าและดูผลการทำงานกลับไปกลับมาจนกว่าจะได้ผลที่พอใจ สำหรับคุณสมบัติของ BLOB ที่นำมาเปรียบเทียบกันนี้ได้แก่

1. มุมของแกนหลัก (Angle of Major Axis)
2. พื้นที่ (Area)
3. เส้นรอบรูป (Perimeter)
4. ขนาดของ Bounding Box (Bounding Box Width and Height)
5. ตำแหน่งศูนย์กลาง (Centroid)
6. ค่าเฉดสีเฉลี่ย (Average Hue)
7. ค่าความอิ่มตัวสีเฉลี่ย (Average Saturation)
8. ค่าเข้มสีเฉลี่ย (Average Intensity)

จากข้อมูลเหล่านี้ หากผู้ใช้เลือกวิธีการทำงานแบบสถิติ ผู้ใช้ก็จะต้องกำหนดค่าสูงสุดและต่ำสุดของคุณสมบัติแต่ละอย่าง เพื่อเป็นการกำหนดความหมายของพื้นผิวงานเสีย ซึ่งการทำงานดังกล่าวต้องใช้เวลาพอสมควรทั้งการเรียนรู้วิธีใช้งานและทำการทดลองหาค่าที่เหมาะสม อีกทั้งเมื่อมองจากมุมมองของผู้ใช้งานนั้น โดยมากแล้ว ผู้ใช้จะมีความคุ้นเคยกับงานของตนเองเป็นอย่างดี อีกทั้งยังสามารถระบุได้อีกด้วยว่า จากภาพของ BLOB ที่จับได้นั้นมีชิ้นใดเป็นพื้นผิวเสีย แต่เมื่อต้องการให้ระบุคุณสมบัติของ BLOB ชิ้นนั้นเป็นเชิงปริมาณหรือเชิงตัวเลขนั้น ดูจะทำให้เกิดความยุ่งยากและกินเวลาในการทำงานของผู้ใช้งานระบบเป็นอย่างมาก

ดังนั้น โครงข่ายประสาทเทียมจึงมาช่วยลดงานในส่วนนี้ ซึ่งถ้ามองจากมุมมองของผู้ใช้ ผู้ใช้เพียงแค่เตรียมข้อมูลตัวอย่างว่า BLOB ใดบ้างที่จัดเป็นพื้นผิวดี และชิ้นใดบ้างเป็นพื้นผิวเสีย (ซึ่งผู้ใช้สามารถจัดเตรียมตัวอย่างเหล่านี้ได้โดยง่าย โดยการกดเม้าส์ลงไปในภาพเพื่อเปลี่ยนกลับไปกลับมาว่า BLOB ในตำแหน่งนั้นๆ เป็นพื้นผิวดีหรือพื้นผิวเสีย) หลังจากป้อนตัวอย่าง

เหล่านี้ให้กับโปรแกรมแล้ว โปรแกรมจะมีความสามารถในการตัดสินใจได้อย่างอัตโนมัติ โดยที่ผู้ใช้ไม่จำเป็นต้องระบุคุณสมบัติของ BLOB ที่เป็นพื้นผิวเสียในเชิงปริมาณแต่อย่างไรเลย นอกจากนั้นแล้ว จะพบว่าด้วยคุณสมบัติที่ไม่เป็นเชิงเส้นของโครงข่ายประสาทเทียมนั้น ทำให้ไม่จำเป็นต้องป้อนกลุ่มตัวอย่างที่เป็นไปได้ทั้งหมด แต่จะเป็นกลุ่มตัวอย่างที่สุ่มมาบางจำนวนก็เพียงพอแล้ว ซึ่งหลังจากนั้น โครงข่ายประสาทเทียมก็จะสามารถคาดเดาได้อย่างอัตโนมัติ แม้ว่าเป็น BLOB ที่ไม่เคยเห็นมาก่อน จึงสามารถกล่าวได้ว่า การนำระบบโครงข่ายประสาทเทียมมีความเหมาะสมที่จะนำมาใช้ตรวจสอบสีของผืนผ้าทั้งในแง่ของความง่ายต่อผู้ใช้ และประสิทธิภาพการทำงานโดยรวมของระบบ นอกจากนั้น จะพบว่า การทำงานของระบบตรวจสอบรอยด่างของผืนผ้าที่นำเสนอนั้น สามารถนำไปปรับใช้กับงานในลักษณะอื่นๆได้อย่างมากมาย เช่น จะเป็นการตรวจนับชิ้นส่วนที่มีลักษณะเฉพาะ การตรวจจับรอยเปื้อนปนบนพื้นผิวงานอื่นๆ เป็นต้น

5.2 ข้อเสนอแนะ

เนื่องจากระบบที่สร้างขึ้น ด้วยเวลาและงบประมาณที่จำกัดนั้น ทำให้มีข้อควรปรับปรุงบางประการที่ควรทำในโครงการในอนาคต เพื่อให้ระบบมีการทำงานที่ถูกต้องและมีประสิทธิภาพสูงสุด ดังนี้

1. แม้ว่าการทำงานของโครงข่ายประสาทเทียมจะสามารถทำให้ระบบตรวจสอบผืนผ้ามีการทำงานที่เป็นอัตโนมัติมากขึ้น เมื่อเมื่อเปรียบเทียบกับวิธีการทางสถิติที่เป็นการเปรียบเทียบแบบ Decision Tree แต่สิ่งหนึ่งที่เป็นข้อด้อยที่เห็นได้อย่างชัดเจนของโครงข่ายประสาทเทียมที่นำมาใช้คือ ผู้ใช้ไม่สามารถเข้าใจวิธีการตัดสินใจของโครงข่ายประสาทเทียมเลย หรือ กล่าวง่ายๆได้ว่า ผู้ใช้ไม่สามารถทราบได้ว่าจากคุณสมบัติของ BLOB ที่ป้อนให้กับโครงข่ายประสาทเทียมนั้น ตัวโครงข่ายใช้เงื่อนไขใดมาตัดสินว่า BLOB นั้นๆเป็นพื้นผิวดีหรือเสีย นอกจากนั้นจะพบว่า ค่าที่อยู่ Weight Matrix และ Bias Vector นั้นเป็นค่าจำนวนที่ไม่สื่อถึงปริมาณทางกายภาพใดๆเลย ซึ่งลักษณะการทำงานดังกล่าวนี้ อาจทำให้ผู้ใช้มีความมั่นใจในการใช้งานระบบที่ทำงานด้วยโครงข่ายประสาทเทียมลดลง ดังนั้น แนวทางในการพัฒนาระบบตรวจสอบผืนผ้าแบบอัตโนมัตินั้น คือการนำเอาตัวจำแนกชนิด Fuzzy Logic มาใช้เพราะเป็นระบบที่มีเงื่อนไขการตัดสินใจที่คล้ายคลึงกับหลักเหตุผลของสมองมนุษย์เป็นอย่างมาก ซึ่งสิ่งนี้เองจะเป็นการเพิ่มความมั่นใจให้กับผู้ใช้งานระบบ

2. ระบบโครงข่ายประสาทเทียมที่ถูก Train โดยระเบียบวิธีแบบแพร่กลับ (Back Propagation Training Algorithm) นั้น แม้ว่าจะเป็นระเบียบวิธีที่มีการใช้งานอย่างแพร่หลาย แต่จัดว่าเป็นวิธีที่ใช้เวลาในการ Train โครงข่ายค่อนข้างนานกว่าที่จะสามารถนำโครงข่ายไปใช้ได้ ดังนั้น ควรปรับปรุงความเร็วในการ Train โครงข่ายไปเป็นวิธีอื่นๆ เช่น Levenberg-Marquardt Algorithm ซึ่งจัดเป็นระเบียบวิธีที่ใช้ในการสอนโครงข่ายประสาทเทียมที่รวดเร็วที่สุด แต่มีการคำนวณที่ซับซ้อนและใช้พื้นที่หน่วยความจำในคอมพิวเตอร์มากที่สุดเช่นกัน

3. การทำงานของระบบที่นำเสนอนั้นเป็นการให้แสงในทิศทางเดียวกับตัวกล้อง และเป็นการตัดสินคุณภาพของผืนผ้าในขณะนั้นด้วยภาพที่รับเข้ามาเพียงภาพเดียว ซึ่งจะพบว่าลักษณะการเป็นพื้นผิวเสียบางอย่างนั้น จะมีสีแล้วคุณสมบัติต่าง ๆ ใกล้เคียงกับผ้าปกติเป็นอย่างมาก เช่น กรณีที่ผ้าเป็นปุ่มนูนหรือหนาผิดปกติ ซึ่งจะพบว่า ระบบที่ออกแบบขึ้นนั้น จะไม่สามารถใช้ตรวจสอบพื้นผิวเสียในลักษณะนี้ได้เลย ดังนั้น ควรจะมีการออกแบบระบบโดยมีการตรวจสอบ 2 ขั้นตอนต่อเนื่องกัน คือ ขั้นตอนที่จับภาพผืนผ้าโดยมีแสงอยู่ในทิศทางเดียวกับกล้อง หลังจากนั้น จึงทำอีกขึ้นตอนหนึ่งโดยให้แสงผ่านผืนผ้าแล้วย้อนเข้าสู่กล้องแทน ซึ่งจะพบว่า ด้วยการทำงานในขั้นตอนที่ 2 นี้เองจะสามารถตรวจจับพื้นผิวเสียในลักษณะดังกล่าวได้อย่างแน่นอน

เอกสารอ้างอิง

1. Ramesh Jain, Rangachar Kasturi and Brian G. Schunck. Machine Vision. n.p., 1995.
2. ศิริลักษณ์ แสนสมบูรณ์สุข และ Nitin V Afzulpurkar. โครงการการประเมินคุณภาพของเมล็ดข้าวโดยใช้ระบบวิเคราะห์ทางภาพ (Image Processing for Rice Kernel Quality Evaluation). Asian Institute of Technology.
3. Awcock, G.J. and Thomas, R. Applied Image Processing. n.p., 1996.
4. Pitas, I. Digital Image Processing Algorithms and Application. n.p., 2000.
5. Parker, J.R. Algorithms for Image Processing and Computer Vision. n.p., 1996.
6. Bryan S. Morse. “Thresholding.” [ออนไลน์] 2000. [สืบค้นวันที่ 11 มกราคม 2549]. จาก http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/MORSE/threshold.pdf
7. Di Stefano, L. and Bulgarelli, A. A Simple and Efficient Connected Components Labeling Algorithm. Image Analysis and Processing, 1999. Proceedings. International Conference on 27-29 Sept. 1999 : 322 – 327.
8. A Simple and Efficient Connected Components Labeling Algorithm [ออนไลน์] 14 มิถุนายน 2006. [สืบค้นวันที่ 13 ธันวาคม 2549]. จาก http://ieeexplore.ieee.org/search/wrapper.jsp?arnumber=797615
9. Hagan, Demuth and Beale. Neural Network Design. n.p., 1995.
10. Li-Xin Wang and Jerry M. Mendel, Generating fuzzy rules by learning from examples. IEEE Transactions on Systems, Man, and Cybernetics, vol. 22, no. 6, pp. 1414-1427, 1992.
11. Li-Xin Wang, Adaptive fuzzy systems and control: design and stability. n.p., 1994.
12. Polygon Scan Conversion [ออนไลน์] 19 กันยายน 2005. [สืบค้นวันที่ 15 ธันวาคม 2549]. จาก http://graphics.stanford.edu/courses/cs248-05/proj2help/assignment2help.pdf

ภาคผนวก ก

พื้นฐานโครงข่ายประสาทเทียม

ภาคผนวก ก
พื้นฐานโครงข่ายประสาทเทียม

ระบบเครือข่ายเส้นประสาท (Neural Network) เป็นอีกแขนงหนึ่งของ AI ที่ได้รับความสนใจ เป็นการเลียนแบบการทำงานของระบบเส้นประสาทและสมองมนุษย์ เพื่อนำมาประยุกต์ในการทำงาน และการประมวลผลของระบบคอมพิวเตอร์ให้มีประสิทธิภาพ ประการสำคัญคือทำให้คอมพิวเตอร์มีพัฒนาการที่ต่อเนื่อง และมีการทำงาน ใกล้เคียงกับการทำงานของสมองมนุษย์มากขึ้น จากการศึกษาเราสามารถอธิบายโดยสรุปได้ว่า ระบบประสาทประกอบด้วย เซลประสาท (Neuron) ซึ่งเปรียบเสมือนหน่วยประมวลผลย่อยที่ตอบสนองต่อการกระตุ้นจากภายนอกหรือเซลประสาทอื่น ซึ่งโดยปกติสมองมนุษย์มีเซลประสาทอยู่ถึงล้านล้านหน่วย โดยที่แต่ละเซลประสาทจะเชื่อมโยงต่อเนื่องเป็นระบบเครือข่ายที่ซับซ้อน และมีประสิทธิภาพในการส่งและรับกระแสประสาท โดยระบบประสาทของมนุษย์จะมีการทำงานผ่านการกระตุ้นของปฏิกิริยา ไฟฟ้าเคมี (Electrochemical Reaction) ที่เกิดขึ้นระหว่างเซลประสาท นอกจากเซลประสาทแล้วระบบเครือข่ายเส้นประสาทจะมีตัวรับ-ส่งสัญญาณ (Dendrite) ทำหน้าที่เชื่อมต่อและส่งข้อมูลระหว่างแต่ละส่วนภายในเครือข่ายเซลประสาทจะมีบริเวณต่อเชื่อมของเซลประสาท (Synapse) เป็นตัวเชื่อม ระหว่าง แต่ละระบบย่อยกับระบบย่อยอื่นภายในเครือข่ายเพื่อให้สามารถทำงานร่วมกัน อย่างสอดคล้อง และมีประสิทธิภาพ ซึ่งส่วนประกอบของระบบเส้นประสาท จะมีความสัมพันธ์ได้ในหลายลักษณะ

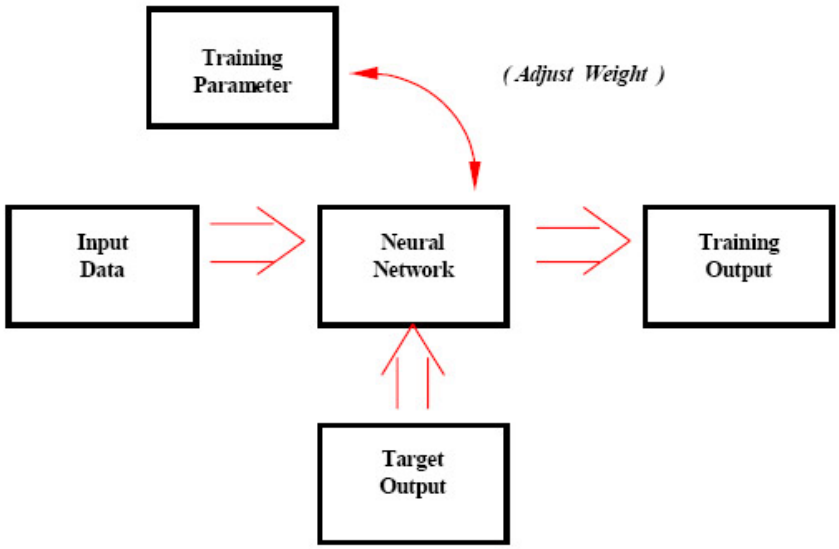
ระบบเครือข่ายเส้นประสาทจะต้องประกอบด้วยเซลประสาทที่ต่อเรียงกันเข้าเป็นระบบอย่างน้อยสองระดับ (Layer) โดยระดับแรก หรือที่เรียกว่า "ระดับนำเข้า (Input Layer)" ทำหน้าที่รับสิ่งนำเข้า (Input) จากสิ่งแวดล้อมภายนอก เข้าสู่ระบบแล้ว ทำการส่งต่อให้เครือข่ายในระดับถัดไปตามหน้าที่และความสัมพันธ์ที่ถูกกำหนด จนกระทั่งถึงระดับสุดท้าย หรือที่เรียกว่า "ระดับ แสดงผลลัพธ์ (Output Layer)" ซึ่งจะประกอบด้วยหน่วยแสดงผลที่ติดต่อสื่อสารกับผู้ใช้

ระบบเครือข่ายเส้นประสาทจะถูกพัฒนาให้เรียนรู้และจดจำจากประสบการณ์เช่นเดียวกันกับการทำงานในสมองมนุษย์ โดยผู้พัฒนาระบบจะต้องดำเนินการออกแบบ และทำการสอนให้ระบบเครือข่ายเส้นประสาทเรียนรู้จากกรณีตัวอย่าง โดยใช้ "กระบวนการถ่ายทอดแบบย้อนกลับ (Back Propagation)" ซึ่งเป็นการเรียนรู้จากผลลัพธ์กลับสู่สาเหตุ ผู้พัฒนาระบบจะต้องทำการสอนระบบเครือข่ายเส้นประสาท โดยพยายามใช้กรณีศึกษาที่หลากหลาย และครอบคลุมรูปแบบของปัญหาจนกระทั่งแน่ใจได้ว่า ระบบสามารถทำการวิเคราะห์ ตัดสินใจ และดำเนินการได้อย่างเหมาะสมเมื่อมีเหตุการณ์เกิดขึ้นจริง โดยระบบเครือข่ายเส้นประสาทที่นิยมนำมาประยุกต์ทางธุรกิจ ได้แก่ "ระบบเครือข่ายแบบไปข้างหน้า (Feed - forward Networks)" โดยระบบเครือข่ายในลักษณะนี้จะมีการเคลื่อนที่ของกระแสประสาทไปในทิศทางเดียวกัน จากระดับนำ

เข้าต่อเนื่องจนกระทั่งถึงระดับแสดงผลลัพธ์ โดยเซลประสาทในแต่ละชั้น จะรับปัจจัยนำเข้าจากเซลประสาทที่อยู่ในชั้นก่อนหน้า เพื่อดำเนินการตามความสัมพันธ์และหน้าที่ที่ถูกกำหนด แล้วส่งต่อไปให้เซลประสาท ในชั้นถัดไป ตามลำดับชั้นจนได้ผลลัพธ์ออกมาตามที่ต้องการ

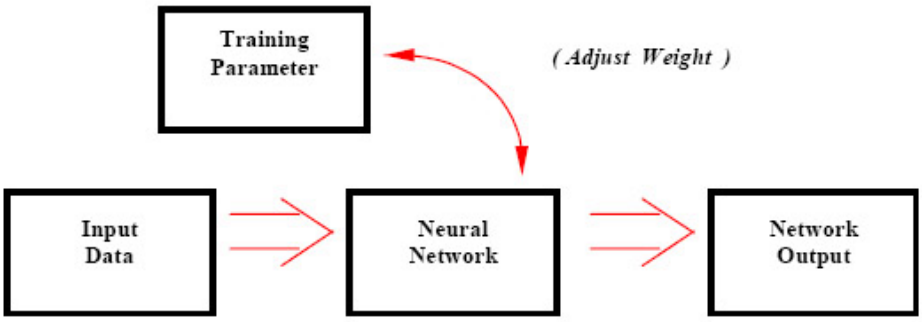
การเรียนรู้ (Learning)

1. การเรียนแบบมีการสอน (Supervised Learning) การเรียนรู้แบบนี้เราจะส่งข้อมูลคู่ของ Input-Output เข้าไป โครงข่ายประสาทเทียมและขบวนการเรียนรู้จะรับ Input เข้าไปทำการคำนวณกับค่า น้ำหนัก (Weight) ของแต่ละเส้นเชื่อม และนำไปผ่านฟังก์ชันบางอย่าง แล้วแต่ขบวนการเรียนรู้ที่เราเลือกใช้ ซึ่งเราจะได้ค่า Output ออกมาค่าหนึ่ง ซึ่งจะถูกนำไปเปรียบเทียบกับค่า Output ที่เราใส่เข้าไปว่าตรงกันหรือไม่ ถ้าตรงกันก็ไม่ต้องมีการเรียนรู้เพิ่มเติม แต่ถ้าไม่ตรงก็ต้องมีการปรับค่าน้ำหนักประจำเส้นเชื่อมต่างๆ เป็นการเรียนรู้ และการนำ Output ที่ใส่เข้าไป เป็นตัวเปรียบเทียบเพื่อระบุว่าโครงข่ายเรียนรู้งานนั้นแล้วหรือยัง นั่นเองที่เราเลยเรียก Output ว่าเป็นครูให้กับโครงข่าย



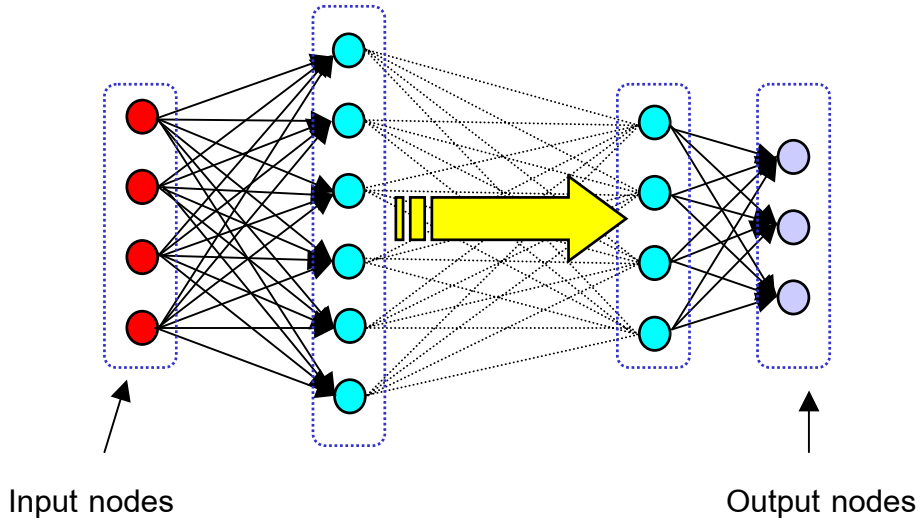
ภาพที่ ก-1 แสดงการเรียนรู้แบบไม่มีผู้สอน (Unsupervised Learning)

2. การเรียนแบบไม่มีการสอน (Unsupervised Learning) Unsupervised Learning นั้นก็คือการที่เราใส่เฉพาะ Input เข้าไปในโครงข่าย แล้วโครงข่ายกับขบวนการเรียนรู้ จะจัดการจัดกลุ่มของข้อมูลให้เอง หลักการที่นำมาใช้ก็จะแล้วแต่ขบวนการเรียนรู้ที่เลือกใช้ เช่น ใช้การวัดระยะทางระหว่าง Input ถ้าใกล้กันกว่าค่าระดับที่ เราตั้งไว้ก็ถือว่าเป็น Input กลุ่มเดียวกัน เป็นต้น



ภาพที่ ก-2 แสดงการเรียนรู้แบบไม่มีผู้สอน (Unsupervised Learning)

Feedforward Network ข้อมูลที่ประมวลผลในวงจรข่ายจะถูกส่งไปในทิศทางเดียวจาก Input Nodes ส่งต่อมาเรื่อยๆ จนถึง Output Nodes โดยไม่มีการย้อนกลับของข้อมูล หรือแม้แต่ Nodes ใน Layer เดียวกันก็ไม่มีการเชื่อมต่อกัน



ภาพที่ ก-3 แสดงสถาปัตยกรรมของ Feedforward Network

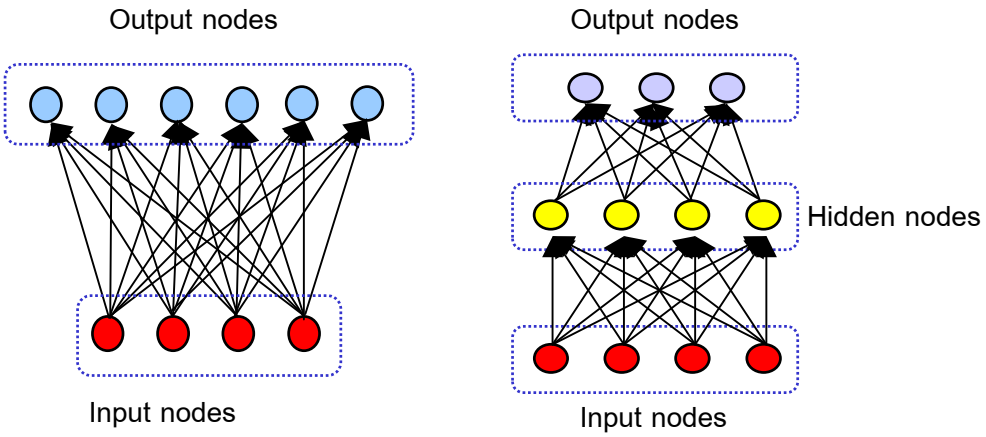
Network Layer

1. พื้นฐานสามัญที่สำคัญของ Artificial Neural Network ประกอบไปด้วย 3 ส่วน หรือ 3 Layer ได้แก่ ชั้นของ Input Units ที่ถูกเชื่อมต่อกับชั้นของ Hidden Units ซึ่งเชื่อมต่อกับชั้นของ Output Unit
2. การทำงานของ Input Unit จะทำหน้าที่แทนส่วนของข้อมูลดิบ ที่จะถูกป้อนเข้าสู่เครือข่าย
3. การทำงานของแต่ละ Hidden Units จะถูกกำหนด โดยการทำงานของ Input Units และค่าน้ำหนักบนความสัมพันธ์ระหว่าง Input Units และ Hidden Units

4. พฤติกรรมการทำงานของ Output Units จะขึ้นอยู่กับการทำงานของ Hidden Units และค่าน้ำหนักระหว่าง Hidden Units และ Output Units

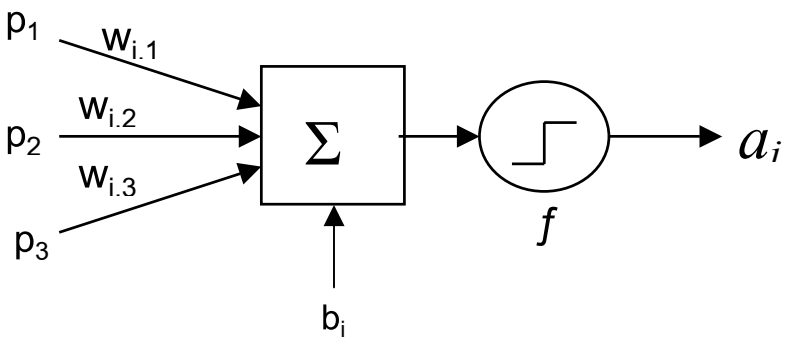
5. ประเภทของเครือข่ายนี้เป็นที่น่าสนใจ เพราะเราสามารถกำหนดการแทนค่าให้แก่ Input Units ได้อย่างอิสระ ค่าน้ำหนักระหว่าง Input Units และ Hidden Units จะถูกกำหนดเมื่อ Hidden Unit กำลังทำงาน ฉะนั้นเวลาที่แก้ไขค่าน้ำหนัก Hidden Units จะสามารถเลือกว่าอะไรคือค่าที่เราแทนเข้ามา

Architecture of Layer สามารถจำแนกสถาปัตยกรรมของชั้น (Layer) ออกเป็น 2 ประเภท คือ Single-layer และ Multi-layer



ภาพที่ ก-4 แสดง Perceptron Network ของ Single Layer และ Multiplayer(for 2 layers)

Single Layer Perceptron Networks



$$
\begin{aligned}
a_i &= f(w_{i,1}p_1 + w_{i,2}p_2 + w_{i,3}p_3 + b_i) \\
&= f(\sum_j w_{ij}x_j + b_i) \\
&= f(\mathbf{W}^T\mathbf{p} + b)
\end{aligned}
\qquad \text{(ก-1)}
$$

เมื่อ p_j = ข้อมูลเข้า (Input Data from node j in the Input Layer)

w_{ij} = ตัวถ่วงน้ำหนักเชื่อมโยง (Connection Weight of Branch (i,j))

b_i = ค่าขีดเริ่มเปลี่ยน (Bias)

f = ฟังค์ชั่นกระตุ้น (Activation Function)

a_i = ข้อมูลออกของนิวรอน (Network Output)

จำนวน Input Nodes ขึ้นอยู่กับจำนวน Components ของ Input Data

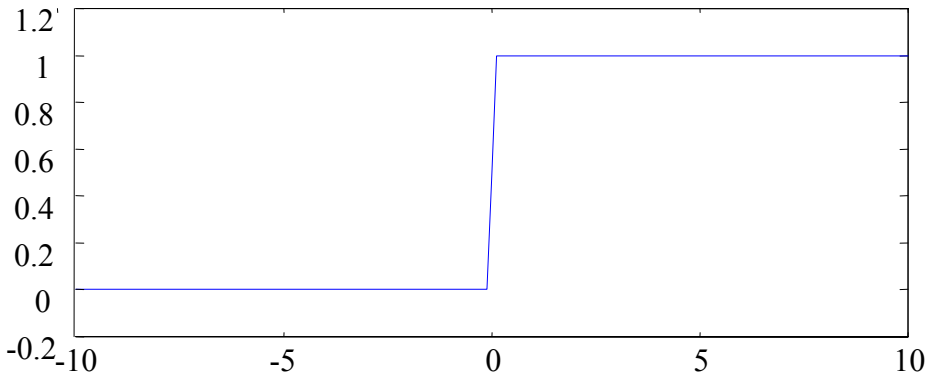
Activation Function ขึ้นอยู่กับลักษณะข้อมูลของ Output เช่น ถ้า Output ที่ต้องการเป็น “ใช่” หรือ “ไม่ใช่” เราจะต้องใช้ Hard Limit Function

$$a = \begin{cases} 1 & \text{if } n \geq 0 \\ 0 & \text{if } n < 0 \end{cases} \qquad \text{(ก-2)}$$

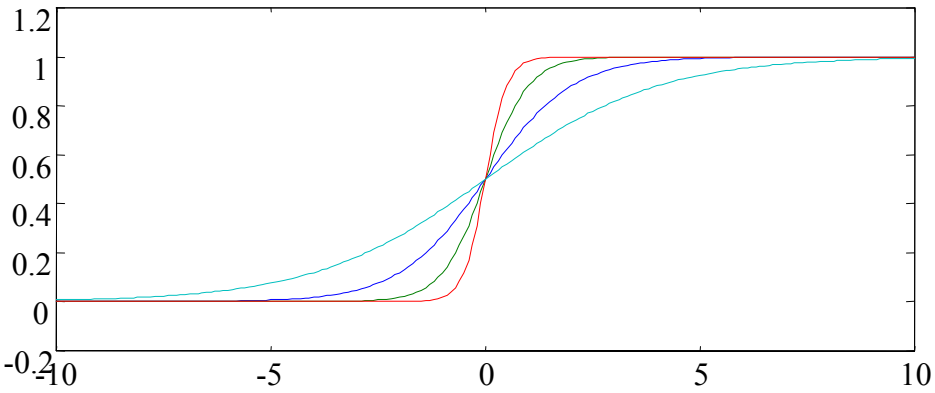
หรือถ้า Output เป็นค่าตัวเลขที่ต่อเนื่อง เราต้องใช้ Continuous Function เช่น Log-Sigmoid Function

$$f(x) = \frac{1}{1+e^{-n}} \qquad \text{(ก-3)}$$

Activation function



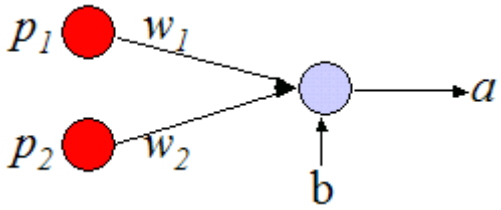
ภาพที่ ก-5 แสดงกราฟ Activation Function ของ Hard Limit function



ภาพที่ ก-6 แสดงกราฟ Activation Function ของ Log-Sigmoid Function

การทำงานของ Single Layer Perceptron

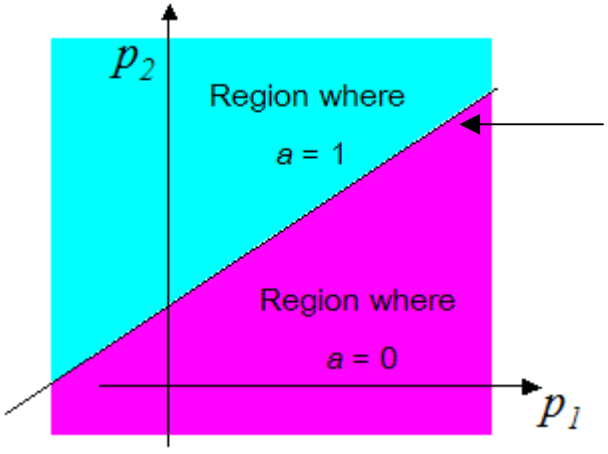
สมมุติว่าเรามีวงจรข่าย Perceptron ที่มี 2 Input Nodes และมี Activation Function เป็น Threshold Function เราจะได้ Binary Output



ภาพที่ ก-7 แสดงวงจรข่าย Perceptron ที่มี 2 Input Nodes

เราได้

$$a = \begin{cases} 1 & \text{if} \quad w_1 p_1 + w_2 p_2 + b \geq 0 \\ 0 & \text{if} \quad w_1 p_1 + w_2 p_2 + b < 0 \end{cases} \qquad \text{(ก-4)}$$



$Linear\ L = w_1 p_1 + w_2 p_2 + b = 0$

ถ้า (p_1,p_2) อยู่เหนือเส้นตรง L จะได้ a = 1

ถ้า (p1,p2) อยู่ใต้เส้นตรง L จะได้ a = 0

ดังนั้นเราเรียกเส้นตรง *L* นี้ว่า

ขอบการตัดสินใจ (Decision Boundary)

ความชันและตำแหน่งของเส้นตรง L : $w_1p_1+w_2p_2-b=0$ ขึ้นอยู่กับพารามิเตอร์ w_1, w_2, และ b เราจะต้องปรับพารามิเตอร์เหล่านี้ให้ได้เส้นตรง L ที่ให้ผลลัพธ์ถูกต้อง

การเทรน Perceptron Networks

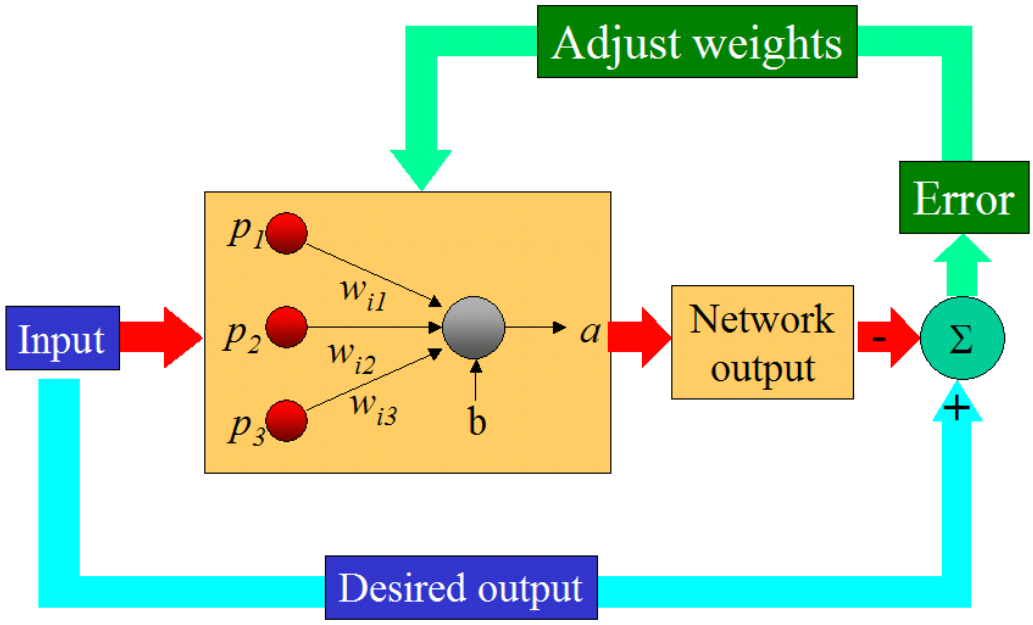
หลักการปรับตัวของวงจรข่าย: ปรับพารามิเตอร์ต่างๆให้ไปในทางที่จะลดค่าความผิดพลาดลงได้

$$e = t - a \qquad \text{(ก-5)}$$

เมื่อ e คือ ค่าความผิดพลาด (Error)
t คือ ข้อมูลออกที่ต้องการ (Target Output)
a คือ ข้อมูลออกของนิวรอน (Network Output)

ขั้นตอนในการฝึกวงจรข่าย Training the Network

1. ป้อน Input เข้า Network — Input (p_1, p_1) และ Target Output t
2. คำนวณค่า Network Output — $a = f(w_1p_1 + w_2p_2 - b)$
3. คำนวณค่า Error — $e = t - a$
4. ปรับค่า Weight ทุกค่า — $w^{new} = w^{old} + ep$, $b^{new} = b^{old} + e$
5. กลับไปทำข้อ 1 ใหม่จนกว่า Error จะต่ำลงจนยอมรับได้ — $|e| < t$



ภาพที่ ก-8 แสดงขั้นตอนในการฝึกวงจรข่าย Training the Network

Higher Dimension Feature Space

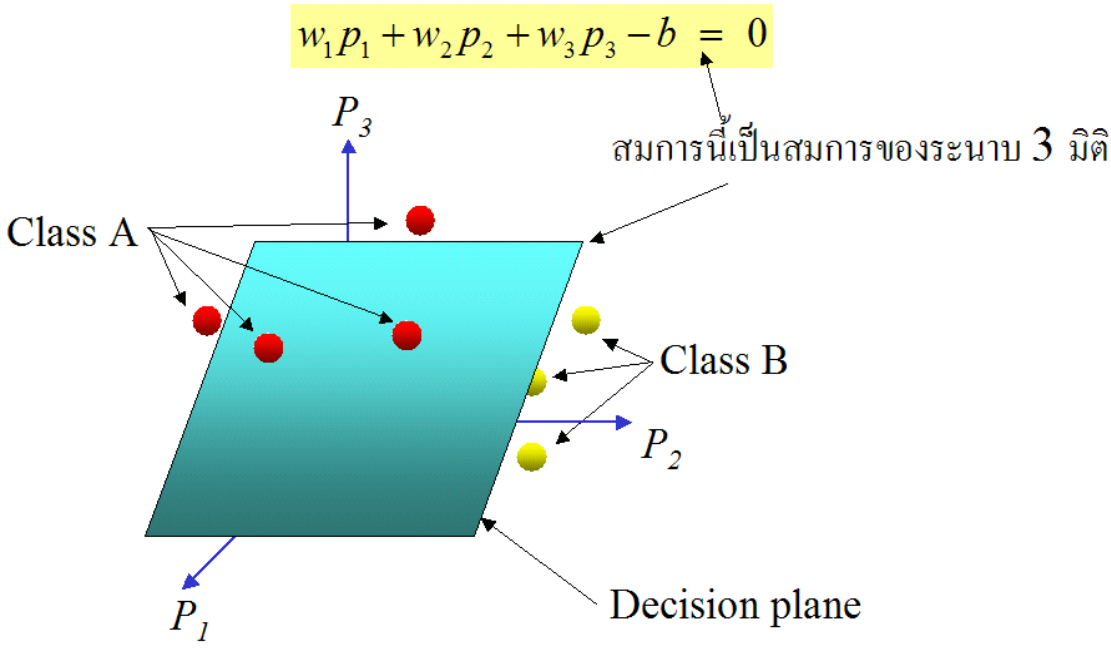
ในกรณีที่ Input Pattern มี Component มากกว่า 1, Output ของ Perceptron จะเป็น

$$a = \begin{cases} 1 & \text{if} \quad w_1 p_1 + w_2 p_2 + \cdots + w_N p_N + b \geq 0 \\ 0 & \text{if} \quad w_1 p_1 + w_2 p_2 + \cdots + w_N p_N + b < 0 \end{cases} \qquad \text{(ก-6)}$$

เราจะได้ Decision Function เป็น

$$w_1 p_1 + w_2 p_2 + \cdots + w_N p_N + b = 0 \qquad \text{(ก-7)}$$

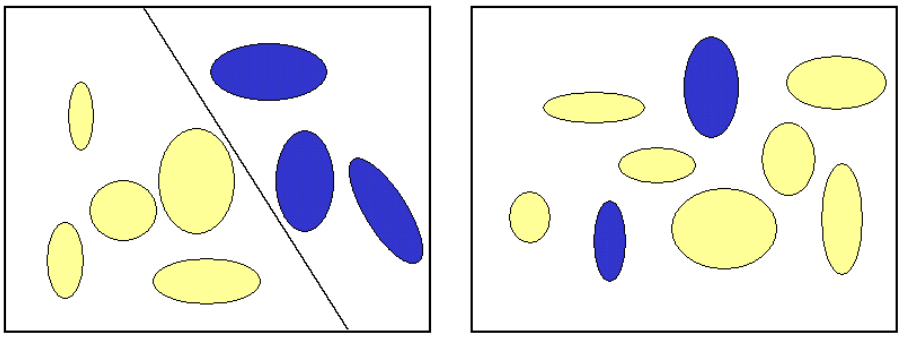
สมการนี้เป็นสมการของระนาบหลายมิติ (Hyperplane) ถ้าในกรณีของ Input 2 มิติ, Decision Boundary จะเป็นเส้นตรง ในกรณีของ input 3 มิติ เราจะได้ Decision Boundary เป็น



ภาพที่ ก-9 แสดงระนาบหลายมิติ (Hyperplane) ในกรณีของ Input 3 มิติ

Linear Separability

- ในกรณีของ a Single Layer Perceptron เรามี Hyperplane เป็น Decision Boundary
- ถ้า Input Pattern ของแต่ละ Class สามารถแบ่งแยกจาก Class อื่นได้โดยใช้ Hyperplane แบ่ง เราเรียกข้อมูลชุดนี้ว่ามีลักษณะเป็น Linearly Separable



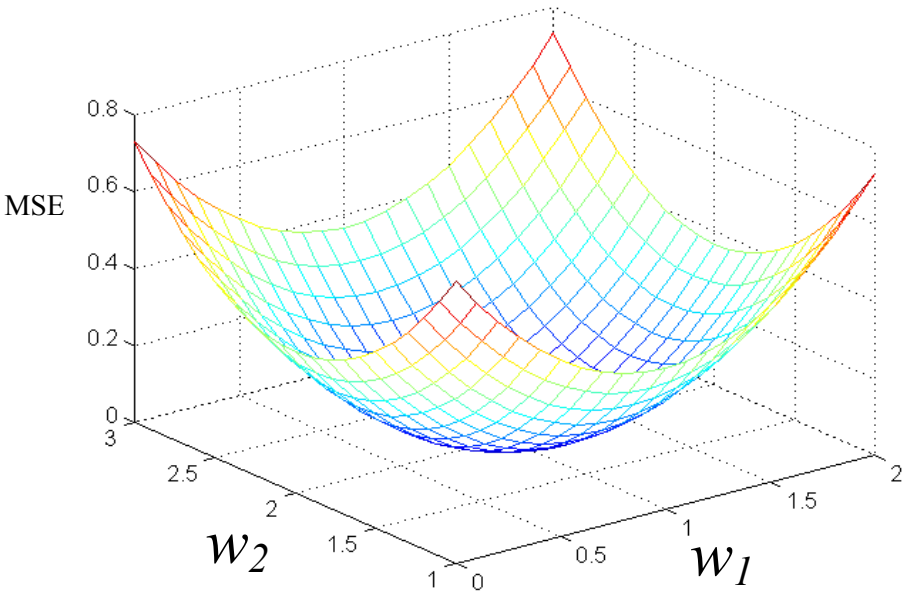
ภาพที่ ก-10 แสดง Linearly Separable และ Not Linearly Separable
(เป็นข้อจำกัดของ A Single Layer Perceptron)

การปรับ weight

พิจารณาค่าเฉลี่ยของค่าความผิดพลาดกำลังสอง (Mean Square Error, MSE)

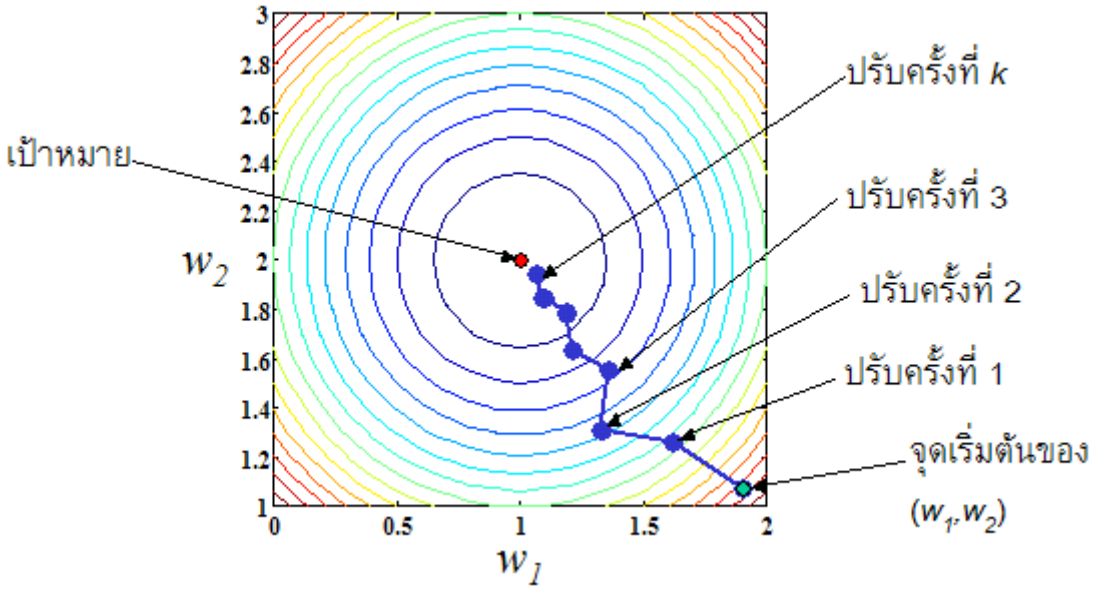
$$\begin{aligned} e^2 &= (t-a)^2 \\ &= (t-w_1x_1-w_2x_2)^2 \end{aligned} \qquad \text{(ก-8)}$$

เราจะได้ e^2 ในรูปของ function ของ w_1 และ w_2 ดังในรูปข้างล่าง



ภาพที่ ก-11 แสดงกราฟของ MSE เราเรียกรูปนี้ว่า Error Surface

ลักษณะการปรับ w_1และ w_2 ให้เข้าสู่จุดต่ำสุดใน Error Surface



ภาพที่ ก-12 แสดงลักษณะการปรับ w_1 และ w_2 ให้เข้าสู่จุดต่ำสุดใน Error Surface

A Single Layer Percentron Case

Network Output คำนวณได้จาก

$$a = f(\sum_{j=1}^{N} w_j p_j + b) \qquad \text{(ก-9)}$$

Square Error e^2 คำนวณได้จาก

$$\begin{aligned} ^2 &= (t-a)^2 \\ &= (t - f(\sum_{j=1}^{N} w_j p_j - b))^2 \end{aligned} \qquad \text{(ก-10)}$$

Slope e^2 เทียบกับ w_j คำนวณได้จาก

$$\begin{aligned} \frac{\partial e^2}{\partial w_j} &= \frac{\partial}{\partial w_j}(t - f(\sum_{j=1}^{N} w_j p_j + b))^2 \\ &= -2(t - f(\sum_{j=1}^{N} w_j p_j + b)) \cdot f'(\sum_{j=1}^{N} w_j p_j + b)) \cdot p_j \end{aligned} \qquad \text{(ก-11)}$$

Slope e^2 เทียบกับ b คำนวณได้จาก

$$\frac{\partial e^2}{\partial b} = 2(t - f(\sum_{j=1}^{N} w_j p_j - b)) \cdot f'(\sum_{j=1}^{N} w_j p_j - b)) \quad \text{(ก-12)}$$

ดังนั้นเราจะได้

$$\sum_{j=1}^{N} w_j p_j - b)) \cdot f'(\sum_{j=1}^{N} w_j p_j - b)) \cdot p_j \quad \text{(ก-13)}$$

$$\Delta b = -\alpha \cdot (t - f(\sum_{j=1}^{N} w_j p_j - b)) \cdot f'(\sum_{j=1}^{N} w_j p_j - b)) \quad \text{(ก-14)}$$

สมการการปรับ Weight จะเป็น

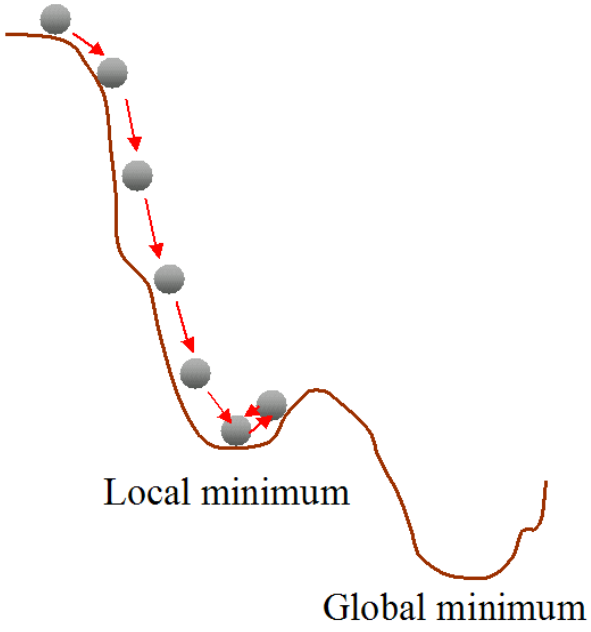
$$w_j^{new} = w_j^{old} + \Delta w_j \quad \text{(ก-15)}$$

$$b^{new} = b^{old} + \Delta b \quad \text{(ก-16)}$$

α = Learning rate

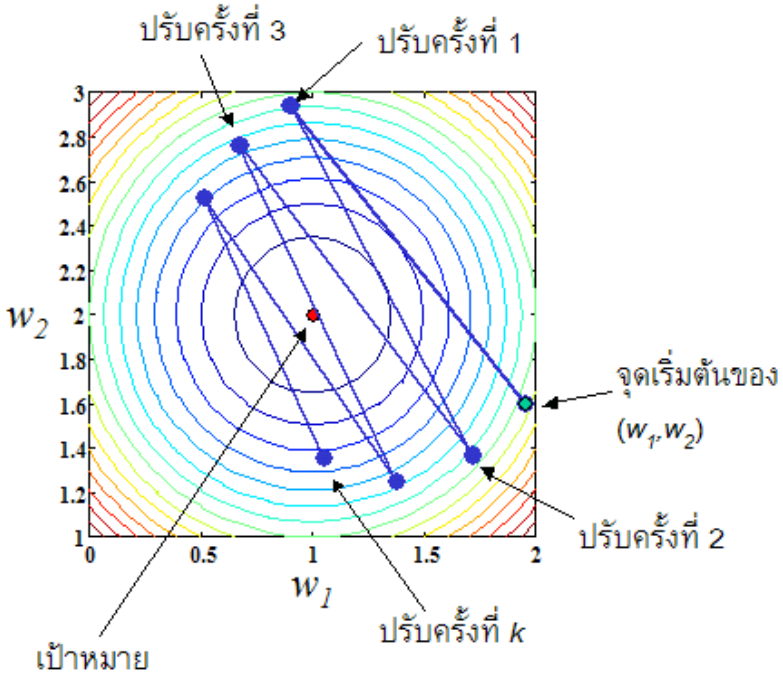
Weak Point of Gradient Descent Method

วิธีการ Gradient Descent นี้ อาจจะทำให้เราติดอยู่ที่ Local Minima ซึ่งยังไม่ใช้จุดที่ต่ำสุด



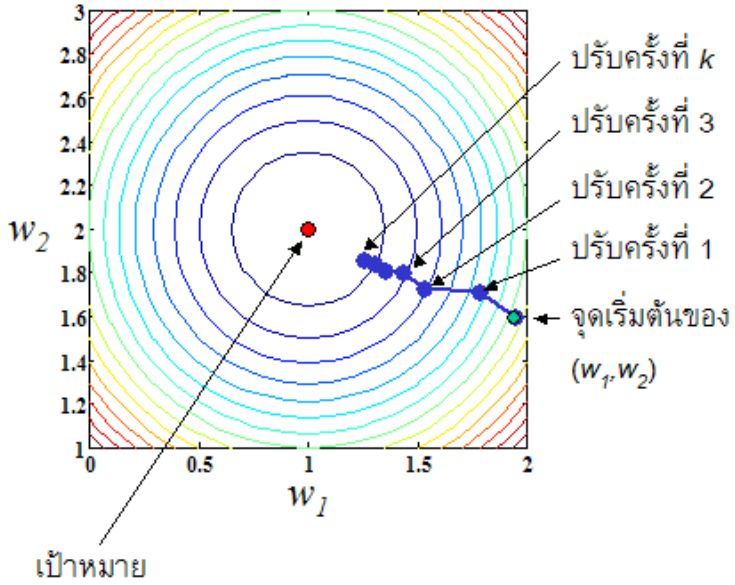
ภาพที่ ก-13 แสดงวิธีการ Gradient Descent Method ซึ่งอาจจะทำให้เราติดอยู่ที่ Local Minima ซึ่งยังไม่ใช้จุดที่ต่ำสุดจริงๆ

การปรับ Weight ถ้าปรับโดยใช้ Learning rate ค่ามากๆจะทำให้ Network ปรับตัวเข้าสู่จุดต่ำสุดได้ช้าหรืออาจไม่ได้เลย (Unstable)



ภาพที่ ก-14 แสดงการปรับ Weight โดยใช้ค่า Learning Rate มากๆ

การปรับ Weight ทีละน้อยจะทำให้ Network ปรับตัวเข้าสู่จุดต่ำสุดได้ดีและช่วยลดเรื่องการไม่เสถียร Unstable ได้



ภาพที่ ก-15 แสดงการปรับ Weight โดยใช้ค่า Learning Rate ต่ำ

การปรับ Weight ของสมการเชิงเส้น

ค่าในการปรับ Weights

$$\Delta w_j = \alpha \cdot (t - f(\sum_{j=1}^{N} w_j p_j - b)) \cdot f'(\sum_{j=1}^{N} w_j p_j - b)) \cdot p_j \qquad \text{(ก-17)}$$

Output error

f'

สำหรับ Linear Unit $f(x) = x$ โดยมี $f'(x) = 1$

เราได้

$$\Delta w_j = \alpha \cdot (t - (\sum_{j=1}^{N} w_j p_j - b)) \cdot p_j \qquad \text{(ก-18)}$$

การปรับ Weight ของสมการไม่เชิงเส้น

การคำนวณ f' สำหรับกรณี Nonlinear Unit

1. Log-Sigmoid

$$a = f(n) = \frac{1}{1+e^{-n}} \qquad \text{(ก-19)}$$

เราได้

$$f'(n) = \frac{e^{-n}}{(1+e^{-n})^2} = \left(1 - \frac{1}{(1+e^{-n})}\right)\left(\frac{1}{1+e^{-n}}\right) = (1-a)(a) \qquad \text{(ก-20)}$$

2. Tan-Sigmoid

$$a = f(n) = \tan sig(n) = \frac{e^{n} - e^{-n}}{e^{n} + e^{-n}} \qquad \text{(ก-21)}$$

เราได้

$$f'(n) = -\frac{e^{n} - e^{-n}}{(e^{n} + e^{-n})^2}(e^{n} - e^{-n}) + \frac{e^{n} + e^{-n}}{e^{n} + e^{-n}} = 1 - \frac{(e^{n} - e^{-n})^2}{(e^{n} + e^{-n})^2} = 1 - (a)^2 \qquad \text{(ก-22)}$$

สรุป Single Layer Perceptron Networks

1. Function ของแต่ละ Unit อยู่ในรูป Function ของ Input Weighted Sum

$$a_i = f(\sum_{j=1}^{N} w_{ij} x_j - b_i)$$

2. ในกรณีของการแยกแยะข้อมูล Decision Boundary จะอยู่ในรูปของ Hyperplane

3. ข้อจำกัดของ A Single Layer Network คือข้อมูลที่วงจรข่ายสามารถแยกแยะได้ จะต้องเป็น Linearly Separable Data

4. การฝึกวงจรข่ายโดยการปรับค่า Weight อาศัยหลักการของ Gradient Descent Method เพื่อที่จะลดค่าความผิดพลาดให้ต่ำที่สุด

5. Gradient Descent Method มีจุดอ่อนคือวิธีการนี้อาจจะทำให้วงจรข่ายติดอยู่ที่ Local Minima ของ Error Surface ได้

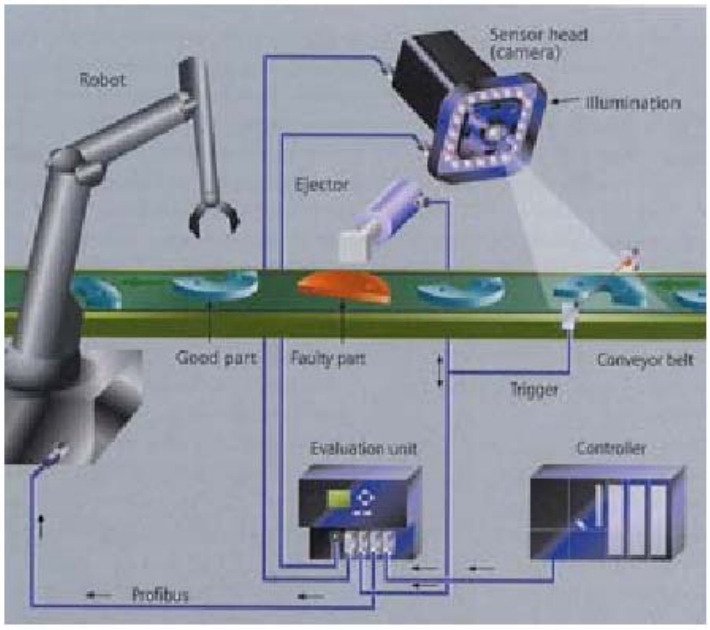
ภาคผนวก ข

พื้นฐานการประมวลผลภาพดิจิตอล

ภาคผนวก ข
พื้นฐานการประมวลผลภาพดิจิตอล

กลไกเคลื่อนไหว (Actuation)

กระบวนการนี้โปรแกรมจะสั่งการส่วนกลไกเคลื่อนไหวต่างๆ ให้กระทำการบางอย่างกับผลิตภัณฑ์ที่ผ่านกระบวนการก่อนหน้าข้างต้น เช่น การสั่งให้สายพานเลื่อนชิ้นงานชิ้นต่อไปเข้ามา หรือการสั่งให้แขนหุ่นยนต์ทำการหยิบจับชิ้นงานที่ผ่านตรวจสอบแล้วไปวางไว้ในที่ๆ จัดไว้เป็นต้น นอกจากนั้นในบางกรณี อาจจะมีสั่งให้ตัวกล้องเคลื่อนที่ไปยังส่วนอื่นๆ ของชิ้นงานที่กำลังพิจารณาอยู่อีกด้วย ซึ่งในส่วนนี้งานหลักๆ จะเป็นการติดต่อและสั่งงานระหว่างอุปกรณ์ประมวลผลและ Programmable Logic Control ที่สามารถใช้สั่งการส่วนเคลื่อนไหวต่างๆ เช่น มอเตอร์แขนหุ่นยนต์ หรืออื่นๆ ได้อย่างง่ายดาย ตัวอย่างการทำงานของส่วนกลไกต่างๆ ซึ่งขึ้นอยู่กับการตัดสินใจของอุปกรณ์ประมวลผลที่รับภาพของผลิตภัณฑ์เข้ามานั้น แสดงไว้ในภาพที่ ข-1



ภาพที่ ข-1 แสดงการทำงานจริงที่ประกอบด้วยกลไกต่างๆ

เมื่อพิจารณาการทำงานของแต่ละกระบวนการแล้ว จะพบว่า เมื่อเสร็จสิ้นแต่ละขบวนการลง ข้อมูลที่ได้จากแต่ละขบวนการจะมีความหนาแน่นของข้อมูลมากขึ้นเรื่อยๆ ตามลำดับของกระบวนการที่เพิ่มขึ้น ดังแสดงไว้ในภาพที่ ข-1 ยกตัวอย่างเช่น ข้อมูลที่ได้จากการประมวลภาพเบื้องต้น (Pre – processing) ก็คือ ภาพที่ได้รับการปรับปรุงคุณภาพแล้ว ซึ่งเมื่อส่งข้อมูลดังกล่าวต่อไปที่กระบวนการแยกบริเวณ (Segmentation) ผลที่ได้คือ บริเวณของวัตถุที่สนใจโดย

ละฉากหลังไว้ นอกจากนั้น แต่ละบริเวณก็มีการบีบอัดพิกัดของพิกเซลไว้ในรูปแบบที่เหมาะสม ซึ่งเมื่อส่งต่อไปให้กระบวนการคำนวณหาคุณสมบัติของวัตถุ (Feature Extraction) ก็จะได้ Feature Vector ซึ่งเป็นคุณสมบัติของแต่ละบริเวณออกมา และท้ายที่สุด เมื่อส่งเวคเตอร์เหล่านี้ไปให้กระบวนการจำแนกและตีความหมาย (Classification and Interpretation) ก็จะได้ข้อมูลที่สำคัญที่สุดออกมาคือ การตัดสินใจว่าจะจัดการกับตัวผลิตภัณฑ์อย่างไร เช่น จัดเป็นชิ้นงานดี–ชิ้นงานเสีย หรือจัดเป็นเกรด 1-เกรด 2 เป็นต้น ซึ่งจากลักษณะการทำงานของกระบวนการย่อยแต่ละขบวนการของระบบตรวจสอบชิ้นงานด้วยภาพแบบอัตโนมัตินั้น ผู้อ่านจะสังเกตเห็นว่าระบบไม่ได้มุ่งเน้นที่การประมวลภาพแบบดิจิตอล (Digital Image Processing, DIP) แต่เพียงอย่างเดียวเท่านั้น หากทว่า การที่จะทำให้ระบบดังกล่าว สามารถทำงานได้ อย่างมีประสิทธิภาพนั้น จะต้องอาศัยความรู้จากหลายๆ สาขาด้วยกัน เช่น

- การประมวลภาพแบบดิจิตอล (Digital Image Processing, DIP)
- ระบบจำแนก (Classification System) ที่ทำให้โปรแกรมสามารถตัดสินใจได้อย่างชาญฉลาด
- ความรู้เรื่อง Computer Graphics ซึ่งจะนำมาใช้ทั้งในส่วนที่ติดต่อกับผู้ใช้ และส่วนที่ทำการตรวจสอบชิ้นงาน
- การติดต่อกับ Programmable Logic Control เพื่อสั่งงานส่วนเคลื่อนไหวต่างๆ เหล่านี้ เป็นต้น

วิธีการดึงข้อมูลภาพภายใต้ระบบปฏิบัติการวินโดวส์

สำหรับวิธีการดึงข้อมูลจากกล้องเข้ามาสู่เครื่องคอมพิวเตอร์ หรืออีกนัยหนึ่งคือ การดึงค่าข้อมูลจากกล้องเข้ามาสู่โปรแกรมนั้น โดยทั่วไปนั้นมีอยู่ด้วยกัน 3 วิธีด้วยกัน

1. การดึงข้อมูลโดยการใช้เครื่องมือในการโปรแกรม (Programming Tool) ที่ผู้ผลิตกล้องให้มา โดยมากแล้ว สำหรับกล้องที่มีความละเอียดสูง (Hi – Resolution Camera) ผู้ผลิตหรือจัดจำหน่ายนั้นอยากให้ผู้ใช้นำกล้องไปใช้งานอยู่แล้วดังนั้น หลังจากซื้อกล้องแล้ว ผู้ผลิตมักจะแถมเครื่องมือในการโปรแกรม เพื่อใช้ดึงข้อมูลภาพออกจากตัวกล้องมาสู่ตัวโปรแกรมหลักเสมอ โดยทั่วไป เครื่องมือดังกล่าวจะมีอยู่ 2 รูปแบบ คือ dll และ activeX นอกจากนั้น ผู้ผลิตยังอาจจะให้ตัวโปรแกรมอย่างง่ายมาให้ด้วย ซึ่งก่อนที่จะซื้อกล้องชนิดนั้น ๆ โปรแกรมเมอร์จะต้องทำการพิจารณาเสียก่อนว่าตัวแปลงภาษา (Compiler) ที่ใช้ในการเขียนโปรแกรมอยู่นั้น รองรับการทำงานของเครื่องมือที่ผู้ผลิตให้มาหรือไม่สำหรับเรื่องนี้ ตอนที่ทำโครงการวิจัยอยู่นั้น ผู้เขียนได้ประสบปัญหาที่ว่า การติดต่อซื้อกล้องจากผู้จัดจำหน่ายในเมืองไทยนั้น ผู้จัดจำหน่ายไม่สามารถให้รายละเอียด เกี่ยวกับเครื่องมือในการโปรแกรมที่ใช้สำหรับดึงข้อมูลภาพมาสู่โปรแกรมที่กำลังพัฒนาอยู่ได้เลย สาเหตุของเรื่องนี้เป็นที่เข้าใจได้ง่าย เพราะผู้จัดจำหน่ายกล้อง

ความละเอียดสูงสำหรับใช้ในการตรวจสอบชิ้นงานในประเทศไทยนั้น โดยมากแล้วจะเป็นผู้จัดจำหน่ายระบบตรวจสอบชิ้นงานทั้งระบบ ที่รวมทั้งส่วนที่เป็นกล้องความละเอียดสูงและโปรแกรมที่ติดมาด้วย ซึ่งจุดมุ่งหมายหลักของผู้จัดจำหน่ายในประเทศไทยคือ ขายทั้งระบบเพื่อให้ลูกค้าสามารถนำไปใช้งานได้เลย โดยไม่ได้มุ่งหวังว่า ลูกค้าจะซื้อกล้องเพื่อไปพัฒนาโปรแกรมเอง ผู้เขียนได้แก้ปัญหาโดยการสั่งซื้อกล้องจาก Website ต่อไปนี้ ซึ่งจะให้เครื่องมือในการโปรแกรมสำหรับดึงข้อมูลภาพมาสู่ตัวโปรแกรมมาด้วย นอกจากนั้น การสนับสนุนในกรณีที่เกิดปัญหาเกี่ยวกับการโปรแกรมก็ทำได้ค่อยข้างดีทีเดียว

2. การดึงข้อมูลโดยการใช้เครื่องมือในการโปรแกรมของระบบปฏิบัติการวินโดวส์ เช่น Video For Window (VFW) หรือ DirectShow

สำหรับ Video For Window (VFW) นั้นเป็นเครื่องมือที่ใช้การดึงข้อมูลภาพจากกล้องและสามารถใช้อ่านไฟล์วีดีโอประเภท AVI (Audio Video Interleave) ได้อีกด้วย เครื่องมือดังกล่าวเป็นเครื่องในการโปรแกรมที่มีมาให้อยู่แล้ว ตั้งแต่ระบบปฏิบัติการวินโดวส์ 3.1 ซึ่งฟังค์ชั่นการรำงานของเครื่องมือในการโปรแกรมชนิดนี้ ถูกฝังอยู่ในไลบรารี่ 2 ตัวด้วยกัน คือ msvfw32.dll และ avicap32.dll และถึงแม้ว่าในปัจจุบัน จะมีเครื่องมือในการโปรแกรมสำหรับดึงข้อมูลภาพจากกล้องมาสู่โปรแกรมตัวใหม่ของระบบปฏิบัติการวินโดวส์ คือ DirectShow มาแทนที่ แต่ระบบปฏิบัติการวินโดวส์รุ่นต่าง ๆ ส่วนใหญ่ก็ยังสนับสนุนการทำงานของ VFW อยู่นั้นเอง จึงอาจกล่าวไว้ว่า ในปัจจุบัน VFW จัดเป็นเครื่องมือที่เหมาะสมที่สุดสำหรับการพัฒนา เนื่องจาก ทั้งผู้ผลิตกล้อง รวมทั้งระบบปฏิบัติการวินโดว์รุ่นต่าง ๆ ยังคงยึด VFW เป็นหลักอยู่ นอกจากนั้น เนื่องจาก VFW มีการใช้งานมานานและมีการพัฒนาอย่างต่อเนื่อง ทำให้ข้อผิดพลาดต่าง ๆ ได้ถูกแก้ไขไปเกือบหมด สำหรับตัวอย่างโปรแกรม รวมทั้งเอกสารต่าง ๆ ที่เกี่ยวกับการเรียกใช้งาน VFW นั้น ผู้อ่านสามารถค้นคว้าเพิ่มเติมได้ที่ The Microsoft Developer Network Online (MSDN) ที่ Website ของไมโครซอฟท์

ข้อสังเกตหนึ่งของการเรียกใช้งานเครื่องมือในการโปรแกรมของระบบปฏิบัติการวินโดวส์คือ สามารถใช้งานได้อย่างยืดหยุ่นมากที่สุด ซึ่งทำให้โปรแกรมเมอร์สามารถเข้าถึงหน่วยความจำทั้งหลัง และก่อนแสดงภาพเลยทีเดียวนอกจากนั้น ด้วยการใช้งานเครื่องมือดังกล่าวนี้ ยังสามารถนำไปพัฒนาเป็นเครื่องมือในการโปรแกรมแบบเดียวกับทางเลือกแรก และทางเลือกที่สามที่สามารถนำไปจำหน่ายในทางการค้าได้อีกด้วย อย่างไรก็ตาม ข้อด้อยของการใช้งานคือ ความยุ่งยากและซับซ้อนของการใช้งานเครื่องมือดังกล่าวนั่นเอง

การดึงข้อมูลภาพโดยการใช้เครื่องมือในการโปรแกรมที่อยู่ในรูปของ activeX หรือ dll ซึ่งมีจำหน่ายในเชิงการค้าหรือที่เรียกกันว่า Third party activeX ในปัจจุบันมีผู้ผลิตและจำหน่ายหลายๆ รายด้วยกัน เช่น videoOCX (http://www.videoocx.del) เป็นต้น ซึ่งโดยทั่วไปแล้ว เครื่องมือในการโปรแกรมลักษณะนี้ มักจะถูกออกแบบให้มีความสามารถหลายๆ อย่างเช่น การดึงข้อมูลภาพ การประมวลภาพเบื้องต้น และการเปิดไฟล์วีดิโอ นอกจากนั้นการใช้งานยัง

สามารถทำได้อย่างง่ายดายอีกด้วย เช่น การอ่านข้อมูลภาพจากกล้อง ที่สามารถทำได้โดยใช้โปรแกรมเพียงไม่กี่บรรทัด อย่างไรก็ตาม การใช้งานเครื่องมือในการโปรแกรมชนิดนี้ มักจะประสบปัญหาบางประการคือ ขาดความยืดหยุ่นในการปรับแก้การทำงาน นอกจากนั้น เมื่อต้องการส่งต่อหรือจำหน่ายโปรแกรมที่เขียนขึ้น ไปสู่ผู้ใช้งานจริง อาจจะต้องจ่ายเงินเพิ่มขึ้น ถ้าเป็นการนำไปใช้ในทางอุตสาหกรรม ที่ไม่ใช่เป็นการใช้เพื่อการศึกษา

การประมวลภาพเบื้องต้น

ภาพก็คือเมตริกซ์ของจำนวนเต็ม ซึ่งถ้าเป็นภาพ Gray Scale ก็จะเป็นเมตริกซ์ของความเข้มแสงเมตริกซ์เดียว และถ้าเป็นภาพสีก็จะประกอบด้วยเมตริกซ์ของสีแดง เขียวและน้ำเงิน รวมทั้งหมด 3 เมตริกซ์ และจะพบว่า ในทางปฏิบัติโปรแกรมเมอร์จะมีเครื่องมือในการโปรแกรมสำหรับดึงข้อมูลภาพเข้ามาที่ตัวโปรแกรมหลากหลายรูปแบบหรือแม้กระทั่งการอ่านข้อมูลภาพจากไฟล์ภาพก็มี “ตัวช่วย” ในการทำงานดังกล่าวอยู่มากมาย ดังนั้นโปรแกรมเมอร์จึงไม่ควรกังวลเท่าใดนัก ในการที่ดึงข้อมูลภาพมาสู่โปรแกรมที่กำลังพัฒนาอยู่ ขั้นตอนต่อมาก็คือแล้วเราจะทำทำอย่างไรก็กับภาพที่ได้มา เพื่อที่จะได้มาตัดสินใจสุดท้ายเกี่ยวกับคุณภาพของชิ้นงาน ในเนื้อหาต่อไปนี้จะเป็นเรื่องการประมวลภาพแบบดิจิตอลเบื้องต้น ซึ่งมุ่งเน้นที่การปรับปรุงคุณภาพของภาพ เพื่อส่งต่อไปให้ขั้นตอนต่อไป

สำหรับเรื่องคุณภาพของภาพนี้ไม่มีมาตรวัดที่เป็นตัวเลขบ่งชี้ว่า ภาพลักษณะใดถึงจะดี อย่างไรก็ตาม การวัดคุณภาพของภาพอย่างง่าย ๆ มีหลักเกณฑ์ว่า ภาพที่มีคุณภาพดีนั้นอย่างน้อยคนหรือผู้ใช้งานระบบ จะต้องสามารถสังเกตเห็นบริเวณที่ต้องการด้วยตาเปล่าผ่านทางหน้าจอแสดงผลเสียก่อน ก่อนที่จะส่งภาพที่ได้ไปให้อุปกรณ์ประมวลผลต่อไป ยกตัวอย่างเช่น ในการตรวจสอบรอยด่างที่ปรากฏอยู่บนโลหะที่มีความมันวาวและมีการสะท้อนแสง ซึ่งส่งผลให้บริเวณที่เป็นโลหะ หากเราส่งภาพดังกล่าวไปให้ขั้นตอนตรวจสอบที่อยู่ในโปรแกรมเลย ระบบตรวจสอบอาจจะมอง “ไม่เห็น” รอยด่างที่เกิดขึ้น ดังนั้น เราจึงต้องทำการปรับปรุงคุณภาพของภาพ โดยการเพิ่มความละเอียดในบริเวณในบริเวณที่มีแสงจ้า เพื่อให้เห็นความแตกต่างระหว่างแสงสะท้อนและรอยด่างที่เกิดขึ้นเสียก่อน ก่อนที่จะส่งไปที่ขั้นตอนการตรวจสอบต่อไป ซึ่งเราจะมาเรียนรู้ร่วมกันในหัวข้อที่จะกล่าวถึงต่อไปนี้

Digital Convolution

Convolution ถ้าแปลตามตัวหมายถึง การพันขดลวดหรือการควันเกลียวเชือกย่อย ๆ เข้าด้วยกัน ซึ่งถ้าเป็นการประมวลภาพแบบดิจิตอลนั้น Convolution เป็นระเบียบวิธีการคำนวณที่กระทำระหว่างภาพ 2 ภาพ ซึ่งการ “พัน” ภาพเข้าด้วยกันนั้น แท้ที่จริงแล้วเป็นการคำนวณผลรวมของผลคูน (sum – of – product) ระหว่างค่าพิกเซลที่อยู่ในทั้งสองภาพ ซึ่ง Convolution

นี้จะใช้ในการกรองสัญญาณภาพ ทั้งการกรองความถี่สูงและการรองความถี่ต่ำซึ่งจะได้กล่าวถึงต่อไป

โดยทั่วไปแล้ว Convolution จะเป็นการกระทำระหว่าง 2 ภาพ โดยมีภาพหลักซึ่งเราต้องการปรับปรุงคุณภาพของภาพดังกล่าวและภาพที่มีขนาดเล็กกว่าภาพหลักมากเรียกกันเฉพาะว่า หน้ากาก (Mask) ซึ่งหลังจากการทำ Convolution แล้ว ผลที่ได้คือภาพที่มีขนาดใกล้เคียงกับภาพหลัก และค่าที่อยู่ในแต่ละช่องใน Mask นี้เอง ที่จะเป็นตัวกำหนดว่า เป็นตัวกรองสัญญาณภาพลักษณะใด

ในภาพที่ ข-2 แสดงส่วนหนึ่งของภาพเม็ดข้าวที่อยู่ในรูป ซึ่งเป็นภาพหลักที่เราต้องการปรับปรุงคุณภาพและหน้ากากที่มีขนาด 3x3 จะพบว่าการ Convolution ก็คือการวางหน้ากาก “ทาบ” ลงไปในภาพหลักไล่ไปทีละพิกเซล แล้วคำนวณผลคูณระหว่างค่าพิกเซลที่อยู่ในภาพหลักด้วยค่าที่อยู่ในช่องตรงกันของหน้ากาก

108	98	98	96	93	101	98	103	96
110	92	118	100	98	98	102	106	89
100	106	98	94	97	97	101	106	118
96	98	110	136	146	148	157	164	172
97	101	160	186	178	172	171	170	168
95	95	153	178	175	186	179	172	171
98	107	111	161	172	180	186	182	173
100	89	97	110	165	180	172	176	182

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

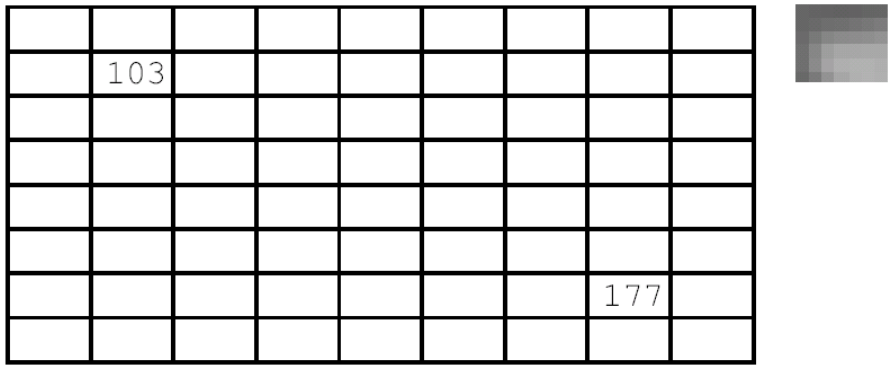
ภาพที่ ข-2 แสดงรูปภาพหลักที่เราต้องการปรับปรุงคุณภาพและหน้ากากขนาด 3x3

ยกตัวอย่างเช่น ถ้าเราทาบหน้ากากลงไปที่มุมซ้ายบนของภาพหลัก ซึ่งหมายความว่าเรากำลังคำนวณผลการ Convolution ของตำแหน่ง 2, 2 ของภาพหลักซึ่งตรงกับจุดกลางของหน้ากากที่ทาบลงไป ผลของการ Convolution จะสามารถคำนวณได้ดังต่อไปนี้

$$\left(108\times\frac{1}{9}+98\times\frac{1}{9}+98\times\frac{1}{9}+110\times\frac{1}{9}+92\times\frac{1}{9}+118\times\frac{1}{9}+100\times\frac{1}{9}+106\times\frac{1}{9}+98\times\frac{1}{9}\right)=103.1111$$

เมื่อแปลงค่าที่คำนวณได้ให้เป็นจำนวนเต็มก็จะเท่ากับ 103 ซึ่งค่าดังกล่าวจะถูกนำไปวางไว้ที่ภาพที่ได้จากการ Convolution ณ ตำแหน่งเดียวกับจุดศูนย์กลางของหน้ากากในภาพตั้งต้น นั่นคือ ค่า 103 นี้ ก็จะถูกนำไปวางไว้ในตำแหน่ง 2, 2 ของภาพที่ได้จากการ Convolution ดังในภาพที่ ข-3 ที่แสดงผลการ Convolution ของภาพตั้งต้นและหน้ากากในรูปที่

14 ซึ่งแสดงเฉพาะมุมซ้ายบนและมุมขวาล่างเท่านั้น ซึ่งในความเป็นจริงจะต้องทำการคำนวณทุกจุดที่อยู่ในภาพตั้งต้น



ภาพที่ ข-3 แสดงผลการ Convolution ของภาพตั้งต้นและหน้ากาก

จากตัวอย่างการคำนวณข้างต้น เรามีประเด็นที่ต้องพิจารณาเกี่ยวกับ Convolution ระหว่างภาพหลักและหน้ากากหลายๆ ประการด้วยกัน ดังต่อไปนี้

1. การทาบหน้ากากลงไปที่ภาพหลักนั้นๆ หากส่วนหน้ากากที่ทาบลงไป “ไม่พอดี” กับภาพหลักแล้ว เราจะทำอย่างไรยกตัวอย่างเช่น ถ้าเราทาบศูนย์กลางของหน้ากากไปที่ตำแหน่ง 1,1 ของภาพหลักเลย จะพบว่า บางส่วนของหน้ากากไม่ได้ทาบลงไปที่ภาพหลัก แล้วการคำนวณตามหลักการเดียวกับสมการที่ข้างต้น เราจะพาพจน์มาคูณกับค่าของหน้ากากไม่ส่วนที่ไม่พอดีนี้ได้อย่างไร สำหรับการประมวลผลภาพโดยทั่วไปแล้ว มีวิธีการจัดการหลาย ๆ วิธี แต่วิธีที่เป็นที่นิยมใช้กันมากที่สุดคือ การไม่คำนวณที่ตำแหน่งดังกล่าวเลย แล้วเติมค่าที่ตำแหน่งดังกล่าวด้วยค่าศูนย์ ซึ่งก็หมายถึง ภาพที่ได้จากการ Convolution จะมีขนาดเล็กกว่าภาพหลัก และจะมีขอบสีดำนั้นเอง ซึ่งขนาดของภาพที่ได้จากการ Convolution จะขึ้นอยู่กับขนาดของหน้ากากนั้นเอง

2. ขนาดของหน้ากากที่นำมาทำการ Convolution นั้น จำนวนแถวและหลักจะต้องเป็นจำนวนคี่เท่านั้น ทั้งนี้ก็เพื่อให้สามารถหาตำแหน่งของจุดศูนย์กลางของหน้ากากได้ลงตัว ซึ่งขนาดของหน้ากากนั้นไม่ได้จำกัดอยู่ที่ขนาด 3x3 เท่านั้น แต่สามารถเป็นได้ตั้งแต่ 5x5, 7x7 เรื่อยไป ซึ่งถ้าหากหน้ากากมีขนาดใหญ่ขึ้น ก็จะส่งผลให้เวลาที่ใช้ในการคำนวณยาวนานมากขึ้น

3. การ Convolute นั้น ค่าที่อยู่ในแต่ช่องของหน้ากากจะเป็นตัวกำหนดรูปแบบการกระทำที่ทำกับภาพหลัก ซึ่งตัวอย่างที่ยกมากนั้น เป็นหน้ากากที่ใช้ในการเฉลี่ยค่า (Average Mask) ที่มีความคล้ายคลึงกับตัวกรองสัญญาณความถี่ต่ำผ่าน (Low Pass Filter) ในทางสัญญาณไฟฟ้า นอกจากนั้น จะพบว่าค่าของภาพที่ได้จากการ Convolution นั้นจะต้องแปลงให้

เป็นจำนวนเต็มที่อยู่ระหว่าง 0 ถึง 225 เท่านั้น ซึ่งจากตัวอย่างที่ยกมา ค่าในแต่ละช่องของหน้ากากเป็นบวกหมดและผลรวมของทุกช่องจะเท่ากับหนึ่ง ทำให้ผลการคำนวณที่ได้จึงเป็นจำนวนจริงที่อยู่ระหว่าง 0 ถึง 255 ซึ่งสามารถแปลงเป็นจำนวนเต็มที่อยู่ระหว่าง 0 ถึง 255 ได้อย่างง่ายดาย แต่ทว่า ในตัวอย่างในหัวข้อถัดๆ ไป ค่าในแต่ละช่องของหน้ากาก อาจติดลบทำให้ผลการคำนวณในรูปแบบเดียวกับสมการที่ข้างต้นนั้น ให้ค่าที่ “เกินช่วง” จำนวน 0 ถึง 255 ซึ่งเราจะมีวิธีการแปลงค่าดังกล่าวอย่างไรนั้น เราจะได้ศึกษาในหัวข้อต่อไป

ตัวกรองสัญญาณความถี่ต่ำผ่าน (Low Pass Filter)

เช่นเดียวกันกับการประมวลสัญญาณ ที่มีตัวกรองสัญญาณความถี่ต่ำผ่าน การประมวลภาพก็มีตัวกรองที่ทำงานในลักษณะดังกล่าวเช่นกัน ซึ่งในทางการประมวลภาพแล้วสัญญาณความถี่สูงหมายถึง การที่ค่าความเข้มแสงหรือความเข้มสี มีค่าสูงกว่าบริเวณโดยรอบ และตัวกรองสัญญาณของการประมวลภาพนั้น แบ่งออกเป็น 2 ประเภทคือ ตัวกรองที่ทำงานกับโดเมนความถี่ (Frequency Domain Filter) และตัวกรองที่ทำงานกับค่าความเข้มแสงของภาพโดยตรง (Spatial Domain Filter) สำหรับตัวกรองประเภทแรกนั้น เราจะสามารถออกแบบความถี่ตัดผ่าน (Cut – Off Frequency) ได้ง่ายดายกว่า ทำให้เราสามารถออกแบบความถี่ตัดผ่านและวามคมของตัวกรองสัญญาณได้ดีกว่า แต่เนื่องจากตัวกรองประเภทนี้ทำงานในโดเมนความถี่ ที่ต้องมีการแปลงฟูเรียร์อย่างรวดเร็วแบบ 2 มิติ (2-Dimension Fast Fourier Transformation) ซึ่งเมื่อเปรียบเทียบความเร็วในการคำนวณแล้วจะพบว่า ตัวกรองประเภทที่ 2 นั้น สามารถทำงานได้อย่างรวดเร็วกว่า สำหรับตัวกรองประเภทที่สองนั้น แท้ที่จริงแล้วก็เป็นการ Convolution ภาพตั้งต้นเข้ากับหน้ากากที่ถูกออกแบบให้เป็นตัวกรองสัญญาณความถี่ต่ำผ่าน (Low Pass Filter) ที่มีผลรวมของทุกตำแหน่งของหน้ากากมีค่าเท่ากับหนึ่งเสมอ ซึ่งถ้ามองอย่างง่ายๆ แล้วการ Convolution ด้วยตัวกรองความถี่ต่ำเหล่านี้ แท้ที่จริงแล้วก็คือ การหาค่าเฉลี่ยของความเข้มแสงหรือความเข้มสีที่อยู่รอบๆ พิกเซลที่กำลังพิจารณาอยู่นั่นเอง ตัวอย่างของหน้ากากที่ถูกออกมาเป็นตัวกรองในลักษณะนี้มีดังนี้

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} \qquad \frac{1}{10}\begin{bmatrix}1 & 1 & 1\\ 1 & 2 & 1\\ 1 & 1 & 1\end{bmatrix} \qquad \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

Average mask 2-to-1 mark factor -4 mask

ภาพที่ ข-4 แสดงหน้ากากชนิดต่างๆ ที่ใช้เป็นตัวกรองความถี่ต่ำผ่าน (Low – Pass Filter)

ในการทำงานจริงนั้น การทำงานค่าการวัดต่างๆ รวมทั้งการกำหนดขั้นตอนการประมวลภาพต่างๆ โดยทั่วไป เป็นการที่ผู้ใช่กำหนดค่าต่างๆ กับภาพนิ่งที่จับมาจากชิ้นงานทั้งสิ้น ซึ่งหลังจากที่ได้กำหนดค่าต่างๆ เรียบร้อยแล้ว โปรแกรมจะประมวลภาพที่รับมาจริงจากกล้อง โดยใช้ค่าต่างๆ ที่ผู้ใช้กำหนดขึ้นจากภาพนิ่ง ดังนั้น ในหัวข้อต่อไปจากนี้ จะเป็นการประมวลภาพที่อยู่ไฟล์ ภาพแทนที่จะต้องรับภาพจากกล้องมาทุกครั้ง

<u>สำหรับโปรแกรมเมอร์และนักพัฒนา</u> ในตอนเริ่มพัฒนาโปรแกรมที่ใช้ในการตรวจสอบชิ้นงานนั้น บางท่านอาจจะไปพัฒนาโปรแกรมส่วนอื่นๆก่อน โดยละเว้นส่วนที่เกี่ยวกับการประมวลผลภาพไว้ เนื่องจากอาจจะเห็นว่า การพัฒนาโปรแกรมในส่วนที่เกี่ยวข้องกับการประมวลผลภาพนั้น จะต้องรอให้ได้กล้องซึ่งให้ข้อมูลภาพ และเครื่องมือในการโปรแกรมที่ใช้ดึงข้อมูลภาพจากผู้ผลิตกล้องมาเสียก่อน ในความเป็นจริง หากเราทำการสร้างคลาส Data ที่กล่าวถึงก่อนหน้านี้ ซึ่งจะทำให้เราสามารถมองภาพเป็นเพียงเมตริกซ์ของจำนวนเต็ม 1 หรือ 3 เมตริกซ์ ขึ้นอยู่กับชนิดของภาพที่กำลังพิจารณานั้นๆ ว่าเป็นภาพ Gray Scale หรือภาพสี และเมื่อเราพัฒนาฟังค์ชั่นที่เกี่ยวกับการประมวลภาพ ก็ให้ฟังค์ชั่นเหล่านั้นกระทำการต่างๆ กับเมตริกซ์ตัวนี้ได้เลย โดยไม่จำเป็นต้องรอให้ได้กล้องมาจริงๆเสียก่อน นอกจากนั้น ในการทดสอบการทำงานของฟังค์ชั่นต่างๆ ที่พัฒนาขึ้น เราก็ใช้ข้อมูลภาพที่อยู่ในไฟล์ภาพเป็นตัวทดสอบ ซึ่งจะพบว่า เรามีเครื่องมือในการโปรแกรมมากมาย ที่ใช้ดึงข้อมูลภาพจากไฟล์ภาพเข้าสู่โปรแกรมของเรา ทั้งนี้โปรแกรมจะต้องจัดเรียงข้อมูลเหล่านั้น ลงไปในเมตริกซ์ก่อนที่ส่งไปประมวลต่อไป จวบจนกระทั่งได้กล้องมา เราก็เขียนโปรแกรมเพิ่มเติมอีกเล็กน้อย เพื่อจัดเรียงข้อมูลภาพที่ได้จากกล้อง ซึ่งอยู่ในรูปแบบของ Byte Stream ลงไปในเมตริกซ์ที่ใช้แทนภาพนั้นๆ

การเพิ่มความคมชัดของขอบวัตถุ

หลังจากที่เราได้เรียนรู้เรื่องการ Convolution แล้ว จะพบว่า การปรับปรุงคุณภาพของภาพ รวมทั้งกระบวนการประมวลภาพพื้นฐานบางอย่าง ต่างก็ตั้งอยู่บนพื้นฐานการ Convolution ภาพตั้งต้นเข้ากับหน้ากากที่ออกแบบมาเฉพาะนั้นเอง ดังจะเห็นได้เช่นเดียวกันจากในหัวข้อนี้

ในการรับภาพเข้ามาประมวลผลนั้น บางกรณีขอบของวัตถุที่อยู่ในภาพอาจจะเลือนไป ทำให้ผู้สังเกตมองเห็นได้ไม่ชัดเจน ซึ่งสำหรับการประมวลภาพแบบดิจิตอลนั้น ภาพในกรณีดังกล่าว สามารถปรับปรุงให้มีคุณภาพดีขึ้นได้ โดยการ Convolution ภาพตั้งต้นเข้ากับหนึ่งในหน้ากากที่ออกแบบมาเฉพาะ ดังต่อไปนี้

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

Mask1 Mask2 Mask3

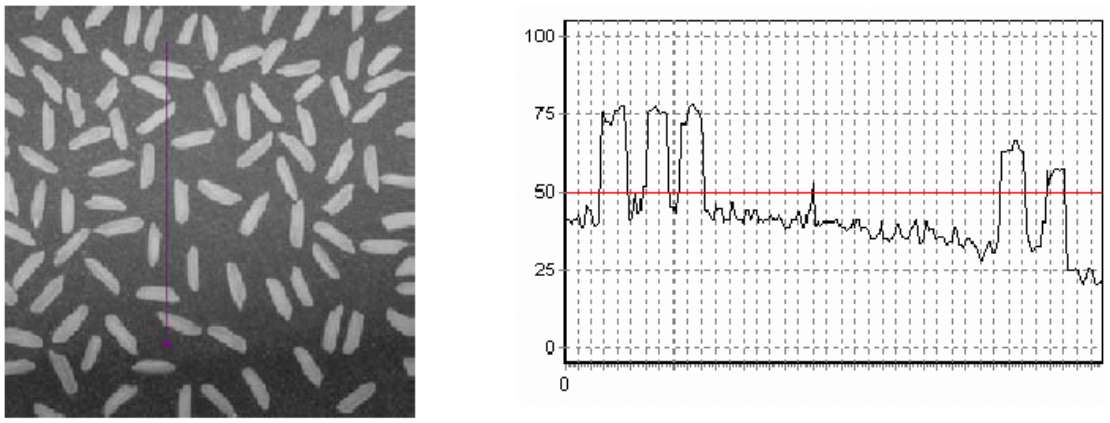
ภาพที่ ข-5 หน้ากากที่ใช้เพิ่มความคมชัดของขอบวัตถุ

การตรวจจับขอบ

ขอบ (Edge) คือ พิกเซลที่มีการเปลี่ยนแปลงความเข้มแสงหรือความเข้มสีเกินกว่าค่า Threshold ที่กำหนดไว้ ซึ่งเปลี่ยนแปลงความเข้มแสงดังกล่าว สามารถตรวจจับได้โดยการใช้ตัวตรวจจับขอบ (Edge detector) ที่มีอยู่ด้วยกันหลากหลายชนิด ถ้าหากกล่าวโดยละเอียดแล้ว การตรวจจับขอบ คือ การตรวจจับการเปลี่ยนแปลงความเข้มแสง หรือความเข้าสีของพิกเซลตำแหน่งนั้น โดยเทียบกับความเข้มแสงหรือความเข้มสีของพิกเซลที่อยู่รอบๆ บริเวณ

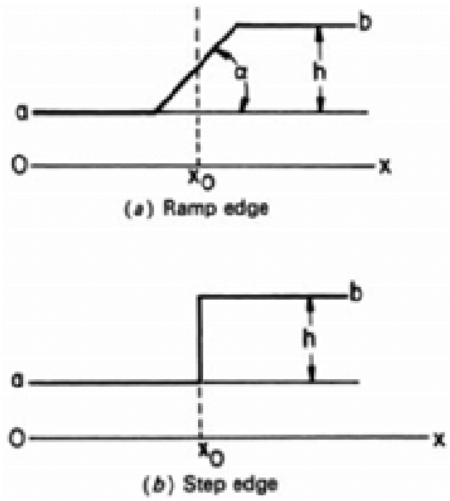
ดังกล่าวไว้แล้วว่า การแบ่งแยกบริเวณ (Segmentation) สามารถทำได้ 2 วิธีคือ การใช้ค่า Threshold เพื่อแปลงภาพ Gray Scale ให้เป็นภาพที่มีเพียง 2 ระดับ เพื่อแยกบริเวณที่เป็นฉากหลังและวัตถุที่สนใจออกจากกัน หรือที่เรียกกันว่า Area Based Segmentation และวิธีที่สองคือ การใช้ขอบของวัตถุเป็นตัวแบ่งแยก หรือที่เรียกกันว่า Edge Based Segmentation ซึ่งวิธีหลังนี้จะถูกนำมาใช้งานตรวจสอบชิ้นส่วนน้อยกว่าวิธีแรก เนื่องจาก ใช้เวลาการคำนวณที่ยาวนานกว่า นอกจากนั้น เนื่องจากสภาพแวดล้อมในการตรวจสอบชิ้นส่วนนั้น โดยมากมักจะมีการกระจายตัวของแสงสม่ำเสมอ ทำให้การแยกบริเวณด้วยวิธีแรกสามารถทำงานได้อย่างมีประสิทธิภาพสูงสุด อย่างไรก็ตาม การตรวจจับขอบก็จัดเป็นหัวข้อที่นักพัฒนาระบบจำเป็นต้องเรียนรู้ไว้ เนื่องจากในสภาพแวดล้อมบางอย่าง การแยกบริเวณด้วยวิธีแรกอาจจะใช้ไม่ได้ผล

ก่อนที่เราจะไปถึงเรื่อง วิธีการใช้ตัวตรวจจับขอบในการตรวจจับการเปลี่ยนแปลงความเข้มแสง เราจะมาพิจารณาถึงลักษณะทางกายภาพของขอบวัตถุที่อยู่ในภาพเสียก่อน ดังในภาพที่ ข-6 ซึ่งแสดง Profile ค่าความแสงตามทิศทางของลูกศร โดยในรูปกราฟนั้น แกนนอนเป็นค่าระยะทางที่มีจุดเริ่มต้น (เลข 0 ที่อยู่ในรูป) อยู่ที่หางของลูกศร และมีจุดสิ้นสุดอยู่ที่หัวลูกศรของในรูป และแกนตั้งเป็นค่าความเข้มแสงที่ถูกปรับสเกลให้อยู่ในช่วง 0 ถึง 100 เปอร์เซ็นต์ เพื่อความสะดวกในการแสดงผล (ความเข้มแสงจริงนั้นมีค่า 0 ถึง 255 แต่ในรูปกราฟนั้น แสดงค่าที่ถูกปรับสเกลไว้ โดยค่าความเข้มแสงเท่ากับ 100 นั้น หมายถึงที่ตำแหน่งนั้น ๆ มีค่าความสว่างหรือความเข้มแสงสูงสุด คือเท่ากับ 255 เลย)



ภาพที่ ข-6 แสดงเส้นตรงและ Profile ของความเข้มแสงที่เส้นตรงลากผ่าน

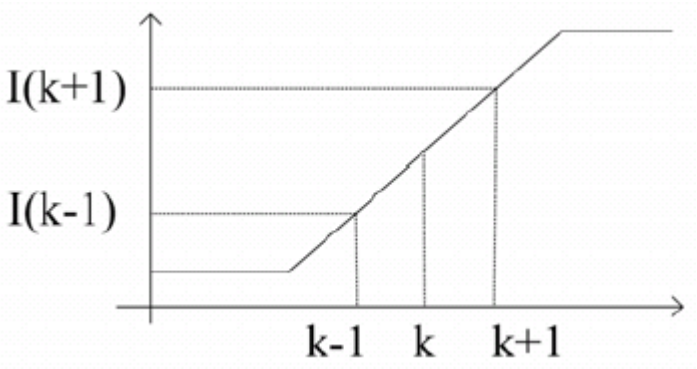
จากภาพที่ ข-6 จะพบว่า ส่วนที่เป็นขอบวัตถุนั้น คือ บริเวณที่มีการเปลี่ยนแปลงความเข้มแสงจากด้านมืดไปสู่ด้านที่สว่างหรือเปลี่ยนจากด้านที่สว่างไปสู่ด้านที่มืด ซึ่งบริเวณที่มืดนั้นหมายถึงบริเวณที่มีค่าความเข้มแสงน้อยกว่าค่า Threshold และในทางกลับกัน ด้านสว่างคือบริเวณที่มีค่าความเข้มแสงมากกว่าค่า Threshold นั่นเอง ซึ่งหากมองจาก Profile ความเข้มแสงแล้ว เราจะพบว่า ในทางปฏิบัติ ค่าความเข้มแสงจะไม่ได้เปลี่ยนแปลงอย่างทันทีทันใด แต่จะค่อยๆ เปลี่ยนแปลงอย่างช้าๆ เมื่อเทียบกับระยะทางที่เพิ่มขึ้น ซึ่งถ้าเราถือว่า พิกเซลที่มีค่าความเข้มแสงตัดผ่านค่า Threshold (ซึ่งในกรณีนี้ ค่า Threhold ก็ถูกปรับสเกลให้อยู่ในช่วง 0 ถึง 100 เช่นกัน) เป็นขอบวัตถุที่อยู่ในภาพแล้ว จะพบว่า ตำแหน่งของขอบวัตถุจะเปลี่ยนแปลงไปตามค่า Threshold ที่เลือกใช้ ดังแสดงไว้ในภาพที่ ข-7



ภาพที่ ข-7 แสดงตำแหน่งขอบของวัตถุ

ภาพ (a) คือ ขอบที่เกิดขึ้นจริง ซึ่งความเข้มแสงมีการเปลี่ยนแปลงอย่างช้าๆ ทำให้ตำแหน่งของของวัตถุขึ้นอยู่กับค่า Threshold ที่เลือกใช้
ภาพ (b) คือ ขอบในอุดมคติ ซึ่งมีการเปลี่ยนแปลงความเข้มแสงอย่างทันทีทันใด ทำให้ตำแหน่งขอบมีเพียงตำแหน่งเดียว

การที่ขอบมีการเปลี่ยนตำแหน่ง เนื่องจากค่า Threshold ที่เปลี่ยนไปนั้น ทำให้มีแนวคิดที่จะตรวจจับขอบวัตถุในภาพด้วยวิธีอื่น แทนที่จะเป็นการพิจารณาตำแหน่งที่ค่าความเข้มแสงตัดผ่านค่า Threshold โดยตรง ซึ่งจะทำให้ตำแหน่งขอบที่ได้คาดเคลื่อนได้ง่าย แนวคิดนี้ก็นำเราไปสู่การตรวจจับขอบวัตถุที่อยู่ในภาพ โดยการตรวจจับความชันของเส้นกราฟแทน (เส้นกราฟที่กล่าวถึงในที่นี้ คือ Profile ความเข้มแสงที่แสดงไว้ในภาพที่ ข-7(a)) ยกตัวอย่างเช่นในภาพที่ ข-8 ค่าความที่ตำแหน่ง k สามารถคำนวณได้จากสมการที่ (ข-1)



ภาพที่ ข-8 รูปประกอบการคำนวณหาค่าความชันที่ตำแหน่ง k

$$\text{Slop}_{@k} = \frac{I_{k+1} - I_{k-1}}{2} \qquad \text{(ข-1)}$$

สมการที่ (ข-1) นั้นเป็นการพิจารณาความชันของเส้นตรง ซึ่งเป็นการพิจารณาการเปลี่ยนแปลงค่าความเข้มแสงใน 1 มิติ เท่านั้นอย่างไรก็ตาม ข้อมูลของรูปภาพนั้นจัดข้อมูล 2 มิติ ดังนั้นจึงมีผู้คิดค้นการตรวจจับการเปลี่ยนแปลงความเข้มแสงแบบ 2 มิติขึ้น โดยมีขั้นตอนดังนี้

1. ตรวจจับการเปลี่ยนแปลงความเข้มแสงในแนวแกน x
2. ตรวจจับการเปลี่ยนแปลงความเข้มแสงในแนวแกน y
3. สำหรับพิกเซลในแต่ละตำแหน่งในภาพ ให้ทำการ “รวม” ค่าการเปลี่ยนแปลงความเข้มแสงของทั้งสองแกน ณ พิกเซลตำแหน่งเดียวกัน ซึ่งค่า “ผลรวม” นี้เรียกว่า ค่าอัตราการเปลี่ยนแปลงความเข้มแสงของภาพ (Intensity Gradient, G) ซึ่งสามารถใช้การคำนวณดังนี้

$$G = \sqrt{G_x^2 + G_y^2} \qquad \text{(ข-2)}$$

เมื่อ G_x คือ ค่าอัตราการเปลี่ยนแปลงความเข้มแสงในแนวแกน x ของพิกเซล ณ ตำแหน่งนั้น

G_y คือ ค่าอัตราการเปลี่ยนแปลงความเข้มแสงในแนวแกน y ของพิกเซล ณ ตำแหน่งเดียวกัน

การคำนวณตามสมการที่ (ข-2) นั้นจะใช้เวลาการคำนวณที่นานมาก เนื่องจากเป็นการถอดรากที่สองของผลบวกกำลังสองของค่าอัตราการเปลี่ยนแปลงความเข้มแสงทั้งสองแกนของทุกพิกเซลที่อยู่ในภาพ ซึ่งเป็นการคำนวณที่ซับซ้อน ดังนั้นเพื่อลดเวลาในการประมวลผล เราจะสามารถประมวลค่าอัตราการเปลี่ยนแปลงความเข้มแสงได้โดยสมการที่ (ข-3) และ (ข-4)

$$G = \max(|G_x|, |G_y|) \qquad \text{(ข-3)}$$

หรือ

$$G = |G_x| + |G_y| \qquad \text{(ข-4)}$$

4. หลังจากที่เราได้ค่าอัตราการเปลี่ยนแปลงความเข้มแสง (Intensity gradient) ของแต่ละพิกเซลที่อยู่ในภาพแล้ว เราก็จะทำการตัดสินว่า พิกเซลใดบ้างที่จัดว่าเป็นขอบของวัตถุ โดยการเปรียบเทียบค่าอัตราการเปลี่ยนแปลงความเข้มแสงของพิกเซลกับค่า Threshold ที่กำหนดขึ้น หากค่าอัตราการเปลี่ยนแปลงของพิกเซลใดมีค่าสูงกว่าค่า Threshold เราก็จะพิจารณาว่า พิกเซลที่ตำแหน่งนั้นจัดเป็นขอบวัตถุที่อยู่ในภาพ

จากขั้นตอนการคำนวณข้างต้น ก็จะมีคำถามว่า แล้วเราจะคำนวณการเปลี่ยนแปลงความเข้มแสงในทั้งสองแนวแกนตามขั้นตอนที่ 1 และขั้นตอนที่ 2 ได้อย่างไร ซึ่งก็ได้มีผู้คิดค้นวิธีการคำนวณอัตราการเปลี่ยนแปลงความเข้มแสงดังกล่าว โดยการ Convolution ภาพตั้งต้นเข้ากับหน้ากากที่ใช้ตรวจจับการเปลี่ยนแปลงความเข้มแสงในแต่ละแนวแกน และชื่อของหน้ากากแต่ละแบบก็จะตั้งตามชื่อของผู้เสนอหน้ากากนั้นๆ ดังแสดงไว้ในตารางที่ ข-1

ตารางที่ ข-1 แสดงหน้ากากที่ใช้ตรวจจับการเปลี่ยนแปลงความเข้มแสงทั้งสองแนวแกน

Name	x kernel	y kernel
Sobel	$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$
Prewits	$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$
Robinson	$\begin{bmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix}$
Kirsch	$\begin{bmatrix} -5 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & 3 & 3 \end{bmatrix}$	$\begin{bmatrix} 3 & 3 & 3 \\ 3 & 0 & 3 \\ -5 & -5 & -5 \end{bmatrix}$

หมายเหตุ จากตารางที่ ข-1 นั้น จะพบว่า หน้ากากที่ใช้ตรวจจับการเปลี่ยนแปลงความเข้มแสงตามแนวแกน x นั้น ก็คล้ายคลึงกับสมการการหาความชันของเส้นตรงที่ภาพที่ ข-8 นั้นคือ ความชันหรืออัตราการเปลี่ยนความเข้มแสงของพิกเซลที่อยู่ตรงกลาง สามารถคำนวณได้โดยการลบค่าความเข้มแสงของพิกเซลที่อยู่ถัดไปด้วยค่าความเข้มแสงของพิกเซลที่อยู่ก่อนหน้า ซึ่งการตรวจจับการเปลี่ยนแปลงความเข้มแสงตามแนวแกน y ก็จะเป็นไปในลักษณะเดียวกัน

การยืดฮิสโตแกรม (Histogram Stretching) และการเกลี่ยฮิสโตแกรม (Histogram Equalization)

ฮิสโตแกรม คือ การนับจำนวนพิกเซลที่มีความเข้มแสงเดียวกัน ดังนั้น สำหรับระบบภาพ Gray scale ที่มีความเข้มแสงเป็นจำนวนเต็มที่ที่อยู่ระหว่าง 0 ถึง 255 แล้ว ฮิสโตแกรมก็จะประกอบด้วยจำนวนนับ 256 จำนวนซึ่งเป็นการนับจำนวนพิกเซลที่มีความเข้มแสงเท่ากับ 0, เท่ากับ 1 เรื่อยไปจนถึงจำนวนพิกเซลที่มีความเข้มแสงเท่ากับ 255 ซึ่งจะพบว่า ฮิสโตแกรมนั้นจะให้ข้อมูลเรื่องลักษณะการกระจายตัวของความเข้มแสงของภาพ ซึ่งสามารถนำไปใช้เพื่อปรับปรุงคุณภาพของภาพได้ด้วยวิธีต่อไปนี้

การยืดฮิสโตแกรม (histogram stretching)

สำหรับระบบภาพ 8 บิตที่มีความเข้มแสงอยู่ระหว่าง 0 ถึง 255 หากข้อมูลภาพที่เข้ามามีฮิสโตแกรมหรือการกระจายตัวของความเข้มแสงในภาพไปอยู่ที่ช่วงความเข้มแสงช่วงหนึ่ง เช่น ค่าความเข้มแสงของทั้งภาพอยู่ในช่วง 120 ถึง 150 เท่านั้น ซึ่งเป็นการใช้ช่วงเข้มแสงไม่เต็มช่วง และจากการที่ความเข้มแสงของวัตถุในภาพมีความแตกต่างกันไม่มากนี่เอง ที่ทำให้มองเห็นวัตถุที่อยู่ในภาพได้ไม่ชัดเจน เนื่องจากความเข้มแสงของวัตถุในภาพแตกต่างกันไม่มาก เพราะจากความเข้มแสงส่วนใหญ่ไป “กอด” กันอยู่ที่ค่า 120 ถึง 150 เท่านั้น การยืดฮิสโตแกรมเป็นการยืดช่วงความเข้มแสงของพิกเซลทั้งหมดออกไปให้เต็มช่วง (ในกรณีของระบบภาพ 8 บิต คือการยืดฮิสโตแกรมออกไปให้เต็มช่วง 0 ถึง 255 นั่นเอง) ซึ่งเป็นการเพิ่มความแตกต่างของความเข้มแสงของพิกเซลที่อยู่ในภาพ ทำให้เราสามารถมองเห็นวัตถุแต่ละชิ้นได้อย่างชัดเจน เนื่องจากหลังจากที่ยืดฮิสโตแกรมแล้วความเข้มแสงของวัตถุในภาพจะมีความแตกต่างกันมากขึ้น

สำหรับการยืดฮิสโตแกรมนั้น แท้ที่จริงก็คือการคำนวณบัญญัติไตรยางค์ธรรมดาที่พิจารณาความเข้มแสงของทุกพิกเซลที่อยู่ในภาพตั้งต้น ดังกรณีที่ยกมา ที่มีค่าฮิสโตแกรมเดิมอยู่ระหว่าง 120 ถึง 150 หากพิกเซลที่กำลังพิจารณามีค่าความเข้มแสงเท่ากับ 125 และอีกพิกเซลมีค่า 128 เมื่อทำการยืดฮิสโตแกรมให้เต็มช่วง 0 ถึง 255 ค่าความเข้มแสงใหม่ของพิกเซลแรกจะเปลี่ยนเป็น และค่าความเข้มแสงของพิกเซลที่กล่าวถึงตัวที่สองจะเปลี่ยนเป็น ซึ่งจะพบว่าความแตกต่างของความเข้มแสงระหว่างพิกเซลนี้เอง ที่จะทำให้เราสามารถภาพได้ชัดเจนขึ้นกว่าของภาพตั้งต้น สำหรับการคำนวณค่าความเข้มแสงใหม่ของแต่ละพิกเซลนั้น จะใช้สมการที่ (ข-5) ต่อไปนี้

$$O = \left(2^{B} - 1\right)\frac{I - \min I}{\max I - \min I} \qquad \text{(ข-5)}$$

เมื่อ B คือ จำนวนบิตที่ใช้แทนค่าความเข้มแสง ซึ่งกรณีทั่วไปมีค่าเท่ากับ 8 บิต
I คือ ค่าความเข้มแสงของพิกเซล ณ ตำแหน่งที่กำลังพิจารณาของภาพตั้งต้น
minI คือ ค่าความเข้มแสงต่ำสุดของภาพตั้งต้น
maxI คือ ค่าความเข้มแสงสูงสุดของภาพตั้งต้น
O คือ ค่าความเข้มแสงของพิกเซล ณ ตำแหน่งเดียวกัน ของภาพขาออกที่ผ่านการยืดฮิสโตรแกรมแล้ว

การเกลี่ยฮิสโตแกรม (Histogram Equalization)

จากการทำงานของกระบวนการยืดฮิสโตแกรมก่อนหน้านั้น จะพบว่า การคำนวณค่าความเข้มแสงของภาพขาออกที่ได้จากการยืดฮิสโตแกรมโดยการใช้สมการที่ (ข-5) นั้นจะพบว่า

จะมีข้อด้อยในทางปฏิบัติ คือ โดยปกติจะมีบางพิกเซลที่ค่าความเข้มแสงเท่ากับหรือใกล้เคียงกับค่าต่ำสุด และบางพิกเซลมีค่าความเข้มแสงเท่ากับหรือใกล้เคียงค่าสูงสุดอยู่แล้ว ซึ่งถึงแม้ว่าในภาพนั้น พิกเซลส่วนใหญ่จะไป “กอด” กันอยู่ที่ช่วงความเข้มแสงใดช่วงความเข้มแสงหนึ่ง แต่เนื่องจากได้มีบางพิกเซลไปตรึงอยู่ที่ค่าขอบของความเข้มแสงทั้งสองด้านไปแล้ว ทำให้ค่าความเข้มแสงของแต่ละพิกเซลในภาพขาออก ที่คำนวณโดยการใช้สมการที่ (ข-5) นั้น มีค่าไม่แตกต่างหรือแตกต่างน้อยมาก เมื่อเทียบกับวามเข้มแสงของแต่ละพิกเซลในตำแหน่งเดียวของภาพตั้งต้น (ทดลองแทนค่าในสมการที่ (ข-5) ถ้าค่า minI เท่ากับ 0, maxI มีค่าเท่ากับ 255 และ I ซึ่งคือค่าความเข้มแสงของพิกเซลที่อยู่ในภาพตั้งต้นมีค่าเท่ากับ 125 พิจารณาดูนะครับว่า ค่าความเข้มแสงของภาพขาออกที่ตำแหน่งเดียวกันมีค่าเท่าใด

จากข้อด้อยของกระบวนการยืดฮิสโตแกรมดังกล่าว จึงทำให้มีผู้คิดค้นวิธีที่จะทำให้วัตถุในภาพมีความเข้มแสงต่างกัน เพื่อที่จะได้มองเห็นได้อย่างชัดเจน โดยวิธีการเกลี่ยฮิสโตแกรม (Histogram Equalization) ขึ้น ซึ่งกระบวนการดังกล่าวมีความคล้ายคลึงกับปรับหน้าดินให้เรียบของงานก่อสร้าง นั้นคือ เมื่อพิจารณาฮิสโตแกรมของภาพที่ได้จากการเกลี่ยฮิสโตแกรม จะพบว่า อัตราส่วนของจำนวนพิกเซลรวมกันตั้งแต่ค่าความเข้มแสงต่ำสุดจนถึงค่าความเข้มแสงที่กำลังพิจารณา หารด้วยค่าความเข้มแสงที่จุดนั้นๆ จะมีค่าคงที่ ไม่ว่าจะพิจารณาที่ค่าความเข้มแสงใดๆ ยกตัวอย่างเช่น หลังจากเกลี่ยฮิสโตแกรมแล้ว จะพบว่าจำนวนพิกเซลที่มีความเข้มแสงตั้งแต่ค่าต่ำสุดคือ 0 จนถึงความเข้มแสงเท่ากับ 20 รวมกัน แล้วหารได้ด้วย 20 จะมีค่าเท่ากับ จำนวนพิกเซลที่มีวามเข้มแสงตั้งแต่ค่าต่ำสุดคือ 0 จนถึงความเข้มแสงเท่ากับ 120 รวมกัน แล้วหารได้ด้วย 120 นั่นเอง หรือกล่าวอย่างง่ายๆ จะได้ว่า ฮิสโตแกรมของภาพที่ผ่านกระบวนการเกลี่ยฮิสโตแกรม จะมีให้ความหนาแน่นของจำนวนพิกเซลต่อค่าความเข้มแสงคงที่ตลอดย่านความเข้มแสง ซึ่งการเกลี่ยฮิสโตแกรมนี้จะให้ผลที่ดีกว่าการยืดฮิสโตแกรม คือทำให้วัตถุในภาพมีความเข้มแสงแตกต่างกันจนเห็นได้อย่างชัดเจน และสามารถนำไปใช้ได้กับทั้งภาพหรือส่วนใดส่วนหนึ่งของภาพ

การเกลี่ยฮิสโตแกรมนั้น มีกระบวนการย่อยอยู่ 2 ขั้นตอนดังนี้ คือ

1. ทำการคำนวณหาฮิสโตแกรม และความถี่สะสมของแต่ละความเข้มแสง
2. คำนวณค่าความเข้มแสงใหม่ของแต่ละพิกเซลของภาพตั้งต้น

ในขั้นตอนที่ 2 ซึ่งเป็นการคำนวณค่าความเข้มแสงใหม่ของแต่ละพิกเซลของภาพตั้งต้น ซึ่งจะนำไปไว้ในภาพขาออกที่ตำแหน่งเดียวกันนั้น สามารถทำได้โดยใช้สมการที่ (ข-6) ดังต่อไปนี้

$$O_i = \left[\sum_{j=0}^{i} N_j \right] \times \frac{\text{Max.IntensityLevel}}{\text{No.ofPixels}} \qquad \text{(ข-6)}$$

เมื่อ Max.Intensity คือ ค่าความเข้มแสงสูงสุดของระบบภาพ ซึ่งเท่ากับ 255 เมื่อเป็นระบบภาพ 8 บิต

No. of pixels คือ จำนวนพิกเซลทั้งหมด เช่น ถ้าภาพมีขนาด 256 x 256 จำนวนพิกเซลก็จะเท่ากับ 65536 พิกเซล

i คือ ค่าความเข้มแสงของพิกเซลที่กำลังพิจารณาของภาพตั้งต้น

O_i คือ ค่าความเข้มแสงใหม่ที่จะนำไปวางในตำแหน่งเดียวกันกับพิกเซลที่ของภาพตั้งต้น

สำหรับพจน์ที่อยู่ในวงเล็บนั้น หมายถึง จำนวนพิกเซลที่มีความเข้มแสงตั้งแต่ต่ำสุดจนถึงค่าความเข้มแสงที่กำลังพิจารณาทั้งหมดรวมกัน เช่น หากพิกเซลของภาพตั้งที่กำลังพิจารณาในขณะนั้นมีค่าเท่ากับ 1 แล้ว พจน์ที่อยู่ในวงเล็บก็จะมีค่าเท่ากับผลรวมของจำนวนพิกเซลที่มีค่าความเข้มแสงเท่ากับ 0 และ 1 หรือถ้าหากพิกเซลของภาพตั้งต้นที่กำลังพิจารณาในขณะนั้นมีค่าเท่ากับ 5 พจน์ที่อยู่ในวงเล็บก็จะมีค่าเท่ากับผลรวมของจำนวนพิกเซล ที่มีค่าความเข้มแสงเท่ากับ 0 1 2 3 4 และ 5 เป็นต้น ซึ่งจากการคำนวณในลักษณะนี้เองทำให้เราจำเป็นต้องทำการคำนวณทั้งฮิสโตแกรมและความถี่สะสมของแต่ละความเข้มแสงควบคู่กันไป

ตัวกรองค่ากลาง (Median Filter)

การทำงานของตัวกรองความถี่ต่ำผ่าน (Low Pass Filter) ในหัวข้อก่อนหน้า เป็นการเฉลี่ยค่าความเข้มแสงของบริเวณที่อยู่รอบข้างของพิกเซลที่กำลังพิจารณาในขณะนั้น ตัวกรองดังกล่าวใช้เพื่อลดสัญญาณรบกวนที่เกิดขึ้นแบบสุ่ม ซึ่งนอกจากจะลดสัญญาณรบกวนความถี่สูงแล้ว ตัวกรองดังกล่าวยังให้ผลข้างเคียงคือ ไปลดความคมชัดของขอบวัตถุที่ใช้แยกวัตถุออกจากพื้นหลังและบริเวณอื่นๆ ดังนั้นเพื่อแก้ปัญหาดังกล่าว จึงมีผู้ออกแบบตัวกรองค่ากลาง (Median Filter) ขึ้น ซึ่งตัวกรองชนิดนี้ จะไม่ได้ใช้การ Convolution ดังเช่นกรณีของตัวกรองความถี่ต่ำผ่าน แต่จะใช้การเลือกค่ากลางแบบเดียวกับการหาค่ากลางของวิธีการทางสถิติ ซึ่งการทำงานของตัวกรองชนิดนี้ จะสามารถขจัดสัญญาณรบกวนที่มีค่าแตกต่างอย่างมากกับบริเวณรอบข้างได้ดี (หรือที่เรียกกันว่าสัญญาณกวนประเภท Salt and Pepper Noise)

สำหรับการทำงานของตัวกรองค่ากลางนั้น จะเป็นการพิจารณาค่าของพิกเซลของบริเวณในภาพตั้งต้นที่หน้ากาก “ทาบ” ลงไปพอดี หลังจากนั้นก็ใช้ค่ากลางของพิกเซลนั้นไปวางไว้ที่ภาพขาออกในตำแหน่งกับพิกเซลที่ศูนย์กลางหน้ากากทาบลงไป ยกตัวอย่างเช่น ถ้าเป็นการทำงานของตัวกรองค่ากลางขนาด 3x3 ซึ่งถ้าศูนย์กลางของหน้ากอกทาบลงไปที่พิกัด (I,j) ใดๆ แล้วตัวกรองค่ากลางจะเลือกค่ากลางจากตำแหน่งนั้นและบริเวณที่อยู่รอบๆ ที่มีพิกัดดังนี้ (i-1,j-1), (i-1,j), (i-1,j+1), (i,j-1), (i,j+1), (i+1,j-1),(i+1,j) และ (i+1,j+1) ซึ่งตำแหน่งตรงกลางและพิกัดที่อยู่รอบๆ ของหน้ากากขนาด 3x3 แสดงไว้ในภาพที่ ข-9 ซึ่งจากค่ากลางที่ได้จะนำไปวางไว้ที่ตำแหน่งเดียวกันกับศูนย์กลางของหน้ากากในภาพขาออกของกระบวนการ

i-1, j-1	i-1, j	i-1, j+1
i, j-1	i, j	i, j+1
i+1, j-1	i+1, j	i+1, j+1

ภาพที่ ข-9 แสดงพิกัดของภาพที่หน้ากากขนาด 3x3 ทาบลงไป โดยมีพิกัด (i,j) เป็นศูนย์กลางของหน้ากาก

การเขียนโปรแกรม C++ สำหรับการเข้ารหัสแบบ Run + Length Encoding และแบบ Chain code

การเข้ารหัสแบบ Run + Length Encoding และแบบ Chain Code นั้น ถ้าได้ Label Matrix มาแล้ว การเข้ารหัสทั้ง 2 รูปแบบก็ไม่ใช่เรื่องยากนัก อย่างไรก็ตาม มีประเด็นที่ต้องนำมาพิจารณาด้วยว่า แล้วจะเก็บข้อมูลที่ได้จากการเข้ารหัสเหล่านี้ไว้ในรูปแบบใด เมื่อเราไม่ทราบจำนวนของวัตถุทั้งหมดที่อยู่ในภาพก่อนหน้า

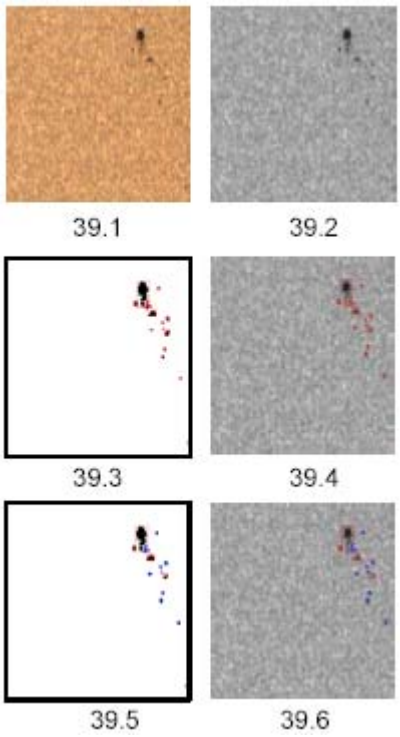
การเข้ารหัส Chain Code นั้น จะพบว่า สำหรับขอบวัตถุแต่ละชิ้นนั้น เราจำเป็นจะต้องเก็บข้อมูล 2 อย่างด้วยกัน คือ พิกัดของพิกเซลแรกที่พบขอบ และรหัสซึ่งใช้แทนการเปลี่ยนแปลงทิศทางการเดินตามเข็มนาฬิกาจนครบรอบ ซึ่งถ้าเป็นการทำงานอย่างง่ายนั้น เราสามารถเก็บพิกัดของพิกเซลแรกที่พบขอบ ในรูปของหมายเลขหลักและหมายเลขแถวที่เป็นตัวแปรประกอบ int และเก็บข้อมูลที่เป็น Chain Code โดยใช้สตริงในการเก็บ ซึ่งถ้าเราต้องการเก็บข้อมูลทั้งสองไว้ในที่เดียวกัน ก็ทำได้โดยการเก็บไว้ในรูปแบบของตัวแปร Structure นั้น เราจะใช้เครื่องหมายเท่ากับในการกำหนดค่าให้กับทั้งตัวแปรไม่ได้ อย่างไรก็ตาม เนื่องจากเราไม่สามารถทราบจำนวนขอบวัตถุที่อยู่ในภาพได้ก่อนหน้า ทำให้ไม่สามารถใช้อาร์เรย์ในการเก็บข้อมูลของขอบวัตถุทุกชิ้นที่อยู่ในภาพได้ จึงเป็นเงื่อนไขบังคับให้เราต้องใช้ Linked List ในการเก็บข้อมูลดังกล่าว

การเข้ารหัสแบบ Run + Length Encoding นั้น จะพบว่า เราจำเป็นต้องเก็บข้อมูลของทุกวัตถุที่อยู่ในภาพเป็นแบบ Linked List ของ Linked List เสมอ (ไม่ใช่ Double Linked List) โดยจะพบว่า สำหรับวัตถุ 1 ชิ้นนั้น จะมีจำนวน Run ที่ไม่ทราบจำนวนมาก่อน จึงเป็น

เงื่อนไขให้ สำหรับวัตถุ 1 ชิ้นเราต้องเก็บข้อมูลของ Run ในรูปแบบของ Liked List และ เนื่องจากเราไม่สามารถทราบจำนวนของวัตถุทั้งหมดที่อยู่ในภาพ ทำให้เราต้องเก็บวัตถุเหล่านั้น ในรูปแบบของ Linked List อีกที่หนึ่ง

ตัวอย่างการตรวจสอบสิ่งปนเปื้อนบนพื้นผิว

ตัวอย่างต่อไปนี้ เป็นการตรวจจับรอยเปื้อนบนผิวชิ้นงาน (Contamination Detection) ซึ่งเป็นการประยุกต์ใช้กระบวนการทั้งหมดที่กล่าวมา ประกอบกับการตัดสินอย่างง่าย เพื่อตัดสินว่า วัตถุแต่ละชิ้นที่ปรากฏอยู่ในภาพนั้น จัดเป็นรอยเปื้อนหรือไม่



ภาพที่ ข-10 แสดงขั้นตอนการตรวจจับรอยเปื้อนบนผิวชิ้นงาน

การตรวจจับรอยเปื้อนผิวชิ้นงานนั้น ในตอนแรกเริ่ม ผู้ใช้งานระบบจะต้องระบุบริเวณที่จะทำการตรวจจับเสียก่อน ในกรณีนี้กำหนดให้เป็นพื้นที่สี่เหลี่ยม ซึ่งจัดเป็นรูปร่างที่ธรรมดาที่สุด หลังจากนั้น จึงเริ่มกระบวนการแปลงภาพสีที่อยู่ในรูปตัวอย่าง 39.1 ให้กลายเป็น Gray Scale เสียก่อนโดยใช้มาตรฐาน NTSC ที่ได้นำเสนอไป ซึ่งในกรณีนี้จะพบว่าภาพที่ได้มีความชัดเจน จึงไม่จำเป็นต้องใช้การประมวลภาพเบื้องต้นมาปรับปรุงคุณภาพของภาพแต่อย่างใด หลังจากนั้น ด้วยการใช้ค่า Threshold ที่เลือกโดยวิธีของ Otsu เราจะได้ภาพ Binary ที่ประกอบด้วยส่วนที่เป็นวัตถุและส่วนที่เป็นฉากหลังสีขาว ซึ่งจะสามารถรู้จำนวนวัตถุตำแหน่ง และพิกัด

ของพิกเซลของวัตถุแต่ละชิ้นที่มีอยู่ในภาพ ได้ด้วยกระบวนการ Connected Components Labeling ดังในรูปตัวอย่าง 39.3 ซึ่งได้เทียบตำแหน่งของวัตถุที่ปรากฏในภาพ Binary กับภาพ Gray Scale ไว้ในรูปตัวอย่าง 39.4 อย่างไรก็ตาม มิได้หมายความว่า วัตถุที่ตรวจจับได้ทุกชิ้นจะเป็นรอยเปื้อนทุกชิ้น ซึ่งถ้าใช้กระบวนการตัดสินอย่างง่ายที่ว่า ขนาดของวัตถุที่ใหญ่กว่า 5 พิกเซลเท่านั้น จึงจะถือว่าเป็นรอยเปื้อน เราจำเป็นจะต้องทำการวัดคุณสมบัติของวัตถุแต่ละชิ้นที่ปรากฏในภาพด้วยกระบวนการ Feature Extraction แล้วเปรียบเทียบกับเงื่อนไขดังกล่าว เพื่อใช้ตัดสินว่า วัตถุชิ้นใดเป็นรอยเปื้อนบ้าง ซึ่งทำให้ท้ายที่สุด เราก็จะได้วัตถุที่จัดเป็นรอยเปื้อนที่ถูกล้อมรอบด้วยกรอบสีแดง และวัตถุที่ไม่จัดเป็นรอยเปื้อนที่ถูกล้อมรอบด้วยสีน้ำเงิน ดังในรูปตัวอย่าง 39.5 และรูปตัวอย่าง 39.6

ซึ่งจะพบว่า หลังจากที่ ผู้ใช้ได้ทำการกำหนดบริเวณที่ต้องการตรวจสอบ กำหนดค่าพารามิเตอร์ที่สำคัญ ๆ ของกระบวนการต่าง ๆ เช่น ค่า Threshold และค่าขนาดวัตถุที่ต่ำสุดที่ถือว่า เป็นรอยเปื้อน หลังจากนั้น โปรแกรมจะใช้ค่าเหล่านี้ เพื่อตรวจสอบชิ้นงานทุกชิ้นที่เข้ามาได้อย่างอัตโนมัติ

จากกระบวนการตรวจสอบรอยเปื้อนอย่างง่ายที่นำเสนอนั้น จะพบว่า เราจำเป็นจะต้องใช้ความรู้ทั้งหมดที่ได้นำเสนอมา ซึ่งกระบวนการตัดสินใจว่า วัตถุที่ตรวจจับได้นั้นจัดเป็นรอยเปื้อนหรือไม่ สามารถทำได้โดยการพิจารณาจากขนาดพื้นที่ของวัตถุเท่านั้น ซึ่งบางกรณีเราอาจจะต้องใช้คุณสมบัติหลาย ๆ ประเภทมาจำแนกว่า วัตถุที่ตรวจจับได้นั้น จัดเป็นรอยเปื้อนหรือไม่ หรือแม้กระทั่งตัวอย่างง่าย ๆ ที่มีแอปเปิ้ลและสาลี่ลำเลียงมาตามสายพาน แล้วเราต้องการจำแนกว่า ผลไม้ที่ผ่านเข้ามานั้นเป็นผลไม้ชนิดใด เพื่อนับจำนวนและนำไปจัดเก็บ ซึ่งจะพบว่า เราจำเป็นต้องใช้คุณสมบัติอื่น ๆ มาใช้พิจารณาด้วย และถ้าคุณสมบัติของผลไม้ทั้งสองมีความแตกต่างกันมาก ๆ เราก็อาจจะใช้การตัดสินใจอย่างง่าย ๆ ที่เรียกกันว่า Decision Tree ซึ่งก็เป็นการเขียนโปรแกรมโดยใช้คำสั่ง if-else if-end เพื่อเปรียบเทียบคุณสมบัติของวัตถุแต่ละชิ้นที่ปรากฏอยู่ในภาพกับลักษณะเด่นของผลไม้แต่ละชนิด แล้วตัดสินใจว่า วัตถุที่กำลังพิจารณานั้นจัดเป็นผลไม้ชนิดใด อย่างไรก็ตาม สำหรับงานจำแนกที่ซับซ้อน ที่คุณสมบัติต่าง ๆ ของวัตถุไม่สามารถแยกออกจากกันได้อย่างชัดเจนนั้น เราจำเป็นจะต้องใช้วิธีที่ดีกว่านี้ ดังที่จะนำเสนอให้หัวข้อต่อไปนี้

ความสำคัญของ Computer Graphics ที่มีต่องานตรวจสอบชิ้นส่วนด้วยภาพแบบอัตโนมัติ

จากเนื้อหาในหัวข้อก่อนหน้านั้น อาจจะทำให้ผู้อ่านอาจจะแปลกใจว่า แค่ความรู้ง่ายๆ ที่ว่ามีพิกเซลใดบ้างในอาร์เรย์ 2 มิติที่เส้นตรงลากผ่าน ก็สามารถนำมาใช้สำหรับงานตรวจสอบชิ้นส่วนด้วยภาพตั้งมากมายหลายประการ ไม่ว่าจะเป็นการใช้วัดระยะห่างระหว่างชิ้นส่วน ใช้นับจำนวนชิ้นส่วนและการเลือกใช้ทำ Threshold แบบอัตโนมัติ ซึ่งการนำความรู้เรื่อง

Computer Graphics มาใช้ในงานตรวจสอบชิ้นส่วนด้วยภาพนั้น ถ้าเป็นผู้ที่มีความรู้เรื่องการประมวลภาพมาก่อนแล้ว อาจจะหลงประเด็นโดยไปมุ่งเน้นที่กระบวนการประมวลภาพแต่อย่างเดียว โดยละเลยที่จะนำความรู้ทางด้าน Computer Graphics มาใช้ก็เป็นไปได้ ดังนั้น ต้องเข้าใจร่วมกันว่า งานตรวจสอบชิ้นส่วนด้วยภาพแบบอัตโนมัติที่เป็นสาขาย่อยของสาขาวิชา Machine Vision นั้น ไม่ได้มุ่งเน้นที่กระบวนการประมวลภาพเพื่อปรับปรุงคุณภาพแต่เพียงอย่างเดียว แต่จำเป็นต้องใช้ความรู้ของสาขาวิชาอื่นๆ มาร่วมด้วย เพื่อให้สามารถปรับปรุงคุณภาพของภาพ ทำการวัด ตีความหมาย และประเมินคุณภาพของผลิตภัณฑ์จากภาพที่รับมาจากกล้องให้ได้ และ Computer Graphics ก็เป็นสาขาหนึ่งที่มีความจำเป็นมากสำหรับงานในสาขานี้ ซึ่งสาเหตุหลักๆ ที่เราจะต้องนำความรู้เรื่อง Computer Graphics มาใช้ในงานตรวจสอบชิ้นส่วนด้วยภาพแบบอัตโนมัติ คือ การตรวจสอบผลิตภัณฑ์ประเภทหนึ่งๆ เท่านั้น จะมีชนิดของการตรวจสอบในแต่ละบริเวณของภาพที่ไม่เหมือนกัน เช่น บางบริเวณจะเป็นการตรวจสอบขนาดและจำนวนของซี่ระบายความร้อน ซึ่งจะถูกระบุเส้นทางในการตรวจสอบโดยวิศวกรคุณภาพหรือผู้ควบคุมคุณภาพของผลิตภัณฑ์นั้นๆ ในลักษณะของเส้นที่มีรูปร่างต่างๆ กัน ในขณะที่บางบริเวณของภาพจะเป็นการตรวจสอบการปนเปื้อนบนพื้นผิวหรือการตรวจจับสีที่ผิดเพี้ยนของผลิตภัณฑ์ ซึ่งจะระบุบริเวณที่ต้องการให้ระบบตรวจสอบเป็นพื้นที่ที่มีรูปร่างต่างๆ กัน และโดยมากแล้วรูปร่างของทั้งเส้นและพื้นที่เหล่านั้นจะไม่เป็นรูปร่างมาตรฐาน เช่น เส้นตรงหรือพื้นที่สี่เหลี่ยมธรรมดาๆ แต่จะมีรูปร่างที่ซับซ้อนกว่านั้นมาก เช่น เป็นเส้นรอบวงของวงกลมหรือ วงรี พื้นที่ที่ล้อมรอบด้วยวงกลมหรือวงรี เส้นหลายเหลี่ยม และที่ซับซ้อนที่สุด คือ พื้นที่ของรูปหลายเหลี่ยม (Solid Polygon) ทั้งนี้เนื่องจาก การที่รูปร่างของผลิตภัณฑ์มีความหลากหลายและซับซ้อนนั้นเอง ซึ่งทำให้เราต้องใช้ความรู้ของ Computer Graphic มาใช้ใน 2 กระบวนการย่อยด้วยกันคือ

1. ใช้ในขั้นตอนการติดต่อระหว่างโปรแกรมกับวิศวกรคุณภาพหรือผู้ควบคุมคุณภาพของผลิตภัณฑ์นั้นๆ โดยทั่วไปแล้วการติดต่อระหว่างโปรแกรมกับผู้ใช้ ซึ่งจะเป็นผู้ระบุเส้นทางในการตรวจสอบหรือพื้นที่ที่ต้องการให้โปรแกรมทำการตรวจสอบนั้น จะเป็นการระบุจุดสำคัญของรูปร่างโดยใช้เม้าส์คลิกลงไปโดยตรงที่ภาพเลย เช่น ถ้าเป็นวงกลมก็จะระบุจุดศูนย์กลางของวงกลมก่อนๆ ที่จะระบุรัศมีของวงกลม หรือหากเป็นพื้นที่ของรูปหลายเหลี่ยม ผู้ใช้ก็ระบุมุมของรูปหลายเหลี่ยมโดยใช้เมาส์คลิกลงไปเลย และเพื่อให้เกิดความสะดวกกับผู้ใช้โปรแกรม จะต้องสามารถแสดงรูปร่างที่ผู้ใช้เลือกและเปลี่ยนรูปร่างนั้นๆ ตามตำแหน่งของเมาส์ซึ่งเปลี่ยนไปได้อย่างทันทีทันใด ซึ่งเทคนิคที่เป็นที่ยอมรับโดยทั่วไป คือ เทคนิคที่เรียกว่า Rubber Band Effect ซึ่งเป็นการวาดเส้นหรือพื้นที่ที่รูปร่างสามารถเปลี่ยนแปลงได้เหมือนหนังยางที่สามารถยืดหดได้นั้นเอง ซึ่งข้อควรระวังสำหรับการเขียนโปรแกรมในลักษณะนี้ คือ ในขณะที่รูปร่างของเส้นตรงหรือพื้นที่ที่การเปลี่ยนแปลงไปตามตำแหน่งของเมาส์นั้น ภาพที่แสดงผลไม่ควรกระพริบ (No Flicking)

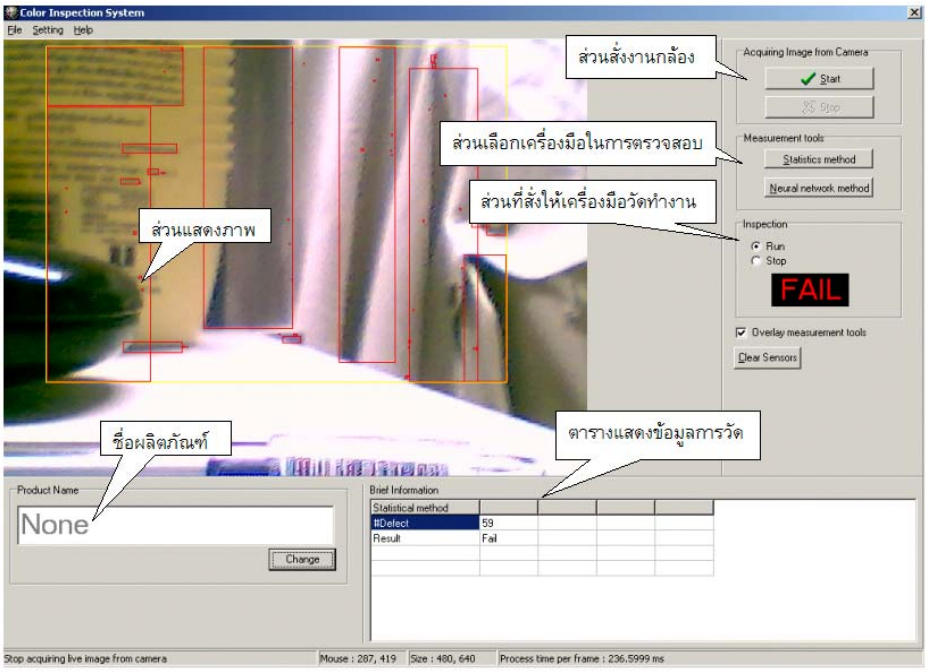
2. ใช้เพื่อระบุว่า จากรูปร่างของเส้นตรงหรือพื้นที่ที่ต้องการให้ระบบตรวจสอบซึ่งระบุโดยผู้ใช้ มีพิกเซลใดบ้างที่จะต้องทำการตรวจสอบ ดังจะเห็นได้จาก โปรแกรมตัวอย่างที่ยกมาที่ผู้ใช้ระบุแค่เพียงจุดเริ่มต้นและจุดปลายของเส้นตรงเท่านั้น โปรแกรมจะต้องทำการหาพิกัดของพิกเซลที่เชื่อมระหว่างจุดทั้งสอง และดึงค่าความเข้มแสงของจุดเหล่านั้นมาแสดงผลเป็น Intensity Profile ให้ได้อย่างอัตโนมัติ และกรณีที่ยุ่งยากที่สุดหัวข้อนี้ คือ หากผู้ใช้ระบุพื้นที่ ที่ต้องการให้ระบบทำการตรวจสอบเป็นรูปหลายเหลี่ยม (Polygon) แล้ว โปรแกรมจะต้องหาให้ได้ว่า พิกเซลใดบ้างที่อยู่ในรูปหลายเหลี่ยมนั้น ก่อนที่จะเริ่มทำการตรวจสอบจริงๆ ซึ่งระเบียบวิธีคำนวณเพื่อให้หาว่า พิกเซลใดบ้างอยู่ใน Graphic เพื่อใช้ระบายสีของพื้นที่รูปหลายเหลี่ยม ซึ่งก่อนจะระบายสีได้จะต้องรู้พิกัดของพิกเซลที่อยู่ในรูปหลายเหลี่ยมเสียก่อน[12]

ภาคผนวก ค

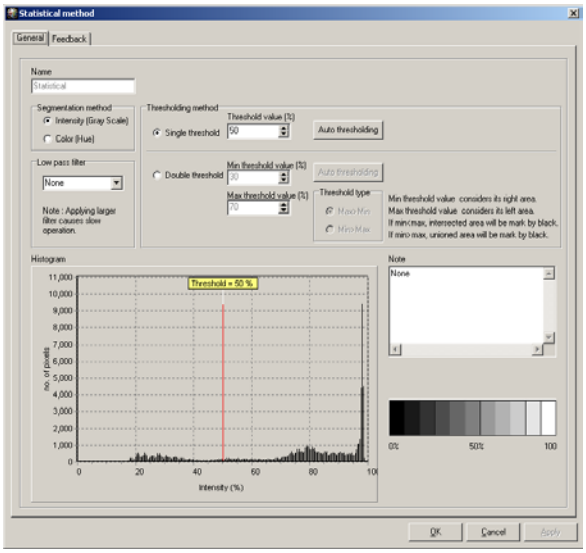
ตัวอย่างการใช้โปรแกรม

ภาคผนวก ค
ตัวอย่างการใช้โปรแกรม

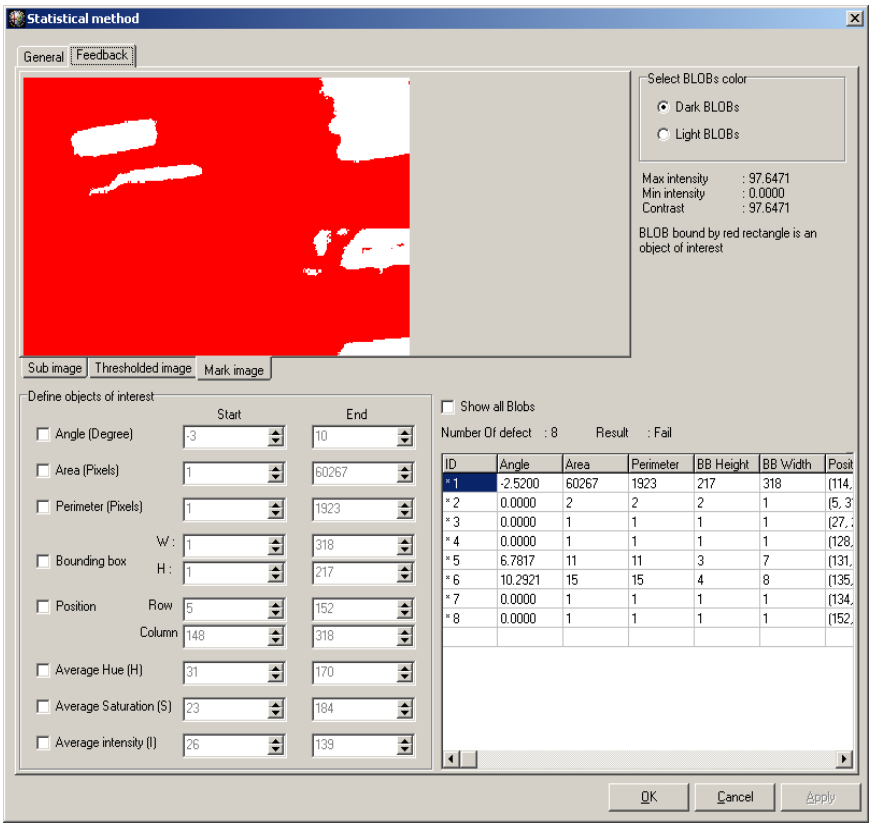
1 ตัวอย่างโปรแกรมที่ได้รับจากงานวิจัยนี้



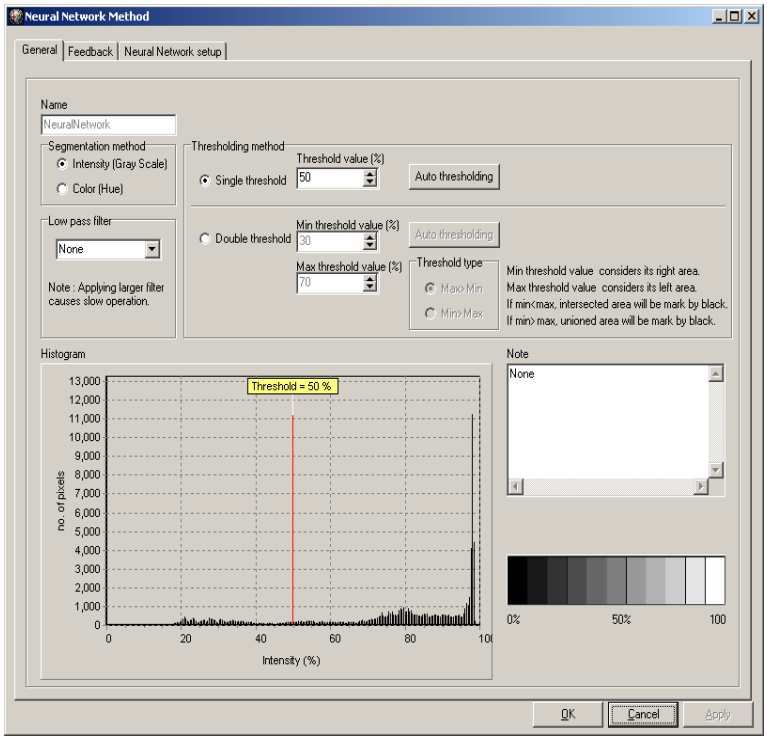
ภาพที่ ค-1 แสดงหน้าต่างหลักของโปรแกรม Color Inspection System



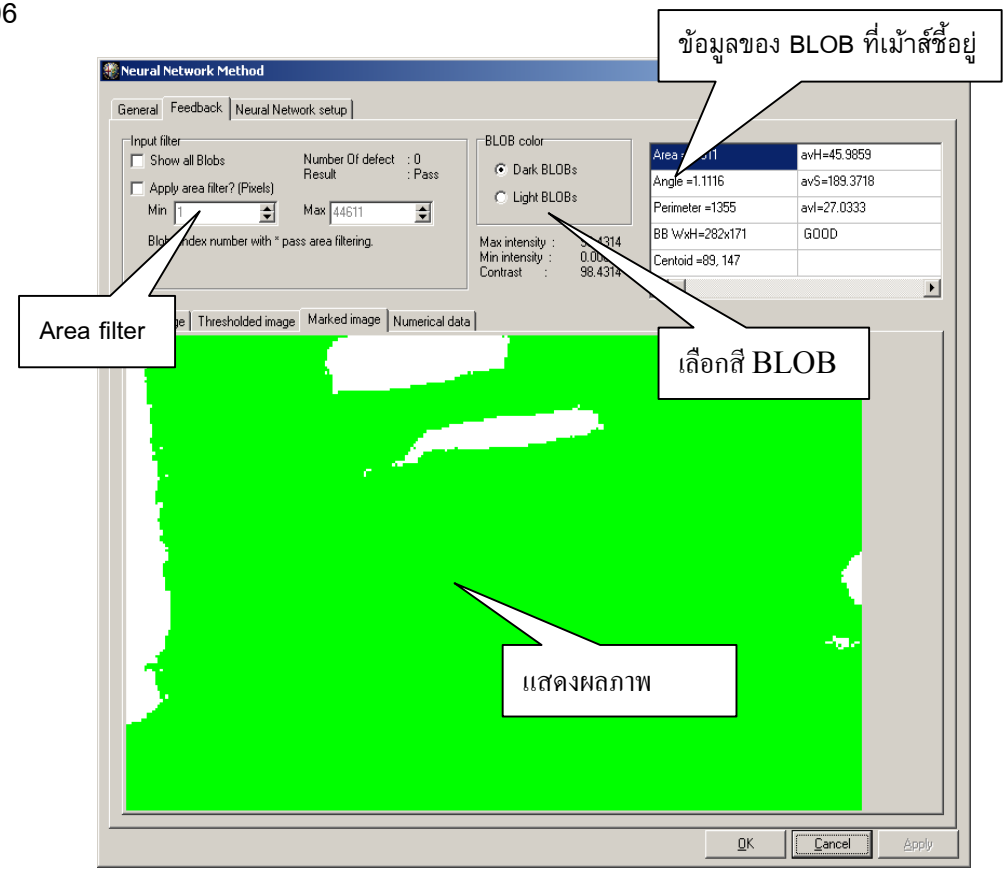
ภาพที่ ค-2 แสดงหน้าข้อมูลทั่วไปของเครื่องมือวัดที่ใช้วิธีทางสถิติ



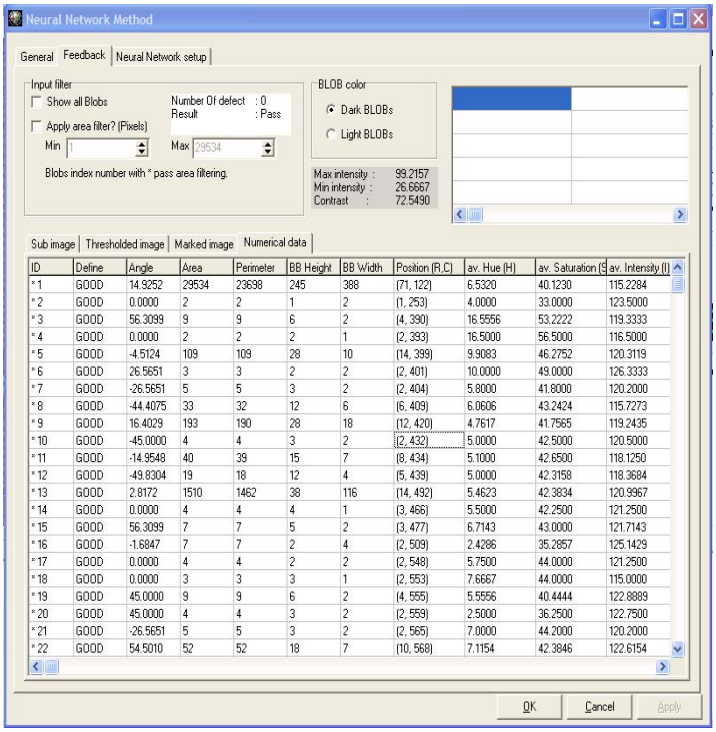
ภาพที่ ค-3 แสดงหน้าป้อนข้อมูลกลับให้กับผู้ใช้ของเครื่องมือวัดที่ใช้วิธีทางสถิติ



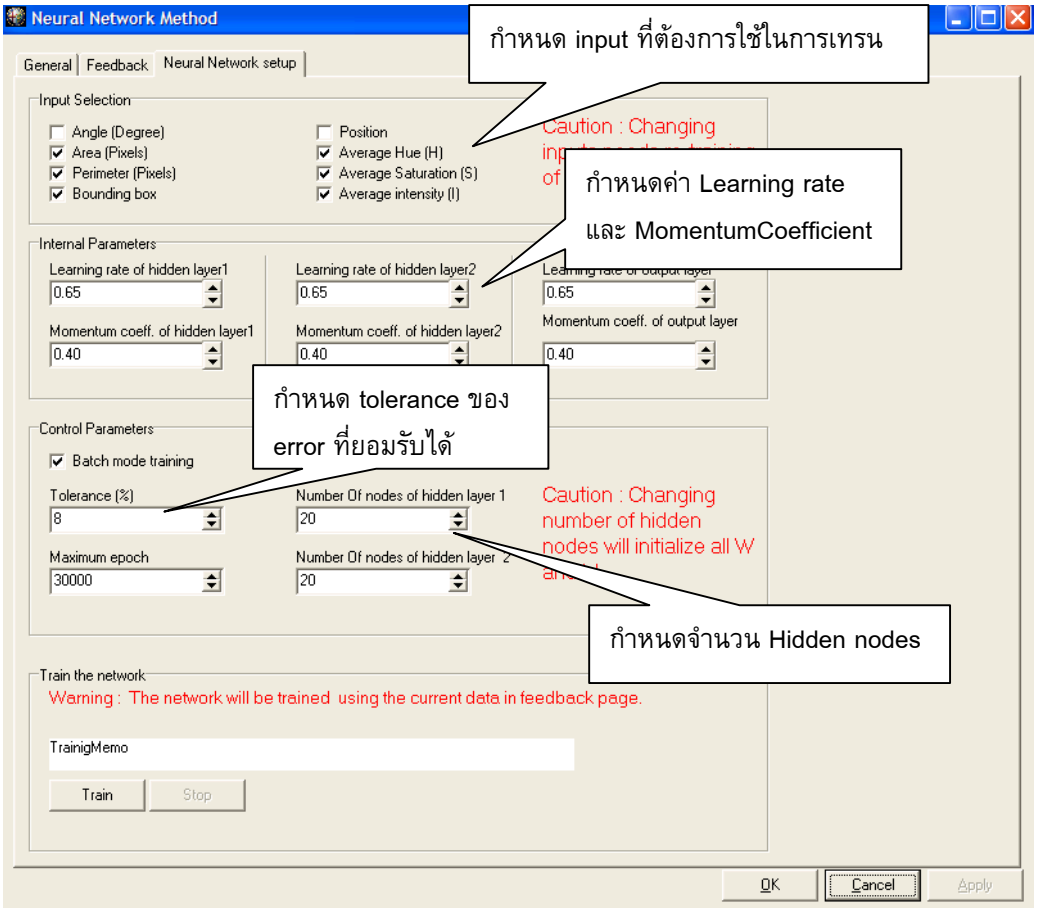
ภาพที่ ค-4 แสดงหน้าข้อมูลทั่วไปของเครื่องมือวัดที่ใช้โครงข่ายประสาทเทียม



ภาพที่ ค-5 แสดงหน้าป้อนข้อมูลกลับให้กับผู้ใช้ของเครื่องมือวัดที่โครงข่ายประสาทเทียม



ภาพที่ ค-6 แสดงหน้าแสดงข้อมูลของ BLOB ทั้งหมดในภาพ



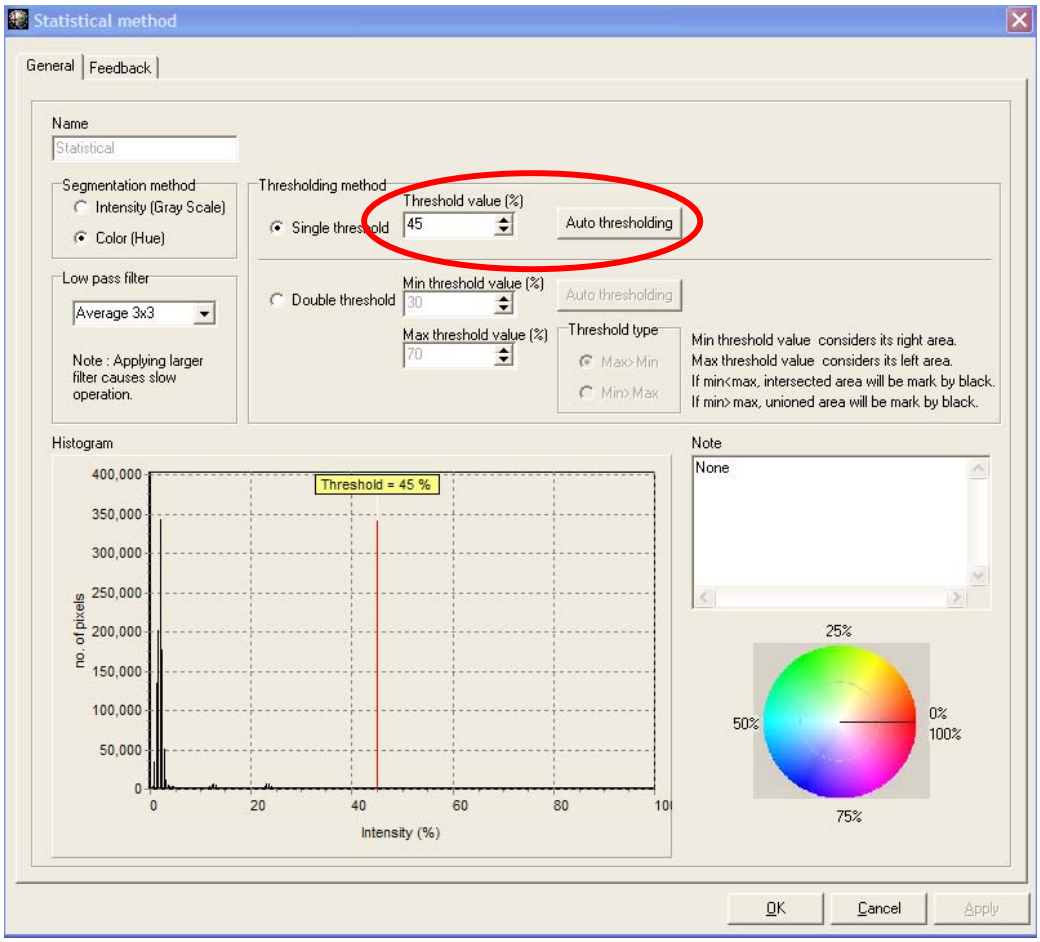
ภาพที่ ค-7 แสดงหน้าที่ใช้สำหรับกำหนดลักษณะการทำงานของโครงข่ายประสาทเทียม (Neural Network Setup)

2 แสดงการทำงานของโปรแกรมกับภาพตัวอย่าง

2.1 การทดลองของเครื่องมือวัดที่ทำงานโดยวิธีการทางสถิติกับภาพตัวอย่าง A ในภาพที่ ค-8

ภาพที่ ค-8 แสดงหน้าภาพตัวอย่าง A ที่มีพื้นผิวเสียด้านมุมขวา

1. ใช้ Auto Threshold ได้ที่ 45 % ดังรูปที่ ค-9

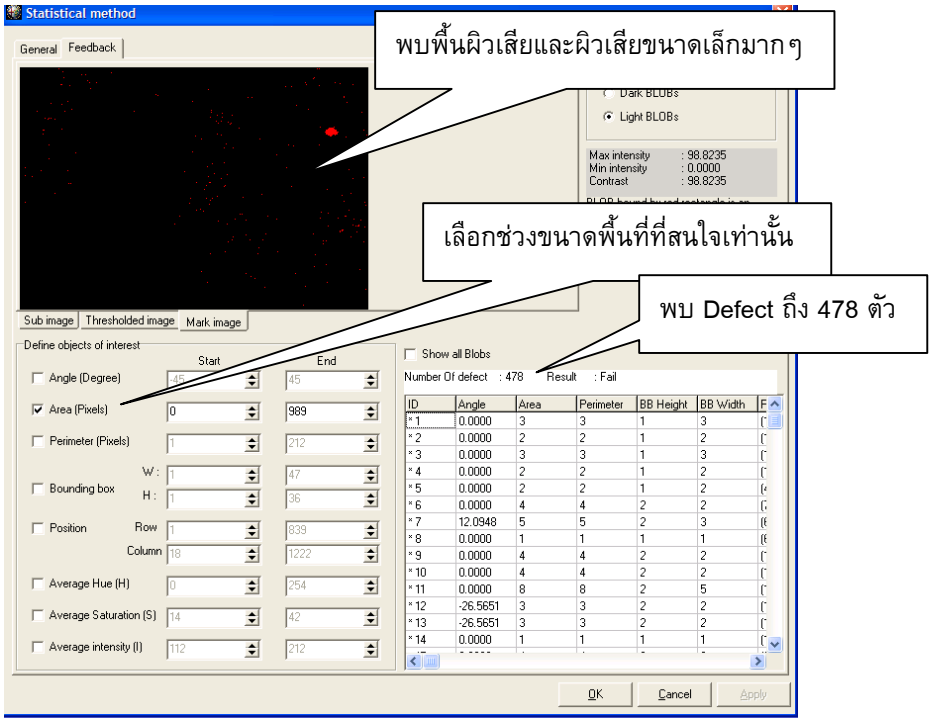


ภาพที่ ค-9 แสดงหน้าการตั้งค่าและ Auto Threshold

2. จะได้ภาพสองระดับ (Binary Image) ซึ่งแยกวัตถุที่เป็นผิวเสียออกจากพื้นหลังดังภาพที่ ค-10

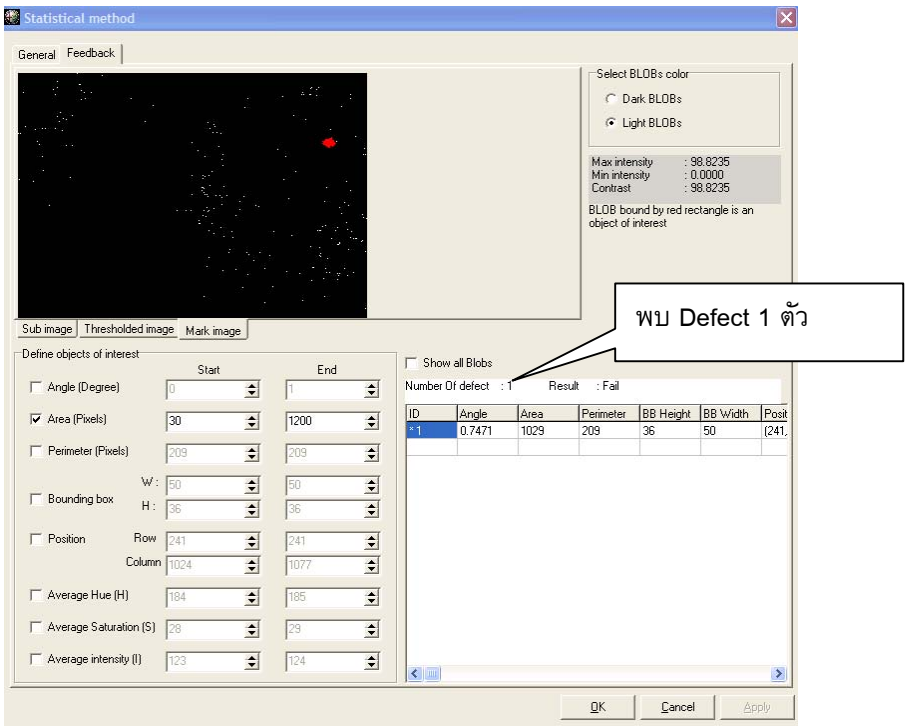
ภาพที่ ค-10 แสดงภาพสองระดับ (Binary Image) ที่ได้จาก Auto Threshold

3. เราจะเห็นว่ามีวัตถุขนาดเล็กมากถูกแยกออกมาด้วย ซึ่งไม่ใช่ขนาดที่เราสนใจ ดังนั้นเราจะทำการกำหนดช่วงคุณสมบัติที่เราสนใจ ในที่นี้คือขนาด (Area) ดังภาพที่ ค-11

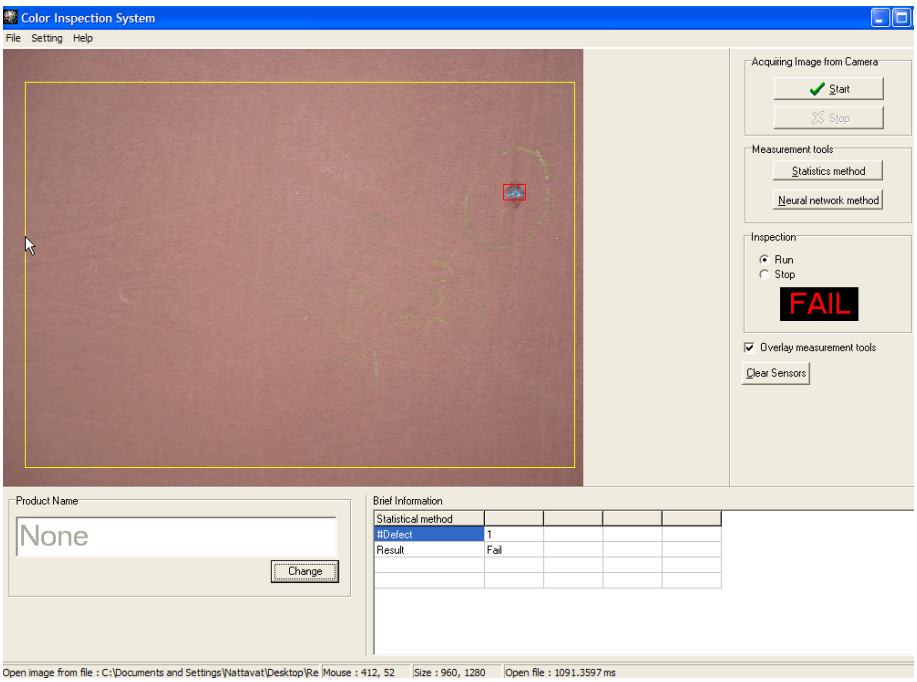


ภาพที่ ค-11 แสดงหน้าการตั้งค่าและผิวเสียทั้งหมดที่พบ

4. กำหนดขนาดพื้นผิวเสียที่สนใจซึ่งมีขนาดไม่ต่ำกว่า 30 พิกเซล เราจะได้ผิวเสียตัวที่เราสนใจ

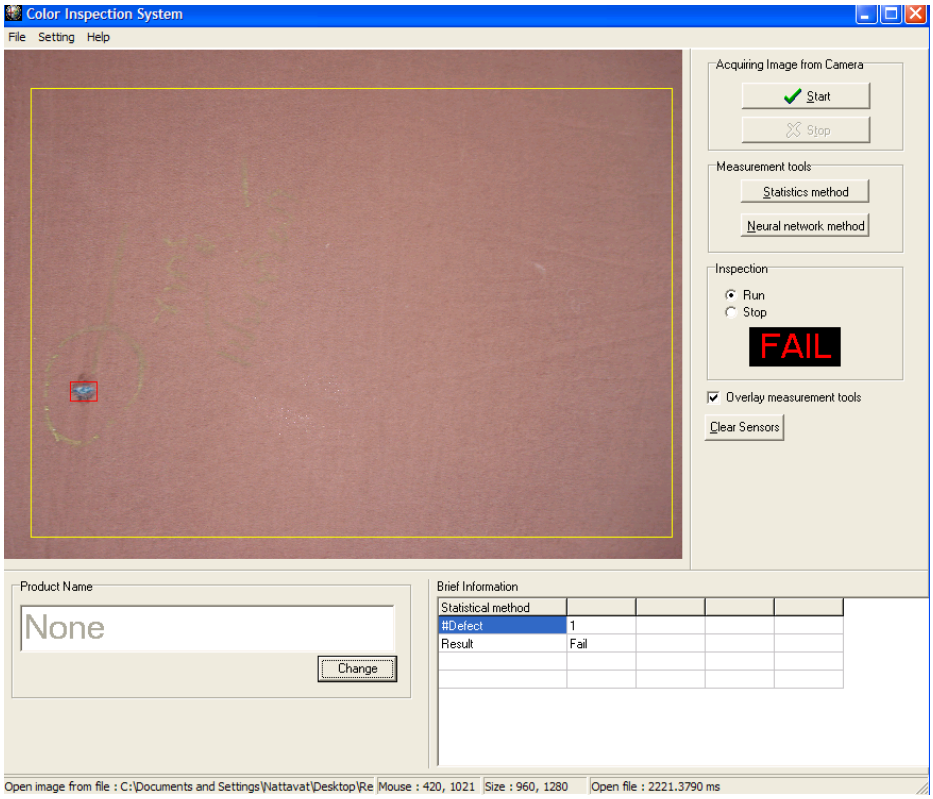


ภาพที่ ค-12 แสดงผลการตรวจสอบหลังจากกำหนดช่วงขนาดผิวเสียที่ต้องการ



ภาพที่ ค-13 แสดงผลการตรวจสอบที่หน้าต่างหลัก

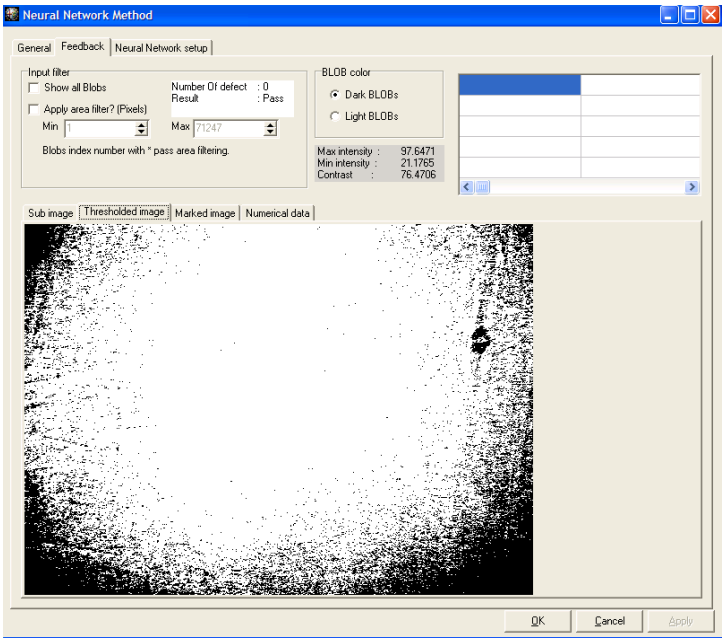
5. สามารถเซฟและนำไปใช้กลับภาพตัวอย่างอื่นที่มีลักษณะใกล้เคียงกันได้



ภาพที่ ค-14 แสดงผลการตรวจสอบที่นำไปใช้กับภาพที่มีลักษณะใกล้เคียงกัน

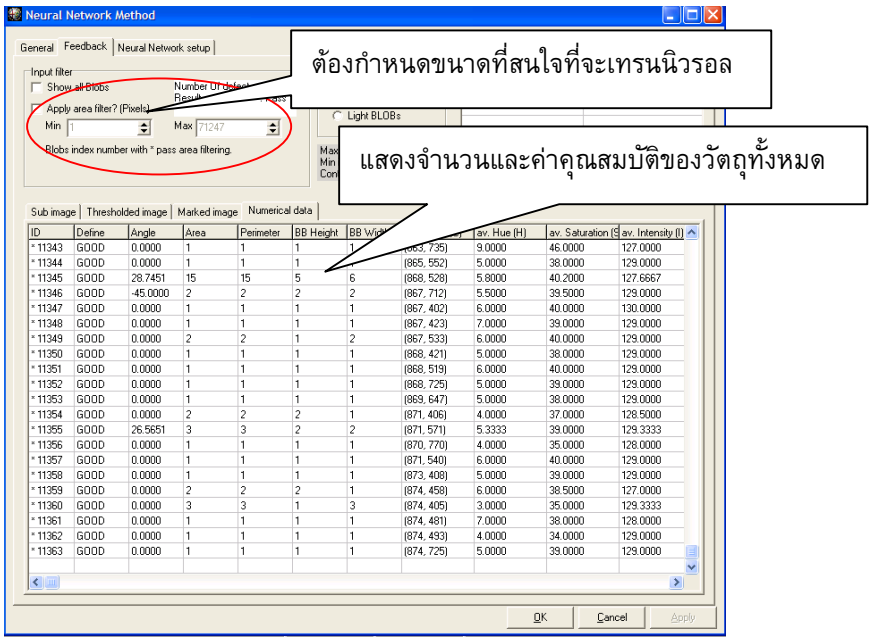
2.2 การทดลองของเครื่องมือวัดที่ทำงานโดยวิธีการทางโครงข่ายประสาทเทียม (Artificial Neural Network) กับภาพตัวอย่าง A ในภาพที่ ค-8

1. หลังจากนำภาพตัวอย่าง A เข้ามาและเลือกวิธีการที่จะใช้เป็นแบบโครงข่ายประสาทเทียม (Artificial Neural Network) เราจะได้ภาพขาวดำ (Binary Image) ที่ผ่านวิธีการแยกบริเวณ (Segmentation) แล้วดังภาพที่ ค-15



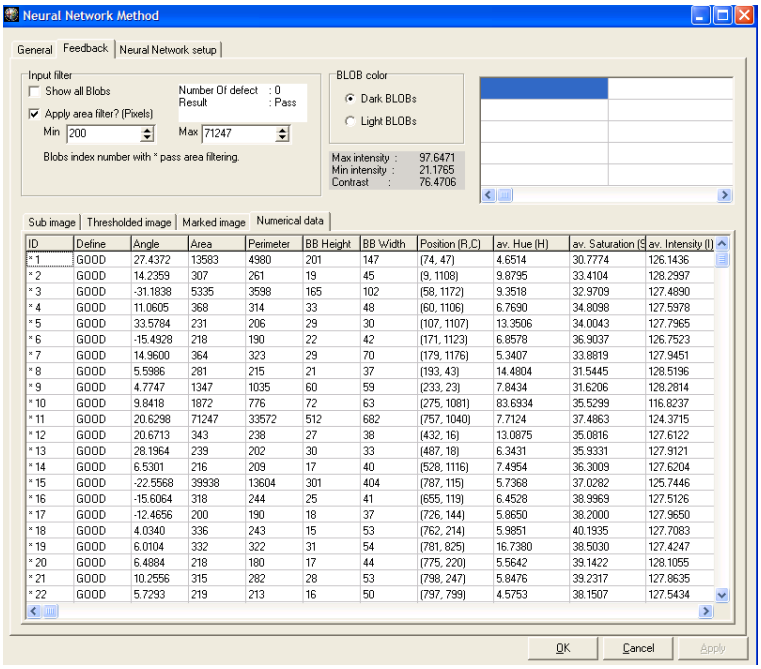
ภาพที่ ค-15 แสดงภาพสองระดับ (Binary Image) ที่ผ่านวิธีการแยกบริเวณ (Segmentation)

2. จำนวนวัตถุในภาพมีถึง11,363 ตัว ดังนั้นเราต้องทำการกำหนดช่วงขนาดที่สนใจก่อน เพราะเราไม่สามารถนำวัตถุทั้งหมดเข้าไปเทรนเนื่องจากจะใช้เวลาเทรนมากและเกินความจำเป็น



ภาพที่ ค-16 แสดงจำนวนและค่าคุณสมบัติของวัตถุทั้งหมดบนภาพ

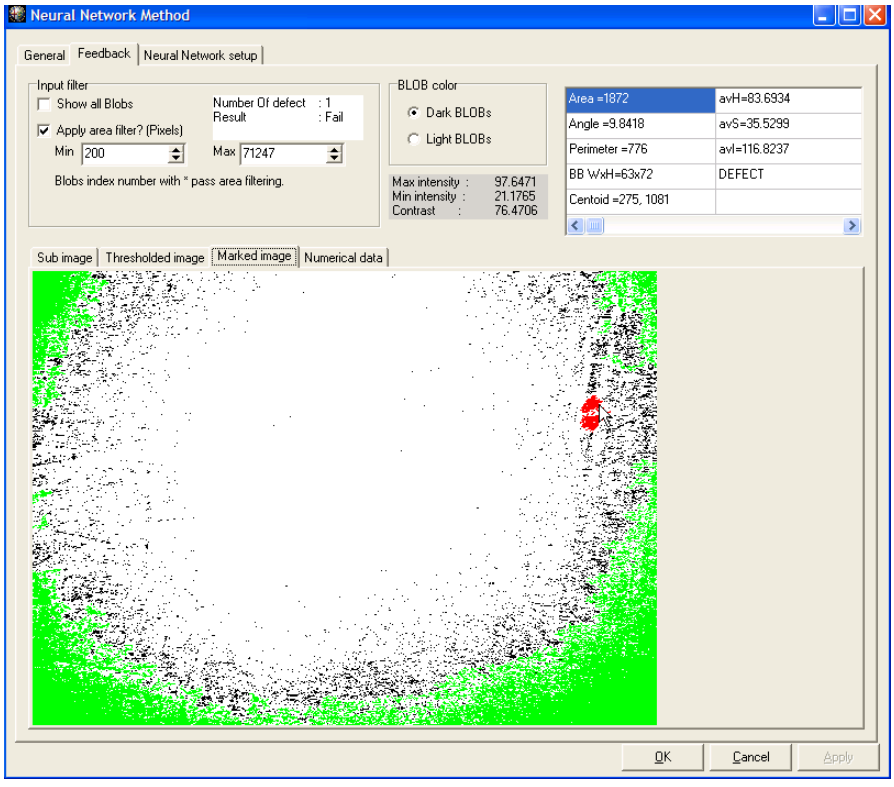
2. หลังจากทำการกำหนดช่วงขนาดที่สนใจ เราได้วัตถุ 27 ตัว



ID	Define	Angle	Area	Perimeter	BB Height	BB Width	Position (R,C)	av. Hue (H)	av. Saturation (S	av. Intensity (I)
* 1	GOOD	27.4372	13583	4980	201	147	(74, 47)	4.6514	30.7774	126.1436
* 2	GOOD	14.2359	307	261	19	45	(9, 1108)	9.8795	33.4104	128.2997
* 3	GOOD	-31.1838	5335	3598	165	102	(58, 1172)	9.3518	32.9709	127.4890
* 4	GOOD	11.0605	368	314	33	48	(60, 1106)	6.7690	34.8098	127.5978
* 5	GOOD	33.5784	231	206	29	30	(107, 1107)	13.3506	34.0043	127.7965
* 6	GOOD	-15.4928	218	190	22	42	(171, 1123)	6.8578	36.9037	126.7523
* 7	GOOD	14.9600	364	323	29	70	(179, 1176)	5.3407	33.8819	127.9451
* 8	GOOD	5.5986	281	215	21	37	(193, 43)	14.4804	31.5445	128.5196
* 9	GOOD	4.7747	1347	1035	60	59	(233, 23)	7.8434	31.6206	128.2814
* 10	GOOD	9.8418	1872	776	72	63	(275, 1081)	83.6934	35.5299	116.8237
* 11	GOOD	20.6298	71247	33572	512	682	(757, 1040)	7.7124	37.4863	124.3715
* 12	GOOD	20.6713	343	238	27	38	(432, 16)	13.0875	35.0816	127.6122
* 13	GOOD	28.1964	239	202	30	33	(487, 18)	6.3431	35.9331	127.9121
* 14	GOOD	6.5301	216	209	17	40	(528, 1116)	7.4954	36.3009	127.6204
* 15	GOOD	-22.5568	39938	13604	301	404	(787, 115)	5.7368	37.0282	125.7446
* 16	GOOD	-15.6064	318	244	25	41	(655, 119)	6.4528	38.9969	127.5126
* 17	GOOD	-12.4656	200	190	18	37	(726, 144)	5.8650	38.2000	127.9650
* 18	GOOD	4.0340	336	243	15	53	(762, 214)	5.9851	40.1935	127.7083
* 19	GOOD	6.0104	332	322	31	54	(781, 825)	16.7380	38.5030	127.4247
* 20	GOOD	6.4884	218	180	17	44	(775, 220)	5.5642	39.1422	128.1055
* 21	GOOD	10.2556	315	282	28	53	(798, 247)	5.8476	39.2317	127.8635
* 22	GOOD	5.7293	219	213	16	50	(797, 799)	4.5753	38.1507	127.5434

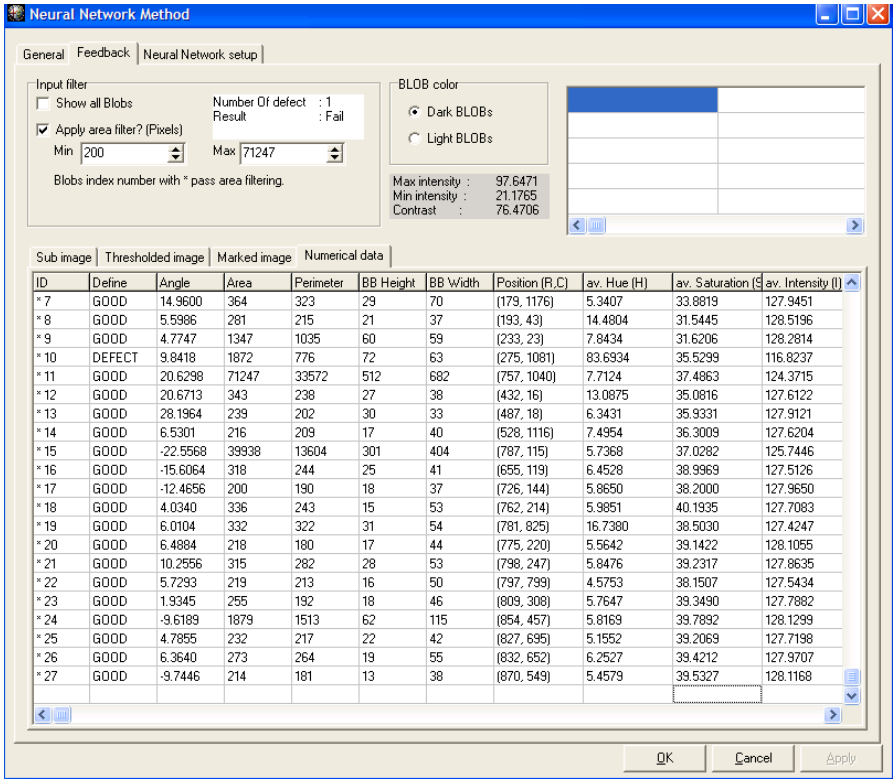
ภาพที่ ค-17 แสดงจำนวนและค่าคุณสมบัติของวัตถุหลังกำหนดช่วงขนาดที่สนใจ

3. เลือกไปที่หน้าสำหรับเลือกวัตถุที่เป็นพื้นผิวเสีย และคลิกที่วัตถุที่จะพิจารณาว่าเป็นผิวเสีย ซึ่งจะปรากฏเป็นสีแดง ส่วนวัตถุที่เป็นสีเขียวนั้นจะพิจารณาว่าเป็นพื้นผิวดี ดังภาพ ค-18



ภาพที่ ค-18 แสดงที่หน้าสำหรับเลือกวัตถุที่เป็นพื้นผิวเสีย

4. เราจะได้วัตถุสำหรับเทรนทั้งหมด 27 ตัว รวมทั้งตัวที่เรากำหนดให้เป็นพื้นผิวเสียด้วย ดังภาพที่ ค-19



ID	Define	Angle	Area	Perimeter	BB Height	BB Width	Position (R,C)	av. Hue (H)	av. Saturation (S	av. Intensity (I)
* 7	GOOD	14.9600	364	323	29	70	(179, 1176)	5.3407	33.8819	127.9451
* 8	GOOD	5.5986	281	215	21	37	(193, 43)	14.4804	31.5445	128.5196
* 9	GOOD	4.7747	1347	1035	60	59	(233, 23)	7.8434	31.6206	128.2814
* 10	DEFECT	9.8418	1872	776	72	63	(275, 1081)	83.6934	35.5299	116.8237
* 11	GOOD	20.6298	71247	33572	512	682	(757, 1040)	7.7124	37.4863	124.3715
* 12	GOOD	20.6713	343	238	27	38	(432, 16)	13.0875	35.0816	127.6122
* 13	GOOD	28.1964	239	202	30	33	(487, 18)	6.3431	35.9331	127.9121
* 14	GOOD	6.5301	216	209	17	40	(528, 1116)	7.4954	36.3009	127.6204
* 15	GOOD	-22.5568	39938	13604	301	404	(787, 115)	5.7368	37.0282	125.7446
* 16	GOOD	-15.6064	318	244	25	41	(655, 119)	6.4528	38.9969	127.5126
* 17	GOOD	-12.4656	200	190	18	37	(726, 144)	5.8650	38.2000	127.9650
* 18	GOOD	4.0340	336	243	15	53	(762, 214)	5.9851	40.1935	127.7083
* 19	GOOD	6.0104	332	322	31	54	(781, 825)	16.7380	38.5030	127.4247
* 20	GOOD	6.4884	218	180	17	44	(775, 220)	5.5642	39.1422	128.1055
* 21	GOOD	10.2556	315	282	28	53	(798, 247)	5.8476	39.2317	127.8635
* 22	GOOD	5.7293	219	213	16	50	(797, 799)	4.5753	38.1507	127.5434
* 23	GOOD	1.9345	255	192	18	46	(809, 308)	5.7647	39.3490	127.7882
* 24	GOOD	-9.6189	1879	1513	62	115	(854, 457)	5.8169	39.7892	128.1299
* 25	GOOD	4.7855	232	217	22	42	(827, 695)	5.1552	39.2069	127.7198
* 26	GOOD	6.3640	273	264	19	55	(832, 652)	6.2527	39.4212	127.9707
* 27	GOOD	-9.7446	214	181	13	38	(870, 549)	5.4579	39.5327	128.1168

ภาพที่ ค-19 แสดงจำนวนและค่าคุณสมบัติของวัตถุที่จะทำการเทรนทั้ง 27 ตัว

5. กำหนดพารามิเตอร์ที่ใช้สำหรับการเทรนดังนี้

ข้อมูลเข้า 6 ค่า คือ ค่าเฉดสี (Hue)

ค่าความเข้มแสง (Intensity)

ค่าความอิ่มตัวของสี (Saturation)

ค่าพื้นที่ (Area)

ค่าจุดศูนย์ถ่วง (Centroid)

ค่า Bounding Box

ค่าเส้นรอบรูป (Perimeter)

ค่าอัตราการเรียนรู้ (Learning Rate) ในปมซ่อนของชั้นที่หนึ่งคือ 0.65

ค่าอัตราการเรียนรู้ (Learning Rate) ในปมซ่อนของชั้นที่สองคือ 0.65

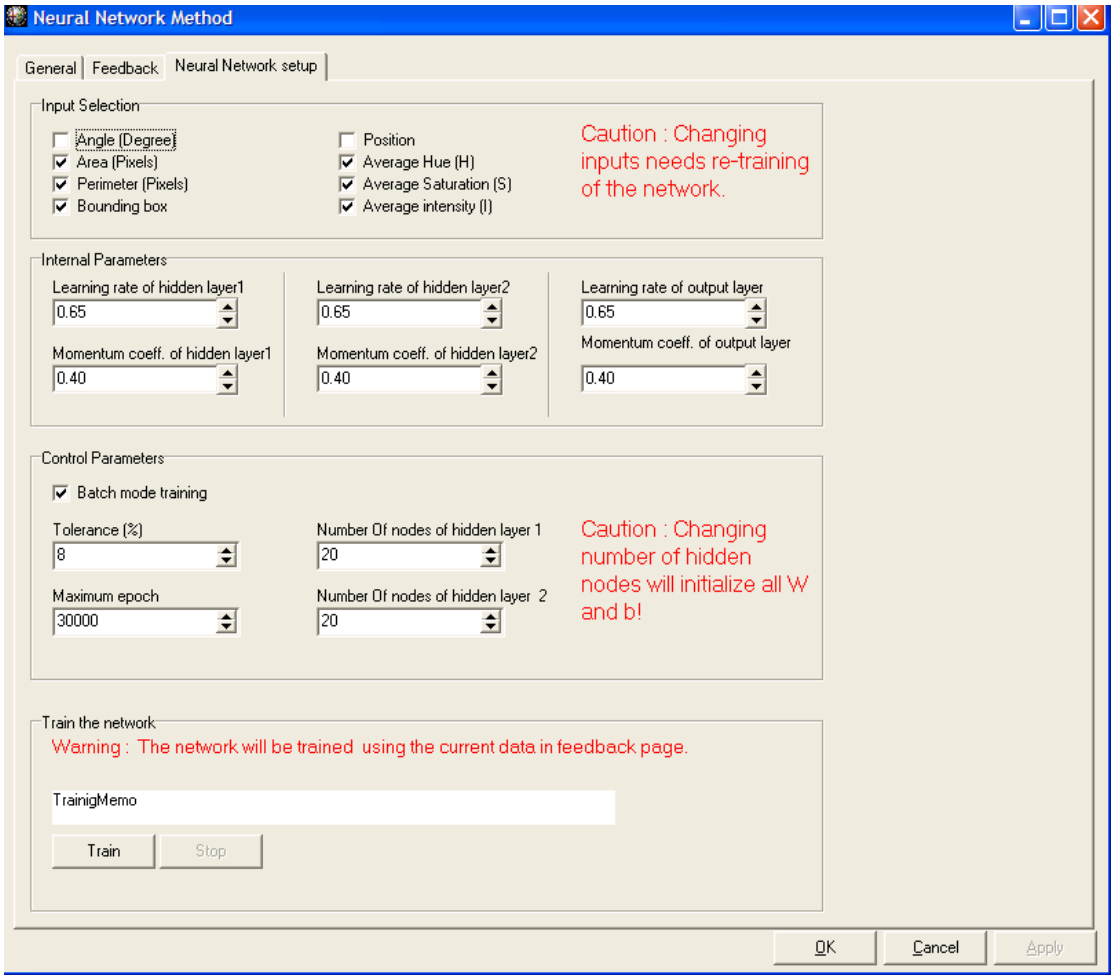
ค่าอัตราการเรียนรู้ (Learning Rate) ในปมซ่อนของชั้นที่สามคือ 0.65

ค่าสัมปะสิทธิ์โมเมนตัม (Momentum Coefficient) ในปมซ่อนของชั้นที่หนึ่งคือ 0.4

ค่าสัมปะสิทธิ์โมเมนตัม (Momentum Coefficient) ในปมซ่อนของชั้นที่สองคือ 0.4

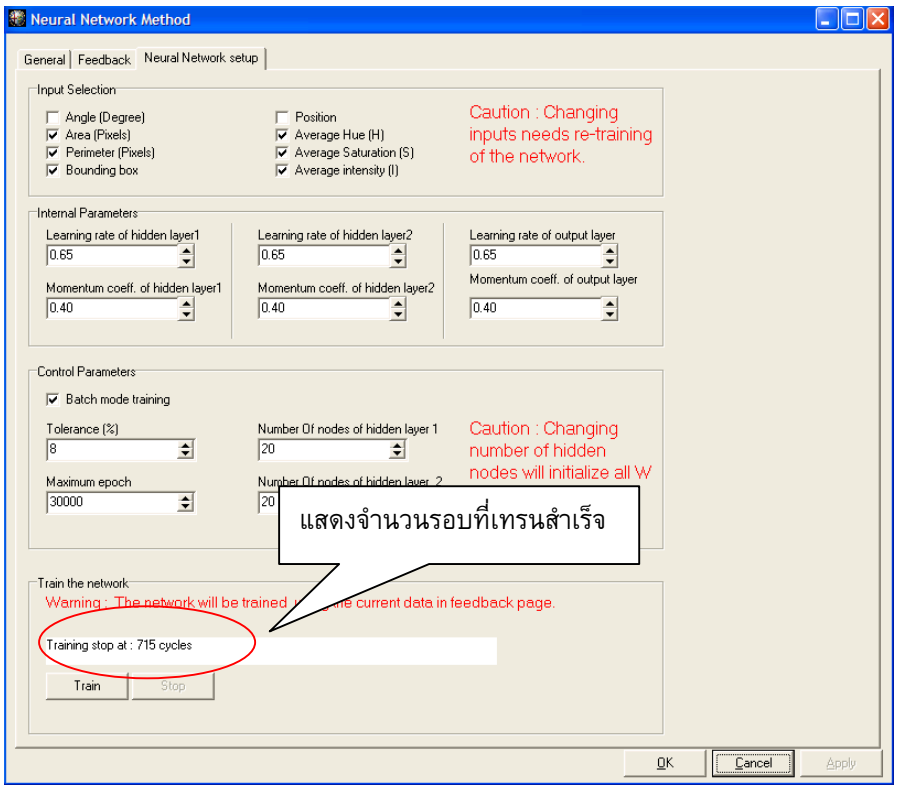
ค่าสัมปะสิทธิ์โมเมนตัม (Momentum Coefficient) ในปมซ่อนของชั้นที่สามคือ 0.4

จำนวนปมซ่อน (Hidden node) ในชั้นที่หนึ่ง คือ 20
จำนวนปมซ่อน (Hidden node) ในชั้นที่สอง คือ 20
ค่าถ่วงน้ำหนักเชื่อมโยง (Connection weight) เริ่มต้น คือ สุ่มระหว่าง –1.5 ถึง 1.5
ค่าขีดเริ่มเปลี่ยน (Bias) เริ่มต้น คือ สุ่มระหว่าง –1.5 ถึง 1.5
โหมดการ Train คือ Batch mode
ช่วงยอมรับ (Tolerance) หรือ ผลต่างความผิดพลาด คือ 8 %
จำนวนรอบสูงสุดในการ Train (Maximum Epoch) คือ 30000

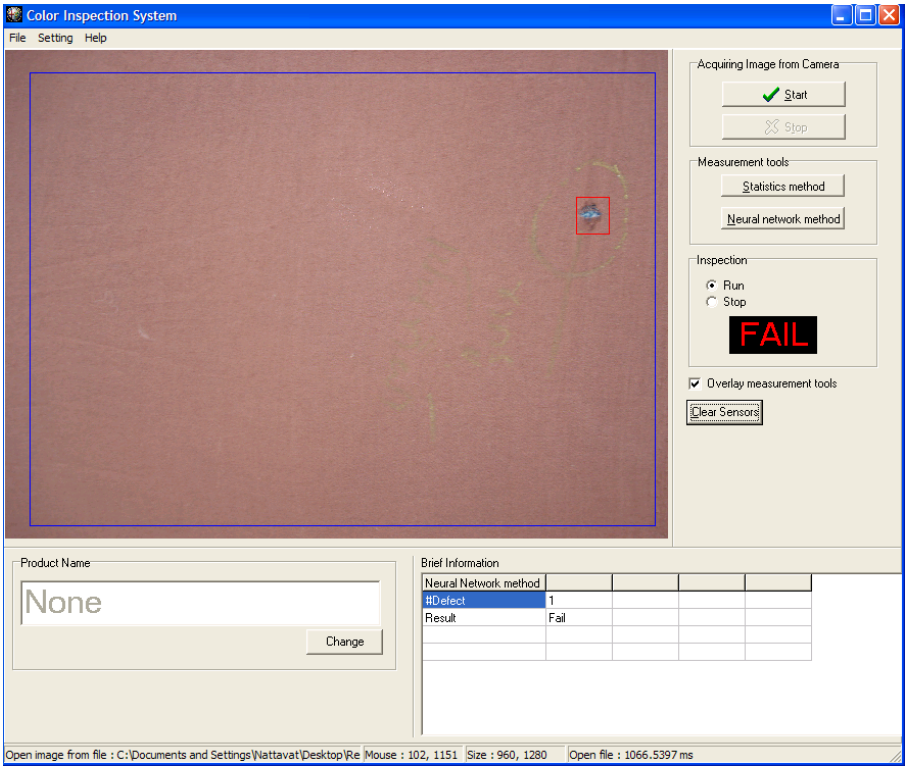


ภาพที่ ค-20 แสดงหน้าที่ใช้สำหรับกำหนดลักษณะการทำงานของโครงข่ายประสาทเทียม (Neural Network Setup)

6. ทำการเทรนโดยคลิกที่ปุ่มเทรน และสามารถเทรนสำเร็จใน 715 รอบ ดังรูปที่ ค-21

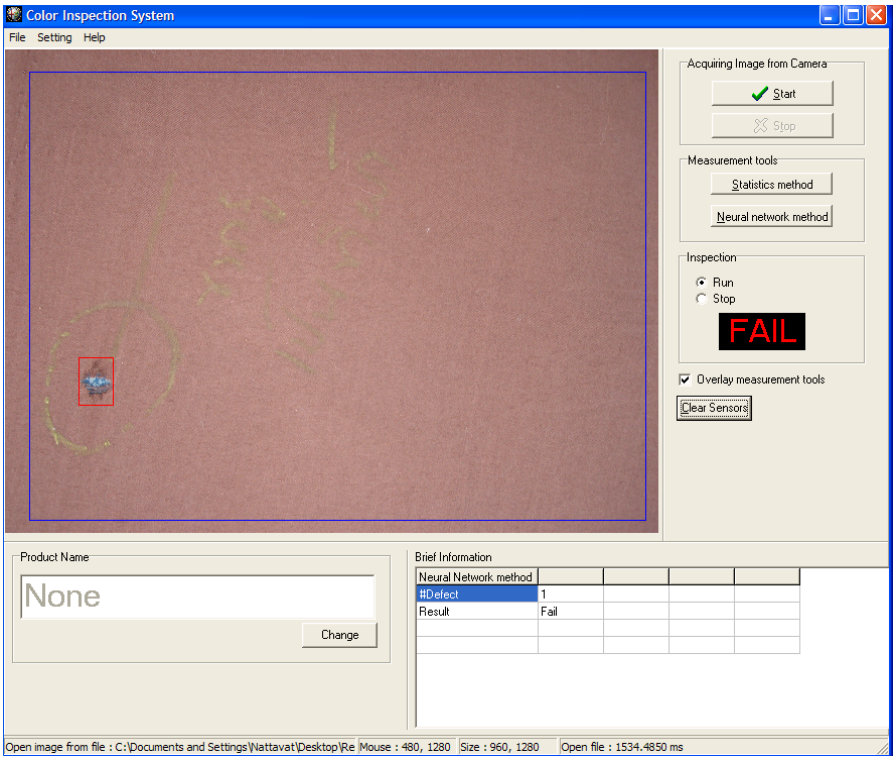


ภาพที่ ค-21 แสดงหน้าที่ใช้สำหรับกำหนดลักษณะการทำงานของโครงข่ายประสาทเทียม

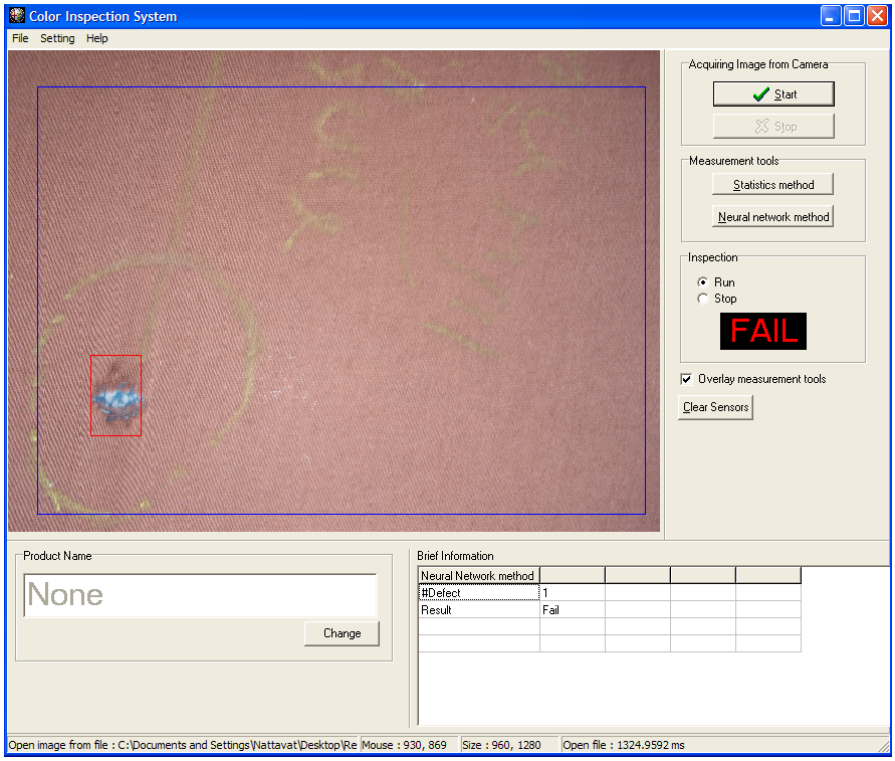


ภาพที่ ค-22 แสดงผลการตรวจสอบที่หน้าต่างหลัก

7. นำไปทดสอบกับภาพที่มีลักษณะพื้นผิวเสียใกล้เคียงกัน จะได้ผลดังภาพที่ ค-23 และ ค-24



ภาพที่ ค-23 แสดงผลการตรวจสอบที่นำไปใช้กับภาพที่มีลักษณะใกล้เคียงกัน



ภาพที่ ค-24 แสดงผลการตรวจสอบที่นำไปใช้กับภาพที่มีลักษณะใกล้เคียงกัน

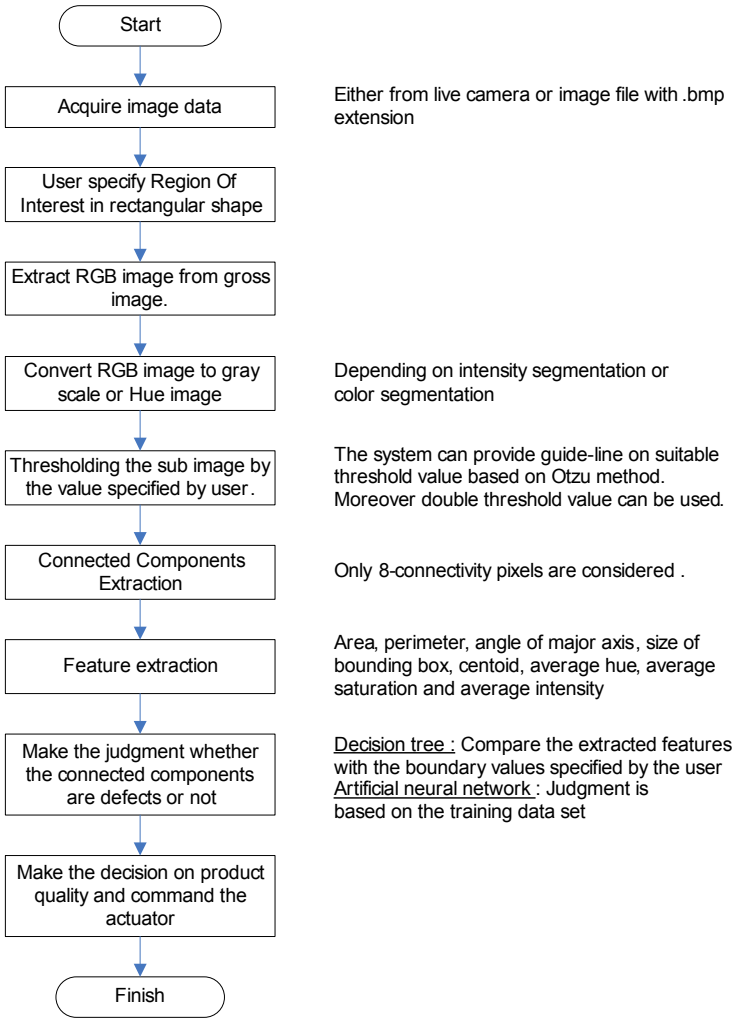
ภาคผนวก ง

คู่มือการใช้งานโปรแกรม (User Manual)

ภาคผนวก ง
คู่มือการใช้งานโปรแกรม (User Manual)

โปรแกรม Color Inspection System นั้นได้ถูกออกแบบมาเพื่อใช้ในการตรวจสอบผ้าสีพื้นแบบเวลาจริง โดยที่ผู้ใช้สามารถเลือกเครื่องมือในการตรวจสอบผืนผ้าได้ ทั้งแบบที่ทำงานแบบสถิติที่ผู้ใช้สามารถกำหนดค่าคุณสมบัติต่างๆ ของ BLOB ที่จัดเป็นผืนผิวเสียของตัวงานได้เอง หรือแบบที่ใช้โครงข่ายประสาทเทียม ซึ่งสามารถใช้งานได้อย่างง่ายดายกว่า โดยทำการระบุแค่ว่าชิ้น BLOB ใดเป็นพื้นผิวเสียหรือดีเท่านั้น สำหรับโปรแกรมที่สร้างขึ้นนั้นมีลักษณะดังนี้

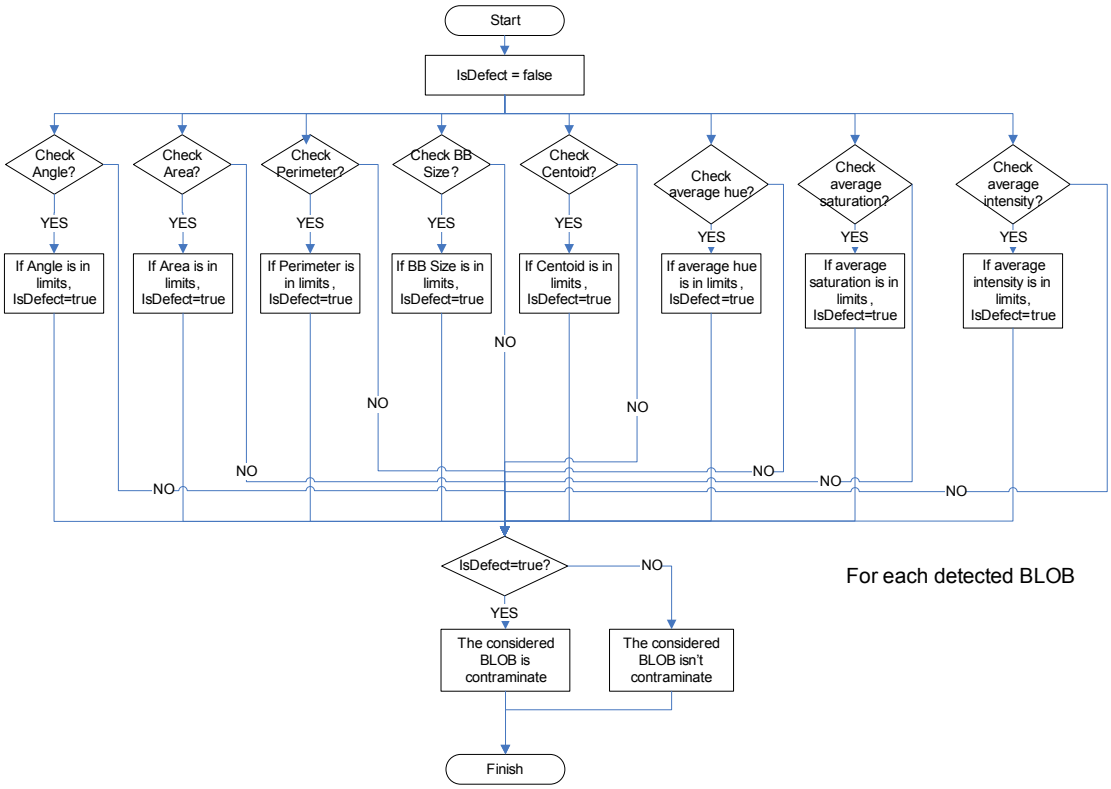
- สามารถต่อเชื่อมได้กับกล้อง USB ใดๆ ก็ได้ที่รองรับ Video For Window (VFW) ซึ่งเป็น Driver มาตรฐานที่มีอยู่ในระบบปฏิบัติการ Windows
- ผู้ใช้สามารถทำการตรวจสอบได้ทั้งภาพที่อยู่ในไฟล์ภาพและข้อมูลภาพที่รับมาจากกล้อง
- สามารถโหลดและบันทึกภาพจากไฟล์ภาพที่มีจุดนามสกุล .bmp
- ผู้ใช้สามารถทำการกำหนดลักษณะการทำงานของกล้องได้ดังนี้
    a. ความละเอียดของภาพ
    b. การตั้งเวลาเปิดปิดหน้ากล้อง (Exposure time)
    c. การลดสัญญาณรบกวนจากระบบไฟบ้าน
    d. การเพิ่มความสว่างให้กับภาพ
    e. การเพิ่มความเข้มสีให้กับภาพ
- ข้อมูลการวัดและรายละเอียดต่างๆ สามารถบันทึกและเปิดอ่านได้จากไฟล์ตัวอักษรซึ่งสามารถเข้าใจได้โดยทั่วไป
- มีเครื่องมือวัดที่ใช้สำหรับตรวจสอบผืนผ้าได้ 2 วิธี คือวิธีการทางสถิติและโดยโครงข่ายประสาทเทียม ซึ่งเครื่องวัดทั้งสองแบบนี้จะมีขั้นตอนการทำงานที่คล้ายคลึงกันทุกประการ แต่จะแตกต่างกันในวิธีที่ใช้ในการตัดสินว่า BLOB ที่ตรวจจับได้มีตัวใดบ้างที่จัดเป็นผืนผิวงานเสีย ซึ่งขั้นตอนการทำงานของโปรแกรมเป็นดังภาพที่ ง-1



ภาพที่ ง-1 แสดงขั้นตอนการทำงานของโปรแกรม Color Inspection System

ในตอนเริ่มแรกนั้น โปรแกรมจะทำการดึงข้อมูลภาพสีไม่ว่าจะมาจากกล้องหรือจากไฟล์ภาพที่เก็บไว้ในคอมพิวเตอร์ และแสดงภาพขึ้นมา เพื่อให้ผู้ใช้ทำการระบุบริเวณที่สนใจ (Region Of Interest) ทั้งนี้เนื่องจาก บริเวณของผลิตภัณฑ์อาจจะไม่คลอบคลุมทั้งภาพ หลังจากนั้น โปรแกรมจะทำการแยกข้อมูลภาพเฉพาะบริเวณที่ระบุโดยผู้ใช้ว่า เป็นบริเวณที่จะต้องทำการตรวจสอบ พร้อมทั้งแปลงให้เป็นภาพ Gray Scale หรือ Hue Image ขึ้นอยู่กับว่า ผู้ใช้ต้องการแยกบริเวณ (Segmentation) โดยความเข้มแสงหรือสี ตามลำดับ และทำการแปลงให้เป็นภาพที่มีสองระดับ (Binary Image) โดยใช้ค่า Threshold ที่ระบุโดยผู้ใช้ ซึ่งโปรแกรมสามารถช่วยระบุค่า Threshold ที่เหมาะสมให้กับผู้ใช้ เพื่อให้ผู้ใช้ช่วยประกอบการตัดสินใจได้ด้วย สำหรับค่า Threshold ที่ระบุโดยโปรแกรมนั้น ได้มาจากการเลือกค่า Threshold โดยวิธีของ Otsu ซึ่งเป็นวิธีการเลือกค่า Threshold แบบอัตโนมัติที่จัดว่า มีการนำไปใช้อย่างแพร่หลายมากที่สุด หลังจากที่ได้ภาพที่มีเพียงสองระดับแล้ว โปรแกรมจะทำกระบวนการ

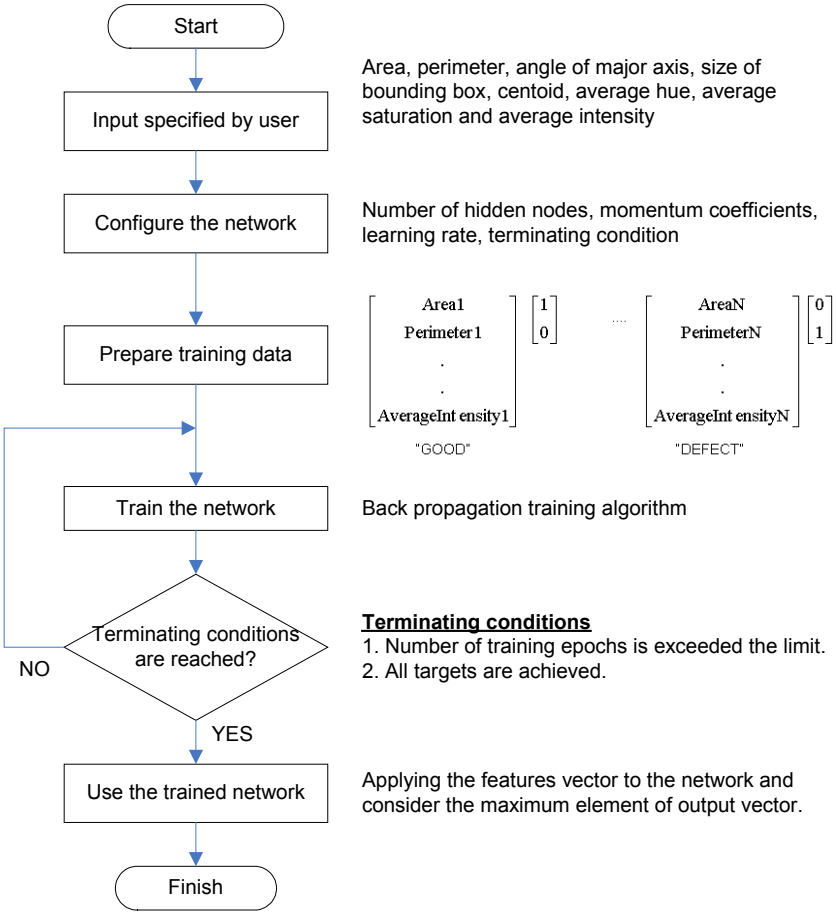
Connected Components Extraction หรือ Connected Components Labeling เพื่อทำการตรวจจับ BLOB ที่ปรากฏขึ้นในภาพ ซึ่งโปรแกรมจะทำการพิจารณาพิกเซลที่มีการเชื่อมต่อแบบ 8 ทิศทางเท่านั้น นอกจากนั้น ผู้ใช้ยังระบุได้ด้วยว่า ให้โปรแกรมทำการตรวจจับ BLOB ที่มีสีดำหรือสีขาว ซึ่งจะขึ้นอยู่กับการใช้งานแต่ละงาน และจากขั้นตอนนี้จะทำให้โปรแกรมทราบว่า มี BLOB จำนวนเท่าใดอยู่ในภาพ นอกจากนั้น จะทำให้โปรแกรมทราบว่า BLOB ที่ตรวจจับได้นั้นครอบคลุมพิกัดของพิกเซลใดบ้าง หลังจากนั้น โปรแกรมจะทำการตรวจวัดคุณสมบัติต่างๆของ BLOB ที่ตรวจจับได้ ซึ่งได้แก่ มุมของแกนหลักของรอยด่างแต่ละชิ้น พื้นที่ เส้นรอบรูป ขนาดของ Bounding box ตำแหน่งของจุดศูนย์กลางของรอยด่าง นอกจากนั้น โปรแกรมยังทำการวัดคุณสมบัติของสีของรอยด่างแต่ละชิ้นด้วย ซึ่งคุณสมบัติของสีที่ทำการตรวจวัด คือ เฉดสีเฉลี่ย ค่า saturation เฉลี่ย ค่าสว่างเฉลี่ยของรอยด่างแต่ละชิ้น จากคุณสมบัติต่างๆของ BLOB แต่ละชิ้น หากผู้ใช้เลือกการตรวจจับโดยวิธีทางสถิตินั้น คุณสมบัติของ BLOB แต่ละชิ้นจะถูกนำไปเปรียบเทียบกับลักษณะของพื้นผิวงานเสีย ซึ่งถูกกำหนดโดยผู้ใช้ และหากมีชิ้นใดที่มีคุณสมบัติตรงกับคุณสมบัติของพื้นผิวงานเสียแม้เพียงชิ้นเดียว ผืนผ้าบริเวณนั้นจะถูกตัดสินว่าเป็นงานเสียทันที สำหรับการเปรียบเทียบดังกล่าว สามารถอธิบายการทำงานได้ดังในรูปต่อไปนี้



ภาพที่ ง-2 แสดงการทำงานของวิธีทางสถิติ

หากผู้ใช้เลือกใช้วิธีตรวจสอบโดยใช้โครงข่ายประสาทเทียม (Artificial Neural Network) นอกจากวิธีการตัดสินว่า BLOB ที่ตรวจจับได้นั้น ชิ้นใดจัดเป็นพื้นผิวงานเสีย ที่นำโครงข่ายประสาทเทียมมาใช้ตัดสินแล้ว ฟังค์ชั่นการทำงานที่เหลือ ต่างก็คล้ายคลึงกับการตรวจสอบโดยวิธีทางสถิติทุกประการ

สำหรับวิธีการตัดสินใจโดยโครงข่ายประสาทเทียม (Artificial Neural Network) นั้น จะทำโดยโครงข่ายที่ถูก Train โดยการใช้กลุ่มตัวอย่าง BLOB ที่จัดเป็นพื้นผิวดีหรือเสีย ซึ่งเป็นข้อมูลที่ผู้ใช้สามารถกำหนดให้ โดยการกดเม้าส์ลงไปในรูปภาพที่เครื่องมือวัดจับได้ เพื่อเป็นการกำหนดว่า BLOB ชิ้นนั้นๆเป็นพื้นผิวดีหรือเสีย ซึ่งผู้ใช้สามารถเปลี่ยนกลับไปกลับมาได้ ซึ่งหลังจากนั้น โปรแกรมจะทำการจัดเตรียมข้อมูลที่ใช้ในการ train โครงข่ายประสาทเทียม และ Train โครงข่ายอย่างอัตโนมัติ ซึ่งหลักการ train โครงข่ายประสาทเทียมเพื่อนำมาใช้เป็นตัวจำแนกว่า BLOB ใดเป็นพื้นผิวงานดีหรือเสียนั้นแสดงไว้ในภาพที่ ง-3 สำหรับระเบียบวิธีที่ใช้ train โครงข่ายประสาทเทียมนั้นใช้วิธีที่มีใช้งานอยู่โดยทั่วไป คือ ระเบียบวิธี Back Propagation Training Algorithm

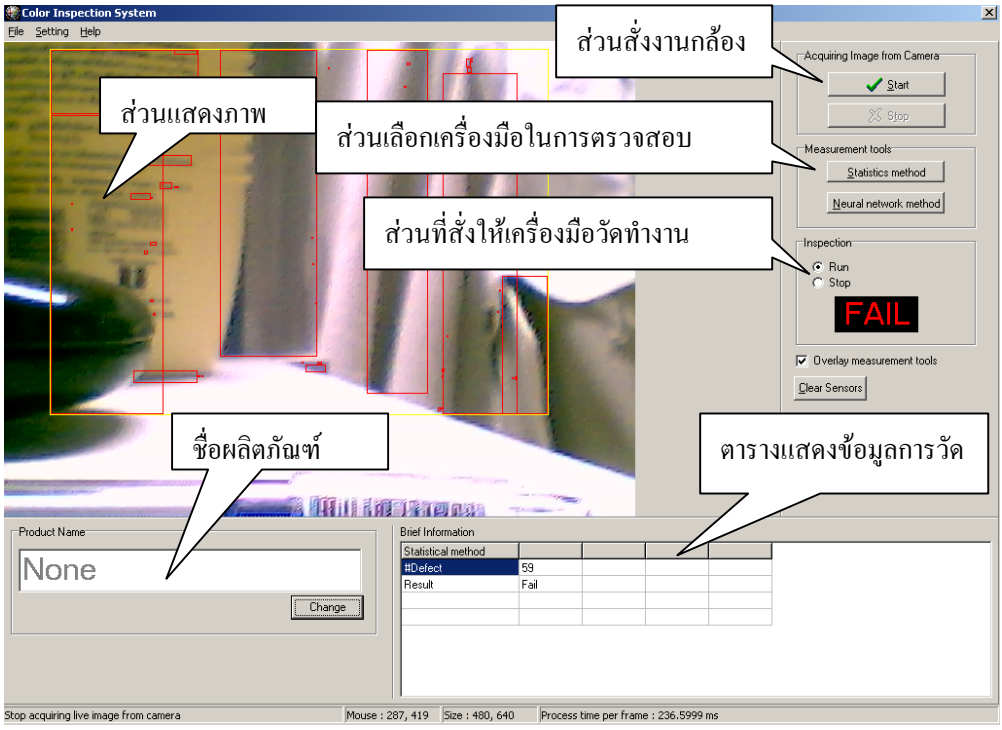


ภาพที่ ง-3 แสดงหลักการทำงานของโปรแกรมเพื่อ Train โครงข่ายประสาทเทียมก่อนนำมาใช้เป็นตัวจำแนก BLOB ว่าเป็นพื้นผิวงานดีหรือเสีย

ในตอนเริ่มแรก โปรแกรมจะทำการตั้งค่าเริ่มต้นต่างๆ ให้กับโครงข่ายประสาทเทียมเสียก่อน หลังจากนั้น ข้อมูลที่ระบุว่า BLOB ตัวใดเป็นพื้นผิวดีหรือเสีย ซึ่งผู้ใช้สามารถทำการคลิ๊กลงไปที่ภาพที่อยู่บนฟอร์มที่ใช้ติดต่อกับผู้ใช้ หลังจากนั้นคุณสมบัติ (Feature) ของ BLOB แต่ละชิ้นจะถูกนำมาทำเป็น input vector ซึ่งเป็นการนำเอาคุณสมบัติต่างๆ ของ BLOB นั้นๆ ที่ผู้ใช้กำหนดมาจัดวางในช่องของเวคเตอร์นั่นเอง ดังนั้นขนาดของ input vector ดังกล่าวจะขึ้นกับจำนวน Feature ที่ผู้ใช้กำหนด นอกจากนั้น โปรแกรมจะทำการสร้าง target vector ตามที่ผู้ใช้กำหนด โดย target vector ในกรณีนี้จะเป็นเวคเตอร์ 2 มิติซึ่งถ้าเป็น target vector ของ BLOB ที่เป็นพื้นผิวดีจะมีแถวแรกเป็น 1 ในขณะที่แถวที่ 2 เป็นศูนย์ และในทางกลับกันหากเป็น target vector ของ BLOB ที่เป็นพื้นผิวเสีย จะมีแถวที่ 1 ของเวคเตอร์เป็นศูนย์และแถวที่ 2 เป็นหนึ่ง ซึ่ง input vector และ target vector ของ BLOB แต่ละชิ้นที่มีอยู่ในภาพตัวอย่างนั้น จะถูกนำไปใช้ train โครงข่ายประสาทเทียม ซึ่งหลังจากที่โครงข่ายถูก train แล้ว ก็จะสามารถนำไปใช้เพื่อจำแนกว่า BLOB ชิ้นใดเป็นพื้นผิวเสียหรือดีได้อย่างอัตโนมัติ

1. หน้าต่างหลักที่ติดต่อกับผู้ใช้

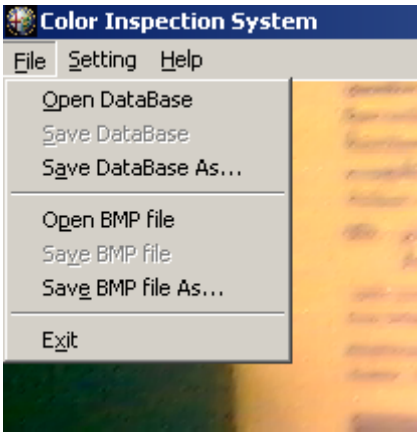
หน้าต่างในภาพที่ ง-4 นั้นเป็นหน้าต่างหลักที่ผู้ใช้สามารถทำการกำหนดเครื่องมือวัดที่จะใช้ในการตรวจสอบผืนผ้า รวมทั้งดูผลการตรวจสอบที่ได้แบบเวลาจริง ซึ่งหน้าต่างนี้ประกอบด้วยส่วนต่างๆ ดังนี้



ภาพที่ ง-4 แสดงหน้าต่างหลักของโปรแกรม Color Inspection System

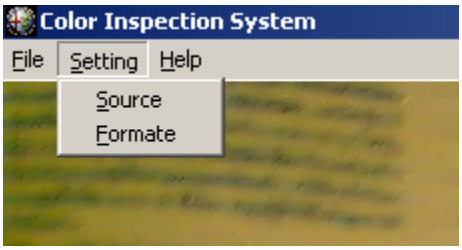
- **ชื่อผลิตภัณฑ์** ผู้ใช้สามารถระบุชื่อผลิตภัณฑ์ได้ด้วยโปรแกรมส่วนนี้ ซึ่งชื่อนี้จะถูกเก็บไว้ในไฟล์ข้อมูลและจะถูกเรียกมาตอนที่เปิดไฟล์ข้อมูลขึ้นมาเช่นกัน
- **ส่วนแสดงภาพ** ภาพที่ได้จากกล้องหรือภาพที่เปิดมาจากไฟล์ภาพนั้น จะถูกแสดงผลไว้ในส่วนนี้ นอกจากนั้น แล้วหากมีการใช้เครื่องมือวัดเพื่อทำการตรวจสอบผืนผ้า ข้อมูลที่เป็นกรอบสี่เหลี่ยมแสดง Region Of Interest และผลการตรวจจับ BLOB ก็จะแสดงให้เห็นดังในรูปที่ ซึ่งหากต้องการให้แสดงข้อมูลแต่ภาพอย่างเดียว โดยไม่ต้องการให้แสดงข้อมูลที่เกี่ยวกับเครื่องมือวัดก็สามารถทำได้โดยการ uncheck ที่ Overlay measurement tools
- **ส่วนสั่งงานกล้อง** การสั่งให้โปรแกรมทำการจับภาพนั้น คอมพิวเตอร์จะต้องต่อกล้อง USB ที่รองรับ Video For Window (VFW) ไว้เสียก่อน และผู้ใช้จะสามารถสั่งให้เริ่มจับภาพโดยการกดปุ่ม Start และสั่งให้หยุดจับภาพโดยปุ่ม Stop
- **ส่วนที่สั่งให้เครื่องมือวัดทำงาน** หากมีการเลือกใช้เครื่องมือวัดที่เป็นแบบสถิติหรือโครงข่ายประสาทเทียม ผู้ใช้สามารถสั่งให้เครื่องมือวัดทำงานหรือหยุดโดยการสั่งงานในบริเวณนี้ ซึ่งหากผู้ใช้สั่งไม่ให้เครื่องมือวัดทำงาน ผลการตรวจสอบจะขึ้นเป็น PASS เสมอ
- **ส่วนเลือกเครื่องมือในการตรวจสอบ** การเลือกใช้เครื่องมือวัดในโปรแกรมนี้ ผู้ใช้สามารถเลือกใช้ได้ทีละหนึ่งอย่างเท่านั้น นอกจากนั้น หากมีการสร้างเครื่องมือวัดชิ้นใหม่ขึ้นมา โปรแกรมจะทำลายข้อมูลของเครื่องมือวัดชิ้นเก่าอย่างอัตโนมัติ ซึ่งผู้ใช้สามารถสร้างเครื่องมือวัดได้โดยการกดปุ่มใดปุ่มหนึ่งในส่วนนี้เพื่อทำการเลือกว่า จะใช้เครื่องมือในการตรวจสอบผืนผ้าที่มีการทำงานโดยวิธีทางสถิติหรือโดยการใช้โครงข่ายประสาทเทียม และหลังจากที่เลือกชนิดของเครื่องมือที่ต้องการแล้ว ผู้ใช้สามารถระบุบริเวณที่ต้องการให้เครื่องมือวัดทำการตรวจสอบโดยการกดลากเม้าส์ลงไปในส่วนที่แสดงภาพ ซึ่งในขณะนั้น โปรแกรมจะแสดงบริเวณที่เครื่องมือวัดจะทำการตรวจสอบ และในทันทีทันใดที่ผู้ใช้ปล่อยปุ่มเม้าส์ รายละเอียด ผลการวัด และการตัดสินใจของเครื่องมือวัด โดยการใช้เงื่อนไขค่าปกติที่ตั้งไว้เป็นตัวตัดสิน ก็จะปรากฏออกมา ซึ่งผู้ใช้สามารถทำการแก้ไขค่าต่างๆ และดูผลการทำงานที่เกิดขึ้นได้อย่างทันที ซึ่งเมื่อใดก็ตามที่ผู้ใช้กด OK หน้าต่างที่ใช้ติดต่อกับผู้ใช้จะหายไป แต่เครื่องมือวัดก็จะฝังตัวอยู่ในโปรแกรม และทำงานอยู่ตลอดที่มีภาพเข้ามา ซึ่งผู้ใช้สามารถเรียกหน้าต่างที่ใช้ติดต่อกับผู้ใช้ขึ้นมาใหม่ได้ตลอดเวลา โดย double click ลงไปในส่วนแสดงภาพหรือส่วนที่เป็นตารางแสดงข้อมูลการวัด
- **ตารางแสดงข้อมูลการวัด** หากมีการสร้างเครื่องมือวัดขึ้น ตารางส่วนนี้จะแสดงข้อมูลเป็นตัวอักษรที่เครื่องวัดนั้นๆ ทำการวัดและวิเคราะห์ได้

- **เมนู File** เมนูส่วนนี้สามารถใช้เปิดไฟล์และบันทึกภาพที่มีจุดนามสกุล .bmp นอกจากนั้นแล้ว ข้อมูลการวัดต่างๆ ทั้งขนาดของรูป ชนิดของเครื่องรวมทั้งค่าพารามิเตอร์ต่างๆ ของเครื่องมือก็สามารถเปิดและบันทึกได้จากเมนูนี้เช่นกัน ซึ่งถ้าผู้อ่านมีความคุ่นเคยกับระบบปฏิบัติการ Windows จะไม่พบความแตกต่างในการทำงานแต่อย่างใด



ภาพที่ ง-5 แสดงส่วนของหน้าต่างหลักขณะเรียกใช้เมนู File

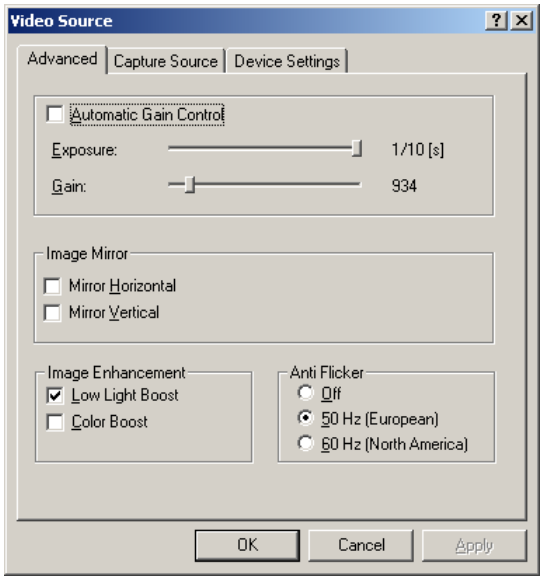
- เมนู Setting เป็นเมนูที่ใช้กำหนดลักษณะการทำงานของกล้อง ซึ่งประกอบไปด้วย 2 เมนูย่อย ดังแสดงไว้ในภาพที่ ง-6



ภาพที่ ง-6 แสดงส่วนของหน้าต่างหลักขณะเรียกใช้เมนู Setting

เมนู Setting นี้จะประกอบด้วย 2 เมนูย่อยด้วยกัน คือ

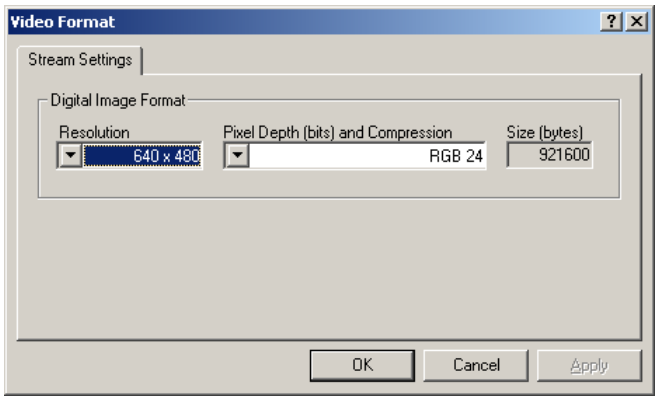
1. Source ซึ่งทำหน้าที่กำหนดค่าการทำงานต่างๆของกล้อง USB ที่ต่อไปกับคอมพิวเตอร์ โดยจะมีหน้าต่างที่ใช้ติดต่อกับผู้ใช้ดังในรูปที่ภาพที่ ง-7 ซึ่งผู้ใช้สามารถกำหนดการทำงานให้กล้องที่ต่อเชื่อมอยู่ในลักษณะต่างๆ ดังนี้



ภาพที่ ง-7 แสดงหน้าต่างย่อยที่จะปรากฏขึ้นหากมีการเลือกใช้เมนูย่อย Source

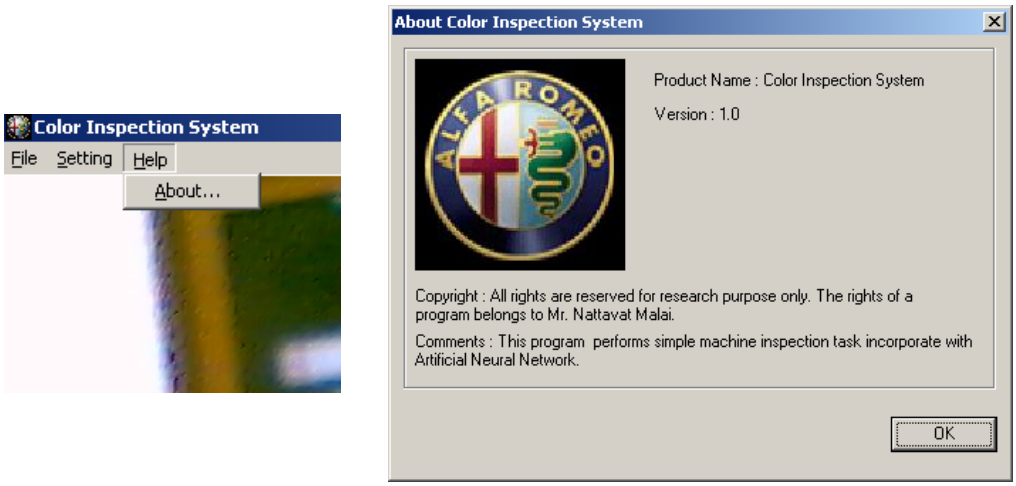
- Image Mirror ในส่วนนี้ผู้ใช้สามารถสั่งให้กล้องทำการกลับภาพได้ทั้งแนวนอนและแนวดิ่ง
- Image Enhancement ผู้ใช้สามารถสั่งให้กล้องเพิ่มความเข้มแสงหรือสีก่อนที่จะส่งมาให้คอมพิวเตอร์ได้อย่างอัตโนมัติ
- Anti Flicker ส่วนนี้จะเป็นการสั่งให้กล้องช่วยลดสัญญาณรบกวนจากไฟบ้าน ซึ่งในที่นี้ให้เลือกความถี่ 50 Hz

2. Format ซึ่งทำหน้าที่กำหนดความละเอียดของข้อมูลภาพ โดยจะมีหน้าต่างที่ใช้ติดต่อกับผู้ใช้ดังในภาพที่ ง-8 ซึ่งผู้ใช้สามารถกำหนดความละเอียดของภาพโดยการเลือกจากหน้าต่างที่ส่วนของ Resolution สำหรับในส่วนของ Pixel Depth นั้น ไม่ว่าผู้ใช้จะกำหนดค่าอะไรไป โปรแกรมจะทำการแก้ให้เป็น 24 บิตเสมอ ทั้งนี้ เนื่องจากเราต้องการให้ภาพเป็น 24 บิตที่แต่ละระนาบสีมีความละเอียด 8 บิตนั่นเอง



ภาพที่ ง-8 แสดงหน้าต่างย่อยที่จะปรากฏขึ้นหากมีการเลือกใช้เมนูย่อย Format

- เมนู Help ผู้ใช้สามารถกดเมนูนี้เพื่อดูรายละเอียดของชื่อผู้เป็นเจ้าของโปรแกรมได้ ดังแสดงไว้ในภาพที่ ง-9



ภาพที่ ง-9 แสดงรายละเอียดของเมนู Help

1. เมนู Help และเมนูย่อย About
2. หน้าต่างย่อยที่จะปรากฏขึ้นหากมีการเลือกใช้เมนูย่อย About

2. เครื่องมือวัดที่ใช้ตรวจสอบผืนผ้า

ดังที่ได้กล่าวไว้แล้ว ในโปรแกรม Color Inspection System ที่สร้างขึ้นนั้น มีเครื่องมือที่ใช้ในการตรวจสอบสีของผืนผ้าอยู่ 2 แบบด้วยกันคือ

1. เครื่องมือวัดที่ทำงานโดยวิธีทางสถิติซึ่งเปรียบเทียบค่าคุณสมบัติของ BLOB ที่ตรวจจับได้กับค่าคุณสมบัติของผืนผิวงานเสียซึ่งเป็นค่าที่ผู้ใช้ต้องกำหนดให้กับโปรแกรม
2. เครื่องมือวัดที่ทำงานโดยการใช้โครงข่ายประสาทเทียมเป็นตัวตัดสินว่า BLOB นั้นๆเป็นพื้นผิวงานเสียหรือไม่ ซึ่งทั้งนี้ผู้ใช้จะต้องทำการระบุใน BLOB ตัวอย่างที่จะนำไป train โครงข่ายประสาทเทียมเสียก่อนว่า มีมีชิ้น BLOB ใดบ้างที่เป็นพื้นผิวเสียหรือพื้นผิวดี

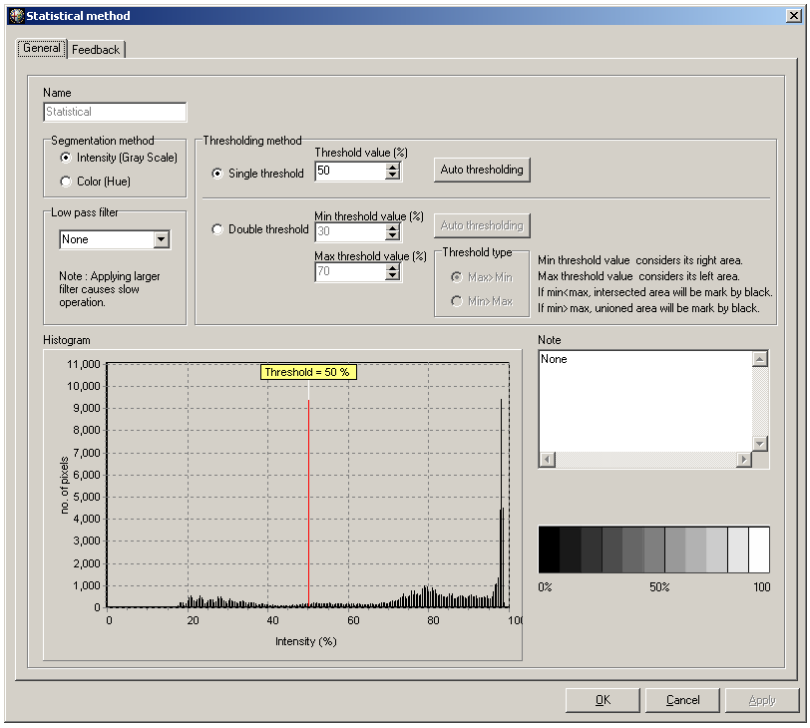
สำหรับการใช้งานเครื่องมือวัดทั้งสองนั้น หลังจากที่กดเลือกว่าต้องการใช้งานเครื่องมือชนิดใดแล้ว ก็ทำการลากเม้าส์ลงไปในส่วนแสดงผลของหน้าต่างหลักของโปรแกรม เพื่อทำการระบุบริเวณที่ใช้ในการตรวจสอบให้กับเครื่องมือ หลังจากที่ปล่อยเม้าส์ ในทันทีทันใดนั้น หน้าต่างที่ใช้ติดต่อกับผู้ใช้ของเครื่องมือวัดก็จะปรากฏขึ้น เพื่อให้ผู้ใช้ใส่ค่าพารามิเตอร์ที่ต้องการลงไป หรือเพื่อปรับเปลี่ยนลักษณะการทำงานของเครื่องมือวัด ซึ่งในตอนแรกทำงานตามค่าปกติที่

ตั้งไว้ให้เป็นค่าที่เหมาะสมกับความต้องการของผู้ใช้ ซึ่งหลังจากที่ป้อนค่าที่ต้องการแล้ว ผู้ใช้สามารถดูผลการทำงานได้ทันทีโยการกดปุ่ม Apply หรือ ถ้าผู้ใช้ไม่ต้องการดูผลการทำงานของค่าพารามิเตอร์ใหม่ แต่ต้องการให้ค่าที่กำหนดไว้ทำงานเลย ก็สามารถทำได้โดยการกดปุ่ม OK ซึ่งหลังจากนั้น เครื่องมือวัดก็จะฝังตัวเองอยู่ในโปรแกรมและคอยทำการตรวจสอบทุกภาพที่เข้ามา นอกจากนั้น ผู้ใช้สามารถบันทึกรายละเอียดการทำงานทุกอย่างของเครื่องมือวัดที่สร้างขึ้นลงไปในไฟล์อักษร เพื่อว่าจะสามารถหลังจากที่ออกจากโปรแกรม Color Inspection System แล้ว เมื่อเปิดโปรแกรมขึ้นมาอีกครั้งก็จะสามารถเปิดไฟล์ดังกล่าวขึ้นมาเรียกใช้เครื่องมือวัดที่เคยสร้างไว้แล้วอีกครั้ง

สำหรับเนื้อหาในส่วนต่อไปนี้จะเป็นการกล่าวถึง การกำหนดค่าพารามิเตอร์ต่างๆของเครื่องมือวัดทั้งสองโยการกำหนดผ่านหน้าต่างที่ใช้ติดต่อกับผู้ใช้ ซึ่งค่าเหล่านี้จะมีผลทั้งการประมวลผลภาพ และเกณฑ์ในการตัดสินว่า BLOB ใดเป็นพื้นผิวงานเสียหรือดีดังนี้

2.1 หน้าต่างของเครื่องมือวัดที่ทำงานโดยวิธีทางสถิติ

หน้าต่างที่ใช้ติดต่อกับผู้ใช้ของเครื่องมือวัดที่ทำงานโดยวิธีทางสถิติประกอบด้วย 2 หน้าย่อย คือ หน้าข้อมูลทั่วไป (General) และส่วนที่ป้อนข้อมูลกลับให้กับผู้ใช้ (Feedback) ซึ่งจะพบว่า หน้าข้อมูลทั่วไปของเครื่องมือชนิดนี้จะเหมือนกันทุกประการกับหน้าข้อมูลทั่วไปของเครื่องมือวัดที่ทำงานโดยโครงข่ายประสาทเทียม ดังนั้น จะขอกล่าวถึงรายละเอียดของหน้านี้เพียงแค่ในส่วนนี้เท่านั้น



ภาพที่ ง-10 หน้าข้อมูลทั่วไปของเครื่องมือวัดที่ใช้วิธีทางสถิติ

- ชื่อของเครื่องมือวัด (Name) ในส่วนนี้จะเป็นบอกให้ผู้ใช้ทราบว่า เป็นเครื่องมือวัดชนิดใดเท่านั้น ผู้ใช้ไม่สามารถทำการเปลี่ยนแปลงใดๆ ได้ ซึ่งสำหรับเครื่องมือวัดที่ทำงานโดยวิธีการสถิตินั้น ในส่วนนี้จะแสดงชื่อเป็น Statistical
- บันทึกช่วยจำ (Note) ในส่วนนี้ผู้ใช้สามารถใส่ตัวอักษรใดๆ ลงไปก็ได้ เพื่อใช้เตือนความจำเกี่ยวกับเครื่องมือวัด เช่น รายละเอียดเกี่ยวกับผลิตภัณฑ์ วันเวลาที่สร้างเครื่องมือวัดขึ้น หรือแม้กระทั่งชื่อผู้มีหน้าที่รับผิดชอบ ซึ่งข้อมูลส่วนนี้จะติดไปกับข้อมูลวัดและสามารถแก้ไขได้ตลอดเวลา เพื่อป้องกันความผิดพลาดในการใช้เครื่องวัดกับผลิตภัณฑ์ที่ผิดนั่นเอง
- วิธีการแยกบริเวณ (Segmentation method) ในส่วนนี้ผู้ใช้สามารถระบุได้ว่า ภาพตั้งต้นที่จะมาใช้แยกบริเวณนั้นจะเป็นภาพที่มีข้อมูลความเข้มแสง, Intensity (Gray Scale) หรือสี Color (Hue)
- ตัวกรองสัญญาณความถี่ต่ำผ่าน (Low-pass filter) หลังจากที่ได้ภาพตั้งต้นไม่ว่าจะเป็นภาพความเข้มแสงหรือสีแล้ว เครื่องมือจะทำการใช้ตัวกรองสัญญาณความถี่ต่ำผ่านกับภาพนั้น เพื่อเป็นการลดความไม่สม่ำเสมอของภาพ ซึ่งผู้ใช้สามารถเลือกตัวกรองสัญญาณความถี่ต่ำผ่านได้ตั้งแต่ขนาด 3x3, 5x5 และ 7x7 นอกจากนั้นผู้ใช้ยังสามารถเลือกที่จะไม่ใช้ตัวกรองสัญญาณความถี่ต่ำผ่านดังกล่าวได้อีกด้วย โดยการเลือกช่องที่ขึ้นตัวอักษรว่า None ซึ่งเป็นค่าปกติที่โปรแกรมเลือกไว้ให้ได้อีกด้วย
- การเลือกวิธีใช้ค่า Threshold (Thresholding method) หลังจากที่เลือกภาพตั้งต้นว่าเป็นความเข้มแสงหรือสี พร้อมทั้งใช้ตัวกรองสัญญาณความถี่ต่ำเพื่อลดความไม่สม่ำเสมอของภาพแล้ว ก็จะมาสู่ส่วนที่สำคัญของเครื่องมือวัด นั่นคือการใช้ค่า Threshold เพื่อเป็นการแยกบริเวณว่า บริเวณใดเป็นส่วนที่สนใจและบริเวณใดที่โปรแกรมไม่สนใจและจัดเป็นฉากหลัง (background) ไป อนึ่งถึงแม้ว่าข้อมูลภาพจะมีค่าอยู่ระหว่าง 0-255 แต่เพื่อความสะดวก โปรแกรมจะใช้ค่า Threshold ที่เป็นค่าร้อยละเท่านั้น ดังนั้น ถ้าค่าเท่ากับ 100 แสดงว่า ค่าจริงในขณะนั้นมีค่าเท่ากับ 255 นั่นเอง และในที่นี้เครื่องมือวัดถูกออกแบบให้มีวิธีการเลือกใช้ค่า Threshold อยู่ 2 แบบด้วยกันคือ
    - การใช้ค่า Threshold เดียว (Single threshold) โปรแกรมจะทำให้บริเวณของภาพตั้งต้นที่มีค่าน้อยกว่าค่า Threshold ที่เลือกใช้เปลี่ยนเป็นด้านมืดไป ส่วนที่เหลือก็ถูกเปลี่ยนเป็นด้านสว่างไป อนึ่งผู้ใช้สามารถเรียกค่า threshold ที่โปรแกรมแนะนำมา โดยการกดที่ปุ่ม Auto Thresholding ซึ่งโปรแกรมจะการหาค่าที่เหมาะสมให้โยการใช้วิธีของ Otzu และหลังจากนั้น ผู้ใช้ก็สามารถทำการปรับเปลี่ยนค่าได้ตามต้องการได้เช่นกัน

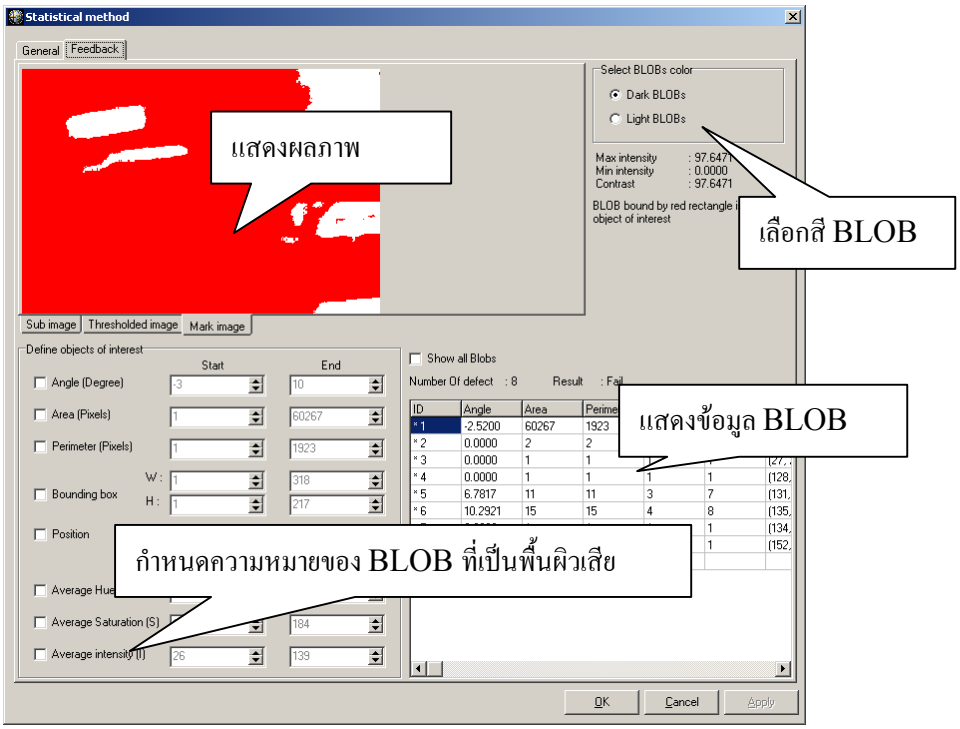
- การใช้ Threshold 2 ค่า (Double threshold) การเลือกใช้งานค่า Threshold ในลักษณะนี้ ก็เพื่อให้โปรแกรมทำการจับเฉพาะบริเวณที่มีค่าความเข้มแสงหรือสีที่สนใจจริงๆ ซึ่งในที่นี้ผู้ใช้จำเป็นต้องระบุค่า Threshold 2 ค่าในช่องของ Min threshold value (เรียกย่อว่า MinT) และช่อง Max threshold value (เรียกย่อว่า MaxT) ซึ่งการทำงานของโปรแกรมจะมีอยู่ 3 รูปแบบขึ้นอยู่กับผลการเปรียบเทียบระหว่างค่าทั้งสองดังนี้
    - กรณีที่ MinT น้อยกว่า MaxT โปรแกรมจะทำการกำหนดให้ภาพขาออกที่ตำแหน่งเดียวกับภาพตั้งต้น ที่มีค่าอยู่ระหว่าง MinT จนถึง MaxT มีค่าเป็นศูนย์หรือเป็นด้านมืดไป ส่วนที่มีค่านอกช่วงดังกล่าวก็จะถูกกำหนดให้มีค่า 255 หรือเป็นด้านสว่างไป
    - กรณีที่ MinT มากกว่า MaxT ในกรณีนี้โปรแกรมจะกำหนดให้ภาพขาออกที่ตำแหน่งเดียวกับภาพตั้งต้นที่มีค่าอยู่ระหว่าง MinT จนถึง 255 และตั้งแต่ 0 ถึง MaxT ให้มีค่าเท่ากับศูนย์หรือเป็นด้านมืดไป ในขณะเดียวกันก็จะกำหนดให้บริเวณอื่นๆที่มีค่านอกเหนือจากนี้เป็นด้านสว่างไป
    - กรณีที่ MinT เท่ากับ MaxT ในกรณีนี้โปรแกรมจะทำการแปลงเป็นภาพ Binary ซึ่งให้ผลเทียบเท่ากับการใช้ค่า Threshold เพียงตัวเดียว

นอกจากนั้นแล้ว ผู้ใช้ยังสามารถกดปุ่ม Auto Thresholding สำหรับกรณีที่ใช้ Threshold 2 ค่าได้อีกด้วย ซึ่งโปรแกรมจะทำการหาค่าสูงสุดและต่ำสุดให้อย่างอัตโนมัติ

- ส่วนแสดงผล Histogram และภาพความเข้มแสงหรือวงกลมสีที่อยู่ข้างๆกัน ในส่วนนี้ผู้ใช้สามารถ Zoom เข้าไปดูค่าที่ต้องการได้โดยการลากเม้าส์คลุมบริเวณที่จะขยายใหญ่ และเมื่อใดก็ตามหากต้องการดู Histogram จากอัตราขยายแบบปกติก็ให้ทำการ double click ที่รูป histogram ซึ่งข้อมูลที่แสดงใน histogram และภาพความเข้มแสงหรือวงกลมสีที่อยู่ข้างๆกันนั้น จะแสดงข้อมูลที่เกี่ยวกับความเข้มแสงหรือสีออกมา ก็อยู่กับว่าผู้ใช้เลือก Segmentation method แบบใดนั่นเอง

หน้าป้อนข้อมูลกลับให้กับผู้ใช้ (Feedback)

ในหน้าข้อมูลทั่วไป (General) นั้น ผลที่ได้คือภาพที่มี 2 ระดับ (Binary image) ซึ่งเป็นภาพที่มีเพียง 2 สีเท่านั้น คือขาวและดำ สำหรับในหน้าป้อนข้อมูลกลับให้กับผู้ใช้ (Feedback) นี้จะใช้สำหรับการกำหนดลักษณะของ BLOB ที่จัดเป็นพื้นผิวเสีย ซึ่งมีผลโดยตรงต่อการตัดสินคุณภาพของผืนผ้า ภาพที่ ง-11 แสดงหน้าป้อนข้อมูลกลับให้กับผู้ใช้ ซึ่งประกอบด้วยส่วนต่างๆ ดังนี้



ภาพที่ ง-11 หน้าป้อนข้อมูลกลับให้กับผู้ใช้ของเครื่องมือวัดที่ใช้วิธีทางสถิติ

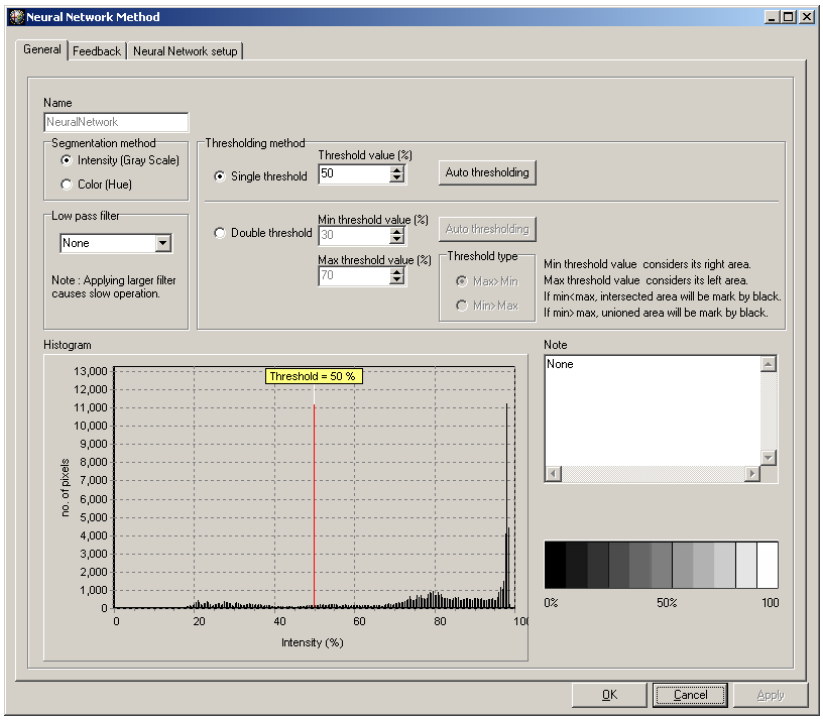
- ส่วนเลือกสี BLOB จากภาพ 2 ระดับ ซึ่งสร้างจากค่าพารามิเตอร์ต่างๆ ในหน้าข้อมูลทั่วไป ผู้ใช้สามารถกำหนดให้โปรแกรมสนใจ BLOB ที่มีสีดำ (Dark BLOB) หรือขาว (Light BLOB) ทั้งนี้ขึ้นอยู่กับความต้องการของงานแต่ละแบบ
- ส่วนแสดงผลภาพ ในส่วนนี้ผู้ใช้จะสามารถเลือกดูภาพย่อย (Sub image) ที่ดึงออกมาจากภาพหลักได้ หรือหากต้องการดูว่าภาพ 2 ระดับที่ได้จากค่าต่างๆในหน้าข้อมูลทั่วไปนั้นมีผลเป็นอย่างไรก็สามารถดูได้เช่นกัน (Thresholded image) และท้ายที่สุกหากต้องการดูการจำแนก BLOB ว่าชิ้นใดเป็นพื้นผิวเสียหรือพื้นผิวดีนั้นก็สามารถดูได้จากส่วนของ Marked image แสดงบริเวณที่เป็นสีแดงและเป็นสีเขียน

โดยหากบริเวณใดมีสีเขียวหมายถึง BLOB ในตำแหน่งนั้นจัดเป็นพื้นผิวดี ในขณะที่หากบริเวณใดเป็นสีแดง BLOB นั้นๆ โปรแกรมจะมองเห็นเป็นพื้นผิวเสียนั่นเอง

- ส่วนกำหนดความหมายของ BLOB ที่เป็นพื้นผิวเสีย ส่วนนี้เองที่เครื่องมือวัดที่ทำงานโดยวิธีทางสถิติ จะใช้เพื่อเปรียบเทียบและตัดสินว่า BLOB ที่ตรวจจับได้นั้นมีชิ้นใดบ้างที่จัดเป็นพื้นผิวงานเสีย ซึ่งหากแม้มีซักชิ้นเดียว เครื่องมือวัดชนิดนี้ก็จะตัดสินให้ผืนผ้าในช่วงนั้นเป็นงานเสียอย่างทันที สำหรับลักษณะของพื้นผิวงานเสียที่ผู้ใช้สามารถกำหนดได้นั้น ประกอบด้วย ลักษณะทางกายภาพ ได้แก่ มุมของแกนหลักของ BLOB (Angle) พื้นที่ (Area) เส้นรอบรูป (Perimeter) ขนาดของสี่เหลี่ยมที่เล็กที่สุดที่สามารถคลุม BLOB ได้ (Bounding BLOB size) รวมทั้งตำแหน่งของจุดศูนย์กลางของ BLOB (Centoid position) นอกจากนั้น ผู้ใช้ยังสามารถระบุคุณสมบัติทางสีของ BLOB ที่จัดเป็นพื้นผิวเสียได้อีกด้วย ซึ่งคุณสมบัติเหล่านี้ได้แก่ ค่าเฉลี่ย Hue ค่าเฉลี่ย Saturation และค่าเฉลี่ย Intensity ของ BLOB นั่นเอง
- ส่วนแสดงข้อมูล BLOB ในส่วนนี้เป็นการแสดงข้อมูลที่เป็นตัวเลขของคุณสมบัติต่างๆที่กล่าวมาแล้วของ BLOB ที่ตรวจจับได้ในภาพย่อย ซึ่งหากผู้ใช้ check ที่ช่อง Show all BLOBs นั้นโปรแกรมจะแสดงข้อมูล BLOB ที่ตรวจจับได้ทุกตัวออกมา โดยที่ตัวที่จัดเป็นพื้นผิวงานเสียนั้น จะมีเครื่องหมายดอกจันทร์ (*) วางไว้ที่หมายเลย BLOB นั้น และหากว่าผู้ใช้ uncheck ช่องดังกล่าว โปรแกรมจะแสดงเฉพาะข้อมูลของ BLOB ที่จัดเป็นพื้นผิวเสียเท่านั้น นอกไปจากนั้น การ check หรือ uncheck ในช่องดังกล่าวนี้ จะมีผลต่อการแสดงผลในส่วนแสดงภาพของหน้าต่างหลักด้วย โดยที่หากผู้ใช้ uncheck ที่ช่องนี้ ในบริเวณของหน้าต่างหลักจะมีสี่เหลี่ยมย่อยสีแดงซึ่งแสดงบริเวณของ BLOB ที่จัดเป็นพื้นผิวเสียเท่านั้น และในทางตรงกันข้ามหากผู้ใช้ Check ในช่องนี้แล้ว ในบริเวณของหน้าต่างหลักจะมีทั้งสี่เหลี่ยมย่อยสีแดง และสีเขียว ซึ่งบ่งชี้บริเวณของ BLOB ที่ตรวจจับได้แต่เป็นพื้นผิวเสียและดีตามลำดับ

2.2 หน้าต่างของเครื่องมือวัดที่ทำงานโดยโครงข่ายประสาทเทียม

สำหรับเครื่องมือวัดที่จำแนก BLOB ว่าเป็นพื้นผิวดีหรือเสียนั้น จะมีหน้าที่เอาไว้ติดต่อกับผู้ใช้อยู่ 3 หน้าด้วยกัน ได้แก่ หน้าข้อมูลทั่วไป (General) หน้าที่ป้อนข้อมูลกลับให้ผู้ใช้ (Feedback) และหน้าที่ใช้สำหรับกำหนดลักษณะการทำงานของโครงข่ายประสาทเทียม (Neural network setup)

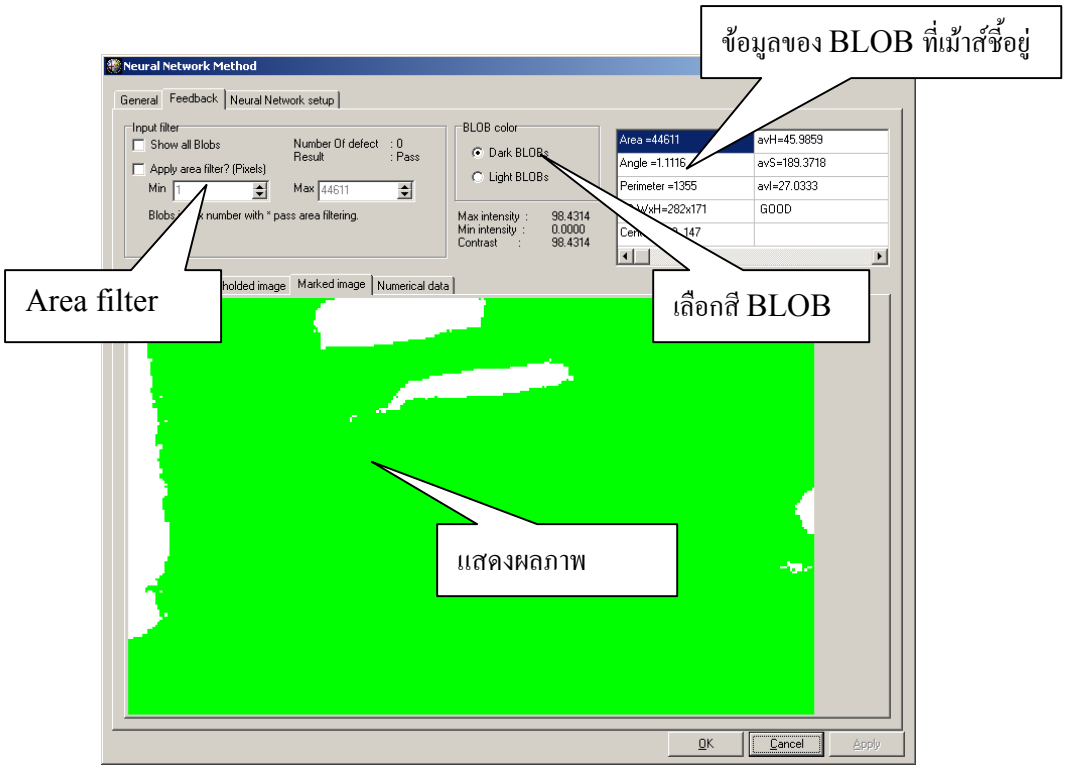


ภาพที่ ง-12 หน้าข้อมูลทั่วไปของเครื่องมือวัดที่ใช้โครงข่ายประสาทเทียม

จะพบว่าหน้าข้อมูลทั่วไป (General) ของเครื่องมือวัดที่ทำงานโดยโครงข่ายประสาทเทียมที่แสดงไว้ในภาพที่ ง-12 นั้น จะเหมือนกันทุกประการกับหน้าข้อมูลทั่วไปของเครื่องมือวัดที่ทำงานโดยวิธีทางสถิติ ที่ได้ให้รายละเอียดไปแล้ว ดังนั้นจึงขอละการอธิบายฟังค์ชั่นการทำงานต่างๆ ของหน้าข้อมูลทั่วไปของเครื่องมือวัดที่ใช้โครงข่ายประสาทเทียม ไว้ ณ ที่นี้

หน้าป้อนข้อมูลกลับให้กับผู้ใช้ (Feedback)

ในหน้านี้จะก็จะคล้ายคลึงกับหน้าป้อนข้อมูลให้ผู้ใช้ (Feedback) ของเครื่องมือวัดที่ทำงานโดยวิธีการทางสถิติ อย่างไรก็ตามจะพบว่า หน้าป้อนข้อมูลกลับให้กับผู้ใช้ของเครื่องมือวัดที่ใช้โครงข่ายประสาทเทียมนี้ นอกจากจะใช้สำหรับแสดงผลการตัดสินใจของโครงข่ายประสาทเทียมให้ผู้ใช้รู้ว่า BLOB ที่ตรวจจับได้นั้น มีชิ้นใดบ้างที่โครงข่ายจัดเป็นพื้นผิวเสียหรือดี ผู้ใช้ยังสามารถสร้างข้อมูลตัวอย่างเพื่อนำไป Train โครงข่ายประสาทเทียมได้จากหน้าป้อนข้อมูลกลับให้กับผู้ใช้นี้ได้เช่นกัน สำหรับหน้านี้ มีองค์ประกอบย่อยๆ ต่างๆ ดังนี้



ภาพที่ ง-13 หน้าป้อนข้อมูลกลับให้กับผู้ใช้ของเครื่องมือวัดที่โครงข่ายประสาทเทียม

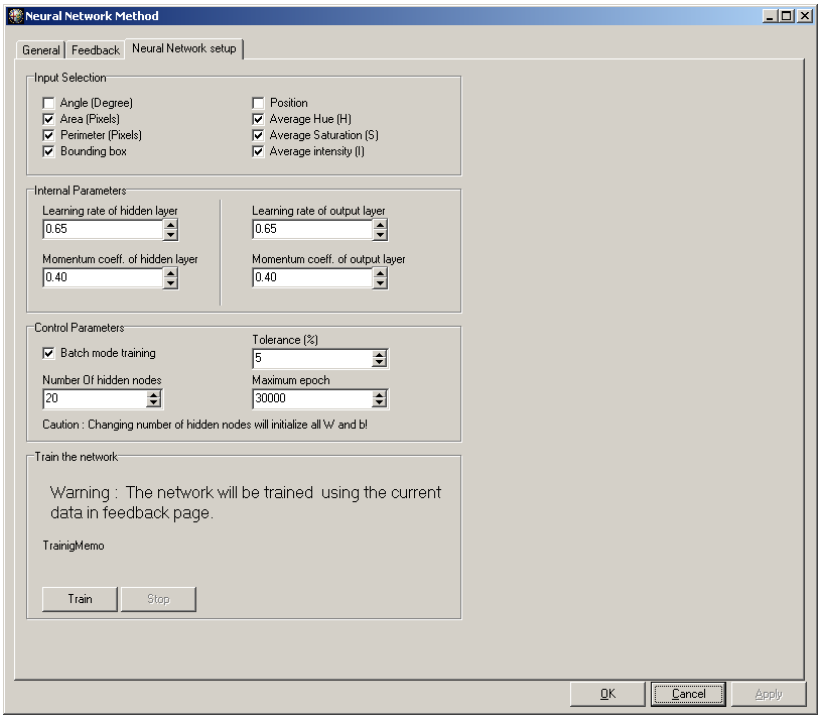
- ส่วนเลือกสี BLOB จากภาพ 2 ระดับ ซึ่งสร้างจากค่าพารามิเตอร์ต่างๆ ในหน้าข้อมูลทั่วไป ผู้ใช้สามารถกำหนดให้โปรแกรมสนใจ BLOB ที่มีสีดำ (Dark BLOB) หรือขาว (Light BLOB) ทั้งนี้ขึ้นอยู่กับความต้องการของงานแต่ละแบบ
- Area filter ในทางทฤษฎีแล้ว เราสามารถป้อนข้อมูลของ BLOB ทุกตัวที่เจอในภาพให้กับโครงข่ายประสาทเทียม เพื่อให้โครงข่ายเรียนรู้ หรือทำการตัดสินว่ามีชิ้นใดบ้างที่จัดเป็นพื้นผิวเสียของผืนผ้า อย่างไรก็ตามในทางปฏิบัตินั้น เราทราบอยู่ก่อนแล้วว่า BLOB ที่มีขนาดเล็กมาก (มีพื้นที่ 5-15 พิกเซล) นั้นไม่มีทางจะเป็นพื้นผิวเสียจริงได้เลย ดังนั้นโปรแกรมส่วนนี้จึงถูกออกแบบและสร้างขึ้นเพื่อกรอง BLOB ที่มีขนาดใหญ่หรือเล็กเกินกว่าจะจัดเป็นพื้นผิวเสียได้ ซึ่งด้วยหลักการเลือกขนาดของ BLOB ดังกล่าว จะลดจำนวน BLOB ที่ป้อนให้กับโครงข่ายประสาทเทียมลงได้มาก ทำให้โครงข่ายประสาทเทียมมีการทำงานรวดเร็วขึ้น นอกจากนั้นยังทำให้ประสิทธิภาพการจำแนกของโครงข่ายเพิ่มขึ้นอีกด้วย
- ส่วนแสดงผลภาพ ในส่วนนี้ผู้ใช้จะสามารถเลือกดูภาพย่อย (Sub image) ที่ดึงออกมาจากภาพหลักได้ หรือหากต้องการดูว่าภาพ 2 ระดับที่ได้จากค่าต่างๆในหน้าข้อมูลทั่วไปนั้นมีผลเป็นอย่างไร ก็สามารถดูได้เช่นกัน (Thresholded image) และหากต้องการดูผลการจำแนก BLOB โดยโครงข่ายประสาทเทียมว่าชิ้นใดเป็นพื้นผิวเสีย

หรือพื้นผิวดีนั้น ก็สามารถดูได้จากส่วนของ Marked image ซึ่งจะแสดงทั้งบริเวณที่เป็นสีแดงและเป็นสีเขียวโดยหากบริเวณใดมีสีเขียวหมายถึง BLOB ในตำแหน่งนั้นจัดเป็นพื้นผิวดี ในขณะที่หากบริเวณใดเป็นสีแดง BLOB นั้นๆ โปรแกรมจะมองเห็นเป็นพื้นผิวเสียนั่นเอง

สำหรับในส่วนของ Marked image นี้ นอกจากจะใช้เพื่อแสดงผลการตัดสินของโครงข่ายประสาทเทียมว่า มี BLOB ชิ้นใดจัดเป็นพื้นผิวงานเสียแล้ว ผู้ใช้ยังสามารถทำการระบุหรือแก้ผลการตัดสินของโครงข่ายประสาทเทียม โดยการกด mouse ลงไปในภาพ Marked image ในบริเวณที่ต้องการแก้ไข BLOB นั้นๆ ซึ่งจะทำให้ BLOB นั้นๆเปลี่ยนกลับไปกลับมาระหว่างพื้นผิวดีและเสีย และหลังจากแก้ไขข้อมูลทั้งหมดแล้ว ผู้ใช้ก็สามารถใช้ข้อมูลชุดเดียวกันนี้ไป train โครงข่ายประสาทเทียมได้อีกด้วย

นอกจากนั้นแล้ว ในส่วนแสดงภาพนั้น ยังมีส่วนแสดงข้อมูล BLOB ที่เป็นตัวเลข (Numerical data) ซึ่งแสดงข้อมูลที่เป็นตัวเลขของคุณสมบัติต่างๆที่กล่าวมาแล้วของ BLOB ที่ตรวจจับได้ในภาพย่อย ซึ่งหากผู้ใช้ check ที่ช่อง Show all BLOBs นั้นโปรแกรมจะแสดงข้อมูล BLOB ที่ตรวจจับได้ทุกตัวออกมา โดยที่ตัวที่จัดเป็นพื้นผิวงานเสียนั้น จะมีเครื่องหมายดอกจันทร์ (*) วางไว้ที่หมายเลย BLOB นั้น และหากว่าผู้ใช้ uncheck ช่องดังกล่าว โปรแกรมจะแสดงเฉพาะข้อมูลของ BLOB ที่จัดเป็นพื้นผิวเสียเท่านั้น นอกไปจากนั้น การ check หรือ uncheck ในช่อง ดังกล่าวนี้ จะมีผลต่อการแสดงผลในส่วนแสดงภาพของหน้าต่างหลักด้วย โดยที่หากผู้ใช้ uncheck ที่ช่องนี้ ในบริเวณของหน้าต่างหลักจะมีสี่เหลี่ยมย่อยสีแดงซึ่งแสดงบริเวณของ BLOB ที่จัดเป็นพื้นผิวเสียเท่านั้น และในทางตรงกันข้ามหากผู้ใช้ Check ในช่องนี้แล้ว ในบริเวณของหน้าต่างหลักจะมีทั้งสี่เหลี่ยมย่อยสีแดง และสีเขียว ซึ่งบ่งชี้บริเวณของ BLOB ที่ตรวจจับได้แต่เป็นพื้นผิวเสียและดีตามลำดับ

หน้าที่ใช้สำหรับกำหนดลักษณะการทำงานของโครงข่ายประสาทเทียม (Neural Network Setup)



ภาพที่ ง-14 หน้าที่ใช้สำหรับกำหนดลักษณะการทำงานของโครงข่ายประสาทเทียม (Neural Network Setup)

ในหน้านี้ จะเป็นส่วนที่ใช้กำหนดรูปแบบการทำงานของโครงข่ายประสาทเทียมที่เครื่องมือวัดใช้เป็นตัวตัดสินว่า BLOB ใดจัดเป็นพื้นผิวเสียหรือพื้นผิวดี นอกจากนั้น ยังใช้สั่งโปรแกรมให้ทำการ train โครงข่ายประสาทเทียมได้อีกด้วย ซึ่งหน้านี้ประกอบด้วยส่วนต่างๆ ดังนี้

- ส่วนที่ใช้เลือกคุณสมบัติของ BLOB ที่จะใช้เป็น Input Vector ให้กับโครงข่ายประสาทเทียม (Input Selection) ซึ่งสามารถเลือกคุณสมบัติต่างๆ ได้ดังนี้
  - มุมของแกนหลักของ BLOB (Angle)
  - พื้นที่ (Area)
  - เส้นรอบรูป (Perimeter)
  - ขนาดของสี่เหลี่ยมที่เล็กที่สุดที่สามารถคลุม BLOB ได้ (Bounding BLOB Size)
  - ตำแหน่งของจุดศูนย์กลางของ BLOB (Centoid Position)
  - ค่าเฉลี่ย Hue
  - ค่าเฉลี่ย Saturation
  - ค่าเฉลี่ย Intensity

จะพบว่า การที่ผู้ใช้ Check ที่หัวข้อเหล่านี้ หมายถึงการกำหนดให้คุณสมบัตินั้นๆ ของ BLOB แต่ละตัวถูกนำไปใส่ไว้ใน input vector ซึ่งหากเลือกหลายๆ คุณสมบัติก็จะทำให้เวคเตอร์ดังกล่าวมีขนาดใหญ่และเสียเวลาในการประมวลผลยาวนานขึ้นเช่นกัน

- ส่วนที่ใช้กำหนด parameter ภายในให้กับโครงข่ายประสาทเทียม ซึ่งได้แก่ค่า Learning rate และ momentum coefficient ของแต่ละ layer
- ส่วนที่ใช้กำหนดค่าพารามิเตอร์ที่ใช้ควบคุมการทำงาน ในส่วนนี้จะประกอบด้วยหลายๆ ส่วนสำคัญที่มีผลต่อประสิทธิภาพการทำงานของโครงข่ายประสาทเทียมเป็นอย่างมาก ซึ่งได้แก่
  a. โหมดการ Train หากผู้ใช้ Check ลงในช่อง Batch mode training การ Train โครงข่ายประสาทเทียมจะเป็นแบบ Batching โดยโปรแกรมจะทำการ update ค่า Weight matrix และ bias vector เพียงครั้งเดียวเท่านั้น หลังจากที่ป้อนกลุ่มตัวอย่างทั้งหมดเข้าไปให้กับโครงข่ายและในทางตรงกันข้าม หากผู้ใช้ uncheck ช่องดังกล่าว การ Train โครงข่ายประสาทเทียมจะเป็นแบบ Incremental training ซึ่งจะทำการ update ค่า Weight matrix และ bias vector ทุกครั้งที่มีการป้อนตัวให้กับโครงข่ายประสาทเทียม
  b. จำนวนของ node ใน Hidden layer (Number Of Hidden Nodes) ซึ่งการเปลี่ยนแปลงค่าใดๆ ในช่องนี้ จะทำให้โครงสร้างของโครงข่ายประสาทเทียมทั้งหมดเกิดการเปลี่ยนแปลง ซึ่งผู้ใช้จำเป็นจะต้อง Train โครงข่ายประสาทเทียมใหม่ทุกครั้งที่มีการเปลี่ยนแปลงดังกล่าว
  c. ค่าความถูกต้อง (Tolerance) ซึ่งระบุในรูปร้อยละค่าดังกล่าว คือค่าสูงสุดของผลต่างระหว่าง target vector และ output vector ของโครงข่ายประสาทเทียม ซึ่งหากตัวอย่างใดมีค่าน้อยกว่าค่าที่กำหนดไว้ จะถือว่าการ Train สำหรับตัวอย่างนั้นเป็นผลสำเร็จ ซึ่งโปรแกรมจะหยุดเทรนก็ต่อเมื่อตัวอย่างทุกตัวมีค่าดังกล่าวน้อยกว่าค่าที่กำหนดไว้ในช่องนี้
  d. จำนวนรอบในการ Train (Maximum Epoch) ในการ Train โครงข่ายประสาทเทียมโดยการใช้ระเบียบวิธีคำนวณแบบ Back Propagation Training Algorithm นั้น การป้อนตัวอย่างทั้งหมดนับเป็นการ Train 1 รอบ (Epoch) ซึ่งผู้ใช้สามารถกำหนดเงื่อนไขหยุด Train ได้จากช่องนี้

ภาคผนวก จ

คู่มือการโปรแกรม (Programming Manual)

ภาคผนวก จ
คู่มือการโปรแกรม (Programming Manual)

ส่วนประกอบของโปรแกรม Color Inspection System สำหรับใช้ในการทดลองของวิทยานิพนธ์นี้ประกอบด้วยองค์ประกอบ 2 กลุ่มหลักๆ ด้วยกันคือ ส่วนประกอบที่เป็น Graphical User Interface ซึ่งเป็นส่วนที่ติดต่อกับผู้ใช้ ทั้งการแสดงผลและการรับค่าต่างๆจากผู้ใช้ เช่น ฟอร์ม ปุ่มกด และ ตัวแสดงอักษร เป็นต้น และส่วนประกอบกลุ่มที่สองคือ คลาสซึ่งทำหน้าที่ควบคุมและคำนวณผลต่างๆ ซึ่งจะพบว่า องค์ประกอบกลุ่มแรกนั้น สามารถหารายละเอียดการใช้งานได้จากหนังสือคู่มือการโปรแกรมโดยทั่วไป แต่สำหรับองค์ประกอบในกลุ่มที่สองนั้น เป็นสิ่งที่ถูกออกแบบและจัดสร้างขึ้นมา เพื่อนำมาใช้กับโปรแกรมนี้โดยเฉพาะ ดังนั้นในเนื้อหาที่จะนำเสนอต่อไปนี้ จะเป็นการอธิบายองค์ประกอบของโปรแกรมในกลุ่มที่สองนี้เท่านั้น

อนึ่ง คลาสต่างๆที่ได้ออกแบบและจัดสร้างขึ้นเองนั้น ไม่มีคลาสใดเลยที่มีความสัมพันธ์แบบสืบทอดต่อจากกัน (Inherit Relationship) แต่จะมีความสัมพันธ์ต่อกันโดยเป็นเพียงส่วนประกอบภายใน class อื่นหรือที่เรียกว่า Composition Relationship เท่านั้น ยกตัวอย่างเช่น ภายในคลาส KRectangle ซึ่งเป็น คลาสที่ทำหน้าที่เก็บพิกัดของบริเวณที่จะทำการตรวจสอบที่จะถูกกำหนดจากผู้ใช้งานโปรแกรม โดยจะทำการเก็บจุดสำคัญเพียง 2 จุด คือ จุดบนซ้ายสุดและล่างขวาของพื้นที่สี่เหลี่ยมเท่านั้น ดังนั้น ในคลาส KRectangle จะประกอบด้วย object ของคลาส KPoint 2 ตัวที่ทำหน้าที่นี้ เป็นต้น นอกไปจากนั้น เนื่องจากตัวแปลงภาษา (Compiler) ที่เลือกใช้สำหรับการพัฒนาโปรแกรม Color Inspection System คือ Borland C++ Builder 5.0 ซึ่งสามารถให้ความรวดเร็วของการทำงานของตัวโปรแกรมที่สร้างจากภาษา C++ และยังใช้เรียนรู้การใช้งานคอมโพแนนต์ต่างๆ สั้นกว่าเมื่อเปรียบเทียบกับ Visual C++ 6.0 จึงทำให้คลาสที่ได้ออกแบบและจัดสร้างขึ้น เพื่อทำหน้าที่ควบคุมและคำนวณผลต่างๆนั้น สามารถแบ่งได้เป็น 2 กลุ่มด้วยกัน คือ กลุ่มที่สามารถนำไปใช้ในตัวแปลงภาษา (Compiler) ใดๆก็ได้ ที่รองรับภาษา C++ ไม่ว่าจะเป็น Microsoft Visual C++ 6.0 หรือ Borland C++ Builder 5.0 และอีกกลุ่มคือคลาสที่สามารถนำไปใช้ได้แค่ใน Borland C++ Builder 5.0 เท่านั้น ซึ่งคลาสที่จัดอยู่กลุ่มหลังนี้มีเพียง 2 คลาสเท่านั้น คือ KBMPInterface ซึ่งใช้ในการโหลดและบันทึกไฟล์ภาพที่มีนามสกุล .bmp และ KDrawer ที่ทำหน้าที่แสดงผลภาพลงไปในคอมโพแนนต์ TImage ซึ่งเป็นคอมโพแนนต์เฉพาะที่มีอยู่ใน Borland C++ Builder 5.0 เท่านั้น สำหรับคลาสต่างๆที่ได้จัดสร้างขึ้น สามารถจัดกลุ่มได้ดังต่อไปนี้

1. KData
2. KBMPInterface

3. KDrawer
4. KPoint และ KRectangle
5. KRun KBLOB และ KListOfKBLOB
6. KVector, KMatrix KVectorPair และ KListOfKVectorPair
7. KBPN
8. KStateBLOB และ KNeuralBLOB

รายละเอียดการทำงานของแต่ละกลุ่มมีดังต่อไปนี้

1. KData คลาสตัวนี้ถูกออกแบบให้มีลักษณะเป็นอาร์เรย์ขนาด 2 มิติ ที่แต่ละตำแหน่งสามารถเก็บข้อมูลจำนวนเต็มที่มีค่าระหว่าง 0 ถึง 255 หรือข้อมูลขนาด 1 byte เท่านั้น คลาสตัวนี้ยังอำนวยความสะดวกแก่โปรแกรมเมอร์ที่จะส่งอาร์เรย์ 2 มิติไปเป็นอาร์กรูเมนต์ของฟังค์ชั่นต่างๆ ซึ่งไม่จำเป็นจะต้องส่งจำนวนแถวและจำนวนหลักแยกกันไปด้วย เนื่องจากคุณสมบัติ 2 อย่างนี้ได้ติดไปกับตัว object ของคลาสนี้อยู่แล้ว นอกจากนั้น เพื่อให้การอ้างอิงระบบพิกัดเป็นไปตามที่มีใช้งานกันอยู่ทั่วไปในงานด้านประมวลภาพแบบดิจิตอล รวมทั้งโปรแกรม MATLAB การอ้างอิงระบบพิกัดใดๆ จะเริ่มด้วยเลข 1 เสมอ โดยที่ค่ามุมซ้ายบนสุดของอาร์เรย์ 2 มิติทั้งหมดที่ถูกออกแบบขึ้นในโปรแกรมนี้ (ซึ่งรวมถึง KMatrix ที่จะได้กล่าวถึงในหัวข้อต่อไปด้วย) จะมีค่าพิกัดเป็น 1, 1 ดังแสดงไว้ในภาพที่ จ-1

|(1,1) (1,2) (1,3)|
|(2,1) (2,2) (2,3)|
|(3,1) (3,2) (3,3)|

ภาพที่ จ-1 ระบบพิกัดของคลาส KData และ KMatrix ที่มีการเก็บข้อมูลเป็นอาร์เรย์ 2 มิติที่พิกัดของจุดซ้ายบนสุดมีค่าเป็น 1,1

สำหรับฟังค์ชั่นสำคัญของคลาสนี้มีดังนี้

1.1 Constructor ซึ่งเป็นการเพิ่มความสามารถในการประกาศตัวแปรแบบอาร์เรย์ 2 มิติ ที่ทำให้ผู้ใช้งานคลาสสามารถประกาศ object ของคลาสนี้ได้ 3 รูปแบบดังนี้

```
KData Im1;
KData Im2(5,6,8);
KData Im3(Im1);
```

จากตัวอย่างโปรแกรมภาษา C++ ข้างต้น จะพบว่า ตัวแปร Im1 นั้น ถ้าไม่มีการระบุค่าพารามิเตอร์ใดๆ จะเป็นตัวแปรที่เป็นอาร์เรย์ 2 มิติขนาด 3 แถว 3 หลัก ที่มีทุกช่องเป็นศูนย์ และสำหรับตัวแปร Im2 นั้นจะเป็นอาร์เรย์ 2 มิติที่มี 5 แถว 6 หลัก โดยที่ทุกช่องในอาร์เรย์ที่สร้างขึ้นนั้น จะมีค่าเท่ากับ 8 และท้ายที่สุดหากกำหนดให้พารามิเตอร์ของการสร้างตัวแปรเป็น object ของคลาส KData แล้วตัวแปรที่สร้างขึ้นให้จะมีลักษณะเหมือนกับ object ที่ใช้เป็นพารามิเตอร์นั้นทุกประการ ดังนั้น Im3 ซึ่งเป็นการทำสำเนาซ้ำของตัวแปร Im1 นั่นเอง

1.2 การเข้าถึงข้อมูลที่เก็บไว้ในอาร์เรย์ 2 มิติที่สร้างขึ้น การเข้าถึงข้อมูลดังกล่าว สามารถทำได้โดยการใช้เครื่องหมายวงเล็บ ดังตัวอย่างต่อไปนี้ ซึ่งเป็นการอ่านค่าข้อมูลในตำแหน่งแถวที่ 3 หลักที่ 2 ของ Im2 มาไว้ในตัวแปรที่ชื่อ Temp1 และกำหนดค่าที่ตำแหน่งแถวที่ 5 หลักที่ 4 ของ Im2 นี้ให้เท่ากับ 255 ตามลำดับ

```
KData Im2(8,8,120);
BYTE Temp1;
Temp1  =  Im2(3,2);
Im2(5,4)  =  255;
```

1.3 การหาจำนวนแถวและหลักของอาร์เรย์ 2 มิตินั้น สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น GetNumberOfRows() และ GetNumberOfColumns() ตามลำดับ อนึ่งจะพบว่า ฟังค์ชั่นทั้งสองนั้นเป็นการเรียกฟังค์ชั่นที่ไม่ต้องการอาร์กรูเมนต์ใดๆ

1.4 ฟังค์ชั่นที่เกี่ยวกับการคำนวณอาร์เรย์ 2 มิติ ฟังค์ชั่นที่สำคัญประการหนึ่งของการประมวลภาพแบบดิจิตอลคือ การแปลงจากภาพสีแดง เขียว น้ำเงิน (RGB Images) ไปเป็นภาพ Gray Scale ซึ่งจะพบว่า จะต้องทำการคำนวณผลคูณระหว่างอาร์เรย์ความเข้มสีแต่ละสีเข้ากับจำนวนจริงที่มีค่าแตกต่างกัน แล้วนำผลการคำนวณที่ได้มารวมกัน ซึ่งจะเป็นการสร้างอาร์เรย์ 2 มิติ 1 อาร์เรย์ขึ้นจากอาร์เรย์ 2 มิติ 3 อาร์เรย์ ซึ่งโปรแกรมได้ถูกออกแบบให้สามารถทำการคำนวณดังกล่าวได้โดยง่าย ดังตัวอย่างต่อไปนี้ ซึ่งเป็นการแปลง RGB Images ไปเป็น Gray scale image ตามมาตรฐาน NTSC อนึ่งจะพบว่า การกำหนดค่าให้กับตัวแปร Gray ซึ่งเป็น object ของคลาส KData โดยใช้เครื่องหมาย = นั้น ตัวแปร Gray จะมีขนาดเท่าใดมาก่อนก็ได้ แต่ด้วยโปรแกรมที่ออกแบบขึ้น จะทำให้ตัวแปร Gray มีการปรับจำนวนแถวและจำนวนหลักได้อย่างอัตโนมัติ ดังนี้

```
KData R(9,9,8),G(9,9,2),B(9,9,10);
KData Gray;
Gray=0.212671*R + 0.715160*G + 0.072169*B;
```

1.5 ฟังค์ชั่นการแปลงจากภาพ Gray Scale ไปเป็นภาพ Binary ซึ่งมีอยู่ได้กัน 2 ฟังค์ชั่นที่มีชื่อซ้ำกัน ซึ่งหากค่า Threshold ที่ป้อนให้กับฟังค์ชั่นนี้มีค่าเดียว โปรแกรมจะทำการแปลงภาพเป็น Binary แบบปกติ คือ ที่ตำแหน่งของภาพตั้งต้นที่มีค่าต่ำกว่าค่า Threshold ที่ตำแหน่งนั้นในภาพขาออกจะถูกกำหนดให้มีค่าเป็นศูนย์หรือเป็นด้านมืดไป และในทางตรงกันข้าม หากมีค่าสูงกว่าหรือเท่ากับค่า Threshold แล้วค่าที่ตำแหน่งเดียวกันของภาพขาออกจะมีค่าเท่ากับ 255 หรือด้านสว่างไป และในกรณีที่มีค่า Threshold 2 ค่าคือ MinT และ MaxT เป็นอาร์กรูเมนต์ของฟังค์ชั่นนี้ตามลำดับ โปรแกรมจะทำการเปรียบเทียบค่าทั้งสองนี้ ซึ่งผลการเปรียบเทียบจะเกิดผลดังต่อไปนี้

1.5.1 กรณีที่ MinT น้อยกว่า MaxT โปรแกรมจะทำการกำหนดให้ภาพขาออกที่ตำแหน่งเดียวกับภาพตั้งต้น ที่มีค่าอยู่ระหว่าง MinT จนถึง MaxT มีค่าเป็นศูนย์หรือเป็นด้านมืดไป ส่วนที่มีค่านอกช่วงดังกล่าวก็จะถูกกำหนดให้มีค่า 255 หรือเป็นด้านสว่างไป

1.5.2 กรณีที่ MinT มากกว่า MaxT ในกรณีนี้โปรแกรมจะกำหนดให้ภาพขาออกที่ตำแหน่งเดียวกับภาพตั้งต้นที่มีค่าอยู่ระหว่าง MinT จนถึง 255 หรือตั้งแต่ 0 ถึง MaxT ให้มีค่าเท่ากับศูนย์หรือเป็นด้านมืดไป ในขณะเดียวกันก็จะกำหนดให้บริเวณอื่นๆที่มีค่านอกเหนือจากนี้ เป็นด้านสว่างไป

1.5.3 กรณีที่ MinT เท่ากับ MaxT ในกรณีนี้โปรแกรมจะทำการแปลงเป็นภาพ Binary ซึ่งให้ผลเทียบเท่ากับฟังค์ชั่นที่มีค่า Threshold เพียงตัวเดียวเป็นอาร์กรูเมนต์
ฟังค์ชั่นเหล่านี้มี Function Prototype ดังต่อไปนี้

```
KData Threshold(BYTE MinT, BYTE MaxT);
KData Threshold(BYTE Val=127);
```

ในโปรแกรมที่ออกแบบขึ้นนั้น จะมีคลาสที่มีความฟังค์ชั่นต่างๆเหมือนกันทุกประการกับคลาส KData ยกเว้นฟังค์ชั่นที่ใช้ในการแปลงเป็น Binary Image นั่นคือคลาส KMatrix ซึ่งจะได้กล่าวถึงรายละเอียดในหัวข้อต่อไป

2. KBMPInterface คลาสนี้ทำหน้าที่ในการโหลดข้อมูลที่อยู่ในไฟล์ภาพที่มีนามสกุล .bmp มาไว้ใน object ของคลาส KData นอกจากนั้น ยังทำหน้าที่บันทึกค่าที่อยู่ใน object ของ KData ลงไปในไฟล์ภาพที่มีนามสกุล .bmp ได้อีกด้วย ซึ่งการโหลดข้อมูลนั้น สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น LoadFromBMPFile และฟังค์ชั่น GetData ต่อเนื่องกัน โดยตัวแรกจะเป็นการเปิดไฟล์ภาพและอ่านข้อมูลเข้ามาในตัวโปรแกรม และตัวต่อมาจะทำหน้าที่จัดเรียงข้อมูลของ

อาร์เรย์ความเข้มสี 3 อาร์เรย์ลงไปในอาร์กรูเมนต์ของฟังค์ชั่น ซึ่งฟังค์ชั่นทั้งสองนั้นมี Function Prototype ดังนี้

```
void LoadFromBMPFile(AnsiString FileName);
void GetData(KData& R, KData& G, KData& B);
```

สำหรับการบันทึกข้อมูลของ object ของคลาส KData ลงไปในไฟล์ภาพนั้น สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น SetData และฟังค์ชั่น SaveToBMPFile ต่อเนื่องกัน โดยฟังค์ชั่นตัวแรกนั้นจะเป็นการทำสำเนาของ object ต่างๆ ที่ใส่เข้าไปเป็นอาร์กรูเมนต์ของฟังค์ชั่น SetData และฟังค์ชั่นตัวที่สอง จึงเป็นการเขียนข้อมูลที่จัดเรียงแล้วให้อยู่ในไฟล์ภาพตามชื่อไฟล์ที่ต้องการ สำหรับ Function Prototype ของทั้ง 2 ฟังค์ชั่นนั้น มีดังนี้

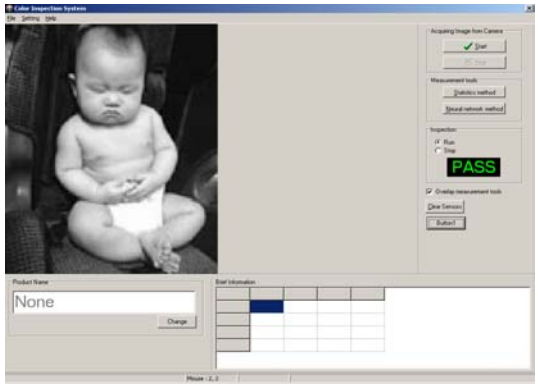
```
void SetData(KData& R, KData& G, KData& B);
void SaveToBMPFile(AnsiString FileName);
```

3. KDrawer ฟังค์ชั่นตัวนี้ถูกออกแบบขึ้น เพื่อทำให้การแสดงผลข้อมูลภาพที่อยู่ใน object ของคลาส KData สามารถทำได้โดยง่าย ซึ่งจะสามารถแสดงผลภาพได้ทั้งภาพสีที่เกิดการรวมกันของอาร์เรย์ความเข้มสี 3 อาร์เรย์ หรือ Gray Scale Image ที่เกิดจากอาเรย์ 2 มิติเพียงตัวเดียวเท่านั้น สำหรับการแสดงผลภาพนั้น จะเป็นการวาดลงไปในคอนโทรล TImage ซึ่งเป็นคอมโพแนนต์ที่มีอยู่ใน Borland C++ Builder เท่านั้น ซึ่งจะต้องเรียกฟังค์ชั่น SetData และ DrawOn ต่อเนื่องกัน โดยฟังค์ชั่นแรกจะมี object ของคลาส KData เป็นอาร์กรูเมนต์ของฟังค์ชั่น ซึ่งอาจจะมีเพียงตัวเดียว หรือ 3 ตัว ขึ้นอยู่ว่าต้องการแสดงเป็นภาพ Gray Scale หรือภาพสี สำหรับ Function Prototype ของฟังค์ชั่นเหล่านี้ มีดังต่อไปนี้

```
void SetData(KData& R, KData& G, KData& B);
void SetData(KData& Gray);
void DrawOn(TImage * M);
```

ตัวอย่างต่อไปนี้ แสดงการทำงานร่วมกันระหว่าง object ของคลาส KData, KBMPInterface และของคลาส KDrawer ซึ่งเป็นการเปิดไฟล์ภาพที่ชื่อว่า boy.bmp แล้วทำการแปลงให้เป็นภาพ Gary Scale และแสดงผลที่ได้ใน Image1 ซึ่งผลที่ได้จะเป็นดังในภาพที่ จ-2

```
KData R,G,B,Gray;
KBMPInterface Loader;
KDrawer Drawer;
Loader.LoadFromBMPFile("c:\\boy.bmp");
Loader.GetData(R,G,B);
Gray=0.212671*R + 0.715160*G + 0.072169*B;
Drawer.SetData(Gray);
Drawer.DrawOn(Image1);
```



ภาพที่ จ-2 ผลที่ได้จากการทำงานของโปรแกรมข้างต้น

4. KPoint และ KRectangle

4.1 คลาส KPoint ใช้เก็บพิกัดของจุดใดๆ ที่อยู่ในระบบภาพ ซึ่งจะเป็นการเก็บหมายเลขหลักและหมายเลขแถวของจุดนั้นๆ ไว้ใน object ดังจะเห็นได้จากสมาชิกของคลาสดังนี้

```
int Row;
int Column;
KPoint();
KPoint(int row, int col) {Row=row; Column=col;};
virtual ~KPoint();
```

4.2 คลาส KRectangle มีหน้าที่ในการเก็บระบบพิกัดของสี่เหลี่ยมใดๆ ซึ่งสามารถเก็บ Region Of Interest ซึ่งระบุโดยผู้ใช้ได้อีกด้วย นอกจากนั้นคลาสนี้มีความสามารถในการดึงภาพย่อยออกจากภาพหลักได้อีกด้วย สำหรับฟังค์ชั่นและสมาชิกมีดังนี้

4.2.1 การเก็บพิกัดของสี่เหลี่ยม คลาสนี้จะทำการเก็บเฉพาะจุดที่อยู่ซ้ายบนสุด และขวาล่างสุดลงไปในสมาชิกของคลาสต่อไปนี้

```
KPoint TopLeft;
KPoint BottomRight;
```

4.2.2 ฟังค์ชั่นที่ใช้ในการดึงภาพย่อยออกจากภาพหลัก จะพบว่า สามารถส่งภาพหลักในรูปของของ object หรือ pointer ก็ได้ และสำหรับขนาดพื้นที่ของภาพย่อยที่ถูกดึงออกมานั้นจะขึ้นอยู่กับพิกัดของสี่เหลี่ยมนั่นเอง

```
KData Crop(KData& m1);
KData Crop(KData* m1);
```

5. KRun KBLOB และ KListOfKBLOB

5.1 คลาส KRun นั้นเป็นการเก็บข้อมูลแบบ Run Length Encoding ซึ่งจะทำที่ในการเก็บบริเวณของวัตถุโดยเก็บหมายเลขแถว หมายเลขหลักเริ่มต้น หมายเลขหลักสุดท้ายและหมายเลขวัตถุไว้ในสมาชิกต่อไปนี้

```
int StartColumn;
int StopColumn;
int Row;
int Label;
```

นอกจากนั้น เพื่อทำการเก็บ Object คลาสในลักษณะของ doubly linked list คลาสนี้ยังได้เตรียมตัวแปรที่ชื่อ Prev และ Next ซึ่งเป็น Pointer ไว้ชี้ object ของคลาสเดียวกันอีกด้วย

5.2 คลาส KBLOB คำว่า BLOB นั้น ย่อมาจาก Binary Large OBject ซึ่งประกอบด้วย KRun ในจำนวนที่ไม่ทราบค่ามาก่อน ดังนั้นคลาส KBLOB จึงถูกออกแบบให้เป็น double linked list หรือ ถ้ากล่าวอย่างง่ายๆ ก็คือ KBLOB นั้นถูกออกแบบมาให้เป็นอาร์เรย์แบบปลายเปิดที่สามารถเก็บ KRun ในจำนวนเท่าใดก็ได้ สมาชิกที่สำคัญของคลาสนี้มีดังต่อไปนี้

5.2.1 Constructor และ Destructor สำหรับฟังค์ชั่น 2 ชนิดจะทำการตั้งค่าเริ่มต้น และลบ linked list ที่สร้างขึ้นอย่างอัตโนมัติ เพื่อเป็นการป้องกันการรั่วไหลของหน่วยความจำ

```
KBLOB();
KBLOB(KBLOB& blob1);
virtual ~KBLOB();
```

5.2.2 การหาจำนวน object ของคลาส KRun ที่มีอยู่ใน linked list ที่สร้างขึ้น สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น GetNumberOfKRun ซึ่งมี function prototype ดังนี้

```
int GetNumberOfKRun(void) {return N;};
```

5.2.3 การเพิ่มจำนวน Node หรือการเพิ่มจำนวน Object ของคลาส KRun บน Linked list ที่สร้างขึ้นสามารถทำได้โดยการใช้ตัวกระทำ += ซึ่งมี function prototype ดังนี้

```
KBLOB& operator+=(KRun *run);
KBLOB& operator+=(KRun &run);
```

5.2.4 การเข้าถึงข้อมูล KRun หรือ Node ที่มีอยู่ใน Doubly linked list ที่สร้างขึ้น สามารถเข้าถึงได้ 2 วิธีคือ การเรียกใช้ฟังค์ชั่น Run() ซึ่งจะทำการคืน object ของคลาส KRun ซึ่งเป็นสำเนาของobject จริงที่อยู่ในตำแหน่งเดียวกับหมายเลขที่เป็นอาร์กรูเมนต์ของฟังค์ชั่น และถ้าหากใช้ฟังค์ชั่น RunPnt() โปรแกรมจะคืนค่าของ pointer ของ object ที่อยู่บน linked list ตามหมายเลขที่ใส่ไว้เป็นอาร์กรูเมนต์ของฟังค์ชั่นให้ ซึ่งผู้ใช้สามารถทำการเปลี่ยนแปลงคุณสมบัติต่างๆ ของ object ของ KRun ที่อยู่บน linked list นั้นๆได้โดยผ่าน Pointer นี้

```
KRun* RunPnt(int Index);
KRun  Run(int Index);
```

อนึ่ง เพื่อให้การอ้างอิงตำแหน่งของ KRun ที่อยู่ใน doubly linked list เป็นไปในแนวทางเดียวกันกับของ KVector และ KMatrix ดังนั้น ตำแหน่งแรกจะมีหมายเลขตำแหน่งเป็น 1 เสมอ

5.2.5 การลบ Node หรือการลบ KRun ที่มีอยู่บน Doubly linked list ซึ่งมีฟังค์ชั่นให้เลือกใช้อยู่ 2 อย่าง คือการลบโหนดในตำแหน่งที่ต้องการ โดยการเรียกใช้ฟังค์ชั่น Delete() และระบุหมายเลขตำแหน่งที่ต้องการออกจาก linked list และการลบ Node ทั้งหมดออกจาก Double Linked List ที่สร้างขึ้น โดยการใช้คำสั่ง DeleteAll()

```
bool Delete(int Index);
void DeleteAll(void);
```

5.2.6 การหาค่าคุณสมบัติทางการภาพทั่วๆไป (Features) ของ BLOB นั้นๆ ได้แก่ พื้นที่ ความยาวเส้นรอบรูป จุด Centoid มุมที่ทำกับแกนพิกัด xy ปกติ การหาความกว้างและความยาวของ bounding box ทำได้โดยการเรียกใช้ฟังค์ชั่นต่อไปนี้

```
int     GetArea(void);
int     GetPerimeter(void);
KPoint  GetCenter(void);
double  GetAngle(void);
int     GetBBWidth(void);
int     GetBBHeight(void);
```

5.2.7 การหาคุณสมบัติทางสีของ BLOB แต่ละชิ้น เนื่องจาก BLOB นั้น ได้มาจากการหาบริเวณของวัตถุแต่ละชิ้นจากภาพ Binary ซึ่งจะทำให้เราทราบแค่บริเวณและคุณสมบัติทางกายภาพของวัตถุเท่านั้น แต่จะไม่สามารถทราบสีเฉลี่ยหรือความเข้มแสงเฉลี่ยของวัตถุทั้งชิ้นได้เลย ดังนั้นก่อนที่จะอ่านค่าเฉลี่ยของ Hue, Saturation และ Intensity ของวัตถุแต่ละชิ้นได้ จำเป็นจะต้องส่งข้อมูลภาพสีไม่ว่าจะอยู่ใน RGB Color Space หรือ HSI Color Space ไปให้ object ของคลาส KBLOB คำนวณค่าเฉลี่ยดังกล่าวเสียก่อน ซึ่งหากข้อมูลภาพสีอยู่ใน RGB Color Space ก็สามารถส่งข้อมูลเพื่อไปคำนวณค่าได้โดยการเรียกใช้ฟังค์ชั่น ComputeAverageHSI() และในทางกลับกัน หากข้อมูลสีอยู่ใน HSI Color Space ก็ให้ทำการเรียกใช้ฟังค์ชั่น ExtractAverageHSI ซึ่งมี Function Prototype ดังนี้

```
void ExtractAverageHSI(KData& H, KData& S, KData& I);
void ComputeAverageHSI(KData& Red, KData& Green, KData& Blue);
```

หลังจากนั้นจึงเรียกฟังค์ชั่นต่อไปนี้ เพื่ออ่านค่าคุณสมบัติทางสีของวัตถุแต่ละชิ้น

```
double  GetAverageHue(void)              {return AvH;};
double  GetAverageSaturation(void)       {return AvS;};
double  GetAverageIntensity(void)        {return AvI;};
```

5.2.8 การระบายพื้นที่ของ BLOB ที่ตรวจจับได้ด้วยสีที่ต้องการลงไปใน object ของคลาส KData โดยการใช้ ฟังค์ชั่น FillArea() ซึ่งมี Function Prototype ต่อไปนี้

```
void    FillArea(KData& Image, BYTE Val);
```

และท้ายที่สุด หากต้องการทราบว่า จุดพิกัดแถวหลักใดๆ นั้น เป็นพื้นที่ของ BLOB ที่ตรวจจับได้หรือ สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น Seek ซึ่งจะหน้าที่ค้นหาและคืนค่า true มาให้ถ้าจุดพิกัดนั้น เป็นพื้นที่ของ BLOB นั้นๆ ฟังค์ชั่น Seek นั้นมี Function Prototype ต่อไปนี้

```
bool    Seek(KPoint &Pnt1);
```

5.3 คลาส KListOfKBLOB เนื่องจากโปรแกรมไม่สามารถทราบจำนวนของ BLOB ที่ปรากฏอยู่ในภาพ ทำให้ คลาสตัวนี้ถูกออกแบบให้เป็น Doubly Linked List ที่สามารถเก็บข้อมูลของ object ของคลาส KBLOB ได้อย่างไม่จำกัดจำนวน นอกไปจากนั้น คลาสตัวนี้ยังมีฟังค์ชั่นที่สำคัญคือ Extract_4C_BLOB และ Extract_8C_BLOB ซึ่งจะทำการดึงข้อมูลของ BLOB ที่มีการเชื่อมต่อกันแบบ 4 ทิศทางหรือ 8 ทิศทางออกจากภาพตามลำดับ ซึ่งถ้ามองจากภาพรวมของการทำงานร่วมกันระหว่าง 3 คลาสในกลุ่มนี้ ก็จะเสมือนว่า แท้ที่จริงแล้วคลาส KListOfKBLOB ก็คือ Doubly Linked List ของ Doubly Linked List ของ object ของคลาส KRun นั่นเอง ซึ่งจะพบว่า ถ้ามองผู้ใช้งานคลาสแล้ว จะมีเพียงคลาสนี้เท่านั้นที่จถูกนำไปใช้งานมากที่สุดเนื่องจากตัวคลาสสามารถดึงข้อมูลของ BLOB ออกจากภาพหลักและเก็บรายละเอียดทั้งหมดที่ดึงมาได้ไว้ในตัวมันเองได้นั่นเอง สำหรับ Function Prototype ของคลาส นี้มีดังนี้ อนึ่งเนื่องจากคลาสนี้มีลักษณะเป็น Doubly Linked List จึงมีฟังค์ชั่นการทำงานต่างๆ ที่ค่อนข้างใกล้เคียงกับคลาส KBLOB ซึ่งมีลักษณะเป็น Doubly Linked List เช่นกัน

5.3.1 Constructor และ Destructor สำหรับฟังค์ชั่น 2 ชนิดจะทำการตั้งค่าเริ่มต้น และลบ Linked List ที่สร้างขึ้นอย่างอัตโนมัติ เพื่อเป็นการป้องกันการรั่วไหลของหน่วยความจำ

```
KListOfKBLOB();
virtual ~KListOfKBLOB();
```

5.3.2 การหาจำนวน object ของคลาส KBLOB ที่มีอยู่ใน Linked List ที่สร้างขึ้นสามารถทำได้โดยการเรียกใช้ฟังค์ชั่น GetNumberOfKBLOB ซึ่งมี Function Prototype ดังนี้

```
int GetNumberOfKBLOB(void) {return N;};
```

5.3.3 การเพิ่มจำนวน Node หรือการเพิ่มจำนวน Object ของคลาส KBLOB บน Linked list ที่สร้างขึ้นสามารถทำได้โดยการใช้ตัวกระทำ += ซึ่งมี Function Prototype ดังนี้

```
KListOfKBLOB & operator+=(KBLOB & blob);
KListOfKBLOB & operator+=(KBLOB* blob);
```

5.3.4 การเข้าถึงข้อมูล KBLOB หรือ Node ที่มีอยู่ใน Doubly Linked List ที่สร้างขึ้นสามารถเข้าถึงได้ 2 วิธีคือ การเรียกใช้ฟังค์ชั่น BLOB() ซึ่งจะทำการคืน object ของคลาส KBLOB ซึ่งเป็นสำเนาของ object จริงที่อยู่ในตำแหน่งเดียวกับหมายเลขที่เป็นอาร์กรูเมนต์ของฟังค์ชั่น และถ้าหากใช้ฟังค์ชั่น BLOBPnt() โปรแกรมจะคืนค่าของ pointer ของ object ที่อยู่บน Linked List ตามหมายเลขที่ใส่ไว้เป็นอาร์กรูเมนต์ของฟังค์ชั่นให้ ซึ่งผู้ใช้สามารถทำการเปลี่ยนแปลงคุณสมบัติต่างๆ ของ object ของ KBLOB ที่อยู่บน linked list นั้นๆได้โดยผ่าน Pointer นี้

```
KBLOB* BLOBPnt(int Index);
KBLOB  BLOB(int Index);
```

อนึ่ง เพื่อให้การอ้างอิงตำแหน่งของ KBLOB ที่อยู่ใน Doubly Linked List เป็นไปในแนวทางเดียวกันกับของ KVector และ KMatrix ดังนั้น ตำแหน่งแรกจะมีหมายเลขตำแหน่งเป็น 1 เสมอ

5.3.5 การลบ Node หรือการลบ KBLOB ที่มีอยู่บน Doubly Linked List ซึ่งมีฟังค์ชั่นให้เลือกใช้อยู่ 2 อย่าง คือการลบโหนดในตำแหน่งที่ต้องการ โดยการเรียกใช้ฟังค์ชั่น Delete() และระบุหมายเลขตำแหน่งที่ต้องการออกจาก Linked List และการลบ Node ทั้งหมดออกจาก Doubly Linked List ที่สร้างขึ้น โดยการใช้คำสั่ง DeleteAll()

```
bool Delete(int Index);
void DeleteAll(void);
```

5.3.6 การดึงข้อมูลของ KBLOB ที่อยู่ใน KData จะพบว่า อาร์กรูเมนต์ของฟังค์ชั่นในกลุ่มนี้จะประกอบด้วยภาพ m1 และค่าความเข้มแสงของวัตถุที่สนใจเท่านั้น val ซึ่งในกรณีนี้ ข้อมูลที่อยู่ในภาพหรือ object ของคลาส KData นั้นจะต้องประกอบด้วยค่าเพียง 2 ค่าที่เป็น 0 หรือ 255 เท่านั้น หากต้องการดึงข้อมูลของ KBLOB ที่มีการเชื่อมต่อแบบ 4 ทิศทางก็สามารถ

ทำได้โดยการเรียกใช้ฟังค์ชั่น และ หากต้องการดึงข้อมูลของ KBLOB ที่มีการเชื่อมต่อแบบ 8 ทิศทางก็สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น ซึ่งมี Function Prototype ดังนี้

```
KListOfKBLOB& Extract_4C_BLOB(KData & m1, BYTE Val);
KListOfKBLOB& Extract_8C_BLOB(KData & m1, BYTE Val);
```

ซึ่งหลังจากการเรียกใช้ฟังค์ชั่นตัวใดตัวหนึ่งในกลุ่มนี้ object ของคลาส KListOfKBLOB จะทำการพิจารณาข้อมูลที่อยู่ภาพใย KData และจะสร้าง object ของคลาส KBLOB ซึ่งเก็บข้อมูล BLOB ที่เจอในภาพแบบอัตโนมัติ อย่างไรก็ตามจะพบว่า ข้อมูลของ KBLOB แต่ละชิ้นนั้น จะมีแค่ขนาดทางกายภาพเท่านั้น เนื่องจาก KBLOB นั้นสร้างขึ้นจากภาพ binary ซึ่งหากต้องการทราบคุณสมบัติสีของ KBLOB แต่ละชิ้นนั้น จำเป็นจะต้องเรียกใช้ฟังค์ชั่นใดฟังค์ชั่นหนึ่งต่อไปนี้

```
void AverageHSI(KData& Red, KData& Green, KData& Blue);
void ExtractAverageHSI(KData& H, KData& S, KData& I);
```

ซึ่งการเรียกใช้ฟังค์ชั่นที่อยู่ในกลุ่มนี้ จะทำให้มีเรียกใช้ฟังค์ชั่นชื่อเดียวกันของ object ของคลาส KBLOB ที่ถูกสร้างขึ้นนั่นเอง

และท้ายที่สุด หากต้องการทราบว่า จุดพิกัดแถวหลักใดๆ นั้น เป็นพื้นที่ของ BLOB ชิ้นใดที่อยู่ใน Doubly Linked List เพื่อจะนำไปใช้งานต่อไป สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น Seek ซึ่งจะหน้าที่ค้นหาและคืนค่าหมายเลข BLOB มาให้ถ้าจุดพิกัดนั้น เป็นพื้นที่ของ BLOB นั้นๆ และหากไม่เจอก็จะคืนค่าศูนย์มาให้ ฟังค์ชั่น Seek นั้นมี Function Prototype ต่อไปนี้

```
int Seek(KPoint &Pnt1);
```

6. คลาส KMatrix, KVector, KVectorPair และ KListOfKVectorPair

6.1 คลาส KMatrix การคำนวณต่างๆ ของโครงข่ายประสาทเทียมนั้น ตั้งอยู่บนพื้นฐานของพีชคณิตของเมตริกซ์และเวคเตอร์ อย่างไรก็ตาม เนื่องจากในโปรแกรมภาษา C++ นั้น ไม่มีชนิดตัวแปรที่เป็นเมตริกซ์ ดังนั้นทางผู้วิจัยจึงได้ออกแบบและจัดสร้างคลาสที่ทำหน้าที่นี้ขึ้น ซึ่งเป็นการอำนวยความสะดวกแก่โปรแกรมเมอร์ในการประกาศตัวแปรที่มีลักษณะเป็นอาร์เรย์ 2 มิติของจำนวนจริง และเช่นเดียวกันกับคลาส KData ที่ต้องการให้การอ้างอิงระบบพิกัดเป็นไปในแนวทางเดียวกัน มุมซ้ายบนสุดของอาร์เรย์ 2 มิติของคลาส KMatrix จะมีระบบพิกัดเป็น 1, 1

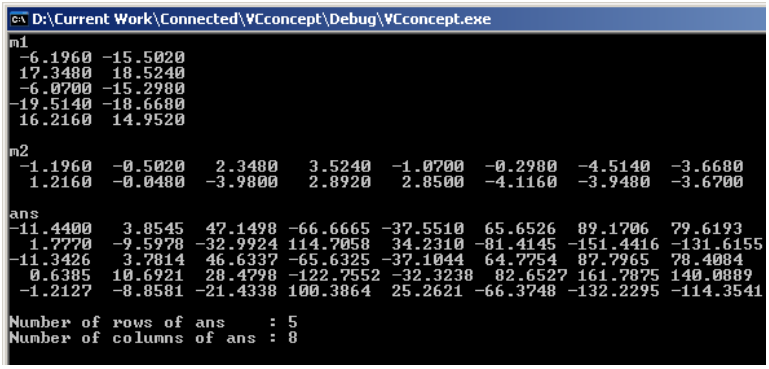
จะพบว่า ฟังค์ชั่นพื้นฐานต่างๆ ที่สามารถเรียกใช้ได้ในคลาส KData ก็สามารถเรียกใช้ได้ในคลาส KMatrix ได้เช่นกัน ทว่าข้อแตกต่างระหว่างคลาส KData และคลาส KMatrix ซึ่งเป็นคลาสที่เก็บข้อมูลของอาร์เรย์ 2 มิติด้วยกันทั้งคู่ คือ

6.1.1 คลาส KData นั้นถูกออกแบบมาเพื่อใช้เก็บข้อมูลที่มีขนาด 1 Byte หรือค่าจำนวนเต็มที่มีค่าระหว่าง 0 ถึง 255 เท่านั้น ในขณะที่คลาส KMatrix นั้นถูกออกแบบมาเพื่อใช้เก็บค่าข้อมูลที่เป็นจำนวนจริง ซึ่งจัดเป็นสิ่งที่จำเป็นในการคำนวณที่เกี่ยวกับโครงข่ายประสาทเทียม

6.1.2 การทำงานของคลาส KData นั้น เป็นการทำงานที่เกี่ยวข้องกับ 2 คลาสอื่น คือ KBMPInterface ที่ทำหน้าที่โหลดและบันทึกค่าข้อมูลที่อยู่ในคลาสเองลงไปในไฟล์ภาพ และ KDisplay ซึ่งทำหน้าที่แสดงผลข้อมูลลงไปในคอนโทรล TImage ซึ่งเมื่อเปรียบเทียบกันแล้ว แต่คลาส KMatrix นั้นเป็นคลาสที่ถูกออกแบบมาเพื่อการคำนวณที่เป็นอาร์เรย์ 2 มิติของจำนวนจริง ที่สามารถทำงานร่วมกับคลาส KVector เช่น การคูณเมตริกซ์เข้ากับเวคเตอร์ ซึ่งเป็นการคำนวณพื้นฐานของระบบโครงข่ายประสาทเทียม นอกจากนั้น ฟังค์ชั่นการคำนวณพื้นฐานของเมตริกซ์ เช่น การกำหนดค่าเริ่มต้นแบบสุ่ม และการ Transpose ก็ได้ถูกออกแบบและจัดสร้างไว้ด้วยเช่นกัน

ตัวอย่างโปรแกรม C++ ต่อไปนี้ แสดงการใช้งานพื้นฐานของคลาส KMatrix ซึ่งแสดงการกำหนดค่าเริ่มต้นแบบสุ่มให้กับตัวแปร m1 และตัวแปร m2 โดยให้ค่าที่อยู่ในตัวแปร m1 มีค่าอยู่ระหว่าง +20 ถึง -20 ในขณะที่ค่าที่อยู่ในตัวแปร m2 นั้นจะมีค่าสุ่มอยู่ระหว่าง +5 ถึง -5 ซึ่งตัวแปร m1 นั้นจะมี 5 แถว 2 หลัก ในขณะที่ตัวแปร m2 มี 8 แถว 8 หลัก หลังจากนั้นก็นำตัวแปรทั้งสองมาคูณกัน แล้วเก็บผลที่ได้ไว้ในตัวแปร ans นอกจากนั้น ในโปรแกรมยังแสดงตัวอย่างการหาจำนวนแถวและหลักของตัวแปร ans อีกด้วย

```
KMatrix m1(5,2), m2(2,8),ans;
m1.Randomize(20);
m2.Randomize(5);
cout<<"m1"<<m1<<endl;
cout<<"m2"<<m2<<endl;
ans=m1*m2;
cout<<"ans"<<ans<<endl;
cout<<"Number of rows of ans    : "<<ans.GetNumberOfRows()<<endl;
cout<<"Number of columns of ans : "<<ans.GetNumberOfColumns()<<endl;
```



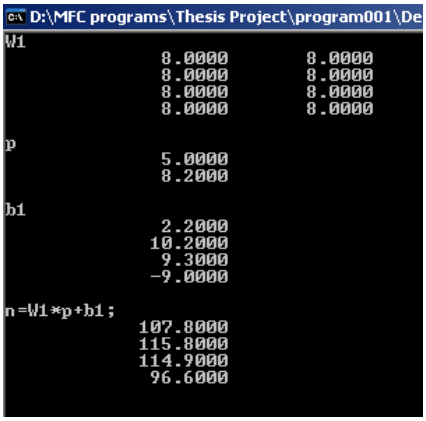
ภาพที่ จ-3 ผลที่ได้จากการทำงานของโปรแกรมข้างต้น

6.2 คลาส KVector ไม่เฉพาะแต่คลาสที่ทำหน้าที่เป็นเมตริกซ์ซึ่งเป็นอาร์เรย์ 2 มิติของจำนวนจริงเท่านั้น ที่ถูกออกแบบและสร้างขึ้นในงานวิจัยนี้ คลาสที่ทำหน้าเหมือนเวคเตอร์หลัก (Column Vector) ซึ่งเป็นอาร์เรย์ 1 มิติของจำนวนจริงนั้นก็ถูกออกแบบและจัดสร้างขึ้นด้วย ถึงแม้ว่าในภาษา C++ เองก็จะมีคลาสมาตรฐานใน Standard Template Library (STL) ที่ทำหน้าที่เป็นเวคเตอร์เช่นกัน อย่างไรก็ตามคลาส “สำเร็จรูป” ที่มีมากับภาษา C++ นั้น ไม่สามารถดำเนินการทางคณิตศาสตร์กับคลาส KMatrix ได้โดยตรง จึงเป็นสาเหตุที่ทำให้ต้องสร้างคลาส KVector ขึ้นมา ซึ่งคลาส KVector นี้ก็จะมีการอ้างอิงระบบพิกัดที่เริ่มต้นด้วยเลข 1 เช่นเดียวกับคลาส KMatrix และคลาส KData ดังแสดงไว้ในสมการต่อไปนี้

$$\begin{bmatrix} v(1) \\ v(2) \\ . \\ . \\ v(n) \end{bmatrix} \qquad \text{(จ-1)}$$

สิ่งที่โดดเด่นของคลาส KVector ที่ออกแบบขึ้นนี้ คือ ความสามารถในการคูณเข้ากับ object ของคลาส KMatrix ซึ่งจัดเป็นพื้นฐานที่สำคัญของการทำงานของโครงข่ายประสาทเทียม ซึ่งสิ่งนี้ที่จะทำให้การทำงานต่างๆ ของโปรแกรมเมอร์ง่ายขึ้น ตัวอย่างต่อไปนี้ แสดงการสร้าง object ของคลาส KMatrix และของคลาส KVector แล้วนำมาคูณเข้าด้วยกันแล้วนำผลการคำนวณที่ได้ไปเก็บไว้ที่ตัวแปร n ซึ่งเป็น object ของคลาส KVector นั่นเอง นอกจากนั้น โปรแกรมยังแสดงการเข้าถึงข้อมูลของ object ของคลาส KVector โดยการใช้เครื่องหมายวงเล็บ และยังแสดงวิธีการอ่านจำนวนแถวของเวคเตอร์หลักให้ทราบอีกด้วย

```
KMatrix W1(4,2,8);
KVector p(2), b1(4), n;
p(1) =5;     p(2) =8.2;
b1(1)=2.2;  b1(2)=10.2;      b1(3)=9.3; b1(4)=-9;
n=W1*p+b1;
cout<<"W1"<<W1<<endl;
cout<<"p"<<p<<endl;
cout<<"b1"<<b1<<endl;
cout<<"n=W1*p+b1;"<<n<<endl;
```



ภาพที่ จ-4 แสดงผลการทำงานของโปรแกรมข้างต้น

6.3 คลาส KVectorPair จากแนวคิดที่ว่า ตัวอย่างที่ใช้ในการสอนโครงข่ายประสาทเทียมนั้น จะมีการป้อนเวคเตอร์เป็นคู่ๆให้กับโครงข่ายประสาทเทียม ดังนั้น คลาสตัวนี้จึงถูกออกแบบและจัดสร้างขึ้น เพื่อใช้เก็บเวคเตอร์หนึ่งคู่ซึ่งประกอบด้วยเวคเตอร์ A และ B ซึ่งโดยทั่วไป ก็จะใช้เพื่อเก็บ object ของคลาส KVector ที่ใช้แทน input vector และ target vector ตามลำดับ สำหรับฟังค์ชั่นที่สำคัญของคลาสตัวนี้มีดังนี้

ฟังค์ชั่นที่ใช้ในการปรับขนาด ซึ่งจะคืนเวคเตอร์ที่ทุกช่องของคู่เวคเตอร์มีค่าระหว่าง 0 ถึง 1 เท่านั้น ทั้งนี้ เนื่องจากการป้อนเวคเตอร์ให้กับโครงข่ายประสาทเทียมนั้น ไม่สามารถป้อนค่าปกติได้ ดังนั้นคู่เวคเตอร์จะต้องถูก Normalize เสียก่อน ซึ่ง Function Prototype ของฟังค์ชั่นนี้เป็นดังต่อไปนี้

```
KVectorPair& Scale(KVectorPair& minvp, KVectorPair& maxvp, bool Scale01=1);
```

ฟังค์ชั่นที่ใช้ในการติดต่อกับไฟล์ข้อมูล จะพบว่า คลาสตัวนี้ถูกออกแบบให้โหลดข้อมูลได้จากทั้งไฟล์และไฟล์สตรีมโดยการใช้ฟังค์ชั่น LoadFromTextFile และ LoadFromTextStream ตามลำดับ นอกจากนั้น คลาสยังถูกออกแบบให้สามารถทำการบันทึกข้อมูลลงไปในไฟล์และไฟล์สตรีมได้ด้วยเช่นกัน ซึ่งสามารถทำได้โดยการเรียกใช้ฟังค์ชั่น SaveToTextFile และ SaveToTextStream ตามลำดับ ซึ่ง Function Prototype ของฟังค์ชั่นเหล่านี้เป็นดังต่อไปนี้

```
void SaveToTextStream(fstream & outFile, bool CheckCode=1);
bool LoadFromTextStream(fstream & inFile,bool CheckCode=1);
void SaveToTextFile(string FileName, bool CheckCode=1);
bool LoadFromTextFile(string FileName, bool CheckCode=1);
```

6.4 คลาส KListOfKVectorPair เนื่องจากคู่เวคเตอร์ที่นำมาใช้เป็น training data ของโครงข่ายประสาทเทียมนั้น มีจำนวนที่ไม่ทราบค่ามาก่อน คลาสตัวนี้จึงถูกออกแบบและสร้างขึ้นเพื่อใช้เก็บข้อมูลของคู่เวคเตอร์ที่ไม่ทราบจำนวนดังกล่าว อนึ่งจะพบว่า คลาสตัวนี้ถูกออกแบบให้มีลักษณะเป็น Doubly Linked List ดังนั้น ฟังค์ชั่นและการเข้าถึงข้อมูลที่อยู่บน Linked List นั้นก็จะคล้ายคลึงกับคลาสตัวอื่นๆที่มีลักษณะกรทำงานเป็น Doubly Linked List เช่นกัน สำหรับฟังค์ชั่นการทำงานของคลาสตัวนี้ มีดังต่อไปนี้

6.4.1 Constructor และ destructor สำหรับฟังค์ชั่น 2 ชนิดจะทำการตั้งค่าเริ่มต้น และลบ Linked List ที่สร้างขึ้นอย่างอัตโนมัติ เพื่อเป็นการป้องกันการรั่วไหลของหน่วยความจำ

```
KListOfKVectorPair();
KListOfKVectorPair(KVectorPair * vp, int Number);
KListOfKVectorPair(KListOfKVectorPair& lvp);
virtual ~KListOfKVectorPair();
```

6.4.2 การหาจำนวน object ของคลาส KVectorPair ที่มีอยู่ใน linked list ที่สร้างขึ้นสามารถทำได้โดยการเรียกใช้ฟังค์ชั่น GetNumberOfVectorPair ซึ่งมี Function prototype ดังนี้

```
int GetNumberOfVectorPair(void)  {return N;}
```

6.4.3 การเพิ่มจำนวน Node หรือการเพิ่มจำนวน Object ของคลาส KVectorPair บน Linked List ที่สร้างขึ้นสามารถทำได้โดยการใช้ตัวกระทำ += ซึ่งมี Function Prototype ดังนี้

```
KListOfKVectorPair& operator+=(KVectorPair *ab);
KListOfKVectorPair& operator+=(KVectorPair ab);
```

6.4.4 การเข้าถึงข้อมูล KVectorPair หรือ Node ที่มีอยู่ใน Doubly linked list ที่สร้างขึ้น สามารถเข้าถึงได้ 2 วิธีคือ การเรียกใช้ฟังค์ชั่น VectorPair () ซึ่งจะทำการคืน object ของคลาส K VectorPair ซึ่งเป็นสำเนาของ object จริงที่อยู่ในตำแหน่งเดียวกับหมายเลขที่เป็นอาร์กรูเมนต์ของฟังค์ชั่น และถ้าหากใช้ฟังค์ชั่น VectorPairPnt() โปรแกรมจะคืนค่าของ pointer ของ object ที่อยู่บน Linked List ตามหมายเลขที่ใส่ไว้เป็นอาร์กรูเมนต์ของฟังค์ชั่นให้ ซึ่งผู้ใช้สามารถทำการเปลี่ยนแปลงคุณสมบัติต่างๆ ของ object ของ KVectorPair ที่อยู่บน Linked List นั้นๆ ได้โดยผ่าน Pointer นี้

```
KVectorPair* VectorPairPnt(int Index);
KVectorPair  VectorPair(int Index);
```

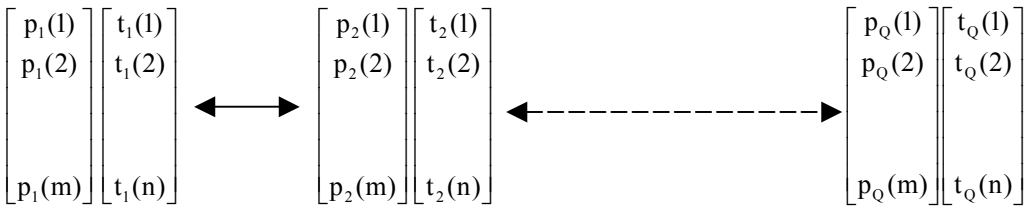
6.4.5 การลบ Node หรือการลบ KVectorPair ที่มีอยู่บน Doubly linked list ซึ่งมีฟังค์ชั่นให้เลือกใช้อยู่ อย่าง คือการลบโหนดในตำแหน่งที่ต้องการ โดยการเรียกใช้ฟังค์ชั่น Delete() และระบุหมายเลขตำแหน่งที่ต้องการออกจาก Linked List และการลบ Node ทั้งหมดออกจาก Doubly Linked List ที่สร้างขึ้น โดยการใช้คำสั่ง DeleteAll()

```
bool Delete(int Index);
void DeleteAll(void);
```

6.4.6 การติดต่อกับไฟล์ข้อมูล หากต้องการบันทึกข้อมูลของคลาส ซึ่งประกอบด้วย object ของคลาส KVectorPair ที่อยู่บน doubly linked list และข้อมูลปลีกย่อยอื่นๆ ลงไปในไฟล์หรือไฟล์สตรีมที่เป็นตัวอักษรนั้น สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น SaveToTextFile หรือ SaveToTextStream ตามลำดับ และในทางตรงกันข้าม หากต้องการอ่านข้อมูลที่อยู่ในไฟล์หรือไฟล์สตรีมก็สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น LoadFromTextFile หรือ LoadFromTextStream ซึ่งฟังค์ชั่นเหล่านี้มี Function Prototype ดังต่อไปนี้

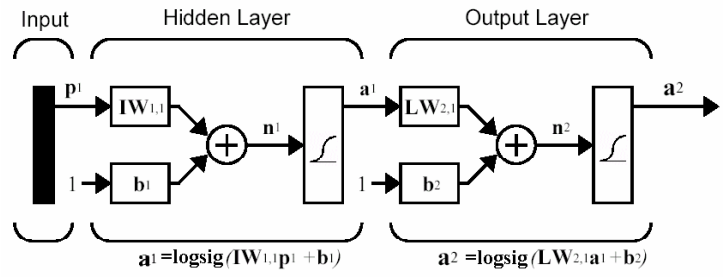
```
void SaveToTextStream(fstream & outFile, bool CheckCode=1);
bool LoadFromTextStream(fstream & inFile, bool CheckCode=1);
void SaveToTextFile(string FileName, bool CheckCode=1);
bool LoadFromTextFile(string FileName, bool CheckCode=1);
```

หากพิจารณาอย่างง่ายๆ แล้วจะได้ว่า



ภาพที่ จ-5 แสดงโครงสร้างของคลาส KListOfKVectorPair

7. คลาส KBPN คลาสตัวนี้เป็นการสร้างโครงข่ายประสาทเทียมแบบ 3 ชั้น ที่ไม่มีการป้อนกลับ (3-layer Feed Forward Artificial Neural Network) ซึ่งมีฟังค์ชั่นของแต่ละชั้นเป็น Log Sigmoid Function ดังแสดงไว้ในภาพที่ จ-6



ภาพที่ จ-6 ระบบโครงข่ายประสาทเทียมที่สร้างขึ้นในคลาส KBPN

สำหรับคลาสตัวนี้เป็นการออกแบบคลาส เพื่อรวบรวมฟังค์ชั่นการทำงานทุกอย่างที่เกี่ยวกับโครงข่ายประสาทเทียม เพื่อให้สามารถทำการ train และใช้งานมันได้อย่างง่ายดาย ซึ่งคลาสตัวนี้ประกอบด้วยฟังค์ชั่นและสมาชิกหลายๆกลุ่มดังนี้

7.1 Constructor และ Destructor จะพบว่า การประกาศตัวแปรประเภทนี้ด้วย default constructor จะไม่สามารถทำได้ ทั้งนี้ เพราะเป็นการออกแบบให้ตัวคลาสทราบขนาดของ input node และ output node จากข้อมูลที่ป้อนให้กับมันนั่นเอง สำหรับ function prototype ของฟังค์ชั่นกลุ่มนี้เป็นดังนี้

```
KBPN(KListOfKVectorPair& TrainData, int NumberOfHidden=10, bool IsBatch=0);
KBPN(KBPN& net);
KBPN(string FileName, bool CheckCode=1);
virtual ~KBPN();
```

7.2 กลุ่มพารามิเตอร์ที่ไม่ทำให้โครงสร้างของโครงข่ายประสาทเทียมที่มีอยู่เกิดการเปลี่ยนแปลง ซึ่งผู้ใช้สามารถทำการเปลี่ยนแปลงและอ่านค่าได้ตลอดเวลาได้ตลอดเวลา สมาชิกของกลุ่มนี้ได้แก่

Learning rate ของ hidden layer (Alpha1)
Learning rate ของ output layer (Alpha2)
จำนวนรอบสูงสุดของการ train (MaxEpoch)
Momentum coefficient ของ hidden layer (Gramma1)
Momentum coefficient ของ output layer (Gramma2)

ชนิดของข้อมูลของสมาชิกในกลุ่มนี้เป็นดังต่อไปนี้

```
double Alpha1, Alpha2;
double Gramma1, Gramma2;
unsigned int MaxEpoch;
```

7.3 กลุ่มฟังค์ชั่นที่ใช้อ่านค่าคุณสมบัติของโครงข่าย ซึ่งจะคืนค่า true ให้หากเงื่อนไขที่เป็นชื่อฟังค์ชั่นนั้นเป็นจริง ฟังค์ชั่นเหล่านี้ได้แก่
IsTrained() ใช้เพื่อตรวจดูว่าโครงข่ายประสาทเทียมที่สร้างขึ้นนั้น โดน train ไปบ้างแล้วหรือยัง
IsDataCorrect() ใช้ตรวจสอบว่าขนาดของคู่เวคเตอร์ที่ใส่เป็นอาร์กรูเมนต์ของฟังค์ชั่นนั้นตรงกับจำนวน node ที่อยู่ทางด้าน input และ จำนวน node ที่อยู่ทาง output หรือไม่
IsBatchMode() การ train โครงข่ายประสาทเทียมเป็นแบบ Batch mode หรือไม่
GetNumberOfHiddenNod() ใช้เพื่ออ่านค่าจำนวน Node ที่มีอยู่ใน hidden layer

สำหรับ Function prototype ของฟังค์ชั่นเหล่านี้ เป็นดังต่อไปนี้

```
bool IsTrained(void)    {return Trained;}
bool IsDataCorrect(KVectorPair & vp);
bool IsDataCorrect(KListOfKVectorPair& Data);
bool IsBatchMode(void);
int GetNumberOfHiddenNode(void);
```

7.4 กลุ่มฟังค์ชั่นที่ทำให้โครงสร้างของโครงข่ายประสาทเทียมเกิดการเปลี่ยนแปลง

```
void SetBatchMode(bool IsBatch=0);
void SetNumberOfHiddenNode(int NumberOfHidden=10);
bool SetTrainingData(KListOfKVectorPair& TrainData);
```

7.5 ฟังค์ชั่นหลักของคลาส ซึ่งเป็นฟังค์ชั่นที่ใช้งานโครงข่ายประสาทเทียมโดยตรง

```
void Train(bool &Run,  TMemo * M=NULL);
KVector Recall(KVector& v);
KMatrix Test(KListOfKVectorPair& TestData);
```

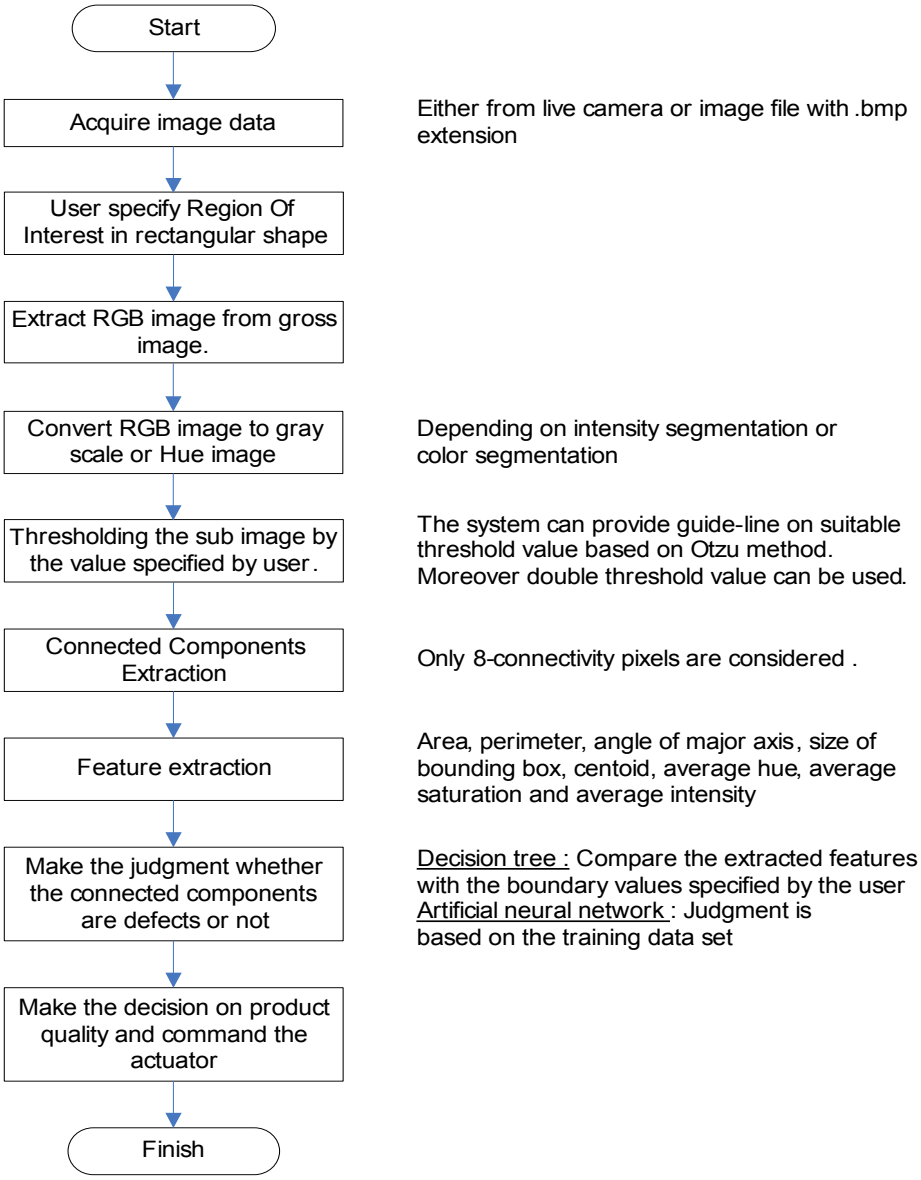
7.6 การติดต่อกับไฟล์ข้อมูล หากต้องการบันทึกข้อมูลของคลาส ซึ่งประกอบด้วยค่าพารามิเตอร์ต่างๆของโครงข่ายประสาทเทียม เช่นค่าของ Weight matrix, bias vector, learning rate, momentum coeeficient และอื่นๆ สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น SaveToTextFile หรือ SaveToTextStream ตามลำดับ และในทางตรงกันข้าม หากต้องการอ่านข้อมูลที่อยู่ในไฟล์หรือไฟล์สตรีมก็สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น LoadFromTextFile หรือ LoadFromTextStream ซึ่งฟังค์ชั่นเหล่านี้มี function prototype ดังต่อไปนี้

```
void SaveToTextStream(fstream & outFile, bool CheckCode=1);
bool LoadFromTextStream(fstream & inFile, bool CheckCode=1);
void SaveToTextFile(string FileName, bool CheckCode=1);
bool LoadFromTextFile(string FileName, bool CheckCode=1);
```

สำหรับการ train โครงข่ายประสาทเทียมด้วยข้อมูลที่อยู่ไฟล์ข้อมูลนั้น สามารถทำได้โดยง่าย ดังตัวอย่างต่อไปนี้

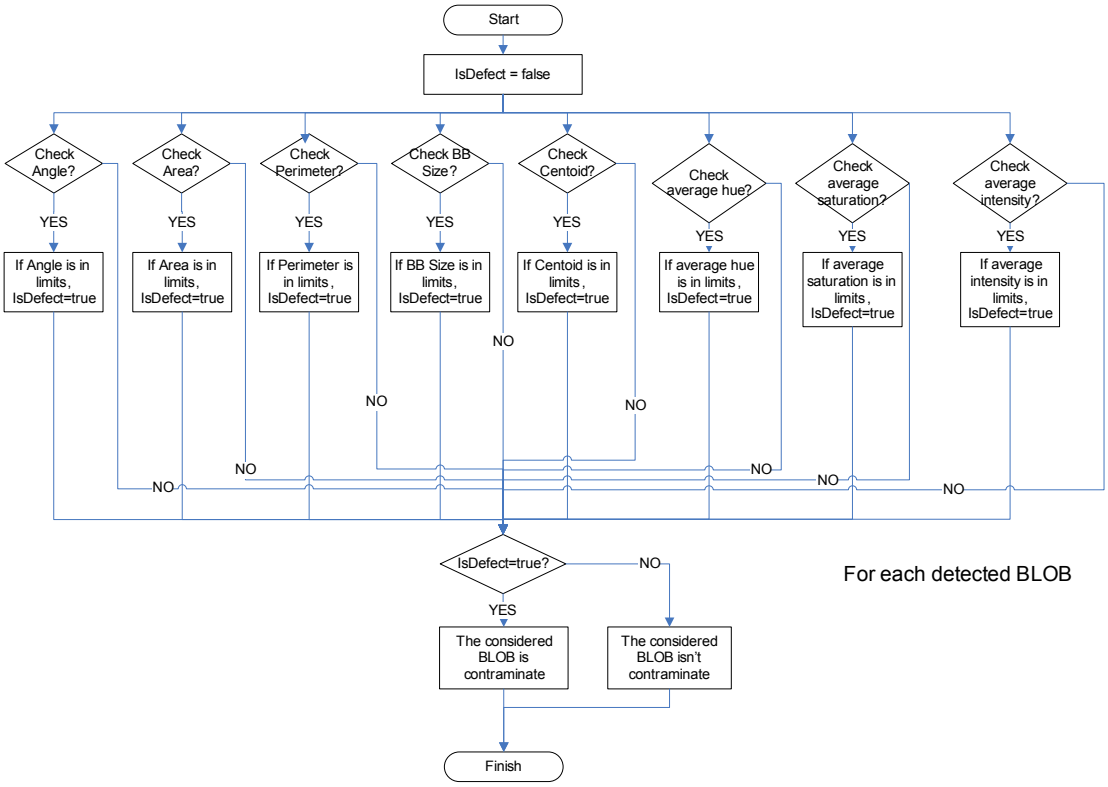
```
KListOfKVectorPair list;
list.LoadFromTextFile("iris.txt",0);
KFFNetwork *net01=new KFFNetwork(list, 8,1);
net01->Train(1);
getch();
delete       net01;
```

8. KStateBLOB และ KNeuralBLOB คลาส 2 ตัวนี้ จัดเป็นหัวใจที่สำคัญของงานวิจัยชิ้นนี้ เนื่องจากคลาสทั้งสองนั้น ทำหน้าที่คำนวณผลและควบคุมการตัดสินใจต่างๆ ซึ่งจะพบว่า การทั้งงานของคลาสทั้งสองตัวจะมีความคล้ายคลึงกัน ซึ่งหมายรวมถึง ชื่อและการเรียกใช้งานฟังค์ชั่นต่างๆ ด้วย อย่างไรก็ตาม จุดที่แตกต่างกันอย่างเห็นได้อย่างชัดเจนระหว่าง 2 คลาสนี้ คือ วิธีการตัดสินใจว่า Connected Component หรือที่มีอีกชื่อหนึ่งว่า BLOB ชิ้นใดบ้างที่เจอในภาพและจัดเป็นพื้นผิวผิดปกติของชิ้นงาน ซึ่งสามารถอธิบายได้จากในภาพที่ จ-7



ภาพที่ จ-7 แสดงการทำงานของคลาส KStateBLOB และคลาส KNeuralBLOB

ในตอนเริ่มแรกนั้น โปรแกรมจะทำการดึงข้อมูลภาพสีไม่ว่าจะมาจากกล้องหรือจากไฟล์ภาพที่เก็บไว้ในคอมพิวเตอร์ และแสดงภาพขึ้นมา เพื่อให้ผู้ใช้ทำการระบุบริเวณที่สนใจ ทั้งนี้เนื่องจากบริเวณของผลิตภัณฑ์อาจจะไม่คลอบคลุมทั้งบริเวณภาพ หลังจากนั้น โปรแกรมจะทำการแยกข้อมูลภาพเฉพาะบริเวณที่ระบุโดยผู้ใช้ว่า เป็นบริเวณที่จะต้องทำการตรวจสอบ พร้อมทั้งแปลงให้เป็นภาพ Gray Scale หรือ Hue Image ขึ้นอยู่กับว่า ผู้ใช้ต้องการแยกบริเวณ (Segmentation) โดยความเข้มแสงหรือความเข้มสี ตามลำดับ และทำการแปลงให้เป็นภาพที่มีสองระดับ (Binary Image) โดยใช้ค่า Threshold ที่ระบุโดยผู้ใช้ ซึ่งโปรแกรมสามารถช่วยระบุค่า Threshold ที่เหมาะสมให้กับผู้ใช้ เพื่อให้ผู้ใช้ช่วยประกอบการตัดสินใจได้ด้วย สำหรับค่า Threshold ที่ระบุโดยโปรแกรมนั้น ได้มาจากการเลือกค่า Threshold โดยวิธีของ Otsu ซึ่งเป็นวิธีการเลือกค่า Threshold แบบอัตโนมัติที่จัดว่า มีการนำไปใช้อย่างแพร่หลายมากที่สุด หลังจากที่ได้ภาพที่มีเพียงสองระดับแล้ว โปรแกรมจะทำการกระบวนการ Connected Components Extraction หรือ Connected Components Labeling เพื่อทำการตรวจจับ BLOB ที่ปรากฏขึ้นในภาพ ซึ่งโปรแกรมจะทำการพิจารณาพิกเซลที่มีการเชื่อมต่อแบบ 8 ทิศทางเท่านั้น และจากขั้นตอนนี้จะทำให้โปรแกรมทราบว่า มี BLOB จำนวนเท่าใดอยู่ในภาพ นอกจากนั้น จะทำให้โปรแกรมทราบว่า BLOB ที่ตรวจจับได้นั้นครอบคลุมพิกัดของพิกเซลใดบ้าง หลังจากนั้นโปรแกรมจะทำการตรวจวัดคุณสมบัติต่างๆของ BLOB ที่ตรวจจับได้ ซึ่งได้แก่ มุมของแกนหลักของรอยด่างแต่ละชิ้น พื้นที่ เส้นรอบรูป ขนาดของ Bounding box ตำแหน่งของจุดศูนย์กลางของรอยด่าง นอกจากนั้น โปรแกรมยังทำการวัดคุณสมบัติของสีของรอยด่างแต่ละชิ้นด้วย ซึ่งคุณสมบัติของสีที่ทำการตรวจวัด คือ เฉดสีเฉลี่ย ค่า Saturation เฉลี่ย ค่าสว่างเฉลี่ยของรอยด่างแต่ละชิ้น จากคุณสมบัติต่างๆของรอยด่างแต่ละชิ้น หากเป็นการทำงานภายในของคลาส KStateBLOB นั้น คุณสมบัติของรอยด่างแต่ละชิ้นจะถูกนำไปเปรียบเทียบกับลักษณะของพื้นผิวงานเสีย ซึ่งถูกกำหนดโดยผู้ใช้ และหากมีชิ้นใดที่มีคุณสมบัติตรงกับคุณสมบัติของพื้นผิวงานเสียแม้เพียงชิ้นเดียว ผืนผ้าบริเวณนั้นจะถูกตัดสินว่าเป็นงานเสียทันที สำหรับการเปรียบเทียบดังกล่าว ซึ่งเป็นการทำงานของคลาส KStateBLOB นั้น สามารถอธิบายการทำงานได้ดังในรูปต่อไปนี้



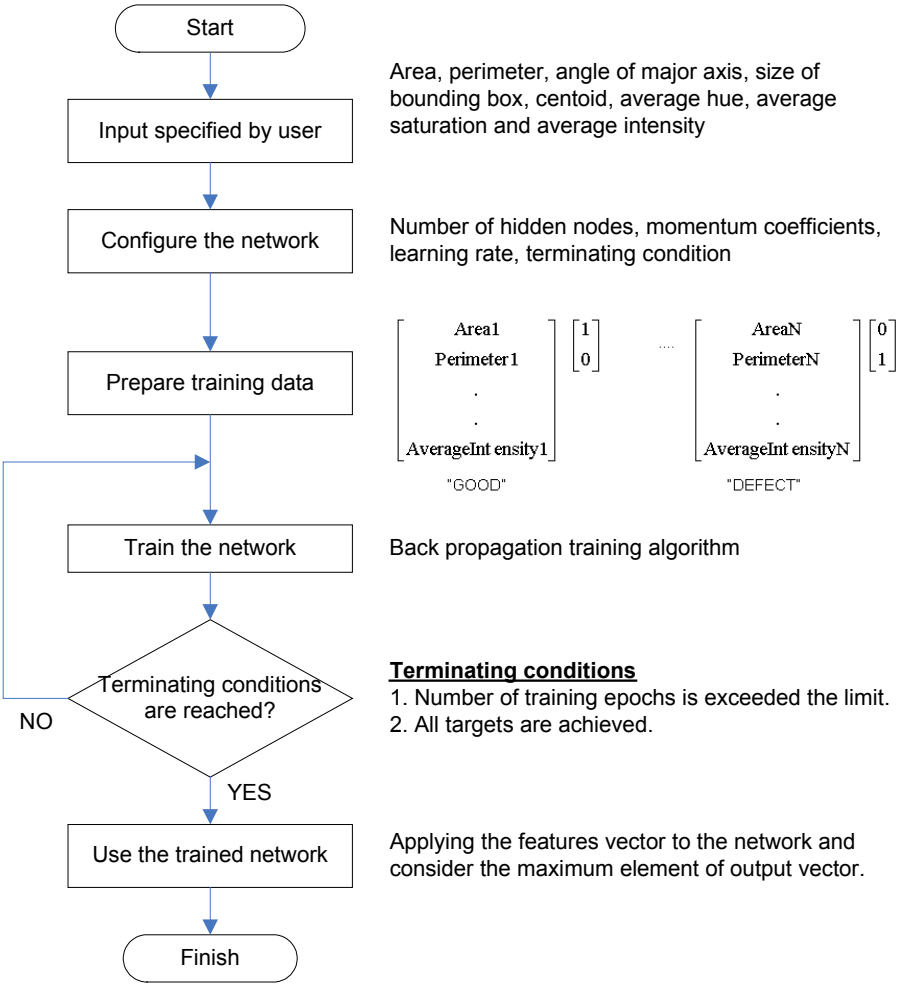
ภาพที่ จ-8 แสดงการทำงานของคลาส KStateBLOB

ซึ่งจะทำการเปรียบเทียบคุณสมบัติของ BLOB แต่ละชิ้น กับคุณสมบัติของพื้นผิวงานเสียซึ่งเป็นค่าที่กำหนดมาจากผู้ใช้งานโปรแกรม ถ้าหากมี BLOB เพียงชิ้นเดียวที่มีลักษณะตรงกับพื้นผิวงานเสีย ผืนผ้าช่วงนั้นจะถูกตัดสินให้เป็นงานเสียทันที

สำหรับการทำงานของคลาส KNeuralBLOB นั้น นอกจากวิธีการตัดสินใจว่า BLOB ที่ตรวจจับได้จัดเป็นพื้นผิวงานเสียหรือไม่ ฟังค์ชั่นการทำงานที่เหลือ ต่างก็คล้ายคลึงกับการทำงานของคลาส KStateBLOB ทุกประการ ซึ่งวิธีการตัดสินใจของ KNeuralBLOB นั้น ทำโดยการใช้โครงข่ายประสาทเทียม (Artificial neural network) ที่ถูก Train โดยการใช้กลุ่มตัวอย่าง BLOB ที่จัดเป็นพื้นผิวดี หรือเสีย ซึ่งเป็นข้อมูลที่ผู้ใช้สามารถกำหนดให้ สำหรับหน้าที่ในการจัดเตรียมข้อมูลที่ใช้ในการ train โครงข่ายประสาทเทียม การตั้งค่าเริ่มต้นต่างๆ และการติดต่อกับฟอร์มและปุ่มกดต่างๆ ก็ทำในคลาสตัวนี้ ดังแสดงไว้ในรูปที่ ซึ่งเป็นการ train และใช้งานโครงข่ายประสาทเทียมหรืออีกนัยหนึ่งคลาส KBPN โดยคลาสตัวนี้ และสามารถอธิบายได้ดังต่อไปนี้

ในตอนเริ่มแรก คลาส KNeuralBLOB จะทำการตั้งค่าเริ่มต้นต่างๆ ให้กับคลาส KBPN เสียก่อน หลังจากนั้น โดยการกำหนดว่า BLOB ตัวใดเป็นพื้นผิวดีหรือเสีย ซึ่งผู้ใช้สามารถทำการคลิ๊กลงไปที่ภาพที่อยู่บนฟอร์มที่ใช้ติดต่อกับผู้ใช้ได้เลย หลังจากนั้นคุณสมบัติ (Feature) ของ BLOB แต่ละชิ้นจะถูกนำมาทำเป็น input vector ซึ่งขนาดของ input vector ดังกล่าวจะขึ้นกับจำนวน

Feature ที่ผู้ใช้กำหนดไว้นั่นเอง นอกจากนั้น โปรแกรมจะทำการสร้าง target vector ตามที่ผู้ใช้กำหนด โดย target vector ในกรณีนี้จะเป็นเวคเตอร์ 2 มิติซึ่งถ้าเป็น target vector ของ BLOB ที่เป็นพื้นผิวดีจะมีแถวแรกเป็น 1 ในขณะที่แถวที่ 2 เป็นศูนย์ และในทางกลับกันหากเป็น target vector ของ BLOB ที่เป็นพื้นผิวเสีย จะมีแถวที่ 1 ของเวคเตอร์เป็นศูนย์และแถวที่ 2 เป็นหนึ่ง คู่เวคเตอร์ของ Training data เหล่านี้จะถูกเก็บไว้ใน KListOfKVectorPair ซึ่งจะส่งไปเป็นอาร์กรูเมนต์ของ KBPN เพื่อใช้ในการ train โครงข่ายประสาทเทียมอีกทีหนึ่ง หลังจากที่โครงข่ายถูก train แล้ว ก็จะสามารถนำไปใช้เพื่อจำแนกว่า BLOB ชิ้นใดเป็นพื้นผิวเสียหรือดีได้ ซึ่งการตัดสินใจของโครงข่ายประสาทเทียมดังกล่าวนั้น จะขึ้นอยู่กับลักษณะของกลุ่มตัวอย่างที่ใช้ในการ train โครงข่ายประสาทเทียมนั่นเอง



ภาพที่ จ-9 การใช้งานโครงข่ายประสาทเทียมหรือคลาส KBPN เพื่อใช้จำแนกว่า BLOB ชิ้นใดจัดเป็นพื้นผิวเสีย ซึ่งมีคลาส KNeuralBLOB คอยทำหน้าควบคุมการทำงานทั้งหมด

สำหรับฟังค์ชั่นการทำงานของคลาสที่มีความคล้ายคลึงกัน ดังนั้น จะขออธิบายฟังค์ชั่นของคลาสทั้งสองไปพร้อมๆกันดังนี้

8.1 Constructor และ Destructor ฟังค์ชั่นกลุ่มนี้จะถูกออกแบบมาให้เป็นฟังค์ชั่น default เท่านั้น ซึ่งผู้ใช้งานคลาสจะไม่สามารถกำหนดค่าคุณสมบัติใดๆของคลาสได้โดยฟังค์ชั่นในกลุ่มนี้ได้เนื่องจากการออกแบบฟังค์ชั่นเพื่อทำงานในลักษณะดังกล่าวนั้น ไม่เหมาะสมในทางปฏิบัติ เพราะค่าคุณสมบัติต่างๆของคลาสนั้นมีอยู่มากมายหลายประการด้วยกัน ซึ่งไม่สะดวกที่จะใช้โปรแกรมเป็นการกำหนดค่า อีกทั้งการกำหนดค่าคุณสมบัติบางอย่างนั้น ผู้ใช้ต้องการที่จะได้เห็นผลการทำงานของคลาสตามค่าที่เปลี่ยนแปลงใหม่อย่างทันทีทันใด ดังนั้นฟังค์ชั่นในกลุ่มนี้จึงถูกออกแบบมาไม่ให้รับค่าใดๆ ทั้งสิ้น สำหรับตัวอย่างฟังค์ชั่นในกลุ่มนี้ที่ยกมากคือ ฟังค์ชั่น

```
KStateBLOB();
virtual ~KStateBLOB();
```

8.2 ฟังค์ชั่นที่ใช้กำหนดบริเวณที่สนใจ (Region Of Interest) ดังที่ได้ทราบแล้วว่า ภาพที่ได้มาจากกล้องนั้น ไม่ได้ประกอบด้วยผืนผ้าทั้งภาพเท่านั้น แต่จะประกอบไปด้วยฉากหลัง ซึ่งเป็นสิ่งที่โปรแกรมไม่จำเป็นต้องให้ความสนใจ ดังนั้น ผู้ใช้งานคลาสสามารถกำหนด ROI ได้โดยตัวเอง ด้วยการเรียกใช้ฟังค์ชั่น SetRectangle () ซึ่งจะจะมีพิกัดของมุมบนซ้ายและล่างขวาของพื้นที่สี่เหลี่ยมที่ต้องการให้คลาสทำการตรวจสอบรอยด่าง นอกจากนั้น หลังจากที่เรียกใช้ฟังค์ชั่นนี้แล้ว ตัวคลาสจะกำหนดให้ค่าตัวแปร ReadyToMeasure มีค่าเป็นจริง ซึ่งแสดงว่าคลาสมีข้อมูลต่างๆ พร้อมที่จำทำการวัดหาคุณสมบัติและตัดสินคุณภาพของผืนผ้าแล้ว สำหรับการอ่านค่าตัวแปรดังกล่าวออกมานั้นจะสามารถทำได้โดยฟังค์ชั่น IsReadyToMeasure ซึ่งจะคืนค่าตรรกะของตัวแปรตัวนี้นั่นเอง สำหรับ function prototype ของฟังค์ชั่นทั้งสองมีดังต่อไปนี้

```
void SetRectangle(KPoint TopLeft, KPoint BottomRight);
bool IsReadyToMeasure(void) {return ReadyToMeasure;};
```

8.3 ฟังค์ชั่นที่สั่งให้ทำการวัด ชื่อ Measure ซึ่งเป็นข้อสังเกตว่า อาร์กรูเมนต์ของฟังค์ชั่นนี้จะประกอบด้วยข้อมูลความเข้มสีทั้งสาม คือ แดง เขียว น้ำเงินนั่นเอง ซึ่งหลังจากการเรียกใช้ฟังค์ชั่นนี้แล้ว โปรแกรมจะทำการวัด และคำนวณต่างๆ และท้ายที่สุดจะทำการตัดสินว่า ผืนผ้าที่อยู่ในภาพ ณ ขณะนั้น เป็นงานเสียหรืองานดี ซึ่งผู้ใช้งานคลาสจะสามารถทราบผลการตัดสินใจดังกล่าว ได้โดยการเรียกใช้ฟังค์ชั่น GetJudgement ซึ่งจะคืนค่า true ให้กับผู้ใช้หากว่าเป็นงานดี และในทางกลับกันฟังค์ชั่นนี้จะคืนค่า false ให้กับผู้ใช้หากว่าเป็นงานเสีย

ซึ่งจะพบว่า หากเริ่มสร้าง object ของคลาสในตอนเริ่มแรกแล้วสั่งให้คลาสทำการวัดภาพเลย จะพบว่า เงื่อนไขการตัดสินว่าเป็นผืนผ้าดีหรือเสียนั้น จะขึ้นอยู่กับค่าเริ่มต้น ซึ่งเป็นสิ่งที่ไม่ถูกต้อง เนื่องจาก งานแต่ละชนิดไม่สามารถใช้เงื่อนไขการตรวจสอบเดียวกันได้ ดังนั้นเพื่อให้ผู้ใช้สามารถกำหนดเงื่อนไขการตรวจสอบต่างๆ ได้ด้วยตัวเอง อีกทั้งสำหรับกรณีของคลาส KNeuralBLOB นั้นผู้ใช้ยังสามารถกำหนดข้อมูล Training data สำหรับโครงข่ายประสาทเทียมได้อีกด้วย ตัวคลาสในกลุ่มนี้จึงถูกออกแบบให้มีฟังค์ชั่น ShowParameterWindow ซึ่งจะทำการแสดงผลการวัดรวมทั้งค่าเงื่อนไขการตรวจสอบต่างๆ ซึ่งผู้ใช้สามารถทำการเปลี่ยนแปลงได้ ซึ่งการแสดงผลดังกล่าวจะอยู่ในรูปฟอร์มที่ผู้สามารถทำการเปลี่ยนแปลงค่าใดๆ ก็ได้ ซึ่งตัวคลาสจะจดจำค่าใหม่ที่กำหนดไว้ให้ได้เสมอ สำหรับ Function Prototype ของฟังค์ชั่นทั้งสามมีดังนี้

```
void Measure(KData& Red, KData& Green, KData& Blue);
bool GetJudgement(void){return Judgement;};
bool ShowParameterWindow(void);
```

8.4 ฟังค์ชั่นที่ใช้ติดต่อกับ main program การแสดงข้อมูลหรือผลการวัดต่างๆ ของคลาส KStateBLOB และ KNeuralBLOB นั้น สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น DrawMeOn ซึ่งจะทำหน้าที่วาดกรอบซึ่งแสดงพื้นที่ๆคลาสทำการตรวจสอบ อีกทั้งยังทำการวาดกรอบสี่เหลี่ยมล้อมรอบ BLOB ที่ตรวจจับได้อีกด้วย นอกจากนั้น ยกสามารถให้คลาสแสดงข้อมูลการวัดต่างๆ ออกมาในรูปของตัวหนังสือในตารางโดยการใช้ฟังค์ชั่น ShowInformation ซึ่ง Function Prototype ของฟังค์ชั่นทั้งสองมีดังต่อไปนี้

```
void DrawMeOn(TImage* Image1);
void ShowInformation(TStringGrid * Grid);
```

8.5 ฟังค์ชั่นที่ใช้ติดต่อกับไฟล์ข้อมูล เนื่องจากข้อมูลที่คลาส KStateBLOB และคลาส KNeuralBLOB มีอยู่นั้นสามารถนำไปใช้กับการตรวจสอบผืนผ้าชนิดเดียวกัน ในเครื่องจักรอื่นๆ ได้ ดังนั้น ฟังค์ชั่นที่ใช้เพื่อบันทึกข้อมูลการวัด รวมทั้งเงื่อนไขการตรวจสอบ พร้อมทั้งฟังค์ชั่นที่ใช้อ่านค่าเหล่านั้นออกมา ก็ถูกออกแบบและจัดสร้างไว้ในคลาสทั้งสองนี้ด้วย ซึ่งจะพบว่า หากต้องการบันทึกข้อมูลของคลาสทั้งสองลงไปในไฟล์สตรีมที่เป็นตัวอักษรนั้น สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น SaveToFileStream และในทางตรงกันข้าม หากต้องการอ่านข้อมูลที่อยู่ในไฟล์สตรีมก็สามารถทำได้โดยการเรียกใช้ฟังค์ชั่น LoadFromFileStream ซึ่งฟังค์ชั่นทั้งสองมี Function Prototype ดังต่อไปนี้

```
void SaveToFileStream(fstream& file);
bool LoadFromFileStream(fstream& file);
```

ประวัติผู้วิจัย

ชื่อ : นายณัฐวัชร มาลัย

ชื่อวิทยานิพนธ์ : การวิเคราะห์และตรวจสอบความบกพร่องของผ้าย้อมด้วยโครงข่ายประสาทเทียม

สาขาวิชา : วิศวกรรมเครื่องกล

ประวัติ

ประวัติส่วนตัว เกิดเมื่อวันที่ 5 สิงหาคม 2523 จังหวัดที่เกิด กรุงเทพมหานคร

ประวัติการศึกษา ปริญญาตรี วิศวกรรมศาสตรบัณฑิต ภาควิศวกรรมเครื่องกล สถาบันเทคโนโลยีนานาชาติสิรินธร มหาวิทยาลัยธรรมศาสตร์ สำเร็จการศึกษาเมื่อปี 2545

ประวัติการทำงาน วิศวกร บริษัทเอกชนแห่งหนึ่ง

สถานที่ติดต่อ 159/41 หมู่ 2 ซ.ติวานนท์ 48 ถ.ติวานนท์ ต.ท่าทราย อ.เมือง จ.นนทบุรี 11000 โทรศัพท์มือถือ 081-904-3466