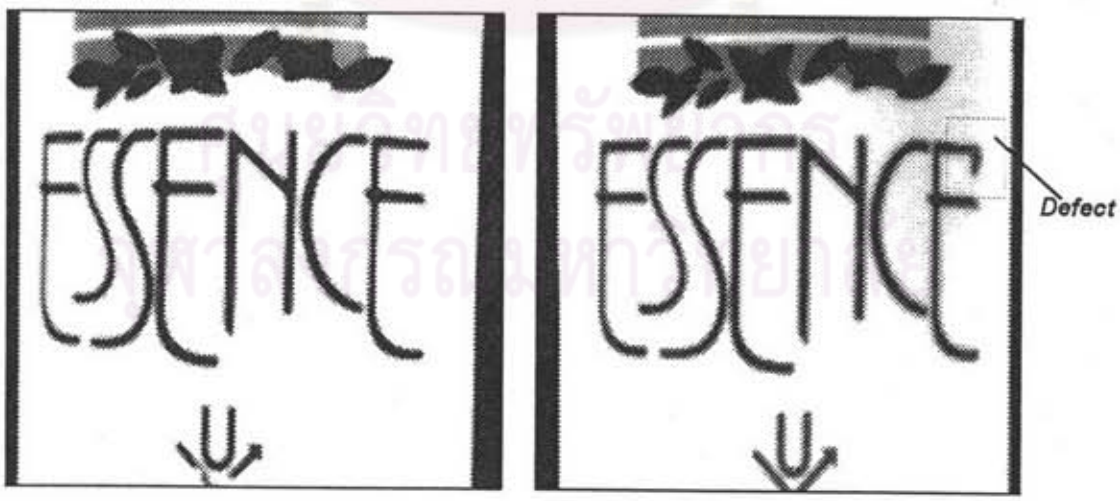
บทที่ 4

อัลกอริทึมในการตรวจสอบ

4.1 นิยามของปัญหาในขบวนการพิมพ์ฉลากบนขวด

ในหัวข้อวิทยานิพนธ์นี้เป็นการพัฒนาระบบตรวจสอบภาพชิ้นงานสำหรับการตรวจสอบหาจุดบกพร่องของฉลากที่พิมพ์อยู่บนขวด (Printed Label on Bottle) โดยตัวอย่างภาพของฉลากบนขวดที่ทำการตรวจสอบแสดงดังรูปที่ 4.1ก โดยที่ลักษณะของฉลากที่พิมพ์นั้นอาจจะเป็นทั้งตัวอักษร, สัญญลักษณ์ หรือรูปแบบต่างๆ โดยที่จุดบกพร่องที่เกิดขึ้นจะขึ้นอยู่กับขั้นตอนในการพิมพ์ว่าสามารถทำให้เกิดจุดบกพร่องแบบใดและตรงจุดใดได้บ้าง ซึ่งความรู้ตรงนี้จะได้มาจากผู้ที่มีความชำนาญในการตรวจสอบชิ้นงานนั้นๆ หรือผู้ที่ควบคุมคุณภาพของระบบการผลิต และหลังจากที่สามารถกำหนดปัญหาของชิ้นงานที่จะทำการตรวจสอบ ซึ่งส่วนใหญ่ก็จะเป็นข้อกำหนดที่ใช้ในการควบคุมคุณภาพการผลิต ก็จะนำปัญหาต่างๆเหล่านี้มาทำการพัฒนาหาอัลกอริทึมที่เหมาะสมสำหรับการตรวจสอบต่อไป



(ก) (ข)

รูปที่ 4.1 ตัวอย่างภาพฉลากบนขวดที่ตรวจสอบ (ก) ฉลากดี (ข) ฉลากบกพร่องเนื่องจากพิมพ์เกินที่บริเวณข้างฉลาก (ตัว E ด้านขวา)

สำหรับในขบวนการพิมพ์บนขวดหลังจากที่สังเกตและศึกษาจากตัวอย่างขวดที่ได้มา สามารถจำแนกจุดบกพร่องออกเป็นประเภทต่างๆ ดังนี้

- ฉลากที่พิมพ์บนขวดนั้นพิมพ์ไม่ครบ (misprint)
- ฉลากที่พิมพ์มีรอยสกปรกและรอยเปื้อนจากสีที่ใช้ในการพิมพ์ (dirty, smudge)
- ฉลากที่พิมพ์นั้นไม่คมชัด (blur)
- ฉลากที่พิมพ์มีการเลื่อนตำแหน่งไปจากตำแหน่งที่ถูกต้อง (misregistration)

จากตัวอย่างจุดบกพร่องดังกล่าวนี้เป็นเพียงบางส่วนของจุดบกพร่องของตัวอย่างขวดที่ได้มา สำหรับลักษณะของจุดบกพร่องแบบอื่นๆจำเป็นต้องได้มาจากการตรวจสอบจากผู้ควบคุมการผลิตและจากปัญหาที่ได้ศึกษามาก็นำมาทำการพัฒนาอัลกอริทึมในการตรวจสอบที่เหมาะสม ซึ่งจะกล่าวถึงในหัวข้อถัดไป

4.2 แนวทางในการพัฒนาอัลกอริทึมในการตรวจสอบ

สำหรับการพัฒนาอัลกอริทึมที่ใช้ในวิทยานิพนธ์นี้ ได้ทำการพัฒนาอัลกอริทึมสำหรับใช้ตรวจสอบขบวนการพิมพ์ฉลากบนขวดซึ่งได้กล่าวไปแล้วในหัวข้อนิยามปัญหาและในหัวข้อเทคนิคที่ใช้ในงานวิจัยทั่วไปในปัจจุบัน และนอกจากนี้ได้ทำการศึกษาระบบที่มีจำหน่ายอยู่ ซึ่งภายหลังการศึกษาแล้วจึงได้สรุปเพื่อนำเสนอการพัฒนาอัลกอริทึมในการตรวจสอบออกเป็น 2 แนวทาง คือ

- ***แนวทางที่ 1*** เป็นการพัฒนาให้ระบบมีอัลกอริทึมในการตรวจสอบแบบที่ไม่ได้เจาะจงไว้สำหรับการใช้งานใดโดยเฉพาะ ซึ่งลักษณะจะคล้ายกับระบบที่มีอยู่แล้วเช่น Image Checker 30 และ VIScanner SE24 ที่ได้เคยกล่าวมาแล้ว โดยที่อัลกอริทึมในการตรวจสอบจะแบ่งออกเป็นฟังก์ชันย่อยๆหลายฟังก์ชันทำงานต่อเนื่องกัน โดยที่ผู้ใช้งานระบบสามารถตั้งค่าพารามิเตอร์ในการตรวจสอบของแต่ละฟังก์ชันเพื่อให้สามารถนำไปตรวจสอบชิ้นงานที่ต้องการได้ โดยที่ในวิทยานิพนธ์นี้จะทำการพัฒนาอัลกอริทึมให้มีลักษณะดังที่กล่าวมาแล้ว หลังจากนั้นก็จะนำไปประยุกต์ใช้ในการตรวจสอบฉลากที่พิมพ์บนขวด

- ***แนวทางที่ 2*** เนื่องจากในอัลกอริทึมแนวทางที่ 1 นั้น ที่พัฒนาขึ้นไม่ได้เจาะจงสำหรับการตรวจสอบฉลากที่พิมพ์บนขวดโดยเฉพาะ ซึ่งอาจจะทำให้ระบบไม่สามารถตรวจสอบได้ดี ดังนั้นในวิทยานิพนธ์นี้จึงทำการพัฒนาอัลกอริทึมเฉพาะสำหรับการตรวจสอบฉลากที่พิมพ์บนขวดโดยพยายามหาวิธีที่เหมาะสมและปัญหาต่างๆ ที่เกิดขึ้น ที่จำเป็นจะต้องแก้ไข เพื่อทำให้ผลการตรวจสอบนั้นมีความถูกต้องและแม่นยำสูงขึ้น

4.3 การพัฒนาอัลกอริทึมสำหรับการตรวจสอบชิ้นงานทั่วไป (แนวทางที่ 1)

4.3.1 ฟังก์ชันในการตรวจสอบที่มีอยู่ในระบบที่มีการพัฒนาขึ้นมาใช้งานแล้ว

ในการพัฒนาอัลกอริทึมในการตรวจสอบของระบบให้สามารถนำไปตรวจสอบชิ้นงานได้ทั่วไป ได้เริ่มต้นจากการสำรวจและศึกษาระบบที่มีการผลิตมาใช้งานและจำหน่ายแล้ว ซึ่งได้ทำการศึกษามา 2 ระบบ คือ Image Checker 30 (Matsushita,1990) และ VIScanner SE24 (Lion Engineering,1992) สำหรับรายละเอียดของแต่ละระบบที่ศึกษามาสรุปได้ดังนี้

Image Checker 30

รายละเอียดของฟังก์ชันการตรวจสอบของระบบนี้นั้นจะมีฟังก์ชันต่างๆจำนวนมาก เพราะนอกจากที่ระบบนี้สามารถนำไปใช้ในระบบการตรวจสอบชิ้นงานโดยทั่วไป ระบบนี้ยังมีฟังก์ชันเกี่ยวกับการวัด (measurement) ที่สามารถทำงานร่วมกับระบบควบคุมแขนกล (Robot Controller) เพื่อนำไปใช้สายงานด้านการประกอบชิ้นงาน (Assembly Line) แต่ในวิทยานิพนธ์นี้เน้นไปในการพัฒนาระบบสำหรับการตรวจสอบหาจุดบกพร่อง (Defect Detection) ซึ่งส่วนใหญ่ต้องการผลลัพธ์สุดท้ายออกมาในรูปผ่านหรือไม่ผ่าน (Go or NoGo) ดังนั้นในหัวข้อนี้จึงจะพูดรายละเอียดของฟังก์ชันในการตรวจสอบส่วนที่เกี่ยวข้องและนำไปใช้ในการตรวจสอบจุดบกพร่องเท่านั้น ฟังก์ชันต่างๆที่มีอยู่ในระบบนี้จะทำการแบ่งออกเป็นฟังก์ชันย่อยๆหลายฟังก์ชัน ทำงานต่อเนื่องกัน และในแต่ละฟังก์ชันย่อยๆนั้นยังแบ่งเป็นชุดของฟังก์ชันหลายๆฟังก์ชันที่สามารถตั้งค่าได้ตามผู้ใช้งาน ซึ่งฟังก์ชันย่อยต่างๆเหล่านั้น ได้แก่ ฟังก์ชันตรวจสอบ (Checker Function), ฟังก์ชันปรับตำแหน่ง (Position Adjustment Function), ฟังก์ชันผลลัพธ์ (Output Function) สำหรับข้อมูลภาพที่ใช้ในการประมวลผลต่างๆของระบบนี้จะเป็นข้อมูลภาพสองระดับ (binary image) แต่ข้อมูลภาพที่ถ่ายเข้ามาได้จะเป็นข้อมูล 64 ระดับ จึงต้องทำการแปลงเป็นข้อมูลภาพสองระดับก่อน โดยใช้ค่าขีดเริ่มเปลี่ยน (threshold) ที่ผู้ใช้กำหนด สำหรับรายละเอียดของแต่ละฟังก์ชันนั้นมีดังต่อไปนี้

- ฟังก์ชันตรวจสอบ จะเป็นฟังก์ชันหลักในการตรวจสอบ ที่จะทำการวัดค่าต่างๆ ออกมาแล้วนำมาเปรียบเทียบกับค่ามาตรฐานที่ตั้งไว้ เพื่อจะตัดสินว่าชิ้นงานนั้นดีหรือเสีย ฟังก์ชันนี้สามารถแบ่งย่อยตามลักษณะของแต่ละฟังก์ชันได้ 3 ฟังก์ชันดังนี้

  - เส้น (Line) เป็นฟังก์ชันสำหรับวัดค่าจำนวนพิกเซลที่กำหนด (พิกเซลสีขาวที่มีค่า 1 หรือพิกเซลสีดำที่มีค่า 0) ในบริเวณของเส้นที่กำหนดโดยผู้ใช้แล้วนำค่านั้นมาเปรียบเทียบกับค่าขอบจำกัดบนและล่าง (upper and lower limit) ที่กำหนดโดยผู้ใช้ ถ้าภาพชิ้นงานใดนับจำนวนพิกเซลมีค่าอยู่ขอบเขตดังกล่าวก็จะถือว่าเป็นชิ้นงานที่ดี มิฉะนั้นแล้วก็จะจัดว่าเป็นชิ้นงานเสีย

- กรอบวินโดวส์ (Window) ฟังก์ชันนี้คล้ายกับฟังก์ชันเส้นในข้อที่แล้วแต่จะเป็นการนับจำนวนพิกเซลในอาณาบริเวณพื้นที่ภายในกรอบวินโดวส์ที่กำหนด

- ฟังก์ชันเปรียบเทียบรูปแบบ (Pattern Matching) เป็นฟังก์ชันสำหรับการตรวจสอบโดยการเปรียบเทียบแบบจุดต่อจุดระหว่างภาพที่ต้องการตรวจสอบกับภาพอ้างอิงภายในบริเวณกรอบวินโดวส์ที่กำหนดแล้ววัดค่าออกมาเป็นจำนวนจุดที่แตกต่างกันของทั้งสองภาพและนำมาตัดสินว่าชิ้นงานนั้นดีหรือไม่

- ฟังก์ชันปรับตำแหน่ง ในระบบนี้การปรับตำแหน่งจะทำการในแนวตั้งและแนวนอนโดยใช้ขอบภาพ (edge data) ของภาพชิ้นงานเป็นจุดอ้างอิง ในการคำนวนตำแหน่งที่เลื่อนไป เพื่อนำไปบวกชดเชยเข้าไปเพื่อปรับตำแหน่งของฟังก์ชันตรวจสอบที่กล่าวมาแล้ว

- ฟังก์ชันเอาท์พุท จะเป็นฟังก์ชันที่จะทำการตัดสินใจจากผลการตรวจสอบของฟังก์ชันตรวจสอบต่าง ๆ ที่ได้เรียกใช้งาน เพื่อจะให้ผลลัพธ์สุดท้ายออกมาในรูปผ่านหรือไม่ผ่าน โดยผลลัพธ์สุดท้ายจะได้จากการรวมกันทางตรรก (logic combination) ของผลลัพธ์ของฟังก์ชันตรวจสอบย่อยต่าง ๆ ที่ใช้งาน

จากรายละเอียดของฟังก์ชันทั้ง 3 ส่วนที่ได้กล่าวมาแล้วนั้น ระบบจะนำไปใช้งานตรวจสอบอะไรก็จะขึ้นอยู่กับผู้ใช้งานทำการปรับแต่งและเซตค่าพารามิเตอร์ต่างๆ ของแต่ละฟังก์ชัน ซึ่งจะขึ้นอยู่กับประเภทของชิ้นงานที่ต้องการตรวจสอบ, เกณฑ์ในการตัดสิน, สภาพแวดล้อมต่างๆ และองค์ประกอบอื่นๆ นอกจากนี้ในระบบนี้ยังมีฟังก์ชันอื่นๆ ที่น่าสนใจ เช่น ฟังก์ชันส่วนคำนวน (Numeric Calculation), ส่วนของการดึงคุณลักษณะสำคัญ (Feature Extraction) ซึ่งเป็นฟังก์ชันสำหรับการดึงคุณลักษณะสำคัญ (feature) ที่ต้องการภายในบริเวณพื้นที่กรอบวินโดวส์ที่กำหนด ซึ่งคุณลักษณะสำคัญได้แก่ จำนวนวัตถุ (number of objects), จุดศูนย์กลางของวัตถุ (center of gravity), พื้นที่, เส้นรอบวง และอื่นๆ

VIScanner SE 24

ลักษณะของเครื่องนี้จะคล้ายกับระบบ Image Checker 30 ที่กล่าวมาแล้ว คือ หลักการในการตรวจสอบจะใช้การตรวจสอบโดยการนับจำนวนพิกเซลของภาพชิ้นงานในบริเวณพื้นที่ที่กำหนด ซึ่งเรียกว่า กรอบวินโดวส์(window) แต่ในระบบนี้จะไม่แบ่งออกเป็นฟังก์ชันย่อยๆที่ทำงานต่อเนื่องกันดังเช่นระบบแรก แต่จะแบ่งเป็นกรอบวินโดวส์ย่อยๆ หลายอัน โดยรวมขั้นตอนต่างๆ ของการตรวจสอบทั้งหมดอยู่ในฟังก์ชันกรอบวินโดวส์อย่างเดียว เช่น การแปลงภาพสองระดับ, การปรับตำแหน่ง, การนับจำนวนพิกเซล ซึ่งจะทำให้การใช้งานค่อนข้างง่ายกว่าระบบแรกที่จะต้องทำการโปรแกรมหลายฟังก์ชันจึงจะใช้ตรวจสอบได้

4.3.2 อัลกอริทึมแบบใช้งานทั่วไปที่พัฒนาขึ้นในวิทยานิพนธ์

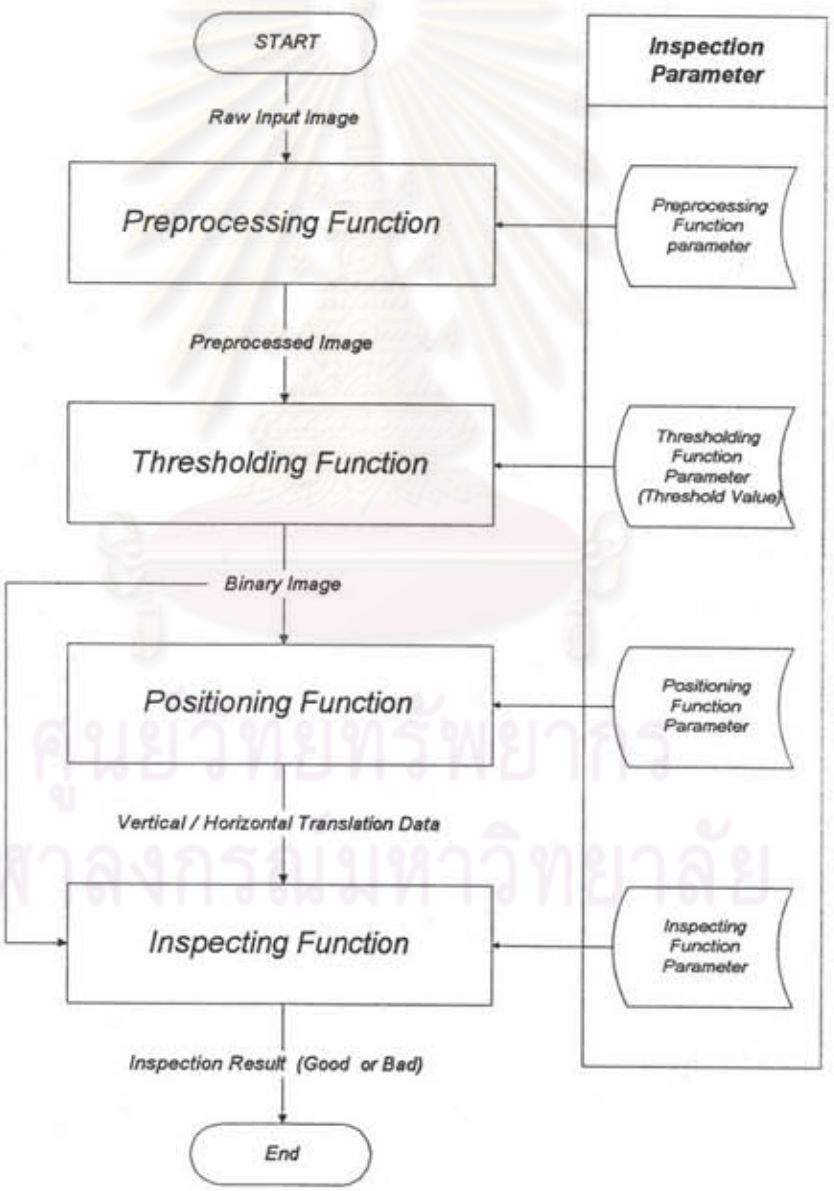
อัลกอริทึมที่พัฒนาขึ้นในแนวทางที่ 1 จะทำการพัฒนาให้เป็นแบบที่ไม่ได้เจาะจงสำหรับการใช้งานในการตรวจสอบชิ้นงานใดโดยเฉพาะ แต่จะทำการพัฒนาให้สามารถใช้งานได้ทั่วไปขึ้นอยู่กับผู้ใช้งานระบบว่าจะนำไปดัดแปลงใช้งาน ซึ่งลักษณะของฟังก์ชันในการตรวจสอบที่พัฒนาขึ้นจะมีลักษณะเช่นเดียวกันกับฟังก์ชันที่มีอยู่ในระบบ Image Checker 30 และ Viscanner SE24 ที่ได้กล่าวมาแล้ว โดยที่จะพยายามลดความยุ่งยากซับซ้อนของแต่ละฟังก์ชันที่ทำงานต่อกันเพื่อให้สามารถนำไปประยุกต์ใช้งานได้ง่ายกว่า และจากการศึกษาพบว่าอัลกอริทึมลักษณะนี้จะอยู่บนพื้นฐานของวิธีกรอบวินโดวส์ (Window Method) ที่ได้กล่าวไว้ใน Ejiri,M.1990 ซึ่งวิธีนี้จะเป็นการตีกรอบบริเวณของพื้นที่ในการตรวจสอบ (Inspection Area) ให้อยู่ในบริเวณภายในพื้นที่กรอบวินโดวส์เท่านั้น หลังจากนั้นก็จะทำการวิเคราะห์ตามขบวนการต่างๆ ซึ่งส่วนใหญ่จะเป็นการวัดค่าคุณลักษณะสำคัญ (feature) เพื่อนำมาเปรียบเทียบกับเกณฑ์ในการตรวจสอบที่ตั้งไว้

สำหรับในวิทยานิพนธ์นี้ ฟังก์ชันในการตรวจสอบก็จะยึดหลักบนวิธีกรอบวินโดวส์นี้ โดยที่คุณลักษณะสำคัญที่ใช้ในการตรวจสอบนี้ คือ จำนวนพิกเซล (ขาว/ดำ) ในบริเวณกรอบวินโดวส์ที่กำหนด นอกจากนี้ยังได้เพิ่มส่วนของฟังก์ชันประมวลผลเบื้องต้น (Preprocessing) ที่ทำการปรับสภาพข้อมูลภาพที่ถ่ายเข้ามาได้ ก่อนที่จะผ่านขบวนการอื่นๆ ต่อไป หลังจากที่ได้พัฒนาและสร้างฟังก์ชันในการตรวจสอบต่างๆ แล้วก็จะนำไปประยุกต์ใช้ตรวจสอบฉลากที่พิมพ์บนขวด เพื่อนำผลมาศึกษาถึงปัญหาและข้อเสียที่เกิดขึ้นเพื่อจะเป็นแนวทางที่จะทำการพัฒนาอัลกอรทึมเฉพาะสำหรับการตรวจสอบฉลากที่พิมพ์บนขวด (แนวทางที่ 2) ต่อไป

ฟังก์ชันในการตรวจสอบที่พัฒนาขึ้นในวิทยานิพนธ์นี้จะทำการออกแบบพัฒนาให้เป็นฟังก์ชันย่อยๆทำงานต่อเนื่องกันตามลำดับ แสดงดังแผนภูมิการไหลของข้อมูล (dataflow diagram) ดังรูปที่ 4.2 โดยที่ผู้ใช้งานระบบสามารถตั้งและกำหนดค่าพารามิเตอร์ในการทำงานของแต่ละฟังก์ชัน (inspection paramater) ให้สามารถนำไปประยุกต์ตรวจสอบชิ้นงานที่ต้องการได้ จากแผนภูมิการไหลของข้อมูลสามารถอธิบายเป็นขั้นตอนคร่าวๆได้ดังนี้

โดยเริ่มต้นจากภาพชิ้นงานที่ถ่ายเข้ามาได้ (raw input image) จะนำมาผ่านขบวนการในฟังก์ชันประมวลผลภาพเบื้องต้น (Preprocessing Function) และได้ผลลัพธ์ออกมาเป็นภาพที่ผ่านการประมวลผลเบื้องต้น (preprocessed image) หลังจากนั้นจึงนำมาแปลงเป็นภาพสองระดับ (binary image) โดยใช้ฟังก์ชันแปลงภาพสองระดับ (Thresholding Function) ภาพชิ้นงานที่ถ่ายเข้ามาได้นั้นอาจจะมีการเลื่อนตำแหน่งไปจากตำแหน่งที่ได้กำหนดไว้ ดังนั้นจะต้องทำการชดเชยการเลื่อนตำแหน่งโดยใช้ฟังก์ชันปรับตำแหน่ง (Positioning Function) ซึ่งจะคำนวณหาค่าการเลื่อน

ตำแหน่งของชิ้นงานทางด้านแนวตั้งและแนวนอน (Vertical/Horizontal Translation data) เพื่อจะนำไปใช้ในการปรับตำแหน่งต่อไป ส่วนฟังก์ชันสุดท้ายก็คือฟังก์ชันตรวจสอบ (Inspecting Function) ซึ่งเป็นฟังก์ชันหลักในการตรวจสอบ ที่จะทำการวัดค่าคุณลักษณะสำคัญ (feature) จากภาพชิ้นงานเพื่อนำมาตัดสินว่าชิ้นงานนั้นดีหรือไม่ โดยที่ก่อนจะทำการวัดค่าต่างๆ นั้นจะต้องทำการปรับตำแหน่งของฟังก์ชันตรวจสอบให้ลงไปยังตำแหน่งที่ถูกต้องโดยใช้ค่าการเลื่อนตำแหน่งที่คำนวณได้จากฟังก์ชันปรับตำแหน่ง สำหรับรายละเอียดของแต่ละฟังก์ชันนั้นสามารถอธิบายได้ดังต่อไปนี้



รูปที่ 4.2 แผนภูมิการไหลของข้อมูลของอัลกอริทึมแนวทางที่ 1

1 ฟังก์ชันประมวลผลภาพเบื้องต้น (Preprocessing Function)

ฟังก์ชันนี้เป็นฟังก์ชันสำหรับการปรับสภาพข้อมูลภาพขึ้นงานที่ถ่ายเข้ามาได้ให้อยู่ในสภาพที่เหมาะสมที่จะผ่านขบวนการอื่นต่อไป เช่น การกำจัดสัญญาณรบกวนแบบต่างๆ (noise reduction), การทำให้ภาพคมชัดขึ้น (sharpening) ,การเน้นขอบภาพ (edge enhancement) ซึ่งขบวนการต่างๆ เหล่านี้จะเป็นการจัดการกับภาพอินพุทและให้ผลออกมาในเป็นภาพเอาท์พุทที่ผ่านการปรับสภาพแล้วสามารถนำไปใช้ในงานต่อไปได้ โดยที่ลักษณะการใช้งานของฟังก์ชันนี้จะให้ผู้ใช้เลือกขบวนการที่จะจัดการกับภาพอินพุทและสามารถโปรแกรมใช้งานได้หลายๆขบวนการต่อเนื่องกัน ซึ่งจะเป็นพารามิเตอร์ในการกำหนดการทำงานของฟังก์ชันนี้

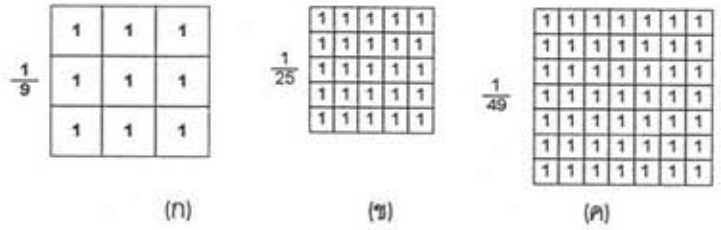
สำหรับวิทยานิพนธ์นี้ ขบวนการ (operation) ต่างๆ ที่นำมาพัฒนาและใช้ในฟังก์ชันนี้ได้นำมาจาก Gonzalez,R.C. and Woods,R.E. 1992 ได้แก่

1.1 ขบวนการในการกำจัดสัญญาณรบกวน

1.1.1 Spatial Lowpass Filter

ขบวนการนี้จะทำหน้าที่ในการลดทอนหรือกำจัดส่วนที่เป็นความถี่สูงออกไป โดยที่ยังคงส่วนที่เป็นความถี่ต่ำไว้ ส่วนที่เป็นความถี่สูงของภาพนี้ก็จะได้แก่ส่วนที่เป็นขอบภาพหรือส่วนที่คมชัด ดังนั้นผลที่ได้จากขบวนการนี้คือ การทำให้ภาพเลือน (image blurring) ซึ่งจะเป็นการลดสัญญาณรบกวนแบบหนึ่ง สำหรับลักษณะของขบวนการนี้จะอยู่ในรูปของหน้ากากกรอบวินโดวส์ (mask window) แสดงดังรูปที่ 4.3 ก โดยที่ขบวนการนี้จะนำเอากรอบวินโดวส์ที่กำหนดด้วยค่าสัมประสิทธิ์ไปทาบลงบนตำแหน่งของภาพอินพุทและคูณค่าสัมประสิทธิ์นั้นกับค่าความสว่าง (gray-level)ของภาพอินพุทแบบจุดต่อจุดและได้ค่าเป็นค่าความสว่างของภาพเอาท์พุทที่ตำแหน่งพิกเซลตรงกลางของกรอบวินโดวส์

จะเห็นได้ว่าฟังก์ชันนี้จะเป็นการเฉลี่ยค่าความสว่างของพิกเซลของตนเองกับพิกเซลรอบข้าง ซึ่งขบวนการนี้สามารถขยายขนาดของกรอบวินโดวส์ เป็น 5x5, 7x7, 9x9 ... ดังรูปที่ 4.3ข แต่ก็จะใช้เวลาในการคำนวนเพิ่มขึ้นด้วย



รูปที่ 4.3 ลักษณะของ Spatial Lowpass Filter ขนาดต่างๆ (ก) 3x3 (ข)5x5 (ค) 7x7

1.1.2 Median Filter

เนื่องจากขบวนการ Spatial Lowpass Filter ที่กล่าวมาแล้วในข้อ 1.1.1 นั้นจะมีข้อเสีย คือจะลดรายละเอียดส่วนที่เป็นขอบภาพและส่วนที่คมชัดของภาพ ซึ่งในกรณีที่ต้องการจะลดสัญญาณรบกวนมากกว่าการทำให้ภาพเลือนก็จะทำให้การใช้งานนั้นไม่เหมาะสม ดังนั้นขบวนการ median filter นี้จะเหมาะสมกว่าในกรณีที่ต้องการลดสัญญาณรบกวนออกและยังคงเก็บรายละเอียดส่วนที่เป็นขอบภาพและคมชัดไว้อยู่ด้วย โดยการทำงานของขบวนการนี้ก็ยังคงใช้ลักษณะของหน้ากากกรอบวินโดวส์อยู่เพียงแต่ว่าค่าความสว่างที่ตำแหน่งพิกเซลตรงกลางของกรอบวินโดวส์ของภาพผลลัพธ์นั้นจะเท่ากับค่ามัธยฐานของพิกเซลทั้งหมดที่อยู่ในบริเวณกรอบวินโดวส์ หรือสามารถอธิบายได้ดังนี้คือจะนำค่าความสว่างของตำแหน่งพิกเซลทั้งหมดในบริเวณกรอบวินโดวส์มาทำการเรียงจากน้อยไปหามาก หลังจากนั้นก็ทำการหาค่าที่อยู่ตรงกลางของเซตข้อมูลนั้น และค่าความสว่างของพิกเซล ณ ตำแหน่งตรงกลางของกรอบวินโดวส์ ก็คือค่านี้นั่นเอง

1.2 ขบวนการในการให้ภาพคมชัด (Sharpening)

หลักการในการทำให้ภาพคมชัดก็คือจะทำการขยาย (enhance) รายละเอียดส่วนที่ถูกทำให้เลือนซึ่งอาจจะเกิดเนื่องจากข้อผิดพลาดทางกายภาพของส่วนที่รับภาพ สำหรับในวิทยานิพนธ์นี้ขบวนการที่ถูกนำมาใช้ในระบบนี้ก็คือ Spatial Highpass Filter ผลของขบวนการนี้จะทำการลดและกำจัดส่วนที่เป็นความถี่ต่ำออกไป ซึ่งลักษณะก็จะคล้ายกับ Spatial Lowpass Filter ที่กล่าวมาแล้ว โดยลักษณะของการทำงานของขบวนการนี้ก็ยังทำงานด้วยหน้ากากกรอบวินโดวส์ แต่ค่าสัมประสิทธิ์ต่างๆนั้นจะแตกต่างกัน โดยค่าสัมประสิทธิ์ของกรอบวินโดวส์แสดงดังรูปที่ 4.4

$\frac{1}{9} \times$

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

รูปที่ 4.4 ลักษณะของ spatial highpass filter ขนาด 3x3

1.3 ขบวนการในการเน้นขอบภาพ (Edge Enhancement)

เนื่องจากการตรวจสอบต่างๆในบางครั้งนั้น ต้องการข้อมูลภาพเฉพาะส่วนที่เป็นขอบภาพเท่านั้น ดังนั้นขบวนการเหล่านี้ก็จะทำหน้าที่ขยายข้อมูลส่วนที่เป็นขอบภาพและกดหรือลดข้อมูลส่วนอื่นลงไป จนทำให้ส่วนที่เป็นขอบภาพเท่านั้นที่เด่นชัดขึ้นมา ซึ่งส่วนที่เป็นขอบภาพนั้นก็คือส่วนที่แบ่งเขตระหว่าง 2 อาณาบริเวณที่มีค่าความสว่างต่างกัน สำหรับขบวนการในการเน้นขอบ

ภาพที่ได้นำมาใช้ในวิทยานิพนธ์นี้ก็อยู่ในรูปแบบของหน้ากากกรอบวินโดวส์ โดยที่ค่าสัมประสิทธิ์ที่แต่ละตำแหน่งนั้นจะเปลี่ยนไปตามแต่ละเทคนิค โดยที่นำมาใช้มี 3 แบบ ก็คือ Sobel,Prewitt และ Laplacian โดยลักษณะของหน้ากากและค่าสัมประสิทธิ์ต่างๆ แสดงดังรูปที่ 4.5

รูปที่ 4.5 ลักษณะของ Edge Detection Mask แบบ (ก) Laplacian (ข) Sobel (ค)Prewitt

นอกเหนือจากขบวนการต่างๆ ในหมวดของฟังก์ชันประมวลผลภาพเบื้องต้นที่กล่าวมาทั้ง 3 ข้อแล้ว ฟังก์ชันนี้สามารถพัฒนาโดยนำขบวนการแบบอื่นๆ เข้ามาเพิ่มและรวมอยู่ในฟังก์ชันนี้ได้ โดยที่ลักษณะของขบวนการจะต้องเป็นการสร้างภาพผลลัพธ์จากภาพอินพุทเดิม ให้มีคุณสมบัติบางอย่างดีขึ้นตามที่เราต้องการ เช่น ขบวนการในการทำให้ภาพสว่าง (histogram equalization), ขบวนการในเน้นความแตกต่าง (contrast streching) เป็นต้น

2. ฟังก์ชันแปลงภาพสองระดับ (Thresholding Function)

เนื่องจากการทำงานของฟังก์ชันต่างๆที่จะกล่าวต่อไป จะเป็นการโปรเซสบนภาพแบบ 2 ระดับ แต่ข้อมูลภาพที่ถ่ายเข้ามาได้และผ่านขบวนการปรับสภาพข้อมูลจากฟังก์ชันประมวลผลภาพเบื้องต้นแล้วยังคงได้ข้อมูลภาพเป็นแบบ 256 ระดับ (grey scale image) ดังนั้นจึงเป็นหน้าที่ของฟังก์ชันนี้ในการแปลงข้อมูลภาพ 256 ระดับ ให้เป็นแบบ 2 ระดับ โดยใช้ค่าขีดเริ่มเปลี่ยน (threshold) ที่กำหนดโดยผู้ใช้งาน ซึ่งสามารถเขียนได้ดังนี้

$$B(x,y) = 1 \text{ (white) if } I(x,y) \geq th$$

$$B(x,y) = 0 \text{ (black) if } I(x,y) < th$$

โดยที่ B(x,y) คือ ค่าความสว่างของภาพผลลัพธ์แบบ 2 ระดับ ที่ตำแหน่ง x,y

$I(x,y)$ คือ ค่าความสว่างของภาพอินพุทแบบ 256 ระดับ ที่ตำแหน่ง x,y

th คือ ค่าขีดเริ่มเปลี่ยนที่กำหนดโดยผู้ใช้งานระบบมีค่าระหว่าง 0-255

3. ฟังก์ชันตรวจสอบ (Inspecting Function)

ฟังก์ชันตรวจสอบนี้ถือเป็นฟังก์ชันหลักในส่วนของฟังก์ชันทั้งหมด เพราะเป็นฟังก์ชันที่จะทำการวัดค่าต่างๆ จากภาพชิ้นงานที่ถ่ายเข้ามาได้ แล้วนำมาเปรียบเทียบกับค่าต้นแบบที่กำหนดเพื่อจะได้ทำการตัดสินว่าชิ้นงานนั้นดีหรือเสีย สำหรับพื้นฐานหลักการของฟังก์ชันตรวจสอบที่ใช้ในวิทยานิพนธ์ คือ วิธีกรอบวินโดวส์ ซึ่งเป็นวิธีที่ใช้กันทั่วไปในระบบแบบใช้งานทั่วไป เช่น ระบบ Image Checker 30 และระบบ VIscanner SE 24 ที่กล่าวมาแล้ว โดยหลักการของวิธีนี้ คือ ทำการลดขอบเขตในการวิเคราะห์ภาพชิ้นงานให้อยู่ในบริเวณขอบเขตกรอบวินโดวส์ที่กำหนดเท่านั้น โดยจะทำการวัดคุณลักษณะสำคัญออกมาจากภาพชิ้นงานที่ถ่ายเข้ามาในบริเวณกรอบวินโดวส์เพื่อนำไปใช้ในการรู้จำหรือเปรียบเทียบต่อไป

เนื่องจากอัลกอริทึมในแนวทางนี้ต้องการให้ฟังก์ชันตรวจสอบมีความยืดหยุ่นในการใช้งาน คือสามารถนำไปใช้ตรวจสอบชิ้นงานได้หลายประเภท จึงเลือกใช้คุณลักษณะสำคัญ คือ จำนวนพิกเซลสีดำ (0) หรือ สีขาว (1) ในภาพสองระดับ ซึ่งใช้ในระบบ Image Checker 30 และ VIscanner-SE24 โดยลักษณะของฟังก์ชันตรวจสอบที่ใช้จำนวนพิกเซลเป็นคุณลักษณะสำคัญนั้นจะทำการนับจำนวนพิกเซลของภาพสองระดับ ซึ่งอาจจะเป็นพิกเซลที่มีค่า 0 หรือ 1 ก็ได้ ในบริเวณพื้นที่กรอบวินโดวส์ที่กำหนด แล้วนำค่าที่ได้มาเปรียบเทียบกับช่วงขอบเขตที่กำหนดโดยค่าขีดจำกัดบนและล่าง (upper and lower limit) โดยที่ถ้าจำนวนพิกเซลที่นับได้อยู่ในช่วงดังกล่าวก็จะถือว่าเป็นชิ้นงานที่ดี มิฉะนั้นแล้วก็จะเป็นชิ้นงานที่เสีย สำหรับค่าขีดจำกัดบนและล่างนั้นจะกำหนดโดยผู้ตั้งระบบ ซึ่งค่านี้จะเป็นที่กำหนดขอบเขตในการยอมรับชิ้นงาน (acceptance range) โดยในวิทยานิพนธ์นี้ ได้เสนอวิธีในการหาค่าขีดจำกัดบนและล่างที่เหมาะสมดังนี้

- ทำการถ่ายภาพชิ้นงานดีที่ใช้เป็นต้นแบบและวัดค่าจำนวนพิกเซลตามแต่ละฟังก์ชันตรวจสอบจากภาพชิ้นงานหลายๆชิ้น หรือใช้ชิ้นงานเดียวแต่ทำการวัดหลายๆครั้ง โดยในแต่ละครั้งของการวัดก็จะทำการบันทึกค่าเก็บไว้ เมื่อทำการถ่ายภาพจนครบจำนวนชิ้นงานที่ต้องการ (หรือจำนวนครั้งที่ต้องการ) ก็นำค่าจำนวนพิกเซลที่บันทึกไว้มาคำนวนหาค่าเฉลี่ย (mean) และค่าเบี่ยงเบนมาตรฐาน (standard deviation)

- ค่าขีดจำกัดบนและล่างที่เหมาะสมจะอยู่บนสมมติฐานที่ว่าเซตของค่าที่วัดจากการทดสอบนั้น มีการกระจายแบบปกติ (normal distribution) ซึ่งข้อมูลที่มีการกระจายแบบนี้ พบว่าค่าที่อยู่ในช่วง mean±3*SD นั้น ครอบคลุม 95% ของเซตของข้อมูลทั้งหมด (Kreszig,E 1988) ดังนั้น ค่าขีดจำกัดบนและล่างที่เหมาะสม คือ

ค่าขีดจำกัดล่างที่เหมาะสม = mean - 3*SD

ค่าขีดจำกัดบนที่เหมาะสม = mean + 3*SD

โดยที่ mean คือ ค่าเฉลี่ยของชุดข้อมูลของจำนวนพิกเซลที่นับได้จากชิ้นงานต้นแบบ

SD คือ ค่าเบี่ยงเบนมาตรฐานของชุดข้อมูลของจำนวนพิกเซลที่นับได้จากชิ้นงานต้นแบบ

ฟังก์ชันตรวจสอบนี้สามารถแบ่งย่อยได้ออกเป็น 3 ฟังก์ชันดังนี้

3.1. ฟังก์ชันนับพิกเซลนับพิกเซลในกรอบวินโดวส์ (Window Pixel Counting)

ฟังก์ชันนี้มีพื้นฐานอยู่บนวิธีกรอบวินโดวส์ที่ได้กล่าวไปแล้ว คือ จะทำการนับจำนวนพิกเซลที่มีค่า 0 หรือ 1 อย่างใดอย่างหนึ่งในบริเวณพื้นที่กรอบวินโดวส์ที่กำหนด แล้วนำมาเปรียบเทียบกับค่าขีดจำกัดบนและล่างที่กำหนด ฟังก์ชันนี้สามารถแบ่งย่อยตามลักษณะรูปร่างของกรอบวินโดวส์ได้เป็น 3 แบบ คือ

3.1.1 กรอบวินโดวส์แบบสี่เหลี่ยมผืนผ้า (Rectangle Window)

กรอบวินโดวส์แบบนี้จะนิยามด้วยค่าจุดพิกัดมุมซ้ายบน (startx,starty) และจุดพิกัดมุมขวาล่าง (endx,endy) ดังรูปที่ 4.6 ก

รูปที่ 4.6 ลักษณะกรอบวินโดวส์แบบต่างๆ (ก) สี่เหลี่ยมผืนผ้า (ข) หลายเหลี่ยม (ค) วงกลม

3.1.2 กรอบวินโดวส์แบบหลายเหลี่ยม (Polygon Window)

ลักษณะของกรอบวินโดวส์แบบนี้จะเป็นแบบหลายเหลี่ยม (polygon) ซึ่งจะนิยามด้วยเซตของจุดเชื่อมต่อ (vertex point) { (Vx1,Vy1) ,(Vx2,Vy2),.....,(VxN,VyN) } ดังตัวอย่างรูปที่ 4.6 ข

สำหรับหลักการในการนับพิกเซลในบริเวณกรอบวินโดวส์แบบนี้จะต้องสามารถระบุได้ว่า บริเวณใดๆ ในภาพนั้นอยู่ในพื้นที่กรอบวินโดวส์นี้หรือไม่ ซึ่งในกรณีของกรอบวินโดวส์แบบหลายเหลี่ยมนั้นจะใช้หลักการในการระบายสีในพื้นที่รูปปิดหลายเหลี่ยม (Polygon Area Filling Algorithm) (Hearn,D. and Baker,M.P. 1994) ซึ่งเป็นอัลกอริทึมพื้นฐานทางคอมพิวเตอร์กราฟิกส์ (computer graphics) ที่ใช้ในการระบายจุดพิกเซลใดที่อยู่ในตำแหน่งในพื้นที่รูปปิดที่กำหนดด้วยเซตของจุดเชื่อมต่อ แต่ในที่นี้จะนำมาใช้โดยการดัดแปลงจากการระบายสีที่ตำแหน่งพิกเซลมา

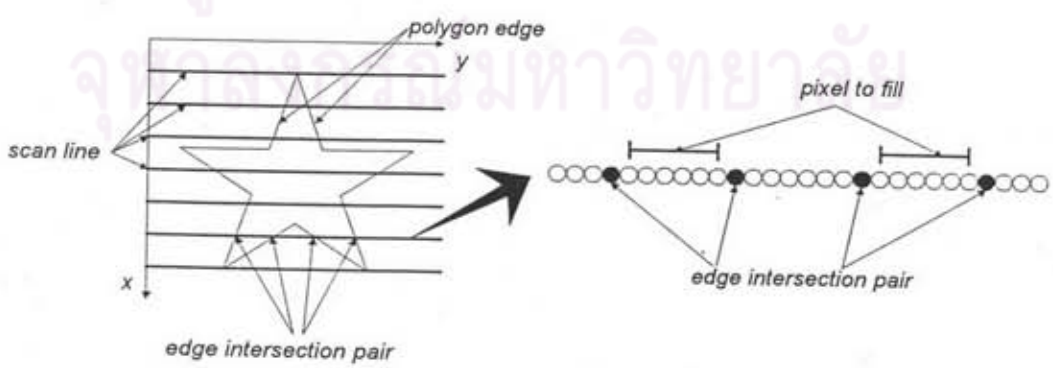
เป็นการอ่านค่าความสว่างที่ตำแหน่งพิกเซลนั้นแทน สำหรับหลักการของอัลกอริทึมในการระบายสีรูปปิด สามารถอธิบายได้ดังนี้

หลักการของการระบายสีรูปปิด

อัลกอริทึมนี้เป็นอัลกอริทึมพื้นฐานที่ใช้ในงานคอมพิวเตอร์กราฟฟิกส์ ซึ่งจะทำการระบายสีรูปปิดที่นิยามด้วยเซตของจุดเชื่อมต่อ โดยจะทำการระบายพิกเซลที่อยู่ในพื้นที่รูปปิดทีละเส้นที่ทำการระบาย (scan line) จากบนลงล่าง ดังรูปที่ 4.7 ดังนั้นวิธีนี้จึงมีชื่อเรียกอีกชื่อว่า *Scan-Line Polygon Fill Algorithm* ซึ่งสามารถอธิบายหลักการเป็นขั้นตอนโดยคร่าวๆ ดังนี้

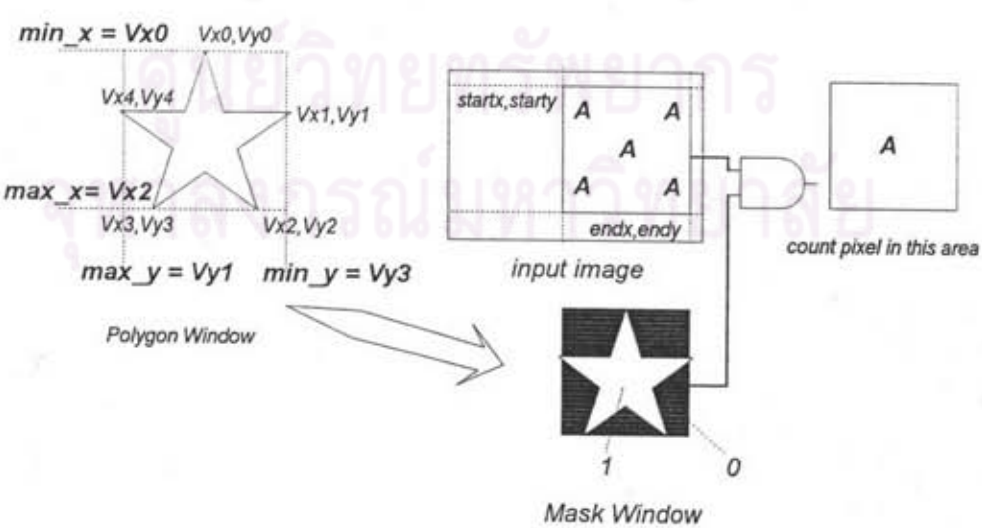
- ที่ตำแหน่งเส้นที่กำลังระบายใดๆ ทำการคำนวนหาค่าของ y ที่ตัดกับขอบของรูปปิด (polygon edge)
- นำเซตของจุดตัดที่คำนวนได้ในขั้นตอนแรก มาทำการเรียง (sort) จากน้อยไปมากโดยใช้ค่าของ y
- จัดกลุ่มของเซตจุดตัดออกเป็นคู่ๆเพื่อหาคู่ของจุดตัดขอบรูปปิดที่จะทำการระบาย (edge intersection pair)
- ระบายสี (Fill) พิกเซลในบริเวณเส้นที่กำลังระบายใดๆระหว่างคู่ของจุดตัดที่จับกลุ่มที่คำนวณได้ จากขั้นตอนที่แล้ว

ถึงแม้ว่าอัลกอริทึมที่อธิบายไปนั้นค่อนข้างง่าย แต่ในแง่ของการเขียนโปรแกรมนั้นค่อนข้างซับซ้อน โดยเฉพาะอย่างยิ่งขั้นตอนในการคำนวนหาจุดตัด (edge intersection) ระหว่างเส้นที่กำลังระบายและขอบของรูปปิด ซึ่งรายละเอียดของอัลกอริทึมและเทคนิคในการเขียนโปรแกรมสามารถศึกษาได้จาก Hearn,D. and Baker,M.P. 1994



รูปที่ 4.7 หลักการในการระบายในพื้นที่สีรูปปิด (Scan-line Polygon Fill Algorithm)

เนื่องจากการทำงานของฟังก์ชันนับพิกเซลในกรอบวินโดวส์แบบหลายเหลี่ยมที่กล่าวมาแล้วนั้นจะต้องทำการคำนวนจุดตัดในแต่ละเส้นที่ทำการระบาย ซึ่งจะต้องผ่านหลายขั้นตอนในการคำนวณ ดังนั้นเพื่อเป็นการลดเวลาในการคำนวนจึงทำการพัฒนาฟังก์ชันกรอบวินโดวส์แบบนี้ ในรูปแบบของหน้ากากกรอบวินโดวส์ (mask window) ที่มีลักษณะดังรูปที่ 4.8 ซึ่งจะทำให้การทำงานของฟังก์ชันแบบนี้คล้ายกับฟังก์ชันนับพิกเซลในกรอบวินโดวส์แบบสี่เหลี่ยมผืนผ้า เพียงแต่ว่าในแต่ละพิกเซลบนหน้ากากกรอบวินโดวส์จะมีค่ากำหนดไว้ว่าจะมีค่าเป็น 0 หรือ 1 อย่างใดอย่างหนึ่ง โดยที่การใช้งานของฟังก์ชันนี้ก็จะเป็นการใช้หน้ากากกรอบวินโดวส์ที่ต้องการไปทาบบนตำแหน่งของภาพที่ต้องการตรวจสอบเพื่อทำการนับพิกเซลในบริเวณกรอบวินโดวส์ โดยถ้าค่าในหน้ากากกรอบวินโดวส์ที่ตำแหน่งพิกเซลที่ตรงกันกับพิกเซลที่กำลังจะทำการนับพิกเซลนั้นมีค่าเป็น 1 ก็จะทำการนับพิกเซลนั้น แต่กรณีที่มีค่าเป็น 0 ก็จะไม่นับพิกเซลนั้น ซึ่งลักษณะเหมือนการกระทำทางลอจิก AND ระหว่างบริเวณพื้นที่สี่เหลี่ยมผืนผ้าในภาพกับภาพของหน้ากากกรอบวินโดวส์ก่อนที่จะทำการนับจำนวนพิกเซลในกรอบวินโดวส์ ซึ่งในกรณีของกรอบวินโดวส์แบบหลายเหลี่ยมก็จะทำการสร้างรูปปิดหลายเหลี่ยมไว้ล่วงหน้าก่อนการตรวจสอบโดยใช้เซตของจุดเชื่อมต่อที่กำหนดดังรูปที่ 4.8 โดยที่ตำแหน่งพิกเซลในบริเวณกรอบวินโดวส์ให้มีค่าเป็น 1 และบริเวณนอกกรอบวินโดวส์ให้มีค่าเป็น 0 และเก็บค่าตำแหน่ง startx,starty,endx,endy ที่ระบุตำแหน่งของกรอบวินโดวส์นั้น จากค่า min_x, min_y ,max_x และ max_y ของรูปปิดหลายเหลี่ยมตามลำดับ ซึ่งทำให้การทำงานของการนับพิกเซลในกรอบวินโดวส์แบบนี้ลดลงเหลือเพียงการใช้หน้ากากกรอบวินโดวส์แบบสี่เหลี่ยมผืนผ้าเท่านั้น



รูปที่ 4.8 การแทนกรอบวินโดวส์แบบหลายเหลี่ยมในรูปของหน้ากากกรอบวินโดวส์ (Mask Window)

3.1.3 กรอบวินโดวส์แบบวงกลม (Circle Window)

ลักษณะของกรอบวินโดวส์แบบนี้จะมีลักษณะเป็นรูปวงกลมที่กำหนดโดยโคออร์ดิเนตของจุดศูนย์กลาง(Xc,Yc) และรัศมี (radius) ดังรูปที่ 4.6(ค) กรอบวินโดวส์แบบนี้ก็จะใช้เทคนิคการระบายสีในพื้นที่ (Area Filling) เช่นเดียวกับกรอบวินโดวส์แบบรูปปิดหลายเหลี่ยมแต่ในกรณีของพื้นที่วงกลมนั้นจะแตกต่างกันโดยที่อัลกอริทึมในการระบายสีในพื้นที่วงกลมจะใช้เทคนิคที่ดัดแปลงมาจากอัลกอริทึมในการกำเนิดจุดบนเส้นวงกลม (Circle line generation algorithm) สำหรับรายละเอียดของอัลกอริทึมเหล่านี้ได้นำมาจาก Hearn,D. and Baker,M.P. 1994 และเพื่อทำให้การทำงานของอัลกอริทึมมีความเร็ว ในวิทยานิพนธ์นี้จึงสร้างให้กรอบวินโดวส์แบบวงกลมนี้อยู่ในรูปของหน้ากากกรอบวินโดวส์แบบสี่เหลี่ยมผืนผ้าเช่นเดียวกันกับในกรณีของกรอบวินโดวส์แบบหลายเหลี่ยม

3.2. ฟังก์ชันนับพิกเซลบนเส้น (Line Pixel Counting)

ฟังก์ชันนี้จะคล้ายกับฟังก์ชันนับพิกเซลในกรอบวินโดวส์แต่จะทำการนับจำนวนพิกเซลในภาพชิ้นงานในบริเวณเส้นที่กำหนดและนำค่าที่นับได้มาเปรียบเทียบกับค่าขีดจำกัดบนและล่างเพื่อทำการตัดสินว่าชิ้นงานนั้นดีหรือเสีย และเช่นเดียวกันกับฟังก์ชันนับพิกเซลในกรอบวินโดวส์ ฟังก์ชันนี้สามารถแบ่งย่อยตามลักษณะของเส้นได้ 4 แบบ คือ

3.2.1 เส้นตรงแนวดิ่ง (Vertical Line)

เป็นเส้นตรงที่กำหนดโดยระยะแนวนอน (horizontal level), จุดเริ่มต้น (startx) และ จุดปลาย (endx) ดังรูปที่ 4.9 ก

3.2.2 เส้นตรงแนวนอน (Horizontal line)

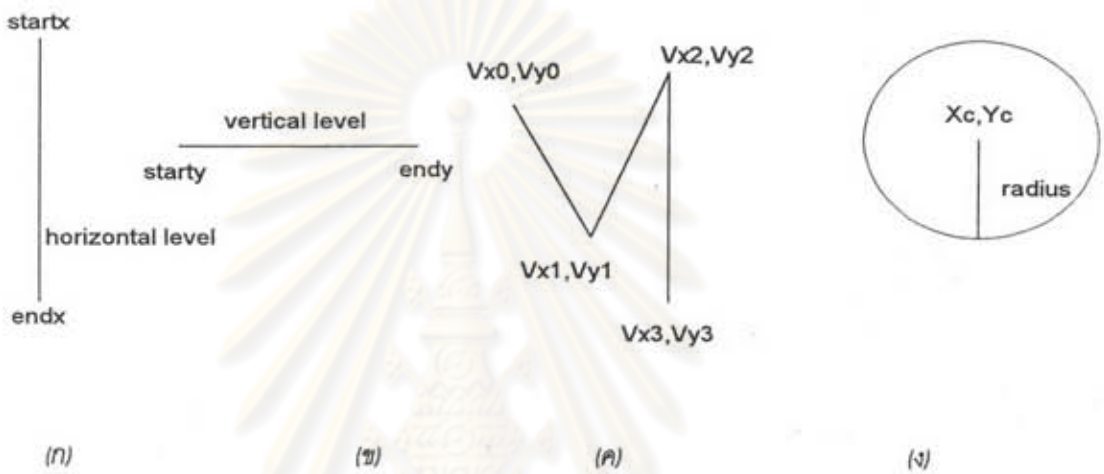
มีลักษณะเดียวกันกับเส้นตรงแนวดิ่ง แต่จะถูกกำหนดโดย ระยะแนวดิ่ง (vertical level) จุดเริ่มต้น (starty) และจุดปลาย (endy) ดังรูปที่ 4.9 ข

3.2.3 เส้นตรงหลายเหลี่ยม (Polygonal line)

เป็นเส้นตรงที่กำหนดโดยเซตของจุดเชื่อมต่อ (vertex point) {(Vx1,Vy1),(Vx2,Vy2),....,(VxN,VyN)} ดังรูปที่ 4.9 ค ซึ่งลักษณะของเส้นตรงแบบนี้จะประยุกต์ใช้หลักการของอัลกอริทึมในการกำเนิดเส้นตรงของ Bresenham (Bresenham's line generation algorithm) ซึ่งเป็นอัลกอริทึมในการคำนวณหาเซตของจุดพิกเซลที่เป็นสมาชิกของเส้นตรงที่กำหนดระหว่าง 2 จุดใดๆ โดยการนำมาใช้ในฟังก์ชันนับพิกเซลนี้ก็ทำการอ่านค่าพิกเซลในตำแหน่งที่กำหนดได้แล้วมาวิเคราะห์ว่าเป็นพิกเซลที่กำหนดหรือไม่ สำหรับรายละเอียดของอัลกอริทึมในการกำเนิดเส้นตรงนี้สามารถดูรายละเอียดได้จาก Hearn,D. and Baker,M.P. 1994

3.2.4 เส้นวงกลม (Circle line)

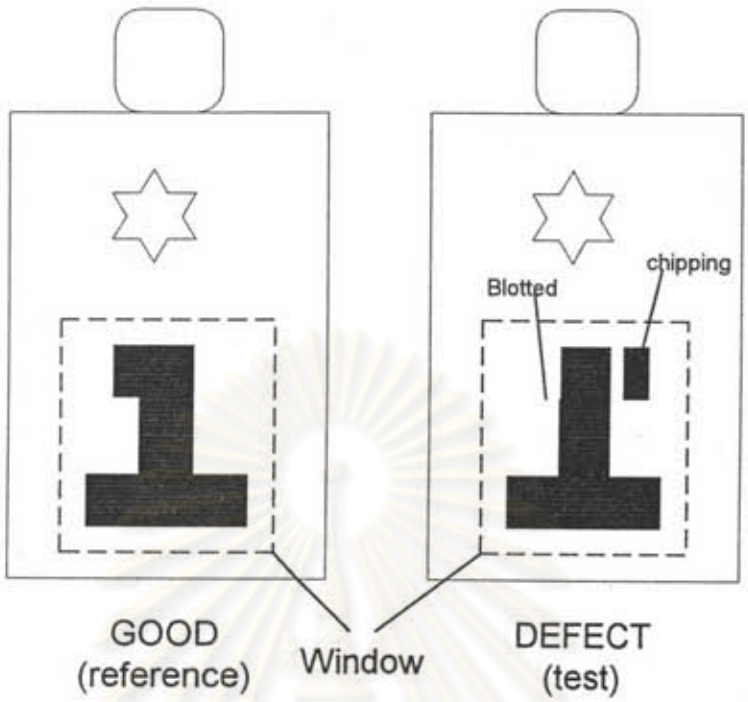
เส้นลักษณะนี้เป็นเส้นโค้งที่กำหนดโดย จุดศูนย์กลาง (Xc, Yc) และรัศมี (radius) ดังรูปที่ 4.9 ง การทำงานของเส้นวงกลมนี้จะใช้หลักการในการกำเนิดเส้นวงกลมของ Bresenham (Bresenham's circle line generation algorithm) เช่นเดียวกันกับเส้นตรงหลายเหลี่ยมซึ่งรายละเอียดของอัลกอริทึมสามารถดูได้จาก Hearn,D. and Baker,M.P. 1994



รูปที่ 4.9 ลักษณะของฟังก์ชันนับพิกเซลบนเส้นแบบต่างๆ
(ก) เส้นตรงแนวดิ่ง (ข) เส้นตรงแนวนอน (ค) เส้นตรงหลายเหลี่ยม (ง) เส้นวงกลม

3.3 ฟังก์ชันเปรียบเทียบรูปแบบในกรอบวินโดวส์ (Window Pattern Matching)

เนื่องจากฟังก์ชันนับพิกเซลในกรอบวินโดวส์และในเส้นในข้อ 3.1 และ 3.2 ที่กล่าวมาแล้วนั้นจะมีข้อเสียในกรณีที่ภาพชิ้นงานที่ถ่ายเข้ามามีจุดเสียในกรณีที่มีจำนวนพิกเซลที่ขาดหายไป (blotted) และเพิ่มเข้ามา (chipping) เท่าๆกัน ก็จะทำให้ไม่สามารถตรวจสอบได้ ดังนั้นฟังก์ชันนี้ก็จะทำการตรวจสอบโดยการเปรียบเทียบแบบจุดต่อจุด (pixel-by-pixel comparison) ระหว่างภาพอ้างอิง (refernce image) กับภาพของชิ้นงานที่ต้องการตรวจสอบ (test image) ในบริเวณพื้นที่กรอบวินโดวส์สี่เหลี่ยมที่กำหนด โดยที่จะนับจำนวนจุดที่แตกต่างกันของทั้ง 2 ภาพ (different count) แล้วนำมาเปรียบเทียบกับค่าขีดจำกัดบนและล่างเพื่อทำการตัดสิน ลักษณะของฟังก์ชันนั้นสามารถแสดงได้ดังรูปที่ 4.10



รูปที่ 4.10 ลักษณะของฟังก์ชันเปรียบเทียบรูปแบบในกรอบวินโดวส์ ซึ่งสามารถตรวจสอบฉลากที่มีจุดเสียในกรณีที่ส่วนที่ขาดหายไป(blotted)และเพิ่มเข้ามา(chipping) มีจำนวนเท่ากัน

4. ฟังก์ชันปรับตำแหน่ง (Positioning Function)

เนื่องจากการทำงานของฟังก์ชันตรวจสอบแบบต่างๆ ที่ได้กล่าวไปแล้วในหัวข้อที่แล้ว จำเป็นอย่างยิ่งที่ตำแหน่งของฟังก์ชันเหล่านั้นจะต้องวางลงไปยังตำแหน่งที่ถูกต้องบนภาพชิ้นงานที่ถ่ายเข้ามาได้จึงจะทำให้การตรวจสอบมีความถูกต้องแม่นยำ ซึ่งในขบวนการผลิตจริงแล้วส่วนของกลไกที่ทำหน้าที่ป้อนชิ้นงานเข้ามาตรวจสอบนั้นอาจจะทำให้เกิดการเลื่อนตำแหน่งของชิ้นงานได้ ถ้าไม่มีการปรับตำแหน่งของฟังก์ชันตรวจสอบให้เลื่อนตามไปด้วยก็อาจจะทำให้การตรวจสอบเกิดความผิดพลาดได้

สำหรับหลักการของการปรับตำแหน่งของฟังก์ชันตรวจสอบนั้นจะแบ่งออกเป็นการปรับตำแหน่งในแนวนอนและแนวตั้ง โดยที่การปรับตำแหน่งของทั้งสองแกนนี้จะเป็นอิสระต่อกันและการปรับตำแหน่งนั้นจะปรับตำแหน่งของฟังก์ชันตรวจสอบให้เลื่อนไปยังตำแหน่งที่ถูกต้องบนภาพชิ้นงาน โดยที่พารามิเตอร์ต่างๆที่กำหนดตำแหน่งของฟังก์ชันตรวจสอบในภาพชิ้นงานนั้นจะสามารถบวกค่าเข้าไปชดเชยการเลื่อนตำแหน่งที่เกิดขึ้นได้ ตัวอย่างเช่น กรณีของฟังก์ชันตรวจสอบโดยการนับพิกเซลในกรอบวินโดวส์แบบสี่เหลี่ยมผืนผ้าดังรูปที่ 4.11 ซึ่งกำหนดตำแหน่งของกรอบวินโดวส์ด้วยโคออร์ดิเนทจุดมุมซ้ายบน (X_s,Y_s) และจุดมุมขวาล่าง (X_e,Y_e) และตำแหน่งใหม่ของกรอบวินโดวส์ (X'_s,Y'_s) , (X'_e,Y'_e) เมื่อเกิดการเลื่อนตำแหน่งของชิ้นงานไป หาได้โดย

$$X'_s = X_s + \Delta V$$

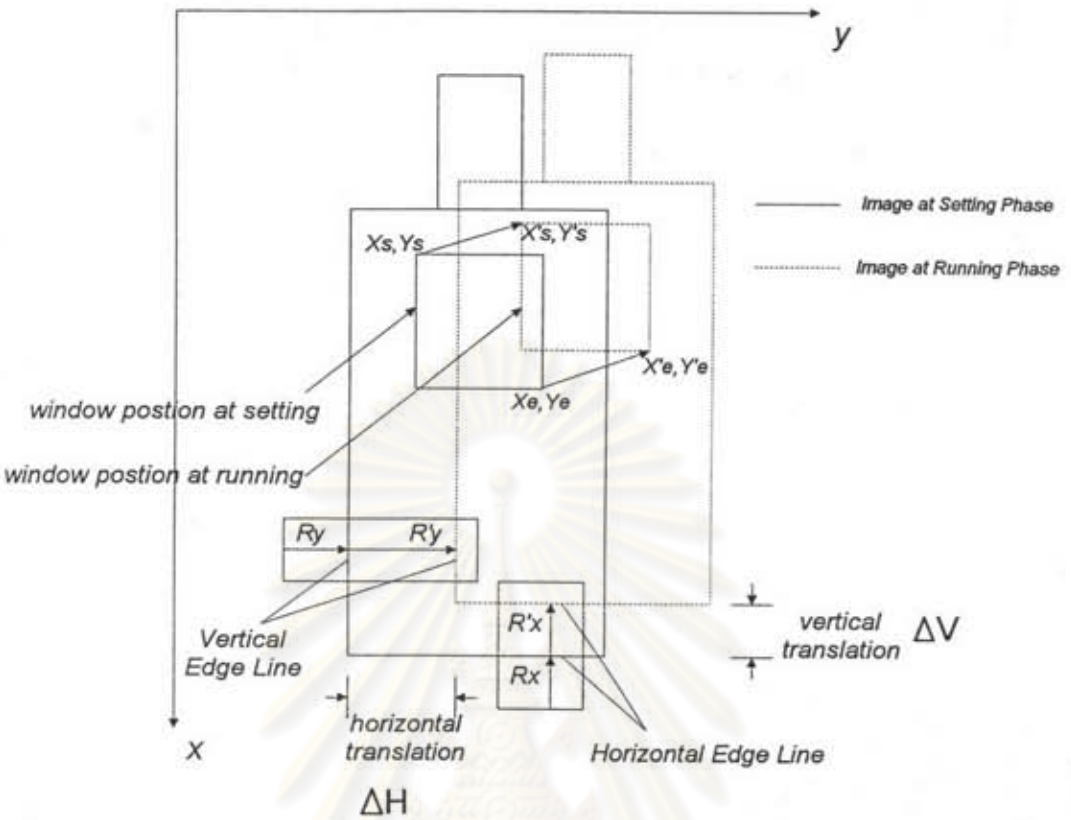
$$Y'_s = Y_s + \Delta H$$

$$X'_e = X_e + \Delta V$$

$$Y'_e = Y_e + \Delta H$$

โดยที่ค่า ΔV และ ΔH คือ ค่าการเลื่อนตำแหน่งของชิ้นงานทางด้านแนวตั้งและแนวนอน (Vertical / Horizontal Translation) ตามลำดับ

ดังนั้นจึงเป็นหน้าที่ของฟังก์ชันปรับตำแหน่งนี้ที่จะทำการคำนวณหาค่าการเลื่อนตำแหน่งของชิ้นงานซึ่งประกอบด้วยค่าการเลื่อนตำแหน่งทางด้านแนวตั้งและแนวนอน (ΔV ,ΔH) จากภาพชิ้นงานที่ถ่ายเข้ามาได้และผ่านการแปลงให้เป็นภาพสองระดับแล้ว หลักการในการหาค่าการเลื่อนตำแหน่งของชิ้นงานนี้จะเป็นการคำนวณหาค่าการเลื่อนตำแหน่งจากภาพชิ้นงานที่ถ่าย



รูปที่ 4.11 ตัวอย่างแสดงการปรับตำแหน่งของกรอบวินโดวส์แบบสี่เหลี่ยมผืนผ้า

เข้ามาได้ในขณะการตรวจสอบ (Running Phase) เทียบกับภาพชิ้นงานที่ถ่ายเข้ามาได้ในขณะตั้งค่า (Setting Phase) โดยจะทำการหาตำแหน่งของจุดอ้างอิงในภาพในขณะการตั้งค่าแล้วเก็บค่านั้นไว้เป็นค่าอ้างอิง และในขณะการตรวจสอบชิ้นงานใดๆ ภาพที่ถ่ายเข้ามาได้ก็จะนำมาคำนวณหาตำแหน่งของจุดอ้างอิงเดียวกันนี้ เพื่อนำมาหาค่าผลต่างของตำแหน่งของจุดอ้างอิงทั้งสองซึ่งก็คือค่าเลื่อนตำแหน่งของชิ้นงานนั่นเอง โดยส่วนของภาพชิ้นงานที่จะใช้เป็นจุดอ้างอิงในการคำนวณหาค่าการเลื่อนตำแหน่งในฟังก์ชันนี้จะใช้ส่วนของขอบภาพที่เป็นเส้นตรง (Edge Line) ซึ่งค่าการเลื่อนตำแหน่งทางแนวนอน (Horizontal Translation) จะใช้ส่วนของขอบภาพที่เป็นเส้นตรงในแนวตั้ง (Vertical Edge Line) เป็นจุดอ้างอิง ส่วนค่าการเลื่อนตำแหน่งทางแนวตั้ง (Vertical Translation) จะใช้ส่วนของขอบภาพที่เป็นเส้นตรงในแนวนอน (Horizontal Edge Line) เป็นจุดอ้างอิง ดังตัวอย่างรูปที่ 4.11 และการคำนวณหาค่าการเลื่อนตำแหน่งนี้สามารถแสดงด้วยสมการดังนี้

$$\Delta V = R'_x - R_x$$

$$\Delta H = R'_y - R_y$$

โดยที่

ΔV = ค่าการเลื่อนตำแหน่งทางแนวตั้ง (Vertical Translation)

ΔH = ค่าการเลื่อนตำแหน่งทางแนวนอน (Horizontal Translation)

R_x , R'_x = ตำแหน่งของเส้นขอบภาพในแนวตั้ง (Vertical Edge Line) ที่หาได้ในขณะตั้งค่า (Setting Phase) และ ขณะตรวจสอบ (Running Phase) ตามลำดับ

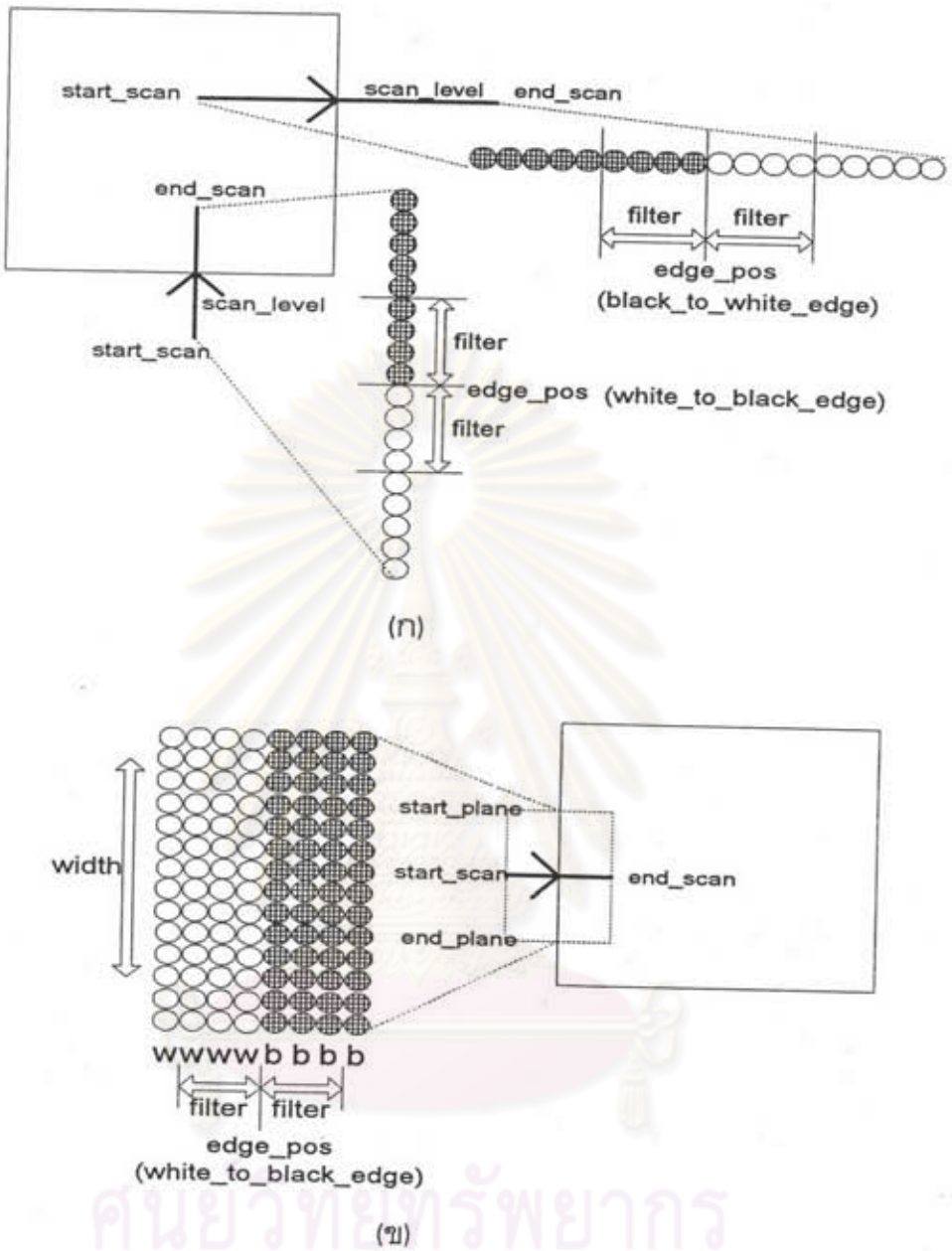
R_y , R'_y = ตำแหน่งของเส้นขอบภาพในแนวนอน (Horizontal Edge Line) ที่หาได้ในขณะตั้งค่า (Setting Phase) และ ขณะตรวจสอบ (Running Phase) ตามลำดับ

สำหรับเทคนิคในการหาตำแหน่งของขอบภาพที่เป็นเส้นตรงในฟังก์ชันนี้จะใช้เทคนิคเดียวกับที่ใช้ในระบบ Image Checker 30 (Matsushiata ,1990) โดยแบ่งออกเป็น 2 แบบ คือ

1.การหาขอบภาพในแนวเส้นตรง (Line Scanning Edge Detection)

วิธีการหาขอบภาพแบบนี้จะทำการหาตำแหน่งที่เป็นขอบภาพในบริเวณเส้นตรงใดๆที่กำหนด, ซึ่งส่วนที่เป็นขอบภาพจะนิยามจากจุดที่เป็นจุดเปลี่ยนแปลงจากพิกเซลที่มีค่า 0 เป็น 1 (black_to_white edge) หรือจาก 1 เป็น 0 (white_to_black_edge) ในเส้นตรงที่ทำการหาขอบภาพ และเพื่อป้องกันการหาขอบภาพผิดเพราะจุดที่เกิดเปลี่ยนของพิกเซลในภาพชิ้นงานนั้นอาจเกิดจากสัญญาณรบกวนได้ ดังนั้นการนิยามจุดที่เป็นขอบภาพในเส้นตรงใดๆจึงนิยามเป็นเซตของจุดที่ต่อเนื่องกัน ที่มีการเปลี่ยนจาก 0 เป็น 1 หรือ 1 เป็น 0 และบริเวณรอบข้างของจุดดังกล่าวจะต้องเป็นพิกเซลชนิดเดียวกันเป็นจำนวนที่กำหนดด้วยค่าพารามิเตอร์ "filter" สำหรับตัวอย่างของการหาขอบภาพแบบนี้ทั้งขอบภาพแนวตั้ง (Vertical Edge Line) และขอบภาพแนวนอน (Horizontal Edge Line) แสดงดังรูปที่ 4.12(ก) ซึ่งโดยสรุปแล้วพารามิเตอร์ที่ต้องการในการหาตำแหน่งของขอบภาพแบบนี้ ได้แก่

- บริเวณพื้นที่ที่ทำการหาขอบ : scan_level (horizontal_level or vertical_level) , start_scan , end_scan
- ชนิดของขอบภาพ : Black_to_White Edge (0→1) , White_to_Black Edge(1→0)
- ค่าพารามิเตอร์ filter (อยู่ในรูปของจำนวนพิกเซล)



รูปที่ 4.12 (ก) ตัวอย่างการหาขอบภาพในแนวเส้นตรงทั้งขอบภาพแนวตั้ง(vertical edge) และแนวนอน(horizontal edge) (ข) ตัวอย่างการหาขอบภาพในแนวระนาบ (ขอบภาพแนวตั้งแบบที่เปลี่ยนจาก '1' เป็น '0')

2. การหาขอบภาพในแนวระนาบ (Plane Scanning Edge Detection)

เนื่องจากการหาขอบภาพแบบแรกที่ทำการหาในแนวเส้นตรงใดๆนั้นมีโอกาสที่จะหาขอบภาพผิดได้ในกรณีที่ภาพชิ้นงานนั้นมีสัญญาณรบกวนเข้ามามาก ดังนั้นการหาขอบภาพในแบบนี้จะนิยามจุดที่เป็นขอบภาพจากตำแหน่งที่เกิดจากการเปลี่ยนแปลงค่า 0 เป็น 1 หรือ 1 เป็น 0 ใน

บริเวณระนาบ (plane) ใดๆ ดังตัวอย่างดังรูปที่ 4.12(ข) ซึ่งเป็นการหาเส้นขอบภาพแนวตั้งที่เปลี่ยนจากพิกเซลที่มีค่า 1 เป็น 0 (White_to_Black Vertical Edge Line) ซึ่งสามารถอธิบายเป็นขั้นตอนได้ดังนี้

- ในแต่ละคอลัมน์ที่ทำการค้นหาตำแหน่งของขอบภาพจะทำการระบุว่าคอลัมน์นั้นเป็นคอลัมน์ของพิกเซล 0 หรือ 1 โดยในแต่ละคอลัมน์จะทำการนับจำนวนพิกเซลที่มีค่า 0 ตั้งแต่ตำแหน่งแถวที่กำหนด โดยค่า start_plane จนถึงค่า end_plane ซึ่งถ้ามีค่าเกินค่าพารามิเตอร์ 'width' ก็จะถือว่าคอลัมน์นั้นเป็นคอลัมน์ที่มีค่า 0 (b,black) มิฉะนั้นแล้วจะถือว่าเป็นคอลัมน์ที่มีค่า 1 (w,white)

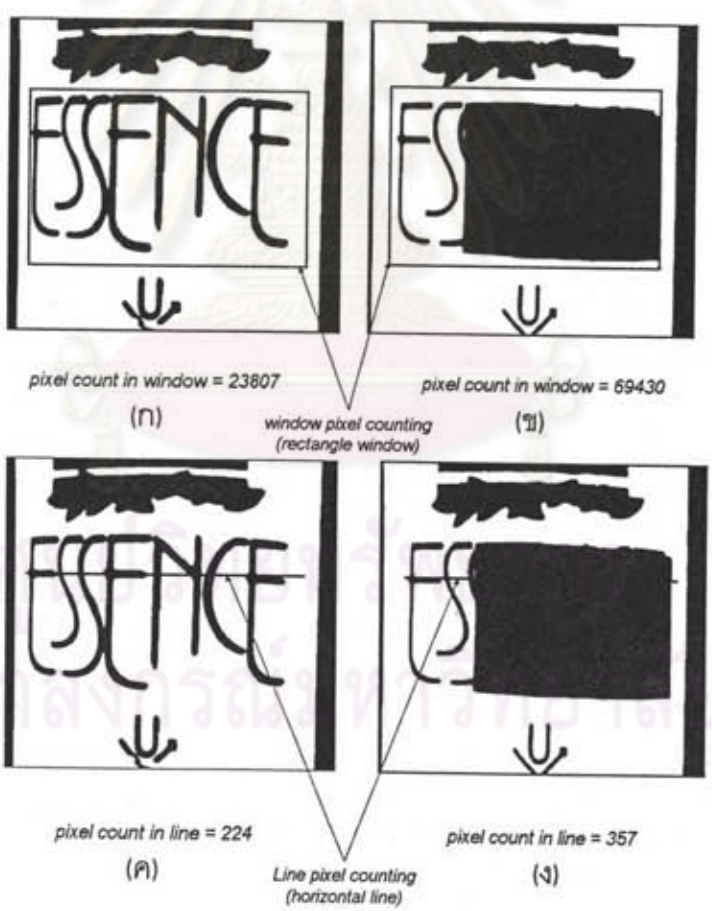
- หลังจากที่ทำการระบุค่าให้แก่แต่ละคอลัมน์ที่กำหนดให้มีค่าเป็น 0 หรือ 1 จากขั้นตอนที่แล้ว ก็จะเป็นการลดข้อมูลจนเหลือเพียง 1 มิติ เช่นเดียวกันกับในกรณีการหาขอบภาพในแนวเส้นตรง (Line Scanning Edge Detection) จึงทำการค้นหาตำแหน่งของคอลัมน์ที่เปลี่ยนจาก 1 เป็น 0 (white→black) และดูค่าของคอลัมน์รอบข้างทั้งทางซ้ายและขวาจะต้องมีค่าเดียวกัน โดยใช้ค่าพารามิเตอร์ "filter" เป็นตัวกำหนดขอบเขตของคอลัมน์รอบข้างที่จะใช้ในการพิจารณาเช่นเดียวกันกับในกรณีของการหาขอบภาพในแบบแรก ซึ่งโดยสรุปแล้วพารามิเตอร์ที่ต้องการในการหาขอบภาพแบบนี้ ได้แก่

  - บริเวณพื้นที่ที่ทำการค้นหา : start_plane , end_plane , start_scan , end_scan
  - ชนิดของขอบภาพ : Black_to_White Edge (0→1) , White_to_Black Edge(1→0)
  - ค่าพารามิเตอร์ filter และ width (อยู่ในรูปของจำนวนพิกเซล)

ซึ่งกล่าวโดยสรุปแล้วการปรับตำแหน่งของฟังก์ชันตรวจสอบให้ลงตรงตำแหน่งที่ถูกต้องในภาพชิ้นงานนั้นจะใช้การทำงานร่วมกันของสองฟังก์ชันคือฟังก์ชันปรับตำแหน่งและฟังก์ชันตรวจสอบ โดยฟังก์ชันปรับตำแหน่งจะแบ่งย่อยออกเป็นฟังก์ชันปรับตำแหน่งด้านแนวตั้งที่จะใช้ส่วนของขอบภาพในแนวนอนเป็นจุดอ้างอิงในการคำนวณค่าการเลื่อนตำแหน่งในแนวตั้ง และฟังก์ชันปรับตำแหน่งด้านแนวนอนที่จะใช้ส่วนของขอบภาพในแนวตั้งเป็นจุดอ้างอิงในการคำนวณค่าการเลื่อนตำแหน่งในแนวนอน หลังจากนั้นฟังก์ชันตรวจสอบแบบต่างๆจะทำการบวกค่าการเลื่อนตำแหน่งนี้ชดเชยเข้าไปกับตำแหน่งพื้นที่ในการตรวจสอบของฟังก์ชันตรวจสอบเหล่านั้น ดังนั้นในการสร้างฟังก์ชันปรับตำแหน่งและฟังก์ชันตรวจสอบเพื่อนำไปใช้งานจะต้องมีส่วนที่สัมพันธ์กันสามารถอ้างถึงกันได้โดยจะสร้างฟังก์ชันปรับตำแหน่งไว้หลายๆฟังก์ชันเพื่อให้สามารถคำนวณค่าการเลื่อนตำแหน่งของชิ้นงานได้หลายๆบริเวณในภาพที่ถ่ายเข้ามาได้ หลังจากนั้นในฟังก์ชัน

ตรวจสอบจะมีพารามิเตอร์ที่อ้างถึงว่าจะใช้ค่าการเลื่อนตำแหน่งที่คำนวณได้จากฟังก์ชันปรับตำแหน่งหมายเลขใด

สำหรับตัวอย่างการนำเอาฟังก์ชันต่างๆไปใช้งานร่วมกันในการตรวจสอบสามารถแสดงได้ดังรูปที่ 4.13 โดยเป็นการนำเอาไปใช้ในการตรวจสอบหาจุดบกพร่องของฉลากที่พิมพ์บนขวด โดยใช้ฟังก์ชันนับพิกเซลในกรอบของวินโดวส์และฟังก์ชันนับพิกเซลบนเส้น จากรูปดังกล่าวจะเห็นได้ว่าในกรณีฉลากที่พิมพ์ดีในรูปที่ 4.13 ก และ ค จะนับจำนวนพิกเซลในกรอบวินโดวส์และเส้นได้เท่ากับ 23807 และ 227 ตามลำดับส่วนกรณีของฉลากที่พิมพ์เสียดังรูปที่ 4.13 ข และ ง จะนับจำนวนพิกเซลในกรอบวินโดวส์และเส้นได้เท่ากับ 69430 และ 357 ตามลำดับ ซึ่งถ้ากำหนดค่าขีดจำกัดบนและล่างที่เหมาะสมของฟังก์ชันตรวจสอบทั้งสองก็จะทำให้สามารถแยกแยะฉลากที่พิมพ์ดีออกจากฉลากที่พิมพ์เสียได้



รูปที่ 4.13 ตัวอย่างการนำฟังก์ชันการตรวจสอบแบบใช้งานทั่วไป ไปตรวจสอบจุดบกพร่องฉลากที่พิมพ์บนขวด (ก) (ข) การใช้ฟังก์ชันนับพิกเซลในกรอบวินโดวส์ตรวจสอบฉลากที่พิมพ์ดีและบกพร่องตามลำดับ (ค) (ง) การใช้ฟังก์ชันนับพิกเซลบนเส้นตรวจสอบฉลากที่พิมพ์ดีและบกพร่องตามลำดับ

4.4 การพัฒนาอัลกอริทึมเฉพาะสำหรับการตรวจสอบฉลากที่พิมพ์บนขวด (แนวทางที่ 2)

เนื่องจากอัลกอริทึมที่พัฒนาขึ้นตามแนวทางที่ 1 ที่ได้อธิบายไปแล้วนั้นเป็นอัลกอริทึมที่ใช้งานได้ทั่วไปไม่ได้เจาะจงสำหรับชิ้นงานใดๆ โดยเฉพาะ ซึ่งหลังจากที่ได้ทำการพัฒนาฟังก์ชันต่างๆ แล้ว ได้นำมาทดลองใช้ในการตรวจสอบฉลากที่พิมพ์บนขวด พบว่าอัลกอริทึมแบบดังกล่าวนั้นมีข้อเสียต่างๆ ที่ไม่เหมาะสำหรับนำมาใช้ในการตรวจสอบฉลากที่พิมพ์บนขวด ดังนี้

1. อัลกอริทึมแบบนี้อยู่บนพื้นฐานของการนับจำนวนพิกเซลซึ่งในกรณีที่ชิ้นงานนั้นมีจุดบกพร่องที่มีส่วนที่ขาดหายไปและเพิ่มเข้ามาในปริมาณของจำนวนพิกเซลที่ใกล้เคียงกันก็จะทำให้ไม่สามารถตรวจสอบได้ และถึงแม้ว่าในอัลกอริทึมแบบนี้จะมีฟังก์ชันการเปรียบเทียบรูปแบบแบบจุดต่อจุดที่ได้กล่าวมาแล้ว ซึ่งสามารถจัดการกับปัญหาดังกล่าวได้ แต่ในความเป็นจริงแล้วพบว่าฟังก์ชันนี้จะทำงานได้ผลถูกต้องก็ต่อเมื่อภาพอ้างอิงกับภาพที่ต้องการตรวจสอบตรงกันสนิทก่อนที่จะทำการเปรียบเทียบกัน ซึ่งพบว่าหลักการในการปรับตำแหน่งของอัลกอริทึมแบบดังกล่าวนั้นยังไม่ละเอียดพอที่จะทำให้ภาพทั้งสองตรงกันสนิทและผลในการเปรียบเทียบถูกต้องแม่นยำพอที่จะนำไปใช้งานจริงได้

2. การประมวลผลต่างๆในการตรวจสอบของอัลกอริทึมนี้เป็นการประมวลผลบนภาพแบบ 2 ระดับ ซึ่งต้องผ่านการแปลงข้อมูลภาพจากภาพ 256 ระดับ โดยใช้ค่าขีดเริ่มเปลี่ยน ซึ่งค่านี้จะมีผลต่อจำนวนพิกเซลที่ปรากฏบนภาพแบบสองระดับ และยังขึ้นอยู่กับสภาพของแสงที่ส่องเข้าหาชิ้นงานด้วยจึงทำให้ค่าจำนวนพิกเซลที่วัดได้ในแต่ละครั้งมีความแปรปรวนสูงกว่าค่าจำนวนพิกเซลที่เพิ่มเข้ามาหรือขาดหายไปของจุดบกพร่องที่มีขนาดเล็กๆ ทำให้ไม่สามารถแยกชิ้นงานที่มีจุดบกพร่องขนาดเล็กออกจากชิ้นงานดีได้เพราะค่าจำนวนพิกเซลที่นับได้มีค่าใกล้เคียงกัน

3. ในการตรวจสอบฉลากที่พิมพ์บนขวดนั้นโดยทั่วไปแล้วจะมีส่วนของกลไกในการป้อนชิ้นงานเข้ามาตรวจสอบ ซึ่งในกรณีของขวดที่มีรูปร่างโค้งกลมนั้นทำให้มีโอกาสที่จะหมุนรอบแกนตั้งของขวดได้หลังจากที่ได้ป้อนขวดเข้ามาตรวจสอบ ผลของการหมุนของขวดนี้ทำให้ภาพชิ้นงานที่ถ่ายเข้ามาได้ในแต่ละครั้งนั้นมีความแตกต่างกัน ทำให้ค่าจำนวนพิกเซลที่นับได้ในแต่ละครั้งก็จะแตกต่างตามไปด้วยจึงมีผลให้ขอบเขตของจำนวนพิกเซลของชิ้นงานที่ดีและชิ้นงานที่มีจุดบกพร่องขนาดเล็กที่มีการหมุนไปนั้นมีค่าใกล้เคียงกัน ทำให้ตรวจสอบผิดพลาดเช่นเดียวกับผลในข้อที่ 2

ด้วยเหตุผลต่างๆดังกล่าวจึงทำให้การพัฒนาอัลกอริทึมเฉพาะสำหรับการตรวจสอบฉลากที่พิมพ์บนขวดนั้นพยายามที่จะหาทางแก้ปัญหาและข้อเสียต่างๆที่ได้กล่าวมาแล้ว เพื่อให้สามารถตรวจสอบฉลากที่พิมพ์บนขวดได้ผลดียิ่งขึ้น

4.4.1 งานวิจัยที่ใกล้เคียง

ในขั้นตอนการพัฒนาอัลกอริทึมในการตรวจสอบนั้นได้เริ่มจากการสำรวจงานวิจัยที่ใกล้เคียง ซึ่งจะได้กล่าวถึงข้างล่าง โดยพบว่าอัลกอริทึมในการตรวจสอบต่างๆ ในงานวิจัยที่ได้ทำการศึกษานั้นอยู่บนพื้นฐานของเทคนิคการเปรียบเทียบภาพแบบจุดต่อจุด (Pixel by Pixel Image Comparison) (Fu,K.S.,1982) โดยในแต่ละงานวิจัยนั้นจะแตกต่างกันไปตามการประยุกต์ใช้งานและเทคนิคในการสร้าง (implementation) เพื่อทำให้การทำงานรวดเร็ว สำหรับบทความและงานวิจัยที่ใกล้เคียงที่ได้ทำการสำรวจและศึกษามาและได้กล่าวสรุปในวิทยานิพนธ์นี้มีดังต่อไปนี้

4.4.1.1 งานวิจัยของ Truchelet,F et.al 1993 (Tampoprint Inspecton by Artificial Vision)

เป็นบทความที่กล่าวถึงการพัฒนาระบบตรวจสอบภาพพิมพ์ที่เกิดจากการปั๊มบนแผ่นโลหะ (Tampoprint Impression) โดยที่อัลกอริทึมที่พัฒนาขึ้นนั้นจะอยู่บนพื้นฐานเทคนิคการเปรียบเทียบภาพ 256 ระดับ (grey-scale image comparison) ระหว่างภาพอ้างอิงกับภาพที่ต้องการตรวจสอบโดยก่อนที่จะทำการเปรียบเทียบนั้นจะทำการปรับตำแหน่งแบบครึ่งพิกเซล(Subpixel Alignment) ระหว่างภาพทั้งสองให้ตรงกันก่อน และในบทความนี้ได้นำเสนอวิธีการแปลงภาพสองระดับแบบยืดหยุ่น (Adaptive Thresholding) เพื่อทำการแปลงภาพที่เกิดจากการลบกันให้เป็นภาพสองระดับ ซึ่งบริเวณที่เป็นจุดบกพร่องจะมีค่าเป็น 1 ส่วนบริเวณอื่นๆ จะมีค่าเป็น 0 หลังจากนั้นก็จะทำการนับจำนวนพิกเซลที่เป็นจุดบกพร่อง(ค่า 1) แล้วนำมาค่านี้มาเปรียบเทียบกับค่าที่กำหนดเพื่อทำการตัดสินว่าชิ้นงานนั้นดีหรือเสีย

4.4.1.2 งานวิจัยของ Sato,K. et.al 1991 (System for inspecting pad-printed character using the normalized correlaion of the segmental character image)

บทความนี้จะทำการพัฒนาระบบสำหรับการตรวจสอบตัวอักษรที่พิมพ์บนตลับเทปวีดีโอ (pad-printed character on video cassette) อัลกอริทึมนี้อยู่บนพื้นฐานเทคนิค Template Matching โดยจะทำการวัดค่า Normalized Correlation ระหว่างภาพตัวอักษรต้นแบบกับภาพตัวอักษรที่ทำการตรวจสอบมาใช้ในการตัดสิน และสามารถแบ่งออกเป็น 3 ส่วน คือ

- การดึงภาพตัวอักษร (Character Extraction)

ทำหน้าที่ในการแยกส่วนของภาพตัวอักษรที่พิมพ์อยู่บนตลับเทปออกมาเพื่อทำการตรวจสอบที่ละตัว โดยจะใช้วิธี Horizontal/Vertical Projection Profile ที่ใช้ในงานวิจัยด้านการรู้จำตัวอักษร (Character Recognition)

- การปรับตำแหน่ง (Positioning)

ตัวอักษรที่ดึงออกมาได้จากส่วนที่แล้ว ถ้านำมาเปรียบเทียบกับตัวอักษรต้นแบบที่เก็บไว้ก็อาจจะผิดพลาดได้เนื่องจากยังไม่ตรงกันสนิท ส่วนนี้จึงจะทำการปรับตำแหน่งให้ตรงกันระหว่างภาพตัวอักษรอ้างอิงกับภาพตัวอักษรที่จะตรวจสอบ โดยการวัดค่า Normalized Correlation ของ Vertical / Horizontal Profile ของตัวอักษรทั้งสองในการหาตำแหน่งที่ตรงกันของภาพทั้งสอง

● การตรวจสอบแบบแยกส่วนโดยใช้ค่าคอร์รีเรชั่น (Segmental Inspection using Correlation)

ส่วนนี้จะทำการตรวจสอบโดยการวัดค่า Normalized Correlation ระหว่างภาพตัวอักษรอ้างอิงกับภาพตัวอักษรที่จะตรวจสอบโดยจะทำการแบ่งตัวอักษรทั้งสองออกเป็นส่วนๆ (segmented character) และหาค่าคอร์รีเรชั่นในแต่ละส่วนนั้นแล้วนำมาเปรียบเทียบกับค่าขีดเริ่มเปลี่ยนที่กำหนดไว้เบื้องต้น เหตุที่ต้องแบ่งตัวอักษรออกเป็นส่วนๆเพื่อที่จะทำให้สามารถตรวจสอบหาจุดบกพร่องขนาดเล็กในแต่ละส่วนของตัวอักษรนั้นได้

4.4.1.3 งานวิจัยของ Mehini,B. 1989 (Fast Visual Inspection for Quality Control)

เป็นงานวิจัยที่ทำการพัฒนาระบบการตรวจสอบสำหรับการตรวจสอบฉลากที่พิมพ์บนถังบรรจุ (Printed Container) โดยได้พัฒนาทั้งส่วนของอัลกอริทึมและส่วนของฮาร์ดแวร์ ซึ่งจะทำให้ระบบทำงานเร็วเท่ากับความเร็วของขบวนการผลิต ในส่วนของอัลกอริทึมในการตรวจสอบนั้นเป็นการรวมกันระหว่างวิธีการเปรียบเทียบภาพแบบจุดต่อจุดและวิธีตรวจสอบคุณลักษณะคือจะนำภาพที่เกิดจากการลบกันระหว่างภาพอ้างอิงและภาพที่จะตรวจสอบและผ่านการทำให้เป็นภาพสองระดับแล้ว มาทำการหา n-tuple vector ที่ได้นิยามไว้เพื่อนำมาทำการตรวจสอบคุณลักษณะโดยวิเคราะห์หาค่าความถี่ (occurence) ในการเกิดของแต่ละสมาชิกใน n-tuple vector และตัดสินจากค่านี้ว่าชิ้นงานใดเป็นชิ้นงานดีหรือเสีย

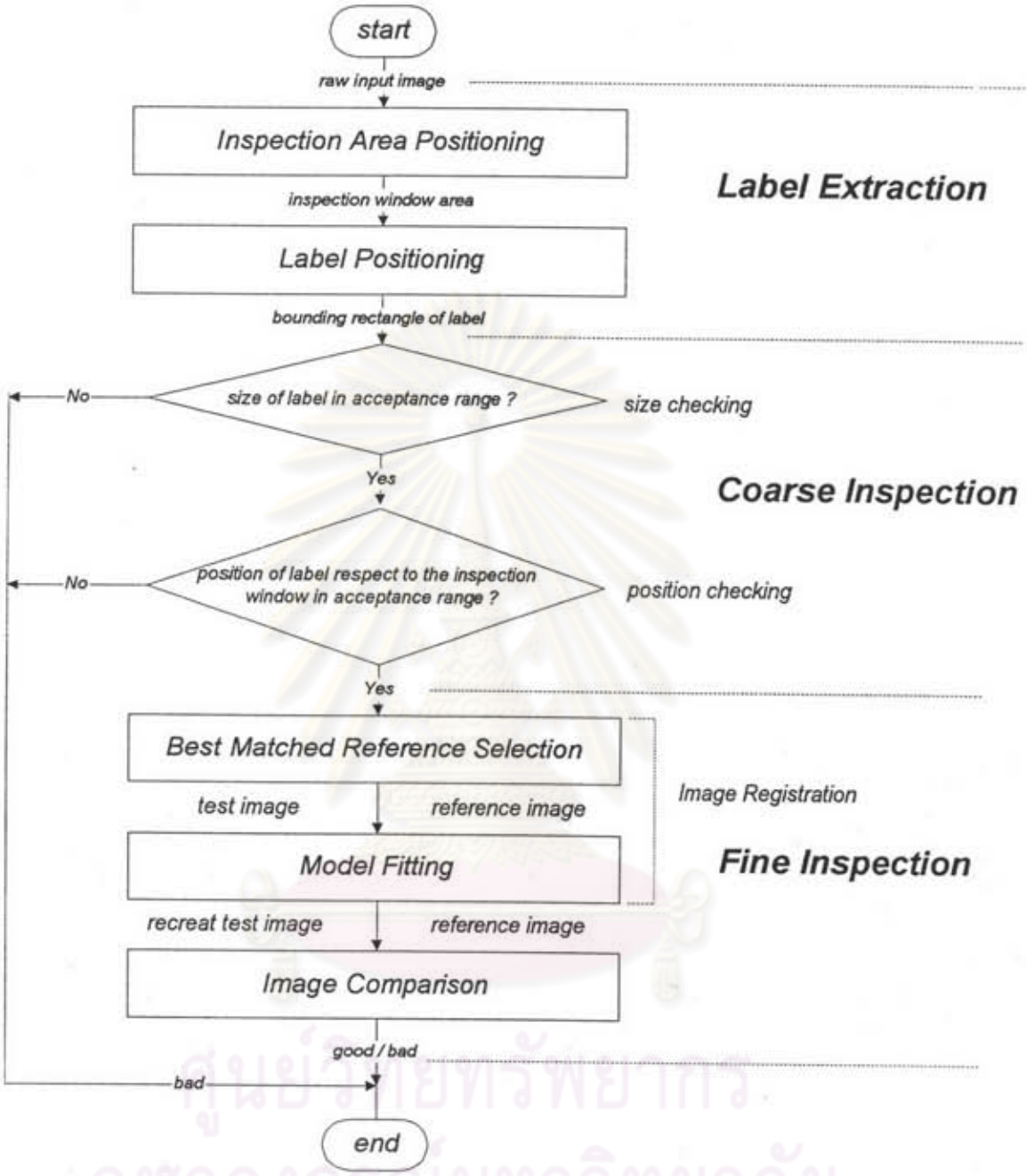
4.4.2อัลกอริทึมเฉพาะสำหรับการตรวจสอบฉลากที่พิมพ์บนขวดที่พัฒนาขึ้นในวิทยานิพนธ์ (แนวทางที่ 2)

จากการสำรวจงานวิจัยที่ใกล้เคียงพบว่าอัลกอริทึมที่พัฒนาขึ้นในบทความที่กล่าวไปแล้วนั้นจะอยู่บนพื้นฐานเทคนิคการเปรียบเทียบภาพแบบจุดต่อจุด โดยจะต้องมีการเก็บภาพอ้างอิงที่ไม่มีจุดบกพร่องไว้ล่วงหน้า แต่ในบทความทั้งหมดที่กล่าวไปแล้วไม่สามารถที่จะจัดการกับปัญหาที่เกิดขึ้นโดยเฉพาะในกรณีของการตรวจสอบฉลากที่พิมพ์บนขวด กล่าวคือเมื่อขวดที่ต้องการตรวจสอบถูกป้อนเข้ามายังแท่นยึดชิ้นงาน (JIG) อาจจะทำให้มีการหมุนของขวดรอบแกนตั้งของขวดได้ ซึ่งจะมีผลทำให้ภาพของฉลากที่ถ่ายเข้ามาได้มีการเพี้ยน (Distortion) ไปจากภาพฉลากอ้างอิงที่เก็บไว้ โดยที่ในแต่ละพิกเซลของภาพจะมีการเลื่อนตำแหน่งไปไม่เท่ากัน ซึ่งอัลกอริทึมในส่วนของการปรับตำแหน่งของภาพทั้งสองให้ตรงกันที่เสนอไว้ใน Sato,K. et.al 1991 และ Truchelet,F et.al 1993 นั้นสามารถที่จะจัดการกับภาพที่มีการเลื่อนตำแหน่งในระนาบ 2 มิติซึ่งแต่ละพิกเซลของภาพจะมีการเลื่อนตำแหน่งไปเท่ากันเท่านั้น แต่ไม่สามารถนำมาใช้กับภาพที่มีการเลื่อนตำแหน่งในระนาบ 3 มิติ ดังเช่นในกรณีของขวดนี้ได้ ส่วนใน Mehini,B. 1989 ไม่ได้กล่าวถึงปัญหาตรงจุดนี้ไว้เพียงแต่ตั้งสมมติฐานไว้ว่าจะไม่มีการเลื่อนตำแหน่งเนื่องจากการหมุนของชิ้นงาน ดังนั้นอัลกอริทึมที่ทำการพัฒนาขึ้นในวิทยานิพนธ์นี้จะพยายามที่จะหาทางแก้ปัญหาดังกล่าวให้ได้ โดยที่หลักการพื้นฐานยังคงอยู่บนวิธีการเปรียบเทียบภาพแบบจุดต่อจุดและจะมุ่งเน้นการพัฒนาอัลกอริทึมในส่วนของเทคนิคการปรับตำแหน่งให้ภาพทั้งสองตรงกันสนิทก่อนที่จะทำการเปรียบเทียบภาพ เพื่อที่จะทำให้ผลในการเปรียบเทียบถูกต้องแม่นยำยิ่งขึ้น

หลักการเบื้องต้นของอัลกอริทึม

สำหรับอัลกอริทึมที่พัฒนาขึ้นนั้นจะประกอบไปด้วย 3 ส่วนหลัก แสดงดังแผนภูมิการทำงานดังรูปที่ 4.14 คือ *การดึงภาพฉลากที่ทำการตรวจสอบ (Label Extraction) การตรวจสอบแบบหยาบ (Coarse Inspection) และการตรวจสอบแบบละเอียด (Fine Inspection)* ในส่วนของการดึงภาพฉลากนั้นจะทำการดึงเฉพาะส่วนภาพฉลากที่สนใจออกจากข้อมูลภาพที่ถ่ายเข้ามาได้ (raw input image) โดยสามารถแบ่งย่อยได้เป็น 2 ขั้นตอน คือ การหาตำแหน่งพื้นที่ในการตรวจสอบ (Inspection Area Positioning) ซึ่งจะใช้การกำหนดพื้นที่สี่เหลี่ยมในการตรวจสอบล่วงหน้า โดยพื้นที่นี้สามารถปรับเลื่อนตำแหน่งไปตามการเลื่อนตำแหน่งของชิ้นงานในแนวตั้งและแนวนอน ขั้นตอนต่อมาก็คือ การหาตำแหน่งของฉลากที่ตรวจสอบ (Label Positioning) ในบริเวณพื้นที่ในการตรวจสอบที่หามาได้จากขั้นตอนที่แล้วซึ่งจะออกมาอยู่ในรูปตำแหน่งของสี่เหลี่ยมที่ล้อมรอบฉลากนั้น (bounding rectangle of label)

หลังจากที่ได้ตำแหน่งของฉลากที่ต้องการตรวจสอบและตำแหน่งของพื้นที่ในการตรวจสอบที่หามาได้แล้วก็จะนำค่าทั้งสองมาทำการตรวจสอบแบบหยาบ ซึ่งแบ่งการตรวจสอบออกเป็นสองส่วนคือ การตรวจสอบขนาดของฉลากและการตรวจสอบตำแหน่งของฉลากเทียบกับตำแหน่งพื้นที่ในการตรวจสอบ ซึ่งทำให้สามารถตรวจสอบฉลากที่มีจุดบกพร่องอย่างชัดเจนได้ เช่น ฉลากที่มีการพิมพ์ขาดหายไปมากๆ หรือ ฉลากที่มีการพิมพ์เลื่อนตำแหน่งไป สำหรับฉลากที่ผ่านการตรวจสอบแบบหยาบมาได้นั้น ก็อาจจะเป็นฉลากที่พิมพ์ดีหรือเสียก็ได้ ดังนั้นจึงต้องนำมาตรวจสอบแบบละเอียดโดยการเปรียบเทียบภาพฉลากที่ตรวจสอบกับภาพฉลากอ้างอิงที่ไม่มีจุดบกพร่องที่เก็บไว้ล่วงหน้าโดยก่อนที่จะทำการเปรียบเทียบจะต้องทำให้ภาพทั้งสองตรงกันสนิทก่อน (Image Registration) และในกรณีของการตรวจสอบฉลากที่พิมพ์บนขวดนั้นจะมีปัญหาเรื่องการหมุนของขวดรอบแกนตั้งที่จะทำให้ภาพฉลากที่ถ่ายเข้ามาได้ในแต่ละครั้งจะมีการเพี้ยนไปจากภาพอ้างอิงที่เก็บไว้ จึงทำให้ยากต่อการทำให้ภาพทั้งสองตรงกันสนิท ดังนั้นในอัลกอริทึมนี้จึงแก้ปัญหาโดยการเก็บภาพฉลากอ้างอิงไว้หลายๆภาพในมุมต่างๆที่ขวดสามารถหมุนไปได้ โดยเมื่อดึงภาพฉลากที่ต้องการตรวจสอบออกมาได้ก็จะนำมาหาภาพฉลากอ้างอิงที่ตรงกับภาพฉลากนี้มากที่สุด (Best Matched Reference Selection) ภาพฉลากอ้างอิงที่ได้มานี้และภาพฉลากที่ตรวจสอบอาจจะยังไม่ตรงกันสนิทดีเนื่องจากว่าไม่สามารถเก็บภาพฉลากอ้างอิงได้ทุกมุมที่ขวดสามารถหมุนไปได้ ดังนั้นจึงต้องทำการปรับภาพให้ตรงกันกับภาพต้นแบบ (Model Fitting) โดยจะทำการปรับภาพฉลากที่ตรวจสอบให้ตรงกันกับภาพฉลากอ้างอิง ซึ่งรายละเอียดในส่วนนี้จะอธิบายต่อไป เมื่อภาพทั้งสองตรงกันสนิทแล้วก็จะทำการเปรียบเทียบภาพ (Image Comparison) โดยการเปรียบเทียบค่าความสว่างของภาพทั้งสองแบบจุดต่อจุด ซึ่งส่วนที่เป็นจุดบกพร่องก็คือจุดที่มีความแตกต่างทางค่าความสว่างมาก โดยจะใช้ค่าขีดเริ่มเปลี่ยนในการตัดสินว่าค่าความแตกต่างเท่าใดจึงจะถือว่าพิกเซลนั้นเป็นจุดบกพร่องและจะทำการนับจำนวนพิกเซลที่เป็นจุดบกพร่องทั้งหมดเพื่อนำมาตัดสินว่าฉลากนั้นพิมพ์ดีหรือเสีย



รูปที่ 4.14 แผนภูมิสายงานของอัลกอริทึมเฉพาะสำหรับการตรวจสอบฉลากที่พิมพ์บนขวด

<u>รายละเอียดของอัลกอริทึม</u>

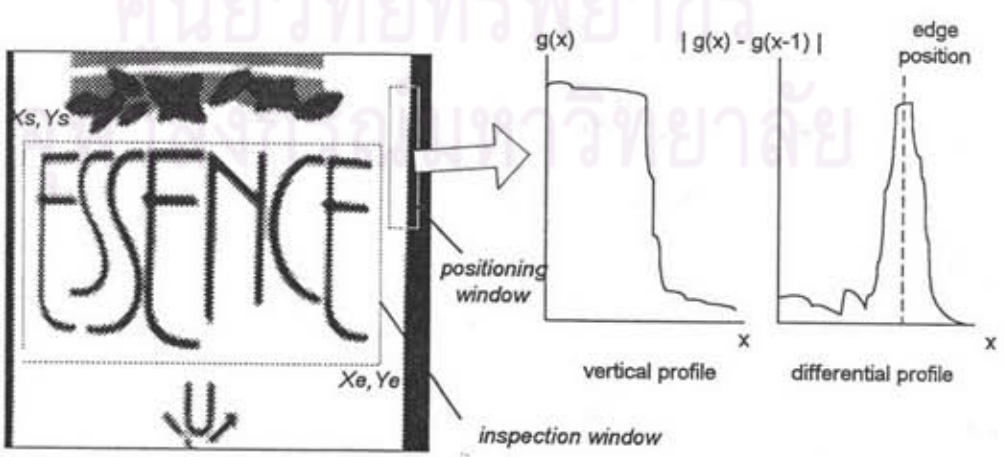
จากหลักการเบื้องต้นของอัลกอริทึมที่ได้กล่าวไปแล้ว ในหัวข้อนี้จะอธิบายรายละเอียดการทำงานของแต่ละส่วน ดังนี้

1.การดึงภาพฉลากที่ตรวจสอบ (Label Extraction)

ส่วนนี้จะทำหน้าที่ในการดึงภาพของฉลากที่ต้องการตรวจสอบออกมาจากภาพขวดที่ถ่ายเข้ามาได้ ซึ่งสามารถแบ่งออกได้เป็น 2 ส่วน คือ

1.1 การหาตำแหน่งของพื้นที่ในการตรวจสอบ(Inspection Area Positioning)

การหาตำแหน่งของพื้นที่ในการตรวจสอบ (inspection area) นี้จะใช้วิธีกรอบวินโดวส์ (window method) (Ejiri,M 1990) ซึ่งเป็นวิธีที่ใช้ในอัลกอริทึมแบบแรก โดยจะเป็นการกำหนดบริเวณที่จะทำการค้นหาส่วนของฉลากที่ทำการตรวจสอบอยู่ภายในพื้นที่กรอบวินโดวส์เท่านั้น สำหรับกรอบวินโดวส์ที่ใช้จะเป็นแบบสี่เหลี่ยมผืนผ้าที่กำหนดโดยตำแหน่งจุดมุมซ้ายบน (Xs,Ys) และมุมล่างขวา (Xe,Ye) ในพิกัดของภาพชิ้นงานที่ถ่ายเข้ามาได้และจะขอเรียกกรอบวินโดวส์นี้ว่า *inspection window* ดังตัวอย่างรูปที่ 4.15 ซึ่งภายในบริเวณกรอบวินโดวส์ที่จะทำการตรวจสอบ (inspection window) จะมีข้อกำหนดว่าจะต้องมีเพียงส่วนของฉลากที่ต้องการตรวจสอบอยู่บนพื้นผิวของขวดเท่านั้นและภายในกรอบวินโดวส์นี้ก็จะนำมาทำการค้นหาส่วนของฉลากในข้อ 1.2 ต่อไป สำหรับวิธีดังกล่าวข้างต้นนั้นจึงทำให้เกิดข้อกำหนดในการตรวจสอบว่า หลังจากที่ได้กำหนดตำแหน่งของกรอบวินโดวส์ที่ทำการตรวจสอบไปแล้วนั้น ขวดที่เข้ามาทำการตรวจสอบที่ป้อนเข้ามาอยู่ต่อหน้ากล้องที่จะถ่ายภาพจะต้องไม่มีการเลื่อนตำแหน่งไปจากตำแหน่งเดิมที่กำหนดไว้ ซึ่งในทางปฏิบัติแล้วในส่วนของกลไกที่ทำการป้อนชิ้นงานเข้ามาตรวจสอบนั้นก็จะป้อนเข้ามาในแท่นยึดขวด (Jig) ที่กำหนดตำแหน่งแน่นอนได้



รูปที่ 4.15 การหาตำแหน่งของพื้นที่ในการตรวจสอบ

การปรับตำแหน่งของกรอบวินโดวส์

จากที่กล่าวมาแล้วว่าหลังจากที่กำหนดตำแหน่งของกรอบวินโดวส์ที่จะทำการตรวจสอบแล้วตำแหน่งของฉลากบนขวดจะต้องอยู่ภายในบริเวณนี้เท่านั้น ซึ่งหมายความว่าขวดที่เข้ามาตรวจสอบจะต้องไม่มีการเลื่อนตำแหน่งไปจากตำแหน่งนี้ อย่างไรก็ตามในทางปฏิบัติแล้วถ้าในส่วนของกลไกที่ทำการป้อนขวดเข้ามาตรวจสอบนั้นไม่มีความถูกต้องแม่นยำก็อาจจะทำขวดที่ถูกป้อนเข้ามามีการเลื่อนตำแหน่งไปได้เล็กน้อย โดยการเลื่อนตำแหน่งของขวดนี้อาจแบ่งได้เป็น การเลื่อนตำแหน่งในแนวตั้ง,การเลื่อนตำแหน่งในแนวนอนและการเลื่อนตำแหน่งโดยการหมุนของขวดรอบแกนตั้งของขวด ในการแก้ปัญหาการเลื่อนตำแหน่งทางแนวตั้งและแนวนอนนั้นสามารถจัดการได้ในอัลกอริทึมในส่วนนี้ ส่วนการแก้ปัญหาการหมุนนั้นจะกล่าวถึงต่อไป

สำหรับหลักการในการแก้ปัญหาการเลื่อนตำแหน่งในแนวตั้งและแนวนอนของขวดนั้นจะใช้การปรับตำแหน่งของกรอบวินโดวส์ที่ได้กำหนดไว้ให้เลื่อนตามขวดที่เลื่อนตำแหน่งไปด้วย ซึ่งใช้หลักการเช่นเดียวกับการปรับตำแหน่งของฟังก์ชันตรวจสอบในอัลกอริทึมแบบแรกที่ได้กล่าวไปแล้วคือจะทำการคำนวณหาค่าการเลื่อนตำแหน่งทางด้านแนวตั้งและแนวนอนของชิ้นงานเพื่อนำไปบวกชดเชยกับตำแหน่งของกรอบวินโดวส์เพื่อให้เลื่อนตำแหน่งตามไปด้วย ในการคำนวณหาค่าการเลื่อนตำแหน่งก็ใช้การเปรียบเทียบตำแหน่งของเส้นขอบภาพ (Edge Line) ที่ใช้เป็นจุดอ้างอิงระหว่างเส้นขอบภาพของภาพที่ทำการกำหนดตำแหน่งของกรอบวินโดวส์ (Setting Phase) เทียบกับภาพของขวดที่กำลังตรวจสอบอยู่ (Running Phase) โดยในการปรับตำแหน่งทางด้านแนวตั้งจะใช้เส้นขอบภาพทางด้านแนวนอน (Horizontal Edge Line) เป็นจุดอ้างอิง ส่วนการปรับตำแหน่งทางด้านแนวนอนก็จะใช้เส้นขอบภาพทางด้านแนวตั้ง (Vertical Edge Line) สำหรับการหาตำแหน่งของเส้นขอบภาพที่ใช้เป็นจุดอ้างอิงนั้นในอัลกอริทึมนี้จะแตกต่างจากอัลกอริทึมแบบแรกซึ่งเป็นการหาเส้นขอบภาพบนภาพแบบสองระดับ แต่ในอัลกอริทึมนี้จะเป็นการหาบนภาพแบบ 256 ระดับ โดยจะใช้วิธีการหาขอบภาพจากค่า Projection Profile ในบริเวณที่กำหนดที่จะหาเส้นขอบภาพ (Okabe,T. et.al 1993)

การหาค่า Vertical / Horizontal Projection Profile (Haralick,R.M. and Shapiro,L.G. 1992) (Jain,R. et.al ,1995) คือ การหาค่าผลรวมของค่าความสว่าง(grey-level) ของพิกเซลในบริเวณคอลัมน์/แถวใดๆ ซึ่งสามารถแสดงดังรูปที่ 4.16 และสมการที่ 4.1 และ 4.2

Vertical Projection Profile :

$$P_V(j) = \sum_{i=0}^{M-1} f(i,j) \quad ; \; 0 \le j \le N-1 \qquad (4.1)$$

Horizontal Projection Profile :

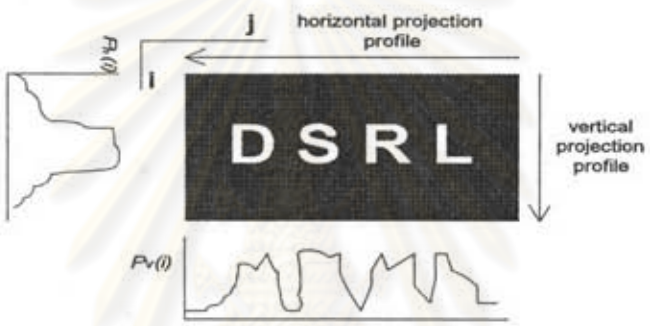
$$P_H(i) = \sum_{j=0}^{N-1} f(i,j) \quad ; 0 \le i \le M-1 \qquad (4.2)$$

โดยที่ $P_v(j)$ คือ อาร์เรย์ผลลัพธ์ขนาด N ของค่า Veritcal Projection Profile ที่คำนวณได้

$P_H(i)$ คือ อาร์เรย์ผลลัพธ์ขนาด M ของค่า Horizontal Projection Profile ที่คำนวณได้

M, N คือ จำนวนแถวและคอลัมน์ของบริเวณในภาพที่ต้องการหาค่า profile

f(i,j) คือ ค่าความสว่างของภาพ ณ ตำแหน่งพิกเซล i,j ซึ่งมีค่าอยู่ระหว่าง 0-255



รูปที่ 4.16 การหาค่า vertical / horizontal projection profile

วิธีการในการหาตำแหน่งของเส้นขอบภาพโดยใช้วิธี projection profile นั้นจะขออธิบายจากตัวอย่างในการหาเส้นขอบภาพแนวตั้งดังรูปที่ 4.15 ซึ่งสามารถอธิบายเป็นขั้นตอนได้ดังนี้

- กำหนดบริเวณพื้นที่ที่จะทำการหาตำแหน่งของเส้นขอบภาพแนวตั้ง ซึ่งจะกำหนดด้วยกรอบวินโดวส์แบบสี่เหลี่ยมผืนผ้าและเรียกกรอบวินโดวส์นี้ว่า *positioning window* โดยมีข้อกำหนดว่าภายในกรอบวินโดวส์นี้จะต้องมีเส้นขอบภาพอ้างอิงที่เด่นชัดเพียงเส้นเดียวเท่านั้น
- ทำการคำนวณหาค่า vertical projection profile ของภาพในบริเวณพื้นที่ที่กำหนดในข้อ 1
- จากอาร์เรย์ของ profile ที่คำนวณได้ในข้อ 2 นำมาคำนวณหาอาร์เรย์ผลต่างของ profile (Differential Profile,D(x)) ซึ่งค่าในอาร์เรย์นี้จะเท่ากับค่าสัมบูรณ์ของผลต่างระหว่างค่า profile ตัวเอง (g(x)) กับค่า profile ข้างๆ (g(x-1)) โดยสามารถเขียนเป็นสมการได้ดังนี้

$$D(x) = \left| g(x) - g(x-1) \right| \quad ; x = 1,2,3,.....N$$

โดยที่ g(x) = ค่าของ vertical (horizontal) projection profile ที่ตำแหน่ง x ใดๆ

D(x) = ค่าของ Differential profile ที่ตำแหน่ง x ใดๆ

N = ขนาดของ profile (อาร์เรย์)

● ค่าที่มากที่สุดในอาร์เรย์ผลต่างของ profile (Differential Profile) ที่คำนวณได้จากขั้นตอนที่แล้ว คือ ตำแหน่งของเส้นขอบภาพอ้างอิงที่ต้องการ

สำหรับการหาเส้นขอบภาพทางด้านแนวนอน นี้จะใช้วิธีการเดียวกันกับการหาขอบภาพแนวตั้ง เพียงแต่เปลี่ยนเป็นการหาค่า horizontal projection profile เข้ามาใช้ในการคำนวณแทน และรายละเอียดในการปรับตำแหน่งของกรอบวินโดวส์นี้ได้กล่าวไว้แล้วในหัวข้อฟังก์ชันปรับตำแหน่งในอัลกอริทึมแบบที่ 1 ซึ่งจะแตกต่างกันเฉพาะวิธีในการหาตำแหน่งของเส้นขอบภาพอ้างอิง จึงไม่ขอกล่าวรายละเอียดในที่นี้

1.2 การหาตำแหน่งของฉลากที่ต้องการตรวจสอบ (Label Positioning)

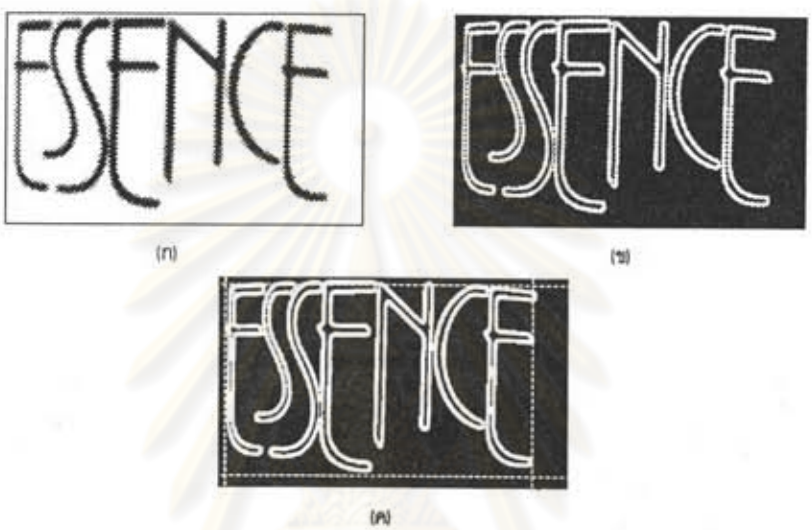
หลังจากทำการระบุตำแหน่งของกรอบวินโดวส์ที่เป็นบริเวณพื้นที่ในการตรวจสอบ (inspection window) จากขั้นตอนที่แล้ว ซึ่งกำหนดว่าภายในพื้นที่นี้จะต้องมีเฉพาะส่วนของฉลากที่ต้องการตรวจสอบอยู่บนพื้นผิวของขวดเท่านั้นดังตัวอย่างรูปที่ 4.17ก หลังจากนั้นอัลกอริทึมในส่วนนี้จะทำการค้นหาตำแหน่งของขอบฉลากในบริเวณพื้นที่นี้ ซึ่งจะออกมาอยู่ในรูปของตำแหน่งของสี่เหลี่ยมที่ล้อมรอบฉลาก(bounding rectangle of label) โดยสามารถแบ่งเป็นขั้นตอนดังนี้

● นำภาพในบริเวณส่วนของกรอบวินโดวส์มาทำการผ่านขบวนการหาขอบภาพแบบโซเบิล (Sobel edge detection) (Gonzalez,R.C. and Woods,R.E. 1992) ซึ่งจะอยู่ในรูปของหน้ากาก (mask) ดังรูปที่ 4.5 ข และนำภาพที่ผ่านการหาขอบภาพมาทำการแปลงเป็นภาพสองระดับโดยใช้ค่าขีดเริ่มเปลี่ยนที่กำหนด แสดงดังตัวอย่างรูปที่ 4.17 ข และ ค

● ทำการหาตำแหน่งของฉลากซึ่งอยู่ในรูปของสี่เหลี่ยมที่ล้อมรอบฉลาก(bounding rectangle) โดยการค้นหาทีละแถวและคอลัมน์ทั้ง 4 ทิศคือ จากบนลงล่าง,จากล่างขึ้นบน,จากซ้ายไปขวาและจากขวาไปซ้าย และในแต่ละแถว(คอลัมน์) ที่ทำการค้นหาก็จะทำการนับจำนวนพิกเซลที่มีค่า '1' ว่ามีค่ามากกว่าค่าที่กำหนดไว้หรือไม่ ถ้ามากกว่าก็จะถือว่าตำแหน่งของแถว (คอลัมน์) นั้นเป็นตำแหน่งของขอบฉลากและจากการค้นหาทั้ง 4 ทิศ ก็จะได้ตำแหน่งสี่เหลี่ยมที่ล้อมรอบฉลากนั้น ซึ่งสามารถแสดงตัวอย่างดังรูปที่ 4.17 ค (เส้นประ)

สำหรับเหตุผลที่นำภาพในพื้นที่ตรวจสอบมาทำการหาขอบภาพแบบโซเบิลก่อนนั้นเป็นเพราะว่าในกรณีของการตรวจสอบฉลากบนขวดนั้น ถ้าสภาพของแสงที่ส่องเข้าหาชิ้นงานไม่ดีพอ

อาจจะทำให้เกิดผลของการกระจายของแสงที่พื้นผิวขวดไม่เท่ากันทำให้ยากต่อการเลือกค่าขีดจำกัดในการแปลงภาพสองระดับเพื่อจะได้ภาพที่เหมาะสมในการหาตำแหน่งของขอบฉลาก ดังนั้นจึงทำการลดผลกระทบตรงนี้โดยการลดส่วนที่เป็นความถี่ต่ำของภาพซึ่งเกิดจากการกระจายของแสงไม่เท่ากัน ด้วยการหาขอบภาพแบบโซเบิลซึ่งเป็นขบวนการในการเน้นขอบภาพที่จะดึงเฉพาะส่วนที่เป็นขอบภาพออกมาและกดส่วนอื่นลงไป



รูปที่ 4.17 (ก) ส่วนของพื้นที่ในการตรวจสอบของภาพฉลากบนขวดที่ดึงออกมาได้ (ข) ภาพผลลัพธ์ที่ผ่านการหาขอบแบบโซเบิลจากภาพ 4.17 ก. (ค) ภาพผลลัพธ์ที่ผ่านการแปลงภาพสองระดับจากภาพ 4.17 ข. และตำแหน่งของขอบฉลากทั้งสี่ด้านที่หาได้ (เส้นประ)

2. การตรวจสอบแบบหยาบ (Coarse Inspection)

จากตำแหน่งของกรอบวินโดวส์ที่ระบุพื้นที่ในการตรวจสอบและตำแหน่งของฉลากซึ่งอยู่ในรูปของค่าตำแหน่งสี่เหลี่ยมที่ล้อมรอบฉลากที่ต้องการตรวจสอบที่หาได้จากในข้อแรก และจากค่าเหล่านี้สามารถนำมาทำการตรวจสอบแบบหยาบๆ โดยใช้เวลาในการคำนวณเพื่อการตรวจสอบไม่นานนัก แต่สามารถตรวจสอบฉลากที่มีจุดบกพร่องที่มีขนาดใหญ่ๆได้ โดยจะแบ่งการตรวจสอบเป็น 2 ส่วน คือ

2.1 การตรวจสอบขนาดของฉลาก (Size Checking)

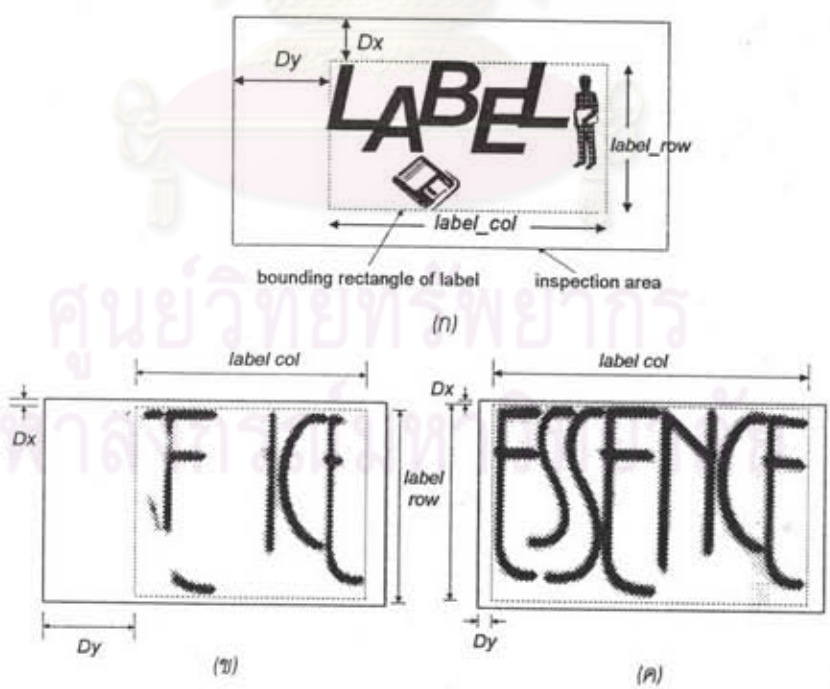
ส่วนนี้จะทำการตรวจสอบขนาดของสี่เหลี่ยมที่ล้อมรอบฉลากซึ่งจะคือค่าพารามิเตอร์ label_row และ label_col ในภาพ 4.18 (ก) ว่าอยู่ในช่วงขีดจำกัดบนและล่างที่กำหนดไว้หรือไม่ ถ้า

ขนาดของสี่เหลี่ยมที่ล้อมรอบฉลากนั้นไม่อยู่ในช่วงขอบเขตที่กำหนดก็จะถือว่าเป็นฉลากที่พิมพ์เสีย แต่ถ้าอยู่ในช่วงขอบเขตดังกล่าวก็จะนำไปตรวจสอบในข้ออื่นๆต่อไป ซึ่งในการตรวจสอบนี้สามารถที่จะตรวจฉลากที่มีจุดเสียประเภทที่พิมพ์ขาดไปมากๆหรือเกิดรอยสกปรกขนาดใหญ่ จนกระทั่งทำให้ขนาดของสี่เหลี่ยมนั้นผิดปกติไปจากขนาดมาตรฐาน

2.2 การตรวจสอบตำแหน่งของฉลาก (Position Checking)

ส่วนนี้จะทำการตรวจสอบตำแหน่งของสี่เหลี่ยมที่ล้อมรอบฉลากเทียบกับตำแหน่งของกรอบวินโดวส์ที่กำหนดพื้นที่ในการตรวจสอบ ซึ่งค่าที่วัดออกมาตรวจสอบ คือ ค่า Dx และ Dy ที่แสดงในรูปที่ 4.18(ก) โดยจะตรวจสอบว่าค่าดังกล่าวอยู่ในขอบเขตที่กำหนดหรือไม่ ถ้าไม่อยู่ในขอบเขตที่กำหนดก็จะถือว่าเป็นฉลากที่เสีย มิฉะนั้นก็จะนำไปตรวจสอบในขั้นอื่นต่อไป ซึ่งการตรวจสอบในข้อนี้สามารถที่จะทำการตรวจพบฉลากที่บกพร่องเนื่องจากการพิมพ์ที่เลื่อนตำแหน่งไป (shift) ได้

สำหรับตัวอย่างฉลากที่บกพร่องที่สามารถตรวจพบด้วยการตรวจสอบแบบหยาบนี้แสดงดังรูปที่ 4.18 (ข) ซึ่งเป็นฉลากที่มีการพิมพ์ขาดหายไปจนทำให้ค่า Dy และ label_col มีค่าแตกต่างจากค่าที่วัดออกมาได้จากภาพฉลากดีในภาพ 4.18 (ค) มาก



รูปที่ 4.18 (ก) พารามิเตอร์ในการตรวจสอบแบบหยาบ (ข) ตัวอย่างฉลากเสียที่สามารถตรวจพบได้ในการตรวจสอบแบบหยาบเทียบกับภาพฉลากดีในภาพ (ค) (ค) ตัวอย่างภาพฉลากดี

3. การตรวจสอบแบบละเอียด (Fine Inspection)

หลังจากที่ผ่านการตรวจสอบแบบหยาบในข้อที่ 2 มาแล้ว ถ้าภาพของฉลากใดสามารถผ่านการตรวจสอบมาได้ก็จะนำมาตรวจสอบแบบละเอียดในขั้นนี้ โดยจะใช้หลักการเปรียบเทียบภาพแบบจุดต่อจุดระหว่างภาพฉลากอ้างอิงซึ่งไม่มีจุดบกพร่องที่เก็บไว้ล่วงหน้ากับภาพฉลากที่ทำการตรวจสอบที่ดึงออกมาได้จากข้อ 1 ขั้นตอนต่างๆ ของการตรวจสอบละเอียดสามารถอธิบายเป็นข้อย่อยๆ ดังนี้

3.1 การทำให้ภาพตรงกัน (Image Registration)

ดังที่กล่าวมาแล้วว่าในกรณีของการตรวจสอบฉลากบนขวดนั้นจะมีปัญหาของกลไกในการป้อนชิ้นงานเข้ามาตรวจสอบ ทำให้เกิดการเลื่อนตำแหน่งเนื่องจากการหมุนของขวดรอบแกนตั้งของขวด ซึ่งมีผลทำให้ภาพฉลากที่ตรวจสอบนั้นมีการเพี้ยน (distortion) ไปจากภาพฉลากอ้างอิงที่เก็บไว้และจะทำให้ภาพทั้งสองไม่ตรงกันสนิทกันก่อนที่จะทำการเปรียบเทียบจนอาจทำให้เกิดข้อผิดพลาดในการตรวจสอบได้ ดังนั้นหน้าที่ของส่วนนี้จึงเป็นการทำให้ภาพทั้งสองทับตรงกันให้มากที่สุดก่อนที่จะทำการเปรียบเทียบต่อไป โดยสามารถแบ่งออกเป็น 2 ขั้นตอน คือ

3.1.1 การเลือกภาพฉลากอ้างอิงที่ตรงกันกับภาพฉลากที่ตรวจสอบมากที่สุด (Best Matched Reference Selection)

เนื่องจากการหมุนของขวดทำให้ภาพฉลากที่ทำการตรวจสอบเพี้ยนไปจากภาพฉลากอ้างอิงที่เก็บไว้ ดังนั้นวิธีแก้ปัญหาในอัลกอริทึมนี้ก็คือ การเก็บภาพฉลากอ้างอิงไว้หลายๆภาพ ที่มุมต่างๆ ที่ขวดนั้นสามารถหมุนไปได้ โดยส่วนของภาพฉลากอ้างอิงที่ทำการเก็บไว้จะใช้วิธีการดึงออกมาเช่นเดียวกันกับการดึงส่วนของภาพฉลากในข้อ 1 สำหรับหน้าที่ของส่วนนี้ก็คือทำการหาภาพฉลากอ้างอิงจากชุดของภาพอ้างอิงที่มีอยู่ที่ตรงกันกับภาพฉลากที่กำลังทำการตรวจสอบมากที่สุดโดยใช้การวัดค่าความคล้ายคลึง (similarity measure) ระหว่างภาพฉลากที่ตรวจสอบกับภาพฉลากอ้างอิงแต่ละภาพ และเลือกภาพอ้างอิงที่มีค่าความคล้ายคลึงกับภาพฉลากที่ตรวจสอบมากที่สุด สำหรับวิทยานิพนธ์นี้ค่าที่ใช้เป็นตัวแทนในการวัดค่าความคล้ายคลึงระหว่างสองภาพ คือ ค่าสัมประสิทธิ์คอร์รีเรชั่น (Correlation Coefficient,γ) (Gonzalez,R.C. and Woods,R.E. 1992) ดังสมการที่ 4.3

$$\gamma = \frac{\sum_x \sum_y \left[f_1(x,y) - \overline{f_1}\right]\left[f_2(x,y) - \overline{f_2}\right]}{\left\{\sum_x \sum_y \left[f_1(x,y) - \overline{f_1}\right]^2 \sum_x \sum_y \left[f_2(x,y) - \overline{f_2}\right]^2\right\}^{1/2}} \qquad (4.3)$$

โดยที่ $f_1(x,y)$, $f_2(x,y)$ คือ ค่าความสว่าง(grey-scale) ของภาพที่ 1 และ 2 ณ ตำแหน่งพิกเซล x,y ตามลำดับ

$\overline{f}_1$, $\overline{f_2}$ คือ ค่าเฉลี่ยความสว่างของภาพที่ 1 และ 2 ตามลำดับ

ค่าคอร์รีเรชั่นที่คำนวณได้นี้จะมีค่าอยู่ระหว่าง -1 ถึง 1 ในกรณีที่ภาพทั้งสองตรงกันมากสุดค่าคอร์รีเรชั่นจะมีค่าเท่ากับ 1 แต่ถ้าต่างกันมากสุดจะมีค่าเท่ากับ -1 ดังนั้นการหาภาพอ้างอิงที่ตรงกันมากที่สุดก็ทำได้โดยการคำนวณหาค่าคอร์รีเรชั่นระหว่างภาพฉลากที่ตรวจสอบกับภาพอ้างอิงต่างๆ และภาพอ้างอิงที่ตรงกันมากที่สุด คือ ภาพอ้างอิงที่ให้ค่าคอร์รีเรชั่นสูงที่สุดนั้นเอง

3.1.2 การปรับภาพให้ตรงกันกับภาพต้นแบบ (Model Fitting)

เนื่องจากภาพฉลากอ้างอิงที่ตรงกันมากที่สุดกับภาพฉลากที่ตรวจสอบที่หามาได้จากข้อที่แล้วอาจจะยังไม่ทับกันสนิทดี เพราะว่าในทางปฏิบัติแล้วการเก็บภาพอ้างอิงนั้นไม่สามารถเก็บได้ทุกความละเอียดขององศาที่ขวดสามารถหมุนไปได้ ดังนั้นจึงต้องทำการปรับภาพทั้งสองให้ทับกันสนิทพอดี โดยในวิทยานิพนธ์นี้ได้ทำการพัฒนาเทคนิคในการปรับให้ภาพทั้งสองตรงกัน ซึ่งเป็นเทคนิคที่ดัดแปลงมาจาก Tran,L.V. and Sklansky,J. 1992 โดยขอเรียกเทคนิคที่พัฒนาขึ้นนี้ว่า *การปรับภาพให้ตรงกับภาพต้นแบบ* หรือ *Model Fitting* โดยเป็นเทคนิคที่ทำการปรับภาพอินพุท (input image) ให้ทับกันสนิท (fit) กับภาพต้นแบบ (model image) การทำงานของอัลกอริทึมในส่วนนี้ถือได้ว่าเป็นส่วนที่สำคัญและค่อนข้างยุ่งยากที่สุด สำหรับรายละเอียดของอัลกอริทึมในการปรับภาพนี้สามารถอธิบายได้ดังนี้

<u>หลักการเบื้องต้นของอัลกอริทึมการปรับภาพให้ตรงกันกับภาพต้นแบบ</u>

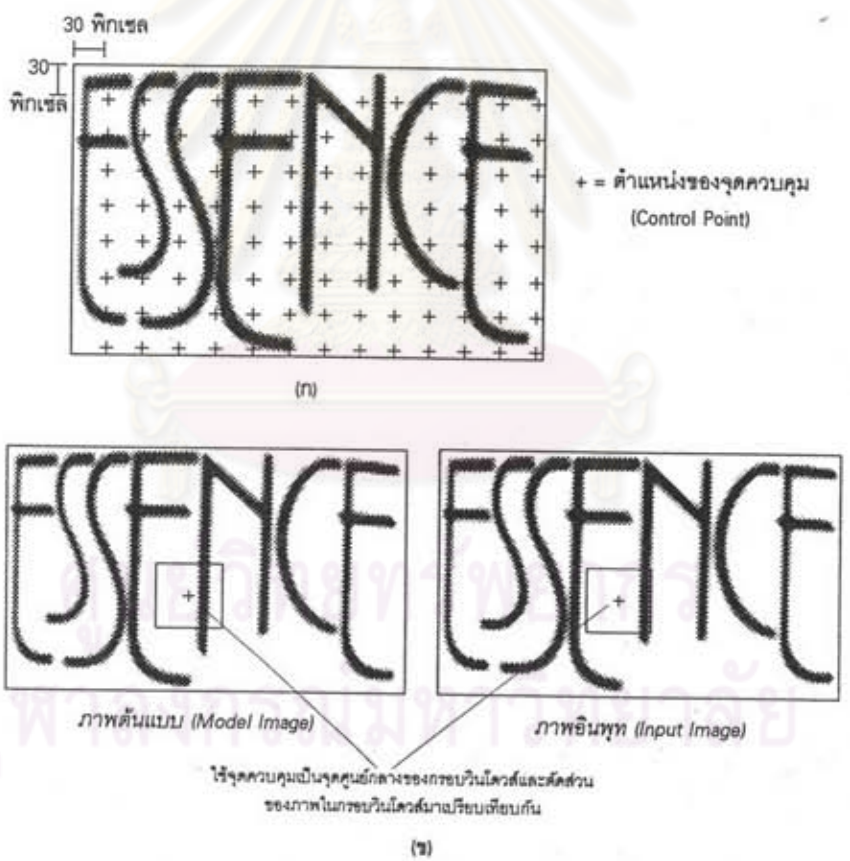
อัลกอริทึมนี้จะทำการปรับภาพอินพุทให้ตรงกันกับภาพต้นแบบโดยการคำนวณหาว่าในแต่ละพิกเซลของภาพอินพุทมีการเลื่อนตำแหน่งไปเท่าใดเมื่อเทียบกับตำแหน่งพิกเซลที่ตรงกันในภาพต้นแบบ ซึ่งค่าการเลื่อนตำแหน่งของแต่ละพิกเซลนั้นจะเรียกว่าเวคเตอร์การเลื่อนตำแหน่ง (translation vector) โดยในการคำนวณเวคเตอร์การเลื่อนตำแหน่งของแต่ละพิกเซลในภาพอินพุทนั้นจะไม่ทำการคำนวณทุกๆพิกเซลในภาพเพราะจะใช้เวลาในการคำนวณนาน แต่จะทำการคำนวณเฉพาะจุดพิกเซลหลักของภาพ ซึ่งจุดพิกเซลเหล่านี้จะเรียกว่าจุดควบคุม (control point) หลังจากนั้นค่าเวคเตอร์การเลื่อนตำแหน่งของจุดพิกเซลที่เหลือในภาพนั้นจะใช้การประมาณค่า (interpolation) จากค่าเวคเตอร์ที่จุดควบคุมเหล่านั้น หลังจากนั้นจะทำการสร้างภาพอินพุทใหม่จากภาพอินพุทเดิมโดยการปรับเลื่อนตำแหน่งพิกเซลในภาพอินพุทไปตามค่าเวคเตอร์การเลื่อนตำแหน่งที่คำนวณได้ ซึ่งผลลัพธ์ก็จะได้ภาพใหม่ที่ทับตรงกันกับภาพต้นแบบมากกว่าภาพอินพุท

เดิม ซึ่งในการคำนวณต่างๆ เหล่านี้อาจจะใช้การทำแบบวนซ้ำ(Iterations) หลายๆ ครั้ง เพื่อจะทำให้ภาพทั้งสองตรงกันสนิทมากยิ่งขึ้น

<u>รายละเอียดของอัลกอริทึมการปรับภาพให้ตรงกันกับภาพต้นแบบ</u>

รายละเอียดการทำงานของอัลกอริทึมในส่วนนี้สามารถอธิบายได้เป็นขั้นตอนดังนี้

<u>*ขั้นตอนที่ 1*</u> คำนวณหาตำแหน่งของจุดควบคุมในภาพต้นแบบและภาพอินพุทเพื่อที่จะนำไปใช้ในการคำนวณหาค่าเวคเตอร์การเลื่อนตำแหน่ง ซึ่งตำแหน่งของจุดควบคุมนั้นจะใช้หลักการกำหนดเช่นเดียวกันกับ Tran,L.V. and Sklansky,J. 1992 คือ เลือกตำแหน่งพิกเซลในภาพที่ทุกๆ N พิกเซล ตามแนวแถวและคอลัมน์ ดังตัวอย่างรูปที่ 4.19 ก ซึ่งเป็นภาพของฉลากที่ต้องการตรวจสอบที่เลือกตำแหน่งของจุดควบคุมทุกๆ 30 พิกเซล



รูปที่ 4.19 (ก) ตัวอย่างของการหาตำแหน่งของจุดควบคุมในภาพ โดยการเลือกทุกๆ 30 พิกเซลและการตัดส่วนของภาพที่จุดควบคุมเพื่อนำมาใช้ในการเปรียบเทียบตำแหน่งที่ตรงที่สุดของแต่ละจุดควบคุม (ข) การหาตำแหน่งของจุดควบคุมของภาพอินพุทในบริเวณรอบๆที่ตรงกันกับจุดควบคุมของภาพต้นแบบมากที่สุดโดยการตัดส่วนของภาพในกรอบวินโดวส์ทั้งสองมาเปรียบเทียบกัน

ตั้งแต่ขั้นตอนที่ 2 ถึง ขั้นตอนที่ 4 จะทำการคำนวณแบบวนซ้ำ (iteration) เท่ากับ จำนวนครั้งที่กำหนด ดังต่อไปนี้

ขั้นตอนที่ 2 คำนวณหาค่าเวคเตอร์การเลื่อนตำแหน่งที่จุดควบคุมต่างๆ ของภาพอินพุทเทียบกับภาพต้นแบบ ซึ่งสามารถแบ่งเป็นขั้นตอนย่อยๆ ดังนี้

(2.1) เลื่อนตำแหน่งของจุดควบคุมของภาพอินพุทไปในบริเวณรอบๆ ที่กำหนดด้วยค่าคงที่ค่าหนึ่ง ซึ่งค่านี้จะเป็นค่าที่กำหนดขอบเขตว่าแต่ละพิกเซลในภาพอินพุทจะต้องไม่มีการเลื่อนตำแหน่งไปเมื่อเทียบกับภาพต้นแบบเกินค่านี้ และที่ตำแหน่งของจุดควบคุมของภาพอินพุทที่เลื่อนไปก็จะคำนวณหาตำแหน่งที่ตรงกันมากที่สุดกับจุดควบคุมของภาพต้นแบบโดยการตัดส่วนของภาพในบริเวณที่ครอบคลุมจุดที่ทำการคำนวณของทั้งสองภาพแล้วนำมาเปรียบเทียบกัน สำหรับในอัลกอริทึมนี้จะใช้จุดควบคุมเป็นจุดศูนย์กลางกรอบวินโดวส์ที่จะทำการตัดส่วนของภาพทั้งสองออกมาเปรียบเทียบกัน ดังตัวอย่างรูปที่ 4.19 ข และค่าที่จะใช้เป็นเกณฑ์ในการวัดความเหมือนกันมากที่สุด (Best Match Criteria, F_{BMC}) ระหว่างส่วนของภาพทั้งสองจะใช้ค่าสัมบูรณ์ของผลต่างน้อยสุดเป็นเกณฑ์ดังสมการที่ (4.4) และตำแหน่งของจุดควบคุมที่เลื่อนไปของภาพอินพุทที่ให้ค่าดังกล่าวน้อยที่สุดก็คือตำแหน่งที่ตรงกันมากที่สุด

$$F_{BMC}(i_c, j_c, \Delta i, \Delta j) = \sum_{k=-\frac{N-1}{2}}^{\frac{N-1}{2}} \sum_{l=-\frac{N-1}{2}}^{\frac{N-1}{2}} \left| I(i_c + \Delta i + k, j_c + \Delta j + l) - M(i_c + k, j_c + l) \right| \quad (4.4)$$

โดยที่ F_{BMC} = ฟังก์ชันในการวัดความเหมือนกันมากที่สุดในการค้นหาตำแหน่งที่ตรงกันระหว่างจุดควบคุม i_c , j_c ในภาพต้นแบบ และ จุดควบคุมในภาพอินพุทที่เลื่อนตำแหน่งไป Δi , Δj

i_c , j_c = ตำแหน่งของจุดควบคุมใดๆ ที่กำลังพิจารณา

Δi, Δj = ระยะเลื่อนตำแหน่งของของจุดควบคุมใดๆ ที่เลื่อนตำแหน่งไปในบริเวณรอบๆ

N = ขนาดของกรอบวินโดวส์ที่ตัดส่วนของภาพออกมาใช้ในการเปรียบเทียบ

I (...,...) = ค่าความสว่างของพิกเซลใดๆ ของภาพอินพุท

M(...,...) = ค่าความสว่างของพิกเซลใดๆ ของภาพต้นแบบ

การทำงานในขั้นตอน 2.1 นี้สามารถอธิบายได้ด้วยสมการทางคณิตศาสตร์ ดังต่อไปนี้ กล่าวคือ ที่ตำแหน่งของจุดควบคุม i_c , j_c ใดๆ ของภาพต้นแบบ จะทำการค้นหาตำแหน่งของจุดควบคุมในภาพอินพุทที่ตรงกันมากที่สุด (i_{BM},j_{BM}) โดยทำการเลื่อนตำแหน่งจุดควบคุมของภาพอินพุทไปในบริเวณรอบๆ $(\Delta i,\Delta j)$ ซึ่งกำหนดช่วงขอบเขตของการเลื่อนตำแหน่ง S ซึ่ง i_{BM}, j_{BM} นี้หาได้โดย

$$i_{BM} = i_c + \Delta i_{BM}$$

$$j_{BM} = j_c + \Delta j_{BM}$$

และ $(\Delta i_{BM} , \Delta j_{BM}) = (\Delta i_o , \Delta j_o) \mid min (F_{BMC}(i_c , j_c , \Delta i , \Delta j))$

$$\Delta i = \{ -S, -S+1, \ldots ,-1, 0, 1, \ldots , S-1, S \} \quad \Delta j = \{ -S, -S+1, \ldots , -1, 0, 1, \ldots , S-1, S \}$$

โดยที่ i_c , j_c = ตำแหน่งของจุดควบคุมของภาพต้นแบบที่กำลังพิจารณา

i_{BM} , j_{BM} = ตำแหน่งของจุดควบคุมของภาพอินพุทที่ตรงกันมากที่สุดกับตำแหน่งของจุดควบคุม i_c , j_c ของภาพต้นแบบ

S = ช่วงขอบเขตของการเลื่อนตำแหน่ง ซึ่งกำหนดบริเวณที่จะทำการค้นหาตำแหน่งที่ตรงกันของจุดควบคุม

(2.2) คำนวณหาค่าเวคเตอร์การเลื่อนตำแหน่งที่จุดควบคุมนั้น $\{V_x(i_c,j_c),V_y(i_c,j_c)\}$ ซึ่งเท่ากับผลต่างของตำแหน่งของจุดควบคุมของภาพต้นแบบ (i_c,j_c) กับตำแหน่งของจุดควบคุมของภาพอินพุทที่เลื่อนไปในบริเวณรอบๆ ที่ตรงกันกับจุดควบคุมของภาพต้นแบบมากที่สุด (i_{BM} ,j_{BM}) และค่าเวคเตอร์นี้จะถูกบวกรวมเก็บไว้ในแต่ละรอบของการวนซ้ำในการคำนวณ ซึ่งสามารถเขียนเป็นสมการได้ดังนี้

$$V_x(i_c,j_c) = i_c - i_{BM}$$

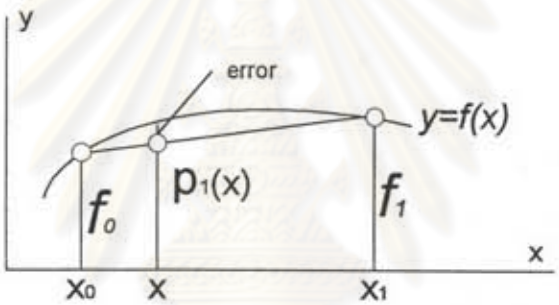
$$V_y(i_c,j_c) = j_c - j_{BM}$$

โดยที่ $V_x(i_c,j_c)$ = ค่าเวคเตอร์การเลื่อนตำแหน่งในแนวตั้ง ที่ตำแหน่งจุดควบคุม i_c,j_c

$V_y(i_c,j_c)$ = ค่าเวคเตอร์การเลื่อนตำแหน่งในแนวนอน ที่ตำแหน่งจุดควบคุม i_c,j_c

สำหรับขั้นตอนที่ 2.1 และ 2.2 จะทำการคำนวณหาเวคเตอร์การเลื่อนตำแหน่งของจุดควบคุมทุกๆจุด ที่หาได้จากขั้นตอนที่ 1

ขั้นตอนที่ 3 คำนวณหาเวคเตอร์การเลื่อนตำแหน่งของจุดพิกเซลที่เหลือของภาพอินพุท ซึ่งการคำนวณนี้จะอยู่ภายใต้สมมติฐานว่าพิกเซลที่อยู่ใกล้กับจุดควบคุมใดก็ควรจะมีค่าใกล้เคียงกับเวคเตอร์การเลื่อนตำแหน่งของจุดควบคุมนั้น และการหาค่าจะใช้วิธีการประมาณค่าจากค่าเวคเตอร์เลื่อนตำแหน่งของจุดควบคุมที่หาได้จากขั้นตอนที่ 2 เพื่อให้การคำนวณเร็วขึ้น การประมาณค่าในอัลกอริทึมนี้ก็จะใช้การประมาณค่าแบบเส้นตรง (Linear Interpolation) (Kreyszig,E. 1988) สำหรับหลักการของการประมาณค่าแบบเส้นตรงนั้น สามารถแสดงตัวอย่างดังรูปที่ 4.20 ซึ่งเป็นการประมาณค่าฟังก์ชัน f(x) ใดๆ เมื่อรู้จุดผ่าน 2 จุด (จุด x_0 และ x_1) และค่าของฟังก์ชันที่จุดทั้งสอง (f_0,f_1) โดยจะทำการหาค่าของฟังก์ชันของจุดที่อยู่ภายในบริเวณ x_0 ถึง x_1 ด้วยการประมาณค่าฟังก์ชันที่ผ่าน 2 จุดนี้ด้วยเส้นตรง $p_1(x)$



รูปที่ 4.20 การประมาณค่าแบบเส้นตรง

และค่าของฟังก์ชันที่จุด x ใดๆ ($p_1(x)$) ในบริเวณ x_0 ถึง x_1 หาได้โดยสมการที่ 4.5

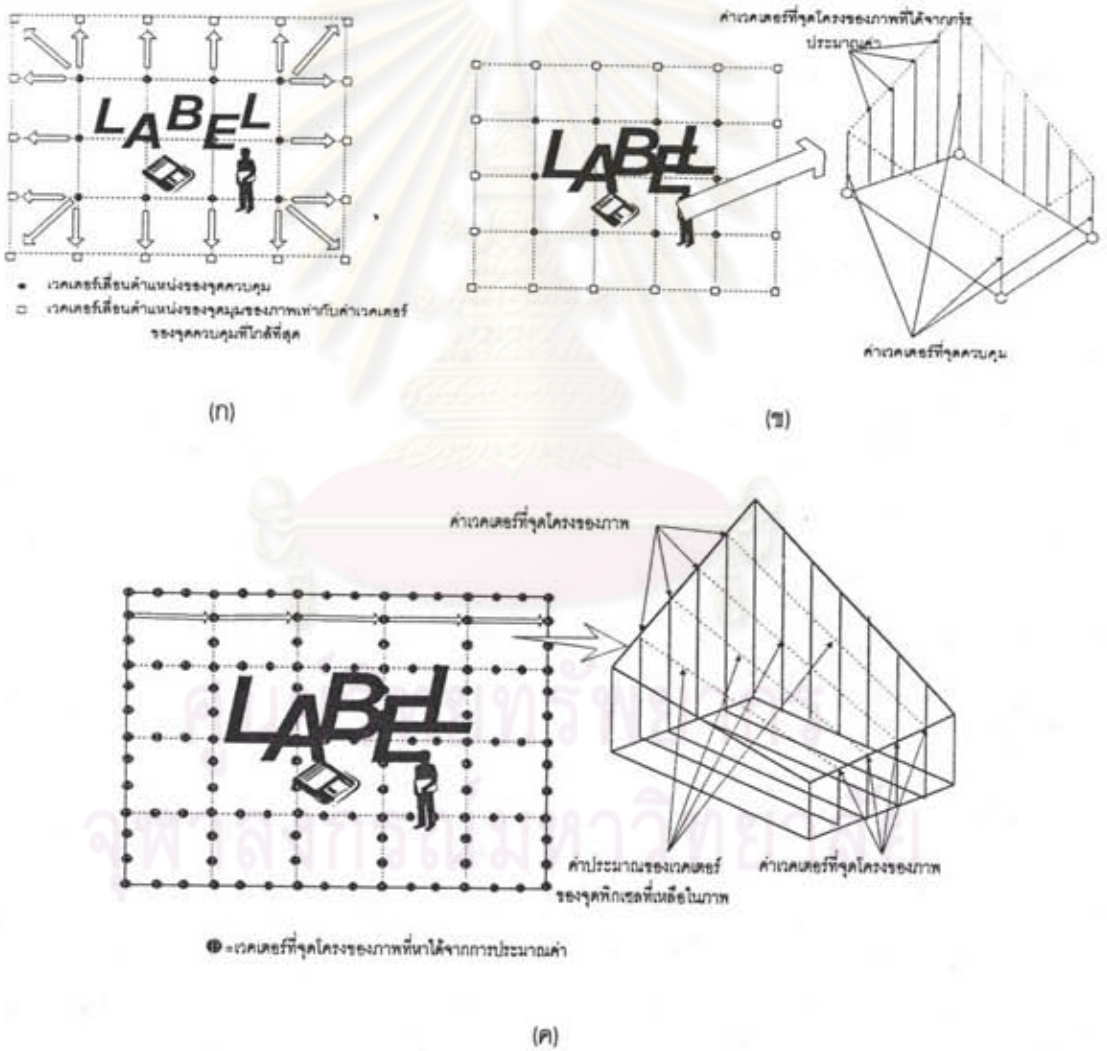
$$p_1(x) = f_0 + (x - x_0) * \frac{f_1 - f_0}{x_1 - x_0} \qquad (4.5)$$

จากหลักการในการประมาณค่าที่กล่าวมาแล้วจะนำมาใช้ในการประมาณค่าเวคเตอร์เลื่อนตำแหน่งที่จุดพิกเซลรอบๆ จุดควบคุม โดยสามารถอธิบายเป็นขั้นตอนดังนี้

(3.1) กำหนดค่าเวคเตอร์เลื่อนตำแหน่งที่ตำแหน่งจุดมุมของภาพให้มีค่าเท่ากับค่าเวคเตอร์เลื่อนตำแหน่งของจุดควบคุมที่อยู่ใกล้ที่สุด ดังตัวอย่างรูปที่ 4.21 n

(3.2) คำนวณหาค่าเวคเตอร์เลื่อนตำแหน่งที่จุดต่างๆบนโครงของภาพที่เป็นจุดเชื่อมต่อระหว่างจุดควบคุมต่างๆ ดังรูป 4.21ข โดยใช้วิธีการประมาณค่าแบบเส้นตรงและใช้ค่าเวคเตอร์เลื่อนตำแหน่งของจุดควบคุม และจุดขอบรูปเป็นค่าของจุดต้น และจุดปลายของฟังก์ชันที่ต้องการประมาณค่า

(3.3) คำนวณหาค่าเวคเตอร์การเลื่อนตำแหน่งของพิกเซลที่เหลือในภาพทีละแถวโดยใช้การประมาณค่าแบบเส้นตรงเช่นเดียวกับข้อ 3.2 และใช้ค่าที่คำนวณได้จากข้อ 3.2 เป็นค่าของจุดต้นและจุดปลายในการคำนวณ ดังรูปที่ 4.21 ค



รูปที่ 4.21 (ก) การให้ค่าเวคเตอร์กับจุดที่มุมของภาพเท่ากับค่าเวคเตอร์ของจุดควบคุมที่ใกล้ที่สุด (ข) การประมาณค่าของเวคเตอร์ที่จุดโครงของภาพ (จุดในเส้นประ) จากค่าเวคเตอร์ของจุดควบคุม (ค) การคำนวณค่าเวคเตอร์การเลื่อนตำแหน่งของจุดที่เหลือในภาพที่ละแถวโดยการประมาณค่าจากค่าเวคเตอร์ที่จุดโครงของภาพ

ขั้นตอนที่ 4 ทำการสร้างภาพใหม่จากภาพอินพุทเพื่อทำการแก้ไขทางตำแหน่ง (geometric correction) ของภาพอินพุทที่มีการเพี้ยนไปให้ตรงกันสนิทกับภาพต้นแบบ โดยใช้ค่าเวคเตอร์การเลื่อนตำแหน่ง{ $V_x(i,j)$, $V_y(i,j)$ } ที่คำนวณได้ในขั้นตอนที่ 2 และ 3 ซึ่งค่านี้จะเป็นค่าที่บอกว่าตำแหน่งพิกเซล (i ,j) ในภาพอินพุทนั้นมีการเลื่อนตำแหน่งไปเท่าใด เมื่อเทียบกับตำแหน่งพิกเซลเดียวกันในภาพต้นแบบ โดยค่าความสว่างที่ตำแหน่งพิกเซล i,j ของภาพอินพุท ($I(i,j)$) จะถูกเลื่อนตำแหน่งไปยังภาพใหม่ที่ตำแหน่ง i',j' ซึ่งค่าตำแหน่งพิกเซลของภาพใหม่ (i',j') ที่เลื่อนตำแหน่งมาจาก ตำแหน่งพิกเซล (i,j) ของภาพอินพุท หาได้โดย

$$i' = i + V_x(i,j)$$
$$j' = j + V_y(i,j)$$

โดยที่ i',j' คือ ตำแหน่งพิกเซลของภาพใหม่ที่เลื่อนตำแหน่งมาจากตำแหน่งพิกเซล i , j ของภาพอินพุท

$V_x(i,j)$, $V_y(i,j)$ คือ ค่าเวคเตอร์การเลื่อนตำแหน่งที่คำนวณได้ ณ ตำแหน่งพิกเซล i, j

แต่เนื่องจากค่าเวคเตอร์การเลื่อนตำแหน่งที่คำนวณได้ซึ่งมาจากการประมาณค่าแบบเส้นตรงนั้นอาจจะไม่ใช่จำนวนเต็ม (non-integer) ดังนั้น ตำแหน่งพิกเซล i', j' ของภาพใหม่ที่คำนวณได้ ก็อาจจะลงไปยังตำแหน่งพิกเซลที่ไม่ใช่จำนวนเต็มดังตัวอย่างรูปที่ 4.22 ซึ่งสมมติว่าตำแหน่งพิกเซลที่คำนวณได้ไปตกอยู่ในบริเวณของพิกเซล (k,l) (k+1,l) (k,l+1) และ (k+1,l+1) ดังนั้นค่าความสว่างของภาพอินพุทที่ตำแหน่ง i,j ($I(i,j)$) ก็จะกระจายอยู่ใน 4 พิกเซลนี้ โดยขึ้นอยู่ว่าพื้นที่ของตำแหน่งพิกเซลดังกล่าวไปตกอยู่ในบริเวณใดมากกว่ากัน ซึ่งสามารถคำนวณได้ดังนี้

พื้นที่ที่ไปตกอยู่ในตำแหน่งพิกเซล k,l : A(k,l) = (1-v)*(1-h)

พื้นที่ที่ไปตกอยู่ในตำแหน่งพิกเซล k+1,l : A(k+1,l) = v*(1-h)

พื้นที่ที่ไปตกอยู่ในตำแหน่งพิกเซล k,l+1 : A(k,l+1) = (1-v)*h

พื้นที่ที่ไปตกอยู่ในตำแหน่งพิกเซล k+1,l+1 : A(k+1,l+1) = v*h

ค่าความสว่างที่กระจายไปอยู่ในตำแหน่งพิกเซล k,l : G(k,l) = A(k,l)*I(i,j)

ค่าความสว่างที่กระจายไปอยู่ในตำแหน่งพิกเซล k+1,l : G(k+1,l) = A(k+1,l)*I(i,j)

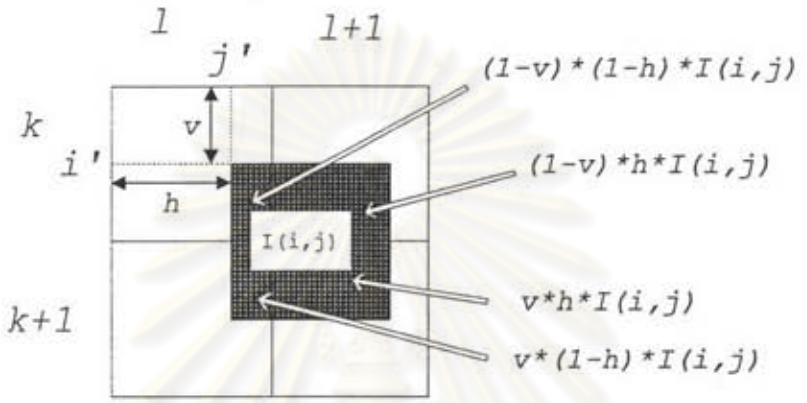
ค่าความสว่างที่กระจายไปอยู่ในตำแหน่งพิกเซล k,l+1 : G(k,l+1) = A(k,l+1)*I(i,j)

ค่าความสว่างที่กระจายไปอยู่ในตำแหน่งพิกเซล k+1,l+1 : G(k+1,l+1) = A(k+1,l+1)*I(i,j)

โดยที่ k , l คือ ค่าจำนวนเต็มของค่า i′,j′ ตามลำดับ

v , h คือ ค่าทศนิยมของค่า i′,j′ ตามลำดับ

I(i,j) คือ ค่าความสว่างของภาพอินพุท ณ ตำแหน่ง i,j



รูปที่ 4.22 ภาพแสดงตำแหน่งของพิกเซลใหม่ที่เป็นเลขที่ไม่ใช่จำนวนเต็มจึงไปตกอยู่ในบริเวณ 4 พิกเซลและการคำนวณหาค่าความสว่างที่แบ่งกระจายไปอยู่พิกเซลต่างๆ

ในการคำนวณนี้จะใช้อาร์เรย์ 2 มิติ ขนาดเท่ากับขนาดของภาพใหม่ที่จะทำการสร้าง จำนวน 2 อาร์เรย์ โดยอาร์เรย์จะเอาไว้สำหรับเก็บผลรวมของค่าความสว่างของพิกเซลใดๆในภาพอินพุทที่มาตกอยู่ในตำแหน่งพิกเซล p,q ใดๆในภาพใหม่ $\{G(p,q)\}$ ส่วนอาร์เรย์ที่สองจะใช้สำหรับเก็บผลรวมของพื้นที่ของพิกเซลใดๆในภาพอินพุทที่มาตกอยู่ในตำแหน่งพิกเซล p,q ใดๆในภาพใหม่ $\{A(p,q)\}$ ซึ่งในการสร้างภาพใหม่นี้ก็จะคำนวณทุกพิกเซลในภาพอินพุทเดิมจากจุดมุมซ้ายบนไปยังจุดมุมขวาล่าง โดยในแต่ละพิกเซลก็จะทำการคำนวณหาค่าของพื้นที่ของพิกเซลทั้งสี่ที่ตำแหน่งใหม่ของพิกเซลไปตกอยู่ $\{A(k,l), A(k+1,l), A(k,l+1), A(k+1,l+1)\}$ และบวกรวมเก็บไว้ในอาร์เรย์ A(p,q) และคำนวณค่าความสว่างที่กระจายไปอยู่ในพิกเซลทั้งสี่ $\{G(k,l),G(k+1,l),G(k,l+1), G(k+1,l+1)\}$ บวกเก็บรวมไว้ในอาร์เรย์ G(p,q) เมื่อทำการคำนวณครบทุกจุดในภาพอินพุทแล้ว ค่าความสว่างของภาพใหม่ที่ตำแหน่ง p,q ใดๆ (N(p,q)) จะเท่ากับ ผลรวมของค่าความสว่างทั้งหมดที่มาตกอยู่ที่พิกเซล p,q หารด้วยผลรวมของพื้นที่พิกเซลที่มาตกอยู่ที่พิกเซล p,q ดังสมการ

$$N(p,q) = G(p,q) / A(p,q)$$

โดยที่ N(p,q) = ค่าความสว่างของภาพใหม่ที่สร้างขึ้น ที่ตำแหน่ง p,q

G(p,q) = ผลรวมของค่าความสว่างที่ตำแหน่งพิกเซล p,q

A(p,q) = ผลรวมของพื้นที่พิกเซลที่ตำแหน่งพิกเซล p,q

<u>การนำเอาฟังก์ชันการปรับภาพให้ตรงกับภาพต้นแบบไปใช้งาน</u>

ฟังก์ชันการปรับภาพให้ตรงกันกับภาพต้นแบบที่ได้อธิบายไปแล้วนั้นจะทำการปรับภาพอินพุทให้ตรงกันสนิทกันกับภาพต้นแบบ สำหรับในการตรวจสอบฉลากที่พิมพ์บนขวดนั้นจะทำการเปรียบภาพระหว่างภาพฉลากอ้างอิงกับภาพฉลากที่ต้องการตรวจสอบ และเพื่อต้องปรับให้ภาพทั้งสองตรงกันสนิทเพื่อนำไปเปรียบเทียบต่อไป ก็จำเป็นที่ต้องเลือกว่าจะให้ภาพใดเป็นภาพอินพุทที่จะทำการสร้างภาพใหม่ออกมา และ ภาพใดจะใช้เป็นภาพต้นแบบ ซึ่งเมื่อลองใช้งานทั้ง 2 แบบ คือ แบบแรกใช้ภาพฉลากอ้างอิงเป็นภาพอินพุทที่ทำการสร้างภาพใหม่ กับ แบบสอง ที่ใช้ภาพฉลากที่ตรวจสอบเป็นภาพอินพุท พบว่าให้ผลในการปรับภาพให้ตรงกันใกล้เคียงกัน แต่ในอัลกอริทึมนี้จะเลือกใช้ภาพฉลากที่ตรวจสอบเป็นภาพอินพุท และภาพฉลากอ้างอิงเป็นภาพต้นแบบในการตรวจสอบ

3.2 การเปรียบเทียบภาพ (Image Comparison)

หลังจากที่ทำให้ภาพทั้งสองที่จะนำมาเปรียบเทียบกันตรงกันสนิทแล้ว ส่วนต่อไปก็จะทำการเปรียบเทียบภาพทั้งสองแบบจุดต่อจุดเพื่อจะตรวจดูว่าภาพทั้งสองนั้นแตกต่างกันอย่างไรบ้าง ถ้าในส่วนใดของภาพฉลากที่ตรวจสอบนั้นแตกต่างจากภาพฉลากอ้างอิงก็จะถือเป็นจุดบกพร่องภาพฉลากนั้น โดยรายละเอียดของอัลกอริทึมในส่วนนี้ สามารถแบ่งเป็นขั้นตอนย่อยดังนี้

3.2.1 การลบภาพ (Image Subtraction)

ภาพฉลากอ้างอิงและภาพฉลากที่ต้องการตรวจสอบที่ผ่านขบวนการปรับภาพมาแล้วจะนำมาลบกันจุดต่อจุดโดยใช้ค่าความสว่างของภาพทั้งสอง แบบค่าสัมบูรณ์ (absolute) ดังสมการที่ 4.6 ซึ่งจะได้ภาพผลลัพธ์เป็นภาพผลต่าง (Difference Image) ดังตัวอย่างรูปที่ 4.23

$$D(i,j) = \left| R(i,j) - T(i,j) \right| \qquad (4.6)$$

โดยที่ D(i,j) = ค่าความสว่าง ณ ตำแหน่งพิกเซล i,j ของภาพผลต่าง (different image) ที่เกิดจากการลบกัน

R(i,j) =ค่าความสว่าง ณ ตำแหน่งพิกเซล i,j ของภาพฉลากอ้างอิง

และ

T(i,j) = ค่าความสว่าง ณ ตำแหน่งพิกเซล i,j ของภาพฉลากที่ตรวจสอบ

รูปที่ 4.23 ภาพผลลัพธ์ที่ได้จากการลบภาพและการดึงพิกเซลที่เป็นจุดบกพร่องโดยการแปลงภาพสองระดับ

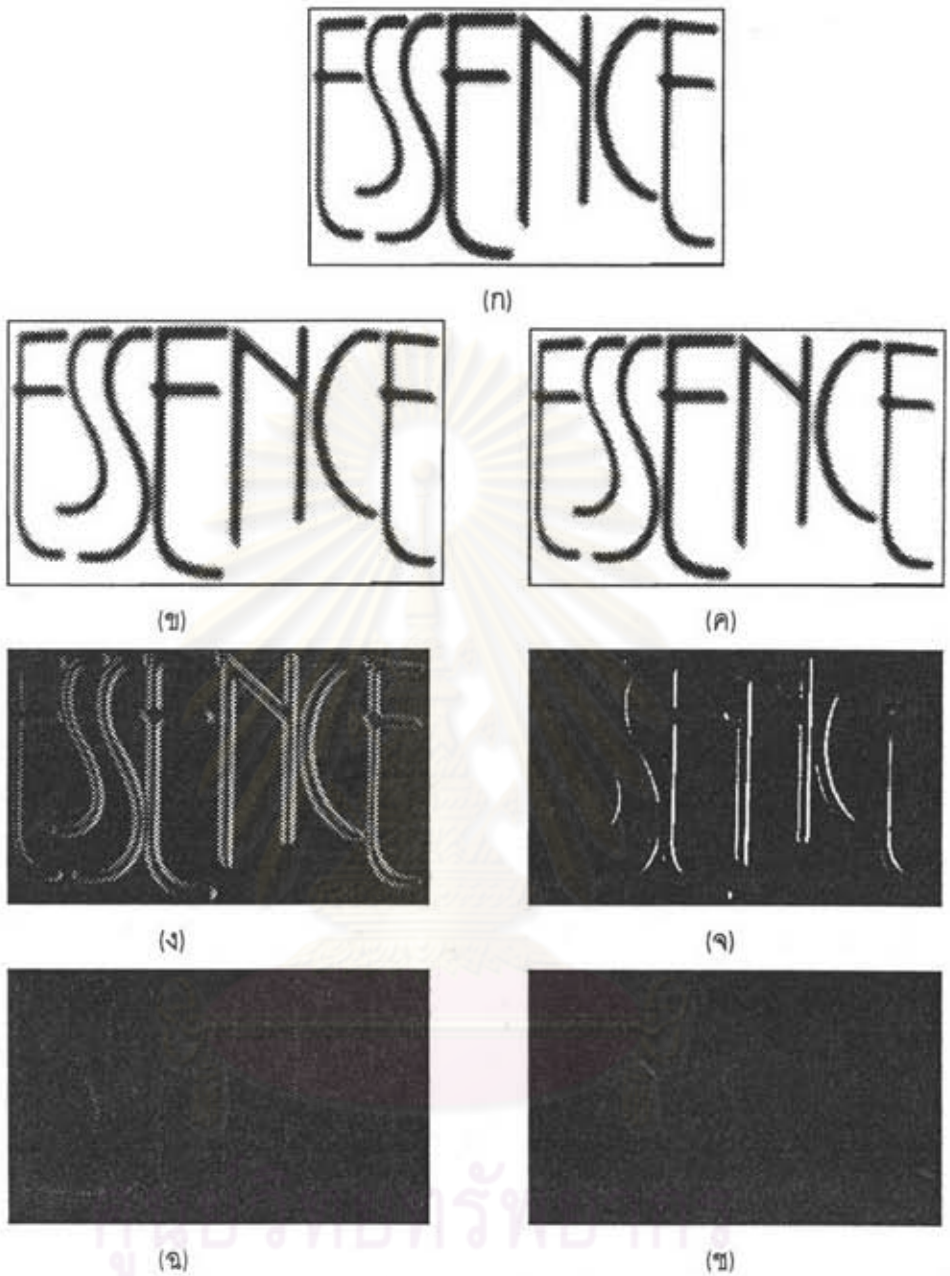
3.2.2 การดึงส่วนที่เป็นจุดบกพร่องและการตัดสิน (Defect Extraction and Judgement)

ภาพผลต่างที่เกิดจากการลบกันจากขั้นตอนที่แล้วในส่วนที่เป็นจุดบกพร่องนั้นจะมีค่าความสว่างสูงกว่าบริเวณที่ไม่ใช่จุดบกพร่อง ดังนั้นส่วนนี้จึงทำการดึงเฉพาะบริเวณที่เป็นจุดบกพร่องออกมานั้นเท่านั้น โดยการแปลงภาพสองระดับ (thresholding) จากภาพที่เกิดจากการลบกันซึ่งเป็นภาพแบบ 256 ระดับให้กลายเป็นภาพสองระดับ โดยใช้ค่าขีดเริ่มเปลี่ยน ซึ่งค่านี้จะมีผลต่อการตรวจสอบดังตัวอย่างรูปที่ 4.23 โดยจะเป็นค่าที่ระบุว่าความแตกต่างของความสว่างในภาพทั้งสองจะต้องมากเพียงใดจึงจะถือว่าเป็นพิกเซลที่เป็นจุดบกพร่อง

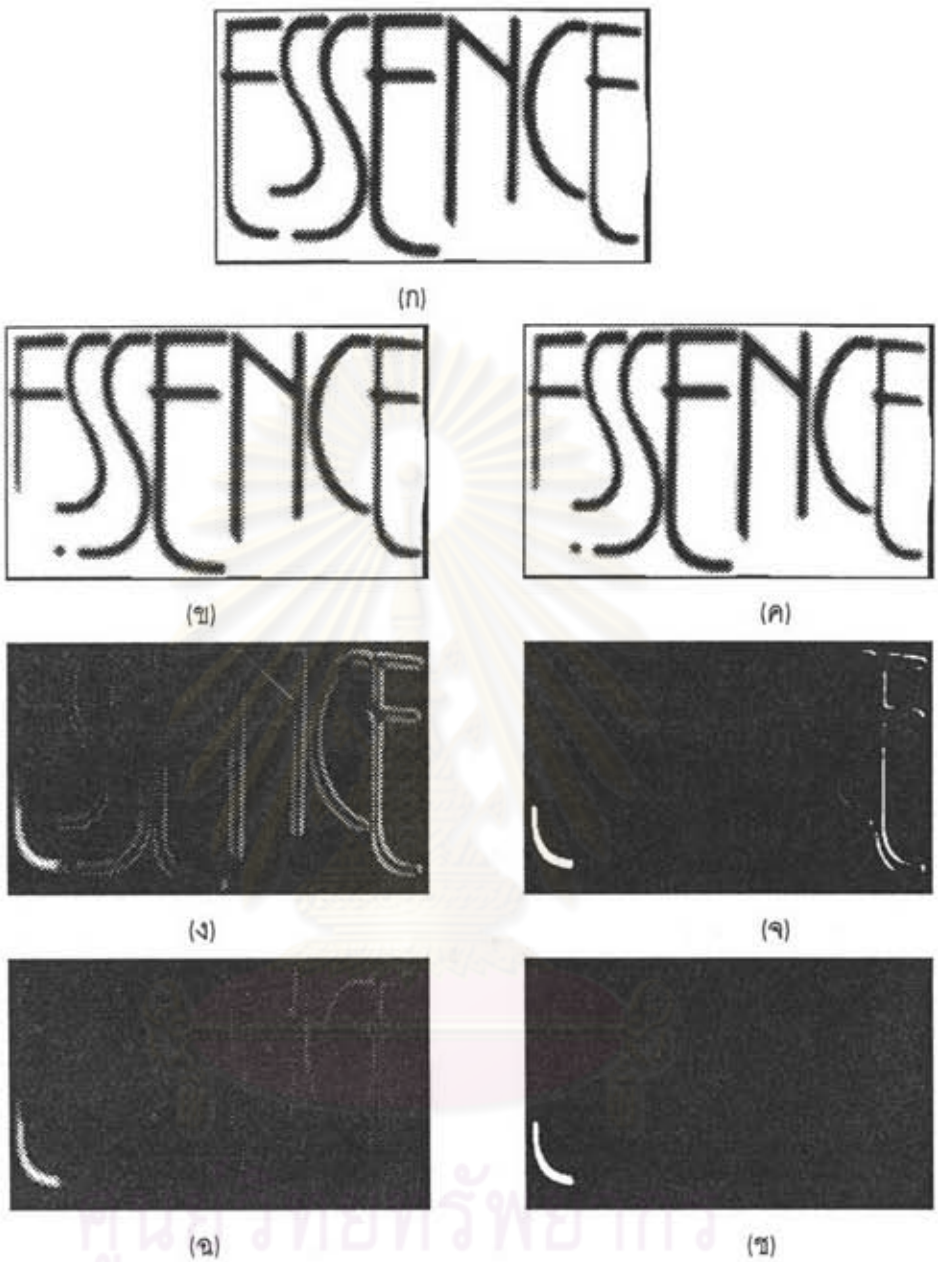
จากภาพสองระดับที่แปลงมาจากภาพที่เกิดจากการลบกันจากข้อที่แล้ว ในส่วนที่เป็นพิกเซลจุดบกพร่องจะมีค่า 1 และส่วนพิกเซลที่ไม่ใช่จุดบกพร่องก็จะมีค่าเท่ากับ 0 สำหรับอัลกอริทึมนี้ในการตัดสินว่าฉลากที่พิมพ์บนขวดนั้นดีหรือบกพร่อง จะใช้จำนวนพิกเซลที่เป็นจุดบกพร่องทั้งหมดในภาพ ซึ่งก็คือจำนวนพิกเซลที่มีค่า 1 ในภาพสองระดับนั่นเอง โดยในการตัดสินก็จะนำจำนวนพิกเซลที่นับได้มาเปรียบเทียบกับค่าขีดเริ่มต้นที่กำหนดไว้เบื้องต้น ถ้าค่าจำนวนพิกเซลเกินค่านี้ก็จะถือเป็นฉลากที่พิมพ์บกพร่อง นอกจากนั้นแล้วก็จะถือว่าฉลากที่พิมพ์ไม่มีข้อบกพร่อง สำหรับค่าขีดเริ่มเปลี่ยนนี้จะขึ้นอยู่กับความละเอียดในการตรวจสอบว่าต้องการตรวจสอบจุดบก

พร่องที่มีขนาดเล็กเพียงใด และค่านี้จะมีผลต่อการความถูกต้องและความละเอียดของการตรวจสอบค่อนข้างสูง

สำหรับตัวอย่างในการทำงานของอัลกอริทึมในการตรวจสอบแบบนี้แสดงดังรูป 4.24 โดยในรูป (ก) (ข) จะเป็นภาพของฉลากอ้างอิงที่เก็บไว้และภาพของฉลากที่ตรวจสอบซึ่งไม่มีจุดบกพร่องตามลำดับ ส่วนในรูป (ค) เป็นภาพฉลากที่ตรวจสอบที่ผ่านการปรับภาพให้ตรงกันกับภาพฉลากอ้างอิง โดยถ้าไม่ทำการปรับภาพให้ตรงกันก่อนที่ทำการลบภาพกันจะทำให้ได้ภาพดังรูป (ง) ซึ่งเมื่อทำการดึงเฉพาะส่วนที่เป็นจุดบกพร่องออกมาก็จะทำให้ส่วนที่ไม่ใช่จุดบกพร่องปรากฏออกมาด้วยดังรูป (จ) แต่ถ้าทำการลบภาพที่ผ่านการปรับภาพแล้วกับภาพอ้างอิงก็จะได้ดังรูป (ฉ) และเมื่อทำการดึงส่วนที่เป็นจุดบกพร่องออกมาจะได้ดังรูป (ช) จะเห็นได้ว่าจะไม่มีพิกเซลส่วนที่ไม่ใช่จุดบกพร่องปรากฏออกมา และเช่นเดียวกันกับกรณีของภาพฉลากที่มีจุดบกพร่องแสดงดังรูปที่ 4.25 ซึ่งจะเห็นได้ว่าถ้าไม่มีการปรับภาพให้ตรงกันระหว่างภาพฉลากที่ตรวจสอบกับภาพฉลากอ้างอิงก่อนที่จะทำการลบภาพกันก็จะทำให้พบว่า นอกจากจะมีส่วนที่เป็นจุดบกพร่องปรากฏออกมาแล้ว ก็ยังมีส่วนที่ไม่ใช่จุดบกพร่องออกมาด้วยดังรูป 4.25 (ง) และ(จ) แต่เมื่อทำการปรับภาพให้ตรงกันแล้ว ผลของการลบกันก็จะเหลือเฉพาะส่วนที่เป็นจุดบกพร่องเท่านั้นดังรูป 4.25 (ฉ)และ(ช)



รูปที่ 4.24 **(ก)** *ภาพฉลากอ้างอิง* **(ข)** *ภาพฉลากที่ตรวจสอบ (ซึ่งไม่มีมีจุดบกพร่อง)* **(ค)** *ภาพฉลากที่ตรวจสอบที่ผ่านการปรับภาพให้ตรงกันกับภาพฉลากอ้างอิงใน (ก)* **(ง)** *ภาพที่เกิดจากการลบกันระหว่างภาพฉลากอ้างอิง (ก) กับภาพฉลากที่ตรวจสอบที่ยังไม่ได้ผ่านการปรับภาพ (ข)* **(จ)** *ภาพของจุดบกพร่องที่ดึงออกมาได้โดยการแปลงภาพสองระดับจากภาพ (ง)* **(ฉ)** *ภาพที่เกิดจากการลบกันระหว่างภาพฉลากอ้างอิง (ก) กับภาพฉลากที่ตรวจสอบที่ผ่านการปรับภาพให้ตรงกันแล้ว (ค)* **(ช)** *ภาพของจุดบกพร่องที่ดึงออกมาได้โดยการแปลงภาพสองระดับจากภาพ (ฉ)*



*รูปที่ 4.25 **(ก)** ภาพฉลากอ้างอิง **(ข)** ภาพฉลากที่ตรวจสอบ (ซึ่งมีจุดบกพร่องบริเวณตัว E ทางด้านขวาล่างของฉลาก) **(ค)** ภาพฉลากที่ตรวจสอบที่ผ่านการปรับภาพให้ตรงกันกับภาพฉลากอ้างอิงใน (ก) **(ง)** ภาพที่เกิดจากการลบกันระหว่างภาพฉลากอ้างอิง (ก) กับภาพฉลากที่ตรวจสอบที่ยังไม่ได้ผ่านการปรับภาพ (ข) **(จ)** ภาพของจุดบกพร่องที่ดึงออกมาได้โดยการแปลงภาพสองระดับจากภาพ (ง) **(ฉ)** ภาพที่เกิดจากการลบกันระหว่างภาพฉลากอ้างอิง (ก) กับภาพฉลากที่ตรวจสอบที่ผ่านการปรับภาพให้ตรงกันแล้ว (ค) **(ช)** ภาพของจุดบกพร่องที่ดึงออกมาได้โดยการแปลงภาพสองระดับจากภาพ (ฉ)*