บทที่ 3

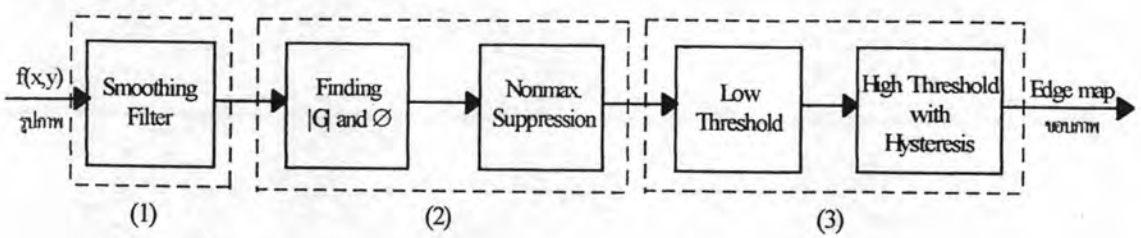
วิธีการหาขอบภาพของ Canny โดยซอฟต์แวร์

การหาขอบภาพของ Canny เป็นวิธีที่มีความซับซ้อน ดังนั้นในการที่จะนำมาออกแบบเป็นฮาร์ดแวร์ จึงจำเป็นอย่างยิ่งที่จะต้องเข้าใจในทุกขั้นตอนของ Canny และวิธีหนึ่งที่ช่วยในการทำความเข้าใจวิธีหาขอบภาพของ Canny ก็ด้วยการเขียนโปรแกรมหาขอบภาพที่ใช้ขั้นตอนของ Canny

ในการเขียนโปรแกรมหาขอบภาพโดยวิธีของ Canny นี้ จะไม่ใช่เป็นการเขียนโปรแกรมให้หาขอบภาพตามวิธีที่ Canny ใช้ทั้งหมด แต่จะเป็นการดัดแปลงเพื่อทำให้เหมาะสมกับการนำไปออกแบบเป็นฮาร์ดแวร์

ขั้นตอนวิธีการหาขอบภาพของ Canny โดยซอฟต์แวร์

จุดประสงค์หลักในการเขียนโปรแกรมเพื่อใช้ในการหาขอบภาพตามวิธีของ Canny ก็เพื่อให้การออกแบบโปรแกรมเป็นแนวทางในการออกแบบฮาร์ดแวร์อีกทีหนึ่ง ดังนั้นโปรแกรมที่ออกแบบจึงได้มีการดัดแปลงการคำนวณเพื่อให้เหมาะสมแก่การนำไปประยุกต์เป็นแนวทางในการออกแบบฮาร์ดแวร์และใช้ในการเปรียบเทียบผลระหว่างซอฟต์แวร์กับฮาร์ดแวร์ ส่วนภาษาที่ใช้ในการเขียนโปรแกรมคือ ภาษา C



รูปที่ 3.1 ขั้นตอนวิธีของ Canny โดยซอฟต์แวร์ในการหาเส้นขอบภาพจากรูปภาพ

จากขั้นตอนวิธีของ Canny ในรูปที่ 2.9 เปลี่ยนมาเป็นซอฟต์แวร์โดยแบ่งออกเป็น 3 โปรแกรมย่อยได้ดังรูปที่ 3.1 คือ

1. โปรแกรมสำหรับการเกลี่ยภาพ โดยใช้ตัวกรองความถี่แบบไบโนเมียล แทนตัวกรองความถี่ที่ใช้ฟังก์ชันเกาส์เซียน

2. โปรแกรมสำหรับการหาค่าขนาดและทิศทางของกราเดียนต์ แล้วนำมาหาจุดภาพที่มีขนาดสูงสุดเฉพาะถิ่น
3. โปรแกรมสำหรับการตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำและค่าสูงแบบฮิสเตอร์เรซิส

โดยทั้ง 3 โปรแกรมจะยึดรูปแบบแกน x และแกน y ตามรูปที่ 2.3(ก) ส่วนเหตุที่แยกออกเป็น 3 โปรแกรม ก็เพื่อให้สามารถตรวจสอบผลลัพธ์ได้เป็นส่วนๆ และสามารถนำไปใช้เพื่อเป็นแนวทางในการออกแบบฮาร์ดแวร์ทีละส่วน

<u>โปรแกรมที่ 1</u> โปรแกรมสำหรับการเกลี่ยภาพ

โปรแกรมสำหรับการเกลี่ยภาพ โดยใช้ตัวกรองความถี่แบบไบโนเมียล แทนตัวกรองความถี่ที่ใช้ฟังก์ชันเกาส์เซียน สาเหตุที่ใช้ตัวกรองความถี่แบบไบโนเมียลแทน ก็เพื่อทำให้เกิดความรวดเร็วในการคำนวณ และทำให้สามารถนำมาออกแบบสร้างเป็นฮาร์ดแวร์ได้ง่ายขึ้น

สำหรับตัวกรองความถี่แบบไบโนเมียลที่ใช้ได้นำมาจากของ ยุทธพงษ์ รังสรรค์เสรี (2537ข) ซึ่งเป็นหน้ากากเพื่อใช้คอนโวลูชันขนาด 3 x 3 ที่ให้ผลตอบสนองได้ใกล้เคียงกับตัวกรองความถี่ที่ใช้ฟังก์ชันเกาส์เซียน ซึ่งมีรูปแบบดังนี้

$$f(x,y,n) = \frac{1}{16^n}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}^{*n}$$

โดยให้สัญลักษณ์ '*n' แทนการทำคอนโวลูชันของตัวหน้ากากกับรูปภาพ n ครั้ง โดยที่ n เป็นพารามิเตอร์ที่ปรับค่าได้ เทียบกับ σ ในฟังก์ชันเกาส์เซียนได้ดังนี้

$$\sigma^2 = 0.5n$$

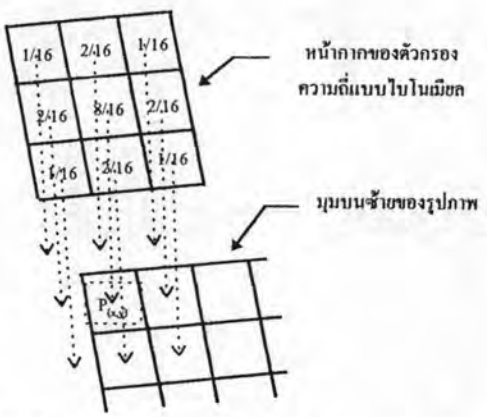
นั่นคือถ้าต้องการ σ เท่ากับ 1 ก็ต้องใช้ n เท่ากับ 2 หรือคือต้องทำคอนโวลูชันของตัวหน้ากากกับรูปภาพ 2 ครั้ง ซึ่งเราจะใช้ σ เท่ากับ 1 ในการออกแบบฮาร์ดแวร์

ซึ่งโปรแกรมที่เขียนใช้ชื่อว่า "ABINO" เวลาเรียกใช้ต้องตามด้วยอาร์กิวเมนต์(argument) สามตัว คือ ไฟล์รูปภาพอินพุต, ไฟล์รูปภาพเอาท์พุต และค่า n (จำนวนครั้งของการทำคอนโวลูชัน) ซึ่งไฟล์รูปภาพอินพุตที่ใช้ ได้มาจากไฟล์รูปภาพที่มากับหนังสือของ Sid-Ahmed (1995) ซึ่งเป็นข้อมูลของรูปภาพขนาด 256 x 256 จุดภาพ 256 ระดับสีเทา นั่นคือใช้ 8 บิตต่อ 1 จุดภาพ เป็นข้อมูลดิบโดยไม่มีรูปแบบ และไม่มีการบีบอัด ดังนั้นทุกไฟล์รูปภาพจะมีขนาด 65,536 ไบต์

เมื่อโปรแกรมเริ่มทำงาน โปรแกรมจะทำการหาขนาดความกว้างและความยาวของรูปภาพ โดยใช้ค่ารากที่สองของไฟล์รูปภาพอินพุตเป็นขนาดความกว้างและความยาวของรูปภาพ

แล้วโปรแกรมก็จะทำการคอนโวลูชันตัวกรองความถี่แบบไบโนเมียลกับจุดภาพของไฟล์รูปภาพอินพุตแล้วเก็บผลลัพธ์ที่ได้ไว้ที่ไฟล์ชั่วคราวที่โปรแกรมสร้างขึ้น ซึ่งใช้ 8 บิตต่อจุดภาพ ทำให้ต้องมีการตัดเศษที่ได้จากการคำนวณทิ้ง (ที่ไม่ทำการปัดเศษก็เพื่อให้ง่ายในการนำไปทำเป็นฮาร์ดแวร์ จะได้ไม่ต้องมีวงจรปัดเศษ) เพื่อให้เก็บเป็นจำนวนเต็มบวกขนาด 8 บิตต่อจุดภาพได้ หลังจากทำเสร็จก็เท่ากับทำคอนโวลูชันไปแล้วหนึ่งครั้ง แล้วก็นำไฟล์ชั่วคราวนี้มาคอนโวลูชันกับตัวกรองความถี่แบบไบโนเมียลอีก เป็นแบบนี้จนกว่าจะครบ n-1 ครั้ง จึงเปลี่ยนชื่อไฟล์ชั่วคราวนี้เป็นชื่อของไฟล์รูปภาพเอาท์พุต

แต่จุดภาพที่ตำแหน่งกรอบรูปภาพ จะได้ค่าจากการคอนโวลูชันที่ไม่ถูกต้อง เพราะว่าหน้ากากของตัวกรองความถี่ที่ทำคอนโวลูชัน ณ บริเวณกรอบภาพที่อยู่นอกกรอบภาพจะต้องให้ค่านั้นเป็นศูนย์แทน ดังเช่นในรูปที่ 3.2



รูปที่ 3.2 แสดงการคอนโวลูชันที่ตำแหน่งขอบของภาพ (ในรูป ณ ตำแหน่งมุมบนซ้ายของรูปภาพ)

ถ้าพิจารณาดูจะพบว่าการคอนโวลูชันในแต่ละครั้งจะทำให้จำนวนจุดภาพที่มีค่าที่ถูกต้องของรูปภาพมีขนาดเล็กลงจาก X*Y เป็น (X-1)*(Y-1) ทั้งนี้เพราะค่าจุดภาพที่อยู่ที่ขอบของภาพ ถ้าต้องการได้ค่าของการคอนโวลูชันที่ตำแหน่งนี้ จะต้องเติมจำนวนจุดภาพเพิ่มต่อจากตำแหน่งขอบของภาพดังรูปข้างบน เพื่อให้มีจำนวนจุดภาพพอดีกับหน้ากากของตัวกรองความถี่ หรืออาจจะเลือกไม่เอาค่าขอบของภาพก็จะเป็นการตัดไม่คำนวณหาค่าคอนโวลูชัน ณ ตำแหน่งขอบของภาพ ภาพที่ได้ก็จะมีขนาดลดลงทุกครั้งที่ผ่านการคอนโวลูชัน แต่ที่นิยมกันจะเลือกให้ขนาดของภาพหลังการคอนโวลูชันมีขนาดเท่าเดิม ดังนั้นจึงต้องหาค่าของจุดภาพมาเติมต่อจากตำแหน่งของขอบภาพ ซึ่งวิธีที่นิยมก็คือใช้ค่าศูนย์และอีกวิธีคือมองเหมือนกับว่าขอบของภาพด้านซ้ายจะติดกับขอบของภาพด้านขวาและขอบของภาพด้านบนติดกับด้านล่าง ซึ่งในที่นี้เราจะเลือกวิธีเติมค่าศูนย์ เพื่อให้จำนวนจุดภาพ(ขนาด)ของภาพที่ได้หลังการคอนโวลูชันจะยังมีจำนวนหรือขนาดเท่าเดิม และสะดวกกว่า

การมองว่าขอบของภาพต่อติดกัน เพราะไม่ต้องอ่านค่าขอบของภาพอีกด้านหนึ่งเพื่อมาใช้เป็นค่าเพิ่มเติม ทำให้ประหยัดเวลาในการอ่านและการคำนวณ เพราะศูนย์คูณกับอะไรก็ได้ศูนย์

<u>โปรแกรมที่ 2</u> โปรแกรมสำหรับการหาค่าขนาดและทิศทางของกราเดียนต์ แล้วหาจุดสูงสุดเฉพาะถิ่น

โปรแกรมสำหรับการหาค่าขนาด |G| และทิศทางของกราเดียนต์ ϕ แล้วนำมาหาจุดภาพที่มีขนาดสูงสุดเฉพาะถิ่น เรียกว่า non-max suppression โดยตัวโปรแกรมจะชื่อว่า "AMAX" ซึ่งเวลาเรียกใช้จะต้องตามด้วยอาร์กิวเมนต์ 2 ตัว คือ ไฟล์รูปภาพอินพุต และไฟล์รูปภาพเอาท์พุต

โปรแกรมจะเริ่มต้นด้วยการหาขนาดความกว้างและความยาวของรูปภาพเช่นเดียวกับโปรแกรมที่ 1 จากนั้นโปรแกรมก็จะใช้หน้ากาก 2 หน้ากากดังข้างล่าง เพื่อหาค่าอนุพันธ์อันดับที่หนึ่งในทิศทางแกน y และแกน x ตามลำดับ (ทิศทางแกนตามรูปที่ 3.2(ก))

$$\begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \quad \text{และ} \quad \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

เพื่อใช้ในการคอนโวลูชันรูปภาพอินพุตให้ได้ค่ากราเดียนต์ในทิศทางแกน y (G_y) โดยใช้หน้ากากทางด้านซ้ายมือ และค่ากราเดียนต์ในทิศทางแกน x (G_x) โดยใช้หน้ากากทางด้านขวามือ เหตุผลที่ใช้หน้ากากทั้งสองแบบนี้แทนหน้ากากแบบที่ Canny ใช้ ก็เพราะว่า Fleck (1992) ได้แนะนำหน้ากากแบบใหม่ โดยให้ใช้หน้ากาก $\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$ ในการคำนวณหาค่า G ทั้ง 4 ทิศทางคือ ในทางแนวนอนให้เป็น H, ในทางแนวตั้งให้เป็น V และในทางแนวเฉียงให้เป็น D_1 และ D_2 ตามลำดับ ซึ่งสามารถคำนวณหา G_y และ G_x ได้ดังนี้

$$G_y = H + \frac{D_1 + D_2}{2}$$

$$G_x = V + \frac{D_1 - D_2}{2}$$

ดังนั้นจึงได้นำหน้ากากแบบที่ Fleck ใช้มาดัดแปลง ให้เหลือแค่หน้ากากในแนวนอน H และแนวตั้ง V ทำให้ G_y และ G_x มีค่าเป็นดังนี้

$$G_y = H$$

$$G_x = V$$

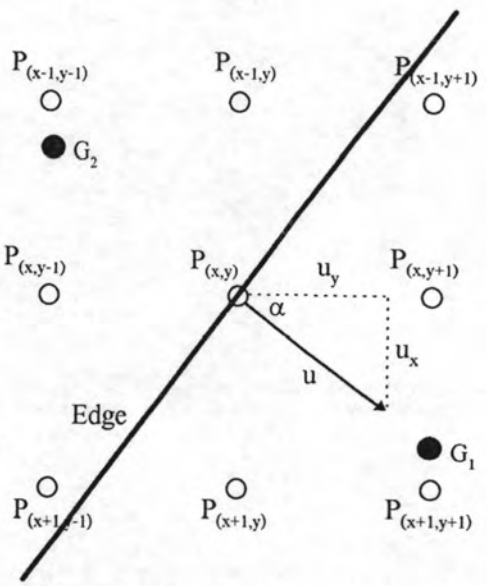
ซึ่งตรงกับวิธีที่ ยุทธพงษ์ รังสรรค์เสรี (2537ก) ได้เสนอไว้

หลังจากที่ได้ G_y และ G_x ก็คำนวณหาขนาดของกราเดียนต์ ($|G|$) และทิศทางของกราเดียนต์ (ϕ) ด้วยสมการดังนี้

$$|G| = \sqrt{G_x^2 + G_y^2}$$

$$\phi = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

แล้วหลังจากนั้นก็นำค่าทั้งสองนี้มาคำนวณหาจุดภาพที่มีค่าสูงสุดเฉพาะถิ่น แสดงวิธีการคำนวณดังตัวอย่างด้วยรูปข้างล่าง



<u>รูปที่ 3.3</u> แสดงการหาทิศทางค่ากราเดียนต์ของขอบภาพ ในรูปคือเวคเตอร์ u

ดังตัวอย่างในรูปข้างบนเป็นกรณี $45^\circ < \phi < 90^\circ$ จะได้

$$G_1 = \frac{u_x}{u_y} G_{(x+1,y+1)} + \frac{u_y - u_x}{u_y} G_{(x,y+1)}$$

และ

$$G_2 = \frac{u_x}{u_y} G_{(x-1,y-1)} + \frac{u_y - u_x}{u_y} G_{(x,y-1)}$$

หรือคือ

$$G_1 = (\tan(\alpha))(G_{(x+1,y+1)}) + (1 - \tan(\alpha))(G_{(x,y+1)})$$

และ

$$G_2 = (\tan(\alpha))(G_{(x-1,y-1)}) + (1 - \tan(\alpha))(G_{(x,y-1)})$$

โดยมุม α ได้มาจาก $\alpha = \tan^{-1}\left(\frac{u_x}{u_y}\right)$

หอสมุดกลาง สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ถ้า $G > G_1$ และ $G \geq G_2$ ก็แสดงว่าจุดภาพนี้เป็นตำแหน่งของเส้นขอบภาพก็ให้มีค่า G คงเดิม แต่ถ้าไม่ใช่แสดงว่าจุดภาพนี้ไม่ใช่ตำแหน่งของเส้นขอบภาพ ก็ให้แก้ค่า G ของจุดภาพให้เป็นศูนย์ นั่นคือ ถ้าค่ากราเดียนต์ของจุดภาพเป็นศูนย์อยู่แล้ว ก็ให้ข้ามไปจุดภาพต่อไปได้เลย ทำอย่างนี้จนกระทั่งครบทุกจุดภาพ

เหตุที่ต้องกำหนดให้ $G > G_1$ และ $G \geq G_2$ แทนที่จะเป็น $G > G_1$ และ $G > G_2$ ก็เพราะว่าบางครั้งค่าของจุดภาพที่ได้มีค่าที่เท่ากัน แทนที่จะมีค่าสูงสุดที่จุดภาพใดจุดภาพหนึ่ง เพื่อป้องกันไม่ให้เส้นขอบขาดหายไปเพราะจุดภาพที่มีค่าที่เท่ากัน

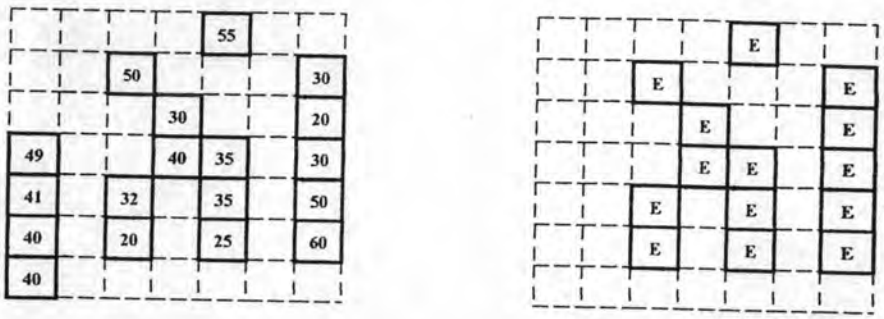
<u>โปรแกรมที่ 3</u> โปรแกรมสำหรับการตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำและค่าสูงแบบฮิสเตอร์เรซิส

โปรแกรมสำหรับการตรวจสอบขีดเริ่มเปลี่ยนทั้งค่าต่ำและค่าสูงแบบฮิสเตอร์เรซิส ใช้ชื่อว่า "ATHD" เวลาเรียกใช้ต้องตามด้วยอาร์กิวเมนต์ 4 ตัว คือ ไฟล์รูปภาพอินพุต, ไฟล์รูปภาพเอาท์พุต, ขีดเริ่มเปลี่ยนค่าต่ำ และขีดเริ่มเปลี่ยนค่าสูง

โปรแกรมนี้เริ่มจากหาขนาดความกว้างและความยาวของรูปภาพอินพุต แล้วทำการจองพื้นที่หน่วยความจำให้เป็นตามขนาดความกว้างและความยาวตามที่คำนวณ จากนั้นก็เริ่มตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำ โดยอ่านจุดภาพรูปภาพเข้ามา พร้อมกับพิจารณาว่าค่าของจุดภาพมีค่าสูงกว่าหรือเท่ากับค่าขีดเริ่มเปลี่ยนค่าต่ำหรือไม่ ถ้าใช่ก็ให้นำค่าจุดภาพนั้นไปเก็บไว้ในหน่วยความจำที่แทนตำแหน่งของจุดภาพนั้น ถ้าไม่ใช่ก็จะใส่ค่าศูนย์ลงไปในหน่วยความจำที่แทนตำแหน่งของจุดภาพนั้นแทน ทำอย่างนี้จนครบทุกจุดภาพ

จากนั้นก็ตรวจสอบขีดเริ่มเปลี่ยนค่าสูง โดยการพิจารณาค่าของจุดภาพที่เก็บในหน่วยความจำว่ามีค่าสูงกว่าหรือเท่ากับขีดเริ่มเปลี่ยนค่าสูงหรือไม่ ถ้าใช่ก็ให้บันทึกตำแหน่งของจุดภาพนั้นเป็นตำแหน่งของเส้นขอบลงในไฟล์รูปภาพเอาท์พุต ถ้าไม่ใช่ก็ให้พิจารณาจุดภาพที่อยู่รอบจุดภาพนี้ว่ามีค่าสูงกว่าหรือเท่ากับขีดเริ่มเปลี่ยนค่าสูงหรือไม่ ถ้าใช่ก็ให้บันทึกตำแหน่งของจุดภาพเริ่มต้นเป็นตำแหน่งของเส้นขอบลงในไฟล์รูปภาพเอาท์พุต ถ้าไม่ใช่ก็ให้พิจารณาหาจุดภาพที่อยู่รอบจุดภาพที่มีค่ามากกว่าศูนย์ แล้วตรวจสอบว่าจุดภาพที่อยู่รอบจุดภาพที่มีค่ามากกว่าศูนย์นี้มีค่ามากกว่าหรือเท่ากับขีดเริ่มเปลี่ยนค่าสูงหรือไม่ ถ้าใช่ก็ให้บันทึกตำแหน่งของจุดภาพเริ่มต้นเป็นตำแหน่งของเส้นขอบลงในไฟล์รูปภาพเอาท์พุต แต่ถ้าไม่ใช่ก็ให้พิจารณาต่อไป ซึ่งวิธีที่โปรแกรมนี้ทำงานจึงเป็นแบบการเวียนเกิด(recursive) โดยกำหนดให้มีการเวียนเกิดได้มากสุดไว้ 25 ครั้ง นั่นคือเท่ากับว่าโปรแกรมนี้สามารถตรวจสอบเส้นขอบภาพได้ยาวสุด 26 จุดภาพ(นับรวมจุดภาพ ณ ตำแหน่งเริ่ม

ต้นด้วย) และทุกครั้งที่มีการเวียนเกิดถอยกลับมา ก็จะทำการเก็บบันทึกตำแหน่งที่เคยมีการเวียนเกิดเข้าไปแล้ว เพื่อไม่ต้องเสียเวลาเข้าไปทำการเวียนเกิดใหม่อีก วิธีนี้จึงพอใช้ทดแทนวิธีกราฟได้โดยการประกันความต่อเนื่องของเส้นขอบไว้ที่จำนวนของการเวียนเกิดได้สูงสุดบวกหนึ่ง (ดูตัวอย่างประกอบในรูปที่ 3.4)



รูปที่ 3.4 แสดงการตรวจสอบขีดเริ่มเปลี่ยนค่าสูง ในรูปด้านซ้ายได้ออกมาเป็นรูปด้านขวา

ในรูปที่ 3.4 เป็นตัวอย่างผลลัพธ์ที่ได้จากการตรวจสอบขีดเริ่มเปลี่ยนค่าสูงแบบฮิสเตอร์เรซิส โดยสมมติให้ขีดเริ่มเปลี่ยนมีค่า '50' จากในรูปด้านซ้ายมือเป็นรูปที่ผ่านการหาค่าสูงสุดเฉพาะถิ่นและตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำมาแล้ว จะเห็นได้ว่าเส้นที่มีค่าสูงสุดเฉพาะถิ่นไม่ถึงค่า '50' (ในรูปมีเส้นเดียวอยู่ซ้ายมือสุด) เมื่อผ่านการตรวจสอบขีดเริ่มเปลี่ยนค่าสูง เส้นนี้ก็จะหายไป เหลืออยู่แต่เส้นที่ค่าสูงสุดเฉพาะถิ่น(จุดภาพ)ที่ประกอบเป็นเส้นนั้นๆมีอย่างน้อยค่าหนึ่งมีค่ามากกว่าหรือเท่ากับค่าขีดเริ่มเปลี่ยนค่าสูง ซึ่งก็คือเส้นขอบภาพที่โปรแกรมหามาได้ โดยจะตั้งให้ค่าขอบภาพนี้มีค่าเป็นเท่าไรก็ได้ ก็แทนลงไปในค่า 'E' ซึ่งในที่นี้จะใช้ค่าสูงสุดที่เป็นไปได้คือค่า '255'

เมื่อใช้ทั้งสามโปรแกรมในการทำงานเรียงต่อกันก็จะได้ไฟล์รูปภาพที่เป็นผลลัพธ์ของการหาภาพลายเส้นขอบ โดยอาศัยขั้นตอนวิธีของ Canny

ซึ่งสามารถทำการรวมโปรแกรมทั้งสามเข้าเป็นโปรแกรมเดียว โดยตั้งชื่อว่า "CANNY" ซึ่งเวลาเรียกใช้จะต้องตามด้วยอาร์กิวเมนต์ทั้งหมด 5 ตัว คือ ไฟล์รูปภาพอินพุต, ไฟล์รูปภาพเอาท์พุต, ค่าจำนวนครั้งของการเกลี่ยภาพ(ค่า n), ขีดเริ่มเปลี่ยนค่าต่ำ และขีดเริ่มเปลี่ยนค่าสูง

สรุปท้ายบท

เนื้อหาในบทนี้ได้แสดงการทำงานของโปรแกรมเพื่อหาขอบภาพตามวิธีของ Canny จุดประสงค์ในการเขียนโปรแกรมก็เพื่อใช้เป็นแนวทางในการออกแบบฮาร์ดแวร์ และใช้ในการเปรียบเทียบผลเพื่อตรวจสอบฮาร์ดแวร์ โดยแบ่งการโปรแกรมออกเป็น 3 โปรแกรม คือ โปรแกรม

สำหรับการเกลี่ยภาพ, โปรแกรมสำหรับการหาค่าขนาดและทิศทางของกราเดียนต์แล้วหาจุดสูงสุดเฉพาะถิ่น และโปรแกรมสำหรับการตรวจสอบขีดเริ่มเปลี่ยนค่าต่ำและค่าสูงแบบฮิสเตอร์เรซิส