บทที่ 3
อัลกอริทึม

บทนี้จะเป็นการอธิบายการสร้าง Input ให้กับกระบวนการเรียนรู้และจดจำตัวอักษร โดยเริ่มภาพของเอกสารที่เกิดขึ้นจากการนำข้อมูลเข้าด้วยเครื่องกวาดภาพ ซึ่งถูกจัดเก็บด้วยรูปแบบไฟล์ .PCX ไปจนถึงผลลัพธ์ที่อยู่ในลักษณะของ text ไฟล์ที่แยกตัวอักษรแล้ว โดยที่มีตัวอักษรแต่ละตัวอยู่ในพื้นที่ขนาด 8*20 มีขั้นตอนต่างๆดังต่อไปนี้

1) *การแปลงภาพ .PCX เป็น Text ไฟล์* เป็นการถอดรหัสไฟล์ .PCX ออกมาเป็น Text ไฟล์ ทีละจุดโดย ตำแหน่งที่เป็นภาพให้แทนด้วย 1 และตำแหน่งที่ไม่เป็นภาพให้แสดงด้วย 0

2) *การตรวจหามุมเอียง* ภาพที่ได้จากการ scan บางครั้งภาพอาจจะเอียงไปมาซ้ายบ้างขวาบ้าง จึงมีการปรับภาพให้ตรง ขั้นตอนนี้เป็นการหาค่ามุมที่เอียงของภาพที่ scan เข้ามา

3) *การปรับแก้มุมเอียง* เป็นการปรับภาพให้ตรง โดยมุมในการปรับให้ตรงได้จากขั้นตอนที่ 2

4) *การตัดตัวอักษร* เมื่อได้ภาพที่ตรงแล้ว ต่อไปจะเป็นการแยกตัวอักษรโดย ขั้นแรกจะเป็นการ แยกตัวอักษรออกเป็นบรรทัดก่อน (ตัดตามแนวนอน) แล้วจึงตัดตัวอักษรแต่ละตัวออกมา (ตัดตามแนวตั้ง) เมื่อตัดตามแนวตั้งแล้วมีตัวอักษรบางตัว เช่น คื ไม่สามารถตัดออกจากกันได้ จะใช้ขั้นตอนต่อไปในการแยก ค. ควาย กับ สระอือ ออกจากกัน

5) *การตัดขอบของตัวอักษร* เป็นการแยกตัวอักษรออกจากกันขั้นสุดท้ายเพื่อให้ได้ตัวอักษรแต่ละตัวออกมา

6) *การ Fitting* เป็นการปรับพื้นที่ให้มีขนาดพอเหมาะกับตัวอักษร เพราะตัวอักษรแต่ละตัวมีขนาดเล็กใหญ่ไม่เท่ากัน เพื่อให้สะดวกในการทำโครงร่าง และ การปรับขนาดต่อไป

7) *การปรับผิวของตัวอักษรให้เรียบ* เป็นการตัดสัญญาณรบกวนของตัวอักษรที่ scan เข้ามาที่ผิวของตัวอักษร จะมีจุดที่เกินออกไปจากผิวของตัวอักษร หรือ ส่วนที่ยุบเข้ามาในตัวอักษร ในขั้นตอนนี้เป็นการตัดจุดที่เกินออกไปและเติมส่วนที่ยุบเข้ามา ทำให้ตัวอักษรมีผิวเรียบ เพื่อไม่ให้มีการรบกวนการทำโครงร่างและปรับขนาด

8) *การทำเส้นโครงร่าง* เป็นการบีบตัวอักษรให้มีขนาดบางเป็นเส้นโครงร่างเพียงเส้นเดียว เพื่อสะดวกในการปรับขนาด

9) *การปรับขนาด* เป็นการปรับขนาดของตัวอักษรที่ได้จากการทำเส้นโครงร่างแล้ว ให้อยู่ในพื้นที่ขนาด 8*20 เพื่อใช้เป็น Input ของกระบวนการเรียนรู้และจดจำตัวอักษร ต่อไป

3.1 การแปลงภาพ PCX เป็น TEXT FILE

เนื่องจากภาพที่ได้มีขนาดใหญ่มากไม่สามารถที่จะแสดงได้บนจอในหน้าจอเดียว ดังนั้นจึงใช้วิธีจัดเก็บลง TEXT FILE โดยเริ่มแรกใช้การเก็บในลักษณะของเลขฐาน 16 ซึ่งจะทำให้ขนาดของ TEXT FILE มีขนาดที่ไม่ใหญ่มากนัก แต่มีปัญหาในการที่จะเข้าไปดึงข้อมูลจาก TEXT FILE เพราะจะต้องเข้าถึงข้อมูลแต่ละตัว จึงได้ใช้วิธีเก็บแบบตรงๆ โดยจะ เก็บเป็นตัวเลข "0" และ "1"

โดยให้ "0" แทนตำแหน่งที่ไม่มีภาพ

โดยให้ "1" แทนตำแหน่งที่มีภาพ

โดยให้ 1 บรรทัดแทน 1 เส้นโดยที่ปลายสุดของบรรทัดจะมีอักขระ RETURN และ LINE FEEDกำกับอยู่ ในขั้นตอนนี้จะทำการตัดบรรทัดที่มีแต่ช่องว่างออกโดยจะทำการตัดออก 1 บรรทัด ถ้ามีบรรทัดว่างติดกัน 3 บรรทัดเพื่อลดขนาดของข้อมูลที่จะนำไปใช้

3.2 การตรวจเช็คหามุมเอียง

ทำการตรวจภาพที่ได้มาจากข้อ 3.1 ว่ามีความเอียงเป็นมุมเท่าใดเพื่อใช้ในการปรับให้ตรงและสะดวกในการตัดตัวอักษรต่อไป การตรวจหามุมเอียงจากด้านล่างของแถว โดยจะใช้การเช็คเป็นแถบเส้นโดยแถบเส้นมีความกว้างประมาณ 30 จุด scan จากด้านล่าง (ขนาดของแถบที่ใช้ในการเช็คควรจะมีความกว้างมากกว่าตัวอักษร เพื่อลดขนาดของจำนวนจุดที่จะเก็บและตัดปัญหาเรื่องจุดที่เข้าไปในตัวอักษร) เช็คว่ามีสีดำหรือไม่ถ้าไม่มีก็จะขยับแถบเส้นมา scan แถบถัดไปอีก จนเจอสีดำหรือจนสุดด้านบนของภาพโดยจะเริ่มจากด้านซ้ายสุดของภาพ มาจนถึงด้านขวาสุดของภาพโดยจะเก็บตำแหน่ง (Coordinate) ที่เจอสีดำในแต่ละแถบ ตำแหน่งที่ได้จะเป็นจุด ที่จะถูกนำมาใช้ในการคำนวณหาค่ามุมระหว่างจุด 2 จุดโดยจะทำการ หามุมระหว่างจุดทุกจุด แล้วนำค่ามุมที่ได้มา ทำการหาว่าค่ามุมที่มีค่าเท่ากันมากที่สุดคือเท่าใด ค่าตัวเลขของมุมที่ได้นั้นจะนำมาใช้เป็น มุมเอียงของภาพ

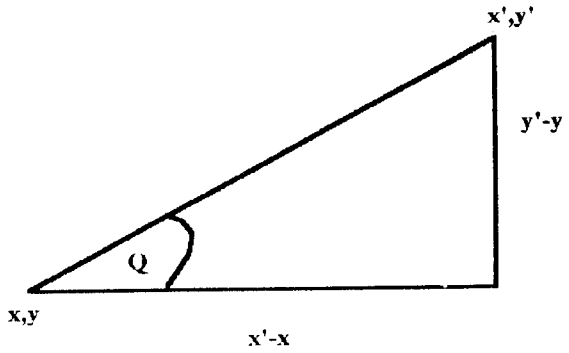
สูตรการหามุมระหว่างจุด 2 จุด

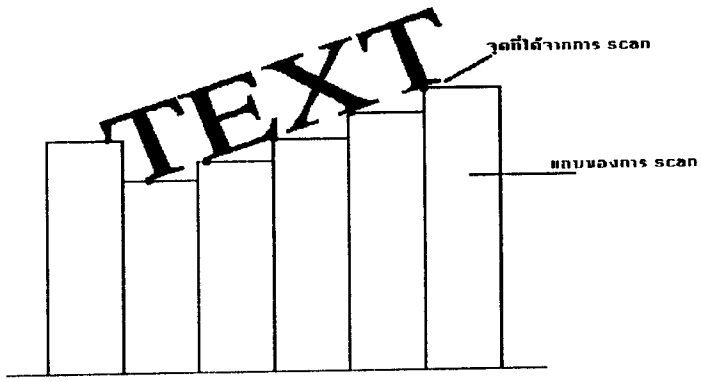
$$\text{มุมเอียง } (Q) = \text{ArcTan}\left((Y'-Y) / (X'-X) \right)$$

โดย จุดใหม่เป็น (X',Y')

จุดเดิมเป็น (X ,Y)

Q คือมุมที่หาได้มีหน่วยเป็น เรเดียน





รูปที่ 3.1 ภาพแสดงแถบของการ scan ตัวอักษร เพื่อหามุมจากจุดที่ได้จากการ scan

3.3 การปรับแก้มุมเอียง

เมื่อได้มุมเอียงของภาพแล้วก็ถึงขั้นตอนการปรับภาพ จะเป็นการสร้างแฟ้มข้อมูลขึ้นมาใหม่เพื่อใช้เก็บภาพที่ทำการมุมแล้ว โดยใช้สูตรการปรับภาพแบบเมตริก โดยให้

ตำแหน่งใหม่เป็น (X',Y')

ตำแหน่งเดิมเป็น (X ,Y)

โดยตำแหน่งใหม่จะได้จากสูตร X' = X*COS(Q)-Y*SIN(Q)

Y' = X*SIN(Q)+Y*COS(Q)

โดย Q เป็นมุมเอียงของภาพเริ่มต้น มีหน่วยเป็นเรเดียน

โดยจะทำการหามุมแบบย้อนกลับ โดยจะใช้หาว่าตำแหน่งใหม่ มาจากตำแหน่งไหนจาก FILE เดิม แล้วทำการอ่านค่าที่ตำแหน่งนั้นมาเขียนลงใน FILE ใหม่ ส่วนในการคำนวณนี้ จะได้ค่าตัวเลขที่เป็นเลขทศนิยมดังนั้นจึงต้องมีการปัดค่าตัวเลขเนื่องจากตำแหน่งใน FILE จะอ้างในลักษณะของเลขจำนวนเต็ม ซึ่งในทำการปัดค่าสามารถปัดได้ 3 วิธี คือปัดลง ปัดขึ้น ปัดขึ้นเมื่อเกินครึ่ง ซึ่งจากการทดลองในตอนแรกได้ทำการใช้วิธีการปัดขึ้นเมื่อเกินครึ่ง แต่ภาพที่มีการปรับจะเกิดมีข้อมูลบางส่วนของตัวอักขระเกินขึ้นมา หรือบางส่วนขาดหายไป จึงเปลี่ยนมาใช้วิธีปัดลง ซึ่งจะได้ภาพที่ดีกว่าส่วนการปัดขึ้นก็จะให้ภาพที่ดีเหมือนกับการปัดลง

แต่ภาพที่ได้จากการปรับภาพ ก็จะเป็นมีความผิดผลาดไปจากเดิมเนื่องจากมีการปัดค่า

เสร็จแล้วก็จะได้ภาพใน text file ที่ปรับตรงแล้วนำไปตัดตัวอักษร

3.4 การตัดตัวอักษร

หลังจากที่ได้ภาพใน text file ที่ถูกปรับให้อยู่ในแนวตรงแล้ว ก็จะนำไปตัดตัวอักษร โดยเริ่มแรกจะทำการตัดออกเป็นบรรทัด ในหนึ่งบรรทัดจะมีทั้งสระบนสระล่างและวรรณยุกต์ ในการตัดจะใช้แถวที่ไม่มีข้อมูลภาพ (เป็น '0' ทั้งแถว) ที่อยู่ติดกันมากกว่า 5 แถวเป็นตัวแบ่งระหว่างบรรทัด ต่อมาจะทำการตัดตัวอักษรในแต่ละบรรทัด โดยใช้การเช็คในแต่ละบรรทัดว่ามี columnใดที่ไม่มีข้อมูลภาพคือมี "0" ทั้ง column ก็จะใช้เป็นตัวแบ่งตัวอักษรแต่ละตัวแล้วนำไปเก็บไว้ใน file ที่มีชื่อว่า temp_cha.dat โดยจะมีการใช้เลข 2 เป็นตัวแบ่งตัวอักษรที่ได้ในแต่ละครั้ง โดยถ้าสิ้นสุดแต่ละบรรทัดจะมีเลข 2 อยู่ติดกัน 2 บรรทัด ซึ่งวิธีนี้จะไม่สามารถแบ่งตัวอักษรบางตัวออกจากกันได้ถ้าตัวอักษรไม่มีช่องว่างในแนว column แบ่งเช่นในตัวอย่าง คี,ข้,มู,มัล ซึ่งจะต้องใช้วิธีการแบ่งโดยใช้การตรวจจับขอบภาพ (Edge detection) เป็นตัวที่จะมาทำการตัดตัวอักษรออกจากกันอีกครั้งหนึ่ง

3.5 การตัดขอบของตัวอักษร (Edging)

การตัดขอบของตัวอักษรใช้วิธีการอ่านข้อมูลจากตัวแปร Arraysหนึ่ง (RawData[ ][ ]) ไปเก็บไว้ในอีกตัวแปร Arraysหนึ่ง (CookData[ ][ ]) โดยมีตัวแปรเก็บค่า Coordinateใหม่ด้วย มีวิธีการดังนี้

1) อ่านข้อมูลเดิมที่ได้จาก Cutchar.exe ใน ตัวแปร Arrays ที่ชื่อ RawData[i][j]

2) พิจารณาจุด RawData[i][j] ตรวจสอบ RawData [i][j] ว่าเป็น '0' หรือ '1'

3) ถ้าเป็น '0' ให้อ่านตัวต่อไป

4) ถ้าเป็น '1' ให้ตรวจสอบจุดทั้ง 8 จุด รอบตัวมันเอง

5) ใน 8 จุด ถ้ามีจุดใดเป็น '1' (สมมุติให้เป็น จุด N) ให้ CookData[Px][Py] เป็น '1' โดย Px และ Py เป็นตัวแปรบอก Coordinate ใหม่ของข้อมูลตัวอักษร

6) ให้ RawData[i][j] ที่พบ '1' เป็น '0'

7) พิจารณาจุด N ที่เป็น '1' ต่อไป (ทำซ้ำข้อ 4-7 ไปเรื่อยๆจน RawData[ ][ ] เป็น '0' หมด)

ตัวอย่างการทำงานของ Function การตัดตัวอักษร พิจารณา รูปที่ 3.2

j	j+1	j+2	j+3	j+4	
0	0	0	0	0	i+0
0	0	1	1	0	i+1
0	1	1	1	0	i+2
0	0	1	0	0	i+3
0	0	0	0	0	i+4

RawData [ ][ ]

	Py-1	Py	Py+1	Py+2
Px-1	0	0	0	0
Px	0	0	0	0
Px+1	0	0	0	0
Px+2	0	0	0	0

CookData [ ][ ]

รูปที่ 3.2 แสดงการ Edging โดยตัวแปร 2 ตัว

พิจารณาจุด RawData[i+2][j+1] = '1' ให้พิจารณา 8 จุดรอบ RawData[i+2][j+1] จะเห็นว่ามี RawData[i+1][j+2] , RawData[i+2][j+2] ,และ RawData[i+3][j+2] เป็น '1' ดังนั้นจะได้ CookData [Px][Py] = '1' และ RawData[i+2][j+1] ='0' จะได้ รูปที่ 3.3

j	j+1	j+2	j+3	j+4	
0	0	0	0	0	i
0	0	1	1	0	i+1
0	0	1	1	0	i+2
0	0	1	0	0	i+3
0	0	0	0	0	i+4

RawData [ ][ ]

	Py-1	Py	Py+1	Py+2
Px-1	0	0	0	0
Px	0	1	0	0
Px+1	0	0	0	0
Px+2	0	0	0	0

CookData [ ][ ]

รูปที่ 3.3 แสดงการ Edging โดยตัวแปร 2 ตัว (ต่อจากรูปที่ 3.2)

ในการพิจารณาจุดต่อไป ในกรณีนี้จะเห็นว่ามี 3 จุดที่อยู่รอบจุดพิจารณาเดิม จะพิจารณาจุดที่พบก่อน ในที่นี้สมมุติให้เป็น RawData[i+3][j+2] พิจารณา 8 จุดรอบ RawData[i+3][j+2] จะเห็นว่ามี RawData[i+2][j+2] และ, RawData[i+2][j+3] เป็น '1' ดังนั้นจะได้ CookData [Px+1][Py+1] = '1' และ RawData[i+3][j+2] ='0' จะได้ รูปที่ 3.4

j	j+1	j+2	j+3	j+4	
0	0	0	0	0	i
0	0	1	1	0	i+1
0	0	1	1	0	i+2
0	0	0	0	0	i+3
0	0	0	0	0	i+4

RawData [ ][ ]

	Py-1	Py	Py+1	Py+2
Px-1	0	0	0	0
Px	0	1	0	0
Px+1	0	0	1	0
Px+2	0	0	0	0

CookData [ ][ ]

รูปที่ 3.4 แสดงการ Edging โดยตัวแปร 2 ตัว (ต่อจากรูปที่ 3.3)

พิจารณาอย่างนี้ไปเรื่อยๆจนกระทั่ง RawData[ ][ ] เป็น '0' ทั้งหมด จะได้ รูปที่ 3.5

j	j+1	j+2	j+3	j+4	
0	0	0	0	0	i
0	0	0	0	0	i+1
0	0	0	0	0	i+2
0	0	0	0	0	i+3
0	0	0	0	0	i+4

RawData [ ][ ]

	Py-1	Py	Py+1	Py+2
Px-1	0	0	1	1
Px	0	1	1	1
Px+1	0	0	1	0
Px+2	0	0	0	0

CookData [ ][ ]

รูปที่ 3.5 แสดงผลสุดท้ายของการ Edging โดยตัวแปร 2 ตัว

3.6 การ Fitting.

หลังจากที่ทำการตัดคำเสร็จเรียบร้อยแล้วจะได้ Array ที่มีขนาดใหญ่แต่มีตัวอักษร ขนาดเล็กซึ่งจะเป็นปัญหา ในการ scaling ดังนั้นจะต้องทำการตัด Array ให้มีขนาดพอดี กับตัวอักษร โดยมีหลักการทำงานดังนี้

1.ทำการจอง array เพื่อใช้ในการเก็บค่าของ input ซึ่งเป็นผลที่ได้จากการตัดคำ

2.กำหนดตัวแปร 4 ตัว เพื่อใช้ในการเก็บจุดสิ้นสุดของแต่ละด้าน (left , right , up , down)

3.ทำการ loop จากแถวแรกลงมาเพื่อทำการหาจุดดำจุดแรก แล้วเก็บค่านั้นเอาไว้ใน up

4.ทำการ loop จากแถวล่างขึ้นมาเพื่อทำการหาจุดดำจุดแรก แล้วเก็บค่านั้นเอาไว้ใน down

5.ทำการ loop จากซ้ายไปขวาเพื่อทำการหาจุดดำจุดแรก แล้วเก็บค่านั้นเอาไว้ใน left

6.ทำการ loop จากขวาไปซ้ายเพื่อทำการหาจุดดำจุดแรก แล้วเก็บค่านั้นเอาไว้ใน right

7.ทำการจอง array ที่มีขนาด

row = down - up + C

col = right - left + D

ซึ่งค่า C,D เป็นค่าคงที่ ที่ใช้จอง array ที่จองเผื่อเอาไว้ ในโปรแกรมกำหนดเป็น 3

8.ทำการcopy ค่าจาก array input มายัง array output ที่ได้ทำการจองเอาไว้โดยเริ่มจากตำแหน่งที่ up,left

3.7 การทำผิวของตัวอักษรให้เรียบ

ภาพที่ได้จากการ SCAN ยังคงไม่สวยงามเท่าไร จึงควรมีการปรับผิวของตัวอักษรให้เรียบเป็นการลดสัญญาณรบกวน(noise) วิธีหนึ่ง โดยมีหลักการพิจารณาดังนี้

P1	**P2**	**P3**
P4	**P**	**P5**
P6	**P7**	**P8**

รูปที่ 3.6 ภาพในการพิจารณาจุดและจุดรอบตัวมันเอง

1) ถ้า P = '0' และ (P1=P2=P3=P4=P5='1') หรือ (P1=P2=P4=P6=P7= '1') หรือ (P4=P5=P6=P7=P8= '1') หรือ (P2=P3=P5=P7=P8= '1') แล้ว P= '1'

2) ถ้า P = '1' และ (P1=P2=P3=P4=P5='0') หรือ (P1=P2=P4=P6=P7= '0') หรือ (P4=P5=P6=P7=P8= '0') หรือ (P2=P3=P5=P7=P8= '0') แล้ว P= '0'

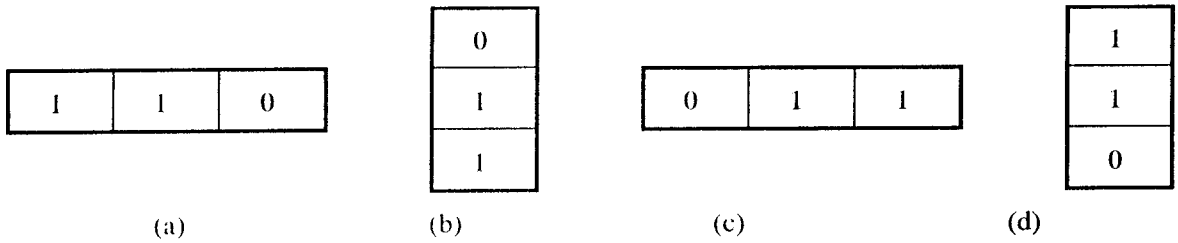
3.8 การทำเส้นโครงร่าง

การทำโครงร่างได้ใช้วิธีการที่เรียกว่า A ROBUST PARALLEL THINNING ALGORITHM FOR BINARY IMAGES โดยมีหลักการทำงานดังนี้

พิจารณา พื้นที่ขนาด 3*3 รอบๆจุด P , เรียกว่า W ดูรูปที่ 3.7 ให้ S เป็น set ของ 8 จุด ใน W คือ P1 , P2 , P3 , P4 , P5 , P6 , P7 ,และ P8

P1	P2	P3
P4	P	P5
P6	P7	P8

รูปที่ 3.7 พื้นที่ขนาด 3*3 รอบจุด P



รูปที่ 3.8 แสดงกรณีต่างๆที่ P สามารถเป็น '1' ได้

พิจารณา W จะเห็นว่า ถ้า P เป็นจุดในกรณี 4 templete ในรูปที่ 3.8 นี้ จะมีเงื่อนไขตาม 4 กรณีข้างล่างต่อไปนี้

a) P เป็น '1' ถ้า P2='0' และ P3='1' หรือ P7='0' และ P8='1'

b) P เป็น '1' ถ้า P4='0' และ P1='1' หรือ P5='0' และ P3='1'

c) P เป็น '1' ถ้า P2='0' และ P1='1' หรือ P7='0' และ P6='1'

d) P เป็น '1' ถ้า P4='0' และ P6='1' หรือ P5='0' และ P8='1'

ถ้า P ไม่อยู่ในกรณีทั้ง 4 นี้นั่นคือ P สามารถเป็น '0' ได้

ตัวอย่างการ thin

พิจารณา กรณี templete a) (ดูรูปที่ 3.8a) ถ้า P4='1', P='1' และ P5='0' แสดงว่า P เป็นขอบ ดังนั้น P สามารถเป็น '0' ได้ แต่ถ้า P2='0' และ P3='1' ถ้า P='0' จะทำให้ P3 และ P4 แยกจากกัน ดังนั้น P ต้องเป็น '1' ทำนองเดียวกัน ถ้า P7='0' และ P8='1'

3.9 การทำ scaling

การทำ scaling เป็นการปรับขนาดของภาพ 1 ตัวอักษร ให้อยู่ในพื้นที่ที่กำหนด ในการทดลองใช้ขนาด 8*20 โดยสามารถปรับขนาดของพื้นที่ได้ที่ตัวแปร NewRow และ NewCol ที่ต้นโปรแกรม เงื่อนไขในการ scale มีดังนี้

สูตรการหา อัตราส่วนการ scale คือ

RowScaleRate = NewRow/OldRow;

ColScaleRate= NewCol/OldCol;

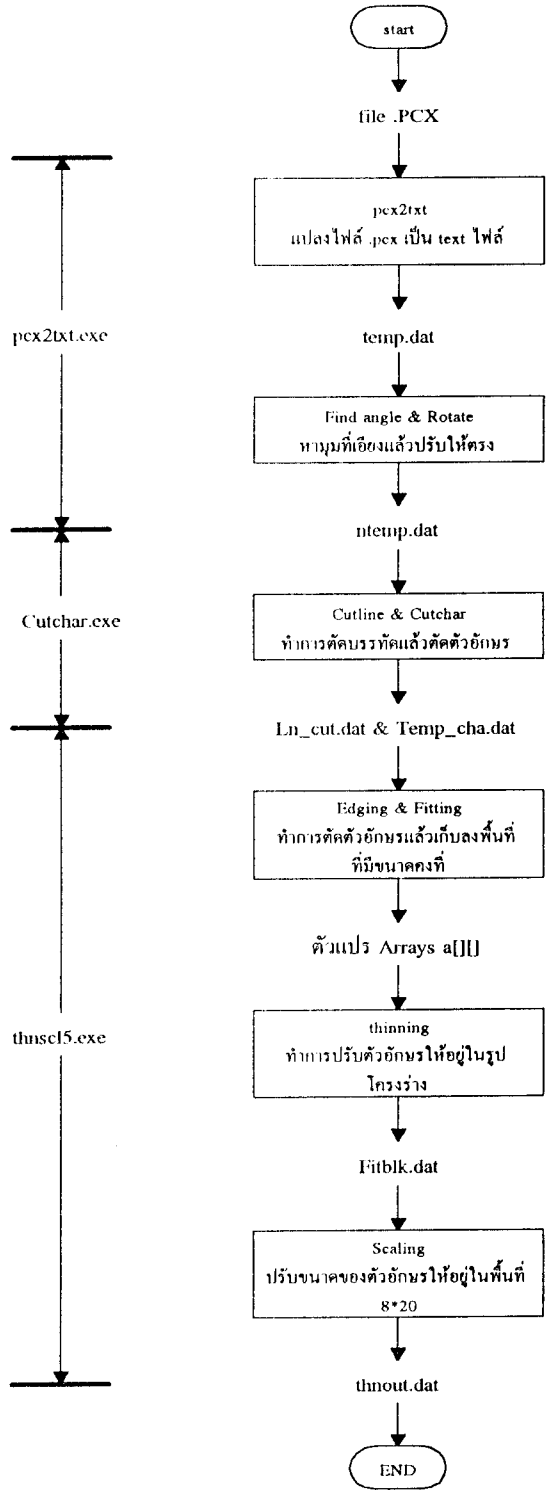
ให้ X เป็น ข้อมูลที่ต้องการ scale และ Y เป็นข้อมูลขนาดที่ scale แล้ว

1) กรณีขยายด้าน Row และ Column พิจารณาว่าแต่ละจุดใน Y ได้มาจาก จุดใดใน X โดยนำค่า Coordinate ใน Y มาคูณกับอัตราส่วนของการ scale จะได้ Coordinate ใน X แล้วดูว่าจุดที่ Coordinate นั้น เป็น '1' หรือไม่ ถ้าเป็น '1' ก็ให้ จุด Coordinate ที่ Y เป็น '1' ด้วย

2) กรณีย่อด้าน Row และ Column ให้เอา Coordinate ของ X มาคูณกับอัตราส่วนของการ scale จะได้ Coordinate ใน Y

3) กรณีย่อด้าน Row และ ขยายด้าน Column ในการย่อด้าน Row ให้พิจารณากรณีที่ 2 ส่วนการขยายด้าน Col ให้พิจารณากรณีที่ 1

4) กรณีขยายด้าน Row และ ย่อด้าน Column ให้พิจารณาเหมือนกรณีที่ 3 แต่ตรงข้ามกัน



รูปที่ 3.9 แสดงลำดับการทำงานของ Program