บทที่ 3

การออกแบบและพัฒนาโปรแกรม

ระบบการแปลงฟอนต์แบบจุดภาพเป็นฟอนต์แบบเวกเตอร์ในที่นี้ จะต้องทำการสร้างฟอนต์แบบจุดภาพเส้นขอบก่อน แล้วจึงแปลงจากฟอนต์แบบจุดภาพเส้นขอบเป็นฟอนต์แบบเวกเตอร์ หลังจากนั้นเราก็จะสามารถนำเวกเตอร์ฟอนต์ที่ได้ ไปประยุกต์ใช้งานในโปรแกรมต่าง ๆ ตามต้องการต่อไป

3.1 การออกแบบ

โครงสร้างการทำงาน ของระบบการแปลงฟอนต์แบบจุดภาพเป็นฟอนต์แบบเวกเตอร์ ประกอบด้วยขั้นตอนต่าง ๆ 4 ขั้นตอน ดังนี้

3.1.1 ขั้นตอนการสร้างฟอนต์แบบจุดภาพ

3.1.2 ขั้นตอนการแปลงฟอนต์แบบจุดภาพเป็นฟอนต์แบบจุดภาพเส้นขอบ

3.1.3 ขั้นตอนการแปลงฟอนต์แบบจุดภาพเส้นขอบเป็นฟอนต์แบบเวกเตอร์ ประกอบด้วยขั้นตอนย่อย คือ

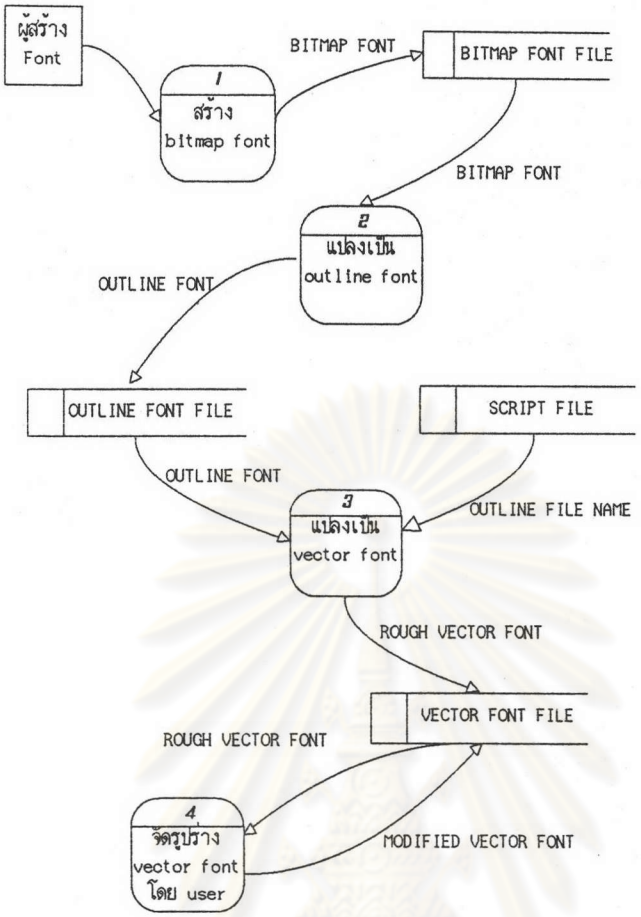
3.1.3.1 ขั้นตอนการตามรอย (trace) ฟอนต์แบบจุดภาพเส้นขอบ

3.1.3.2 ขั้นตอนการตัดแบ่งฟอนต์แบบจุดภาพเส้นขอบเป็นเส้นโค้งย่อย

3.1.3.3 ขั้นตอนการแปลงเส้นโค้งย่อยเป็นเส้นโค้งแบบเบซิเยร์

3.1.4 ขั้นตอนการตัดแปลงรูปร่างเวกเตอร์ฟอนต์ที่ได้ให้สวยงามตามต้องการ

หลังจากได้ทำการแปลงฟอนต์แบบจุดภาพเป็นฟอนต์แบบเวกเตอร์แล้ว ก็จะเป็นขั้นตอนในการนำฟอนต์แบบเวกเตอร์ไปประยุกต์ใช้งานต่อไป



รูปที่ 3.1 แผนผังแสดงการไหลของข้อมูล (DFD) ตามขั้นตอนการทำงานต่าง ๆ ในระบบการแปลงฟอนต์แบบจุดภาพเป็นฟอนต์แบบเวกเตอร์

3.2 โปรแกรมสร้างฟอนต์แบบจุดภาพ

ในส่วนของการรับข้อมูล (input) ของงานวิจัยนี้ จะเป็นการสร้างฟอนต์สัญลักษณ์แบบจุดภาพขึ้น โดยใช้อุปกรณ์เมาส์ (mouse) ในการสร้างจุดบนตาราง ซึ่งจะประกอบกันขึ้นเป็นฟอนต์สัญลักษณ์ โดยอุปกรณ์นี้จะเชื่อมต่ออยู่กับเครื่องไมโครคอมพิวเตอร์ และต้องมีการเรียกโปรแกรมไดร์เวอร์ของเมาส์ก่อน เราจึงสามารถใช้งานโปรแกรมส่วนของการสร้างฟอนต์แบบจุดภาพในงานวิจัยนี้ได้

ฟอนต์แบบจุดภาพที่สร้างขึ้นจะทำการจัดเก็บในรูปแบบที่สะดวกในการตรวจสอบ และสามารถแก้ไขผ่านทางโปรแกรมบรรณาธิกรณ์ (editor) ทั่วไป เพื่อความเหมาะสมในการใช้

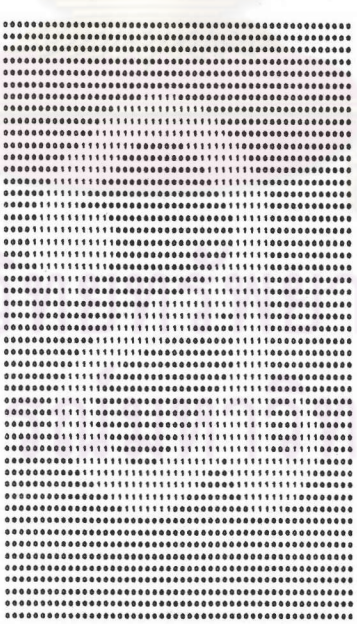
งานด้านการวิจัย โดยการจัดเก็บจะต้องจัดเก็บแยกฟอนต์แบบจุดภาพแต่ละฟอนต์ออกเป็นแต่ละแฟ้มข้อมูล และมีรูปแบบการจัดเก็บ คือ

แถวที่ 1 - FONTWIDTH FONTHEIGHT
แถวที่ 2 - B_{11} B_{12} B_{13} B_{14} B_{15} . . . . B_{1m}
. B_{21} B_{22} B_{23} B_{24} B_{25} . . . . B_{2m}
. B_{31} B_{32} B_{33} B_{34} B_{35} . . . . B_{3m}
. . . . . .
. . . . . .
แถวที่ n - B_{n1} B_{n2} B_{n3} B_{n4} B_{n5} . . . . B_{nm}

รูปที่ 3.2 รูปแบบการเก็บแฟ้มข้อมูลของฟอนต์แบบจุดภาพ

โดยที่ FONTWIDTH และ FONTHEIGHT คือ ความกว้างและความสูงของฟอนต์จุดภาพนั้นตามลำดับ และ B_i คือค่าตัวเลขแสดงจุดปรากฏของฟอนต์จุดภาพ โดยที่ค่า 1 จะแสดงว่าจุดนั้นเป็นจุดปรากฏ และ 0 แสดงว่าเป็นจุดว่าง ส่วน m และ n คือความกว้างและความสูงของฟอนต์ (font width และ font height) ตามลำดับ

ในงานวิจัยนี้ได้กำหนดขนาดความกว้างและความสูงของฟอนต์ คือ 50x50 ตามลำดับ และในรูปที่ 3.3 จะเป็นตัวอย่างหนึ่งของแฟ้มฟอนต์จุดภาพที่สร้างขึ้น



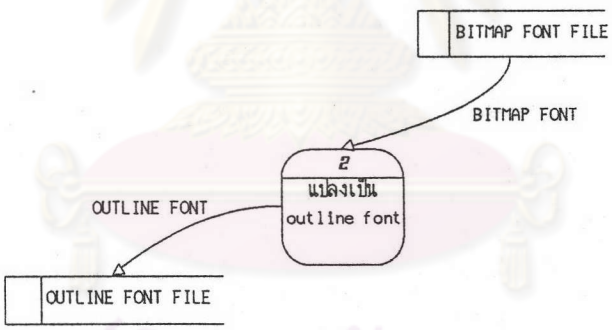
รูปที่ 3.3 ตัวอย่างฟอนต์สัญลักษณ์แบบจุดภาพที่ใช้ในงานวิจัย

3.3 โปรแกรมการแปลงฟอนต์แบบจุดภาพเป็นฟอนต์แบบจุดภาพเส้นขอบ

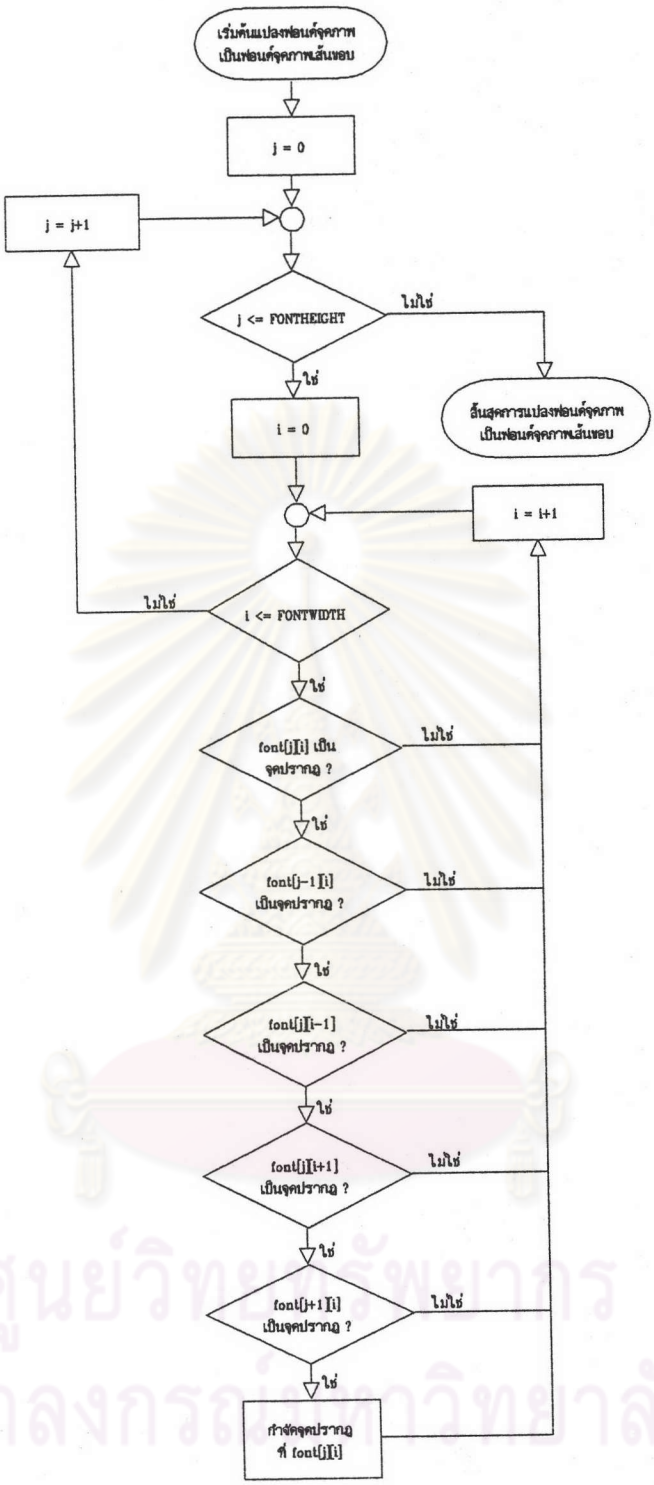
โปรแกรมนี้จะทำการอ่านแฟ้มข้อมูลแบบจุดภาพ ที่ได้จากโปรแกรมการสร้างฟอนต์แบบจุดภาพเข้ามา แล้วจึงทำการกำจัดจุดภาพปรากฏที่เป็นพื้นที่ส่วนทึบของฟอนต์จุดภาพนั้น ให้เหลือแต่จุดภาพที่เป็นเส้นขอบ โดยใช้หลักการตามหัวข้อ 2.2.2 ซึ่งจะทำให้เราได้ฟอนต์แบบจุดภาพเส้นขอบ จากนั้นจึงทำการเก็บฟอนต์จุดภาพเส้นขอบนี้ลงไปในแฟ้มข้อมูลซึ่งมีรูปแบบเช่นเดียวกับฟอนต์แบบจุดภาพเดิม (ตามรูปที่ 3.2)

อนึ่ง ในกรณีที่เราสร้างฟอนต์แบบจุดภาพซึ่งปรากฏแต่เส้นขอบอยู่แล้ว เราอาจไม่ต้องผ่านขั้นตอนนี้ก็ได้ แต่โดยปกติควรผ่านขั้นตอนนี้เสมอ เพื่อกำจัดจุดภาพที่เป็นส่วนทึบออกไป

ในรูปที่ 3.4 จะแสดงให้เห็นถึงการเข้าและออกของแฟ้มข้อมูลผ่านโปรแกรมการแปลงฟอนต์แบบจุดภาพเป็นฟอนต์แบบจุดภาพเส้นขอบ



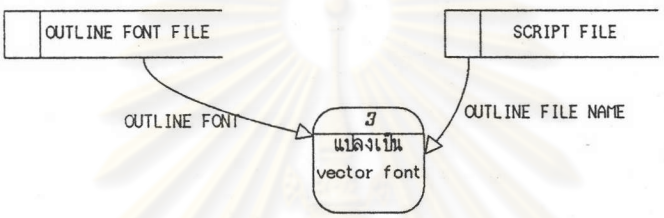
รูปที่ 3.4 แสดงขั้นตอนการไหลของข้อมูลที่เข้าและออก จากโปรแกรมแปลงฟอนต์แบบจุดภาพเป็นฟอนต์แบบจุดภาพเส้นขอบ



รูปที่ 3.5 ผังงานแสดงการทำงานของโปรแกรมการแปลงฟอนต์แบบจุดภาพเป็นฟอนต์แบบจุดภาพเส้นขอบ

3.4 โปรแกรมการแปลงฟอนต์แบบจุดภาพเส้นขอบเป็นฟอนต์แบบเวกเตอร์

โปรแกรมนี้จะทำการอ่านแฟ้มข้อมูลแบบจุดภาพเส้นขอบเข้ามา เพื่อแปลงฟอนต์แบบจุดภาพเส้นขอบให้อยู่ในรูปแบบของฟอนต์แบบเวกเตอร์ โดยที่การอ่านแฟ้มของฟอนต์แบบจุดภาพเส้นขอบจะทำการอ่านครั้งละหลาย ๆ แฟ้มตามที่กำหนด และรวมให้อยู่ในแฟ้มของฟอนต์แบบเวกเตอร์เพียงแฟ้มเดียว ซึ่งการเลือกแฟ้มของฟอนต์แบบจุดภาพเส้นขอบจะเป็นไปตามแฟ้มกำกับ (script file) ว่าจะเลือกแฟ้มใดบ้าง และแปลงให้อยู่ในรหัสแอสกีใด ดังรูปที่ 3.6 ซึ่งจะแสดงถึงขั้นตอนการทำงานของโปรแกรม



รูปที่ 3.6 แสดงขั้นตอนการไหลของข้อมูลเข้าและออกจากโปรแกรมแปลงฟอนต์แบบจุดภาพเส้นขอบเป็นฟอนต์แบบเวกเตอร์

รหัสแอสกี	ชื่อแฟ้มข้อมูลแบบจุดภาพเส้นขอบ
รหัสแอสกี	ชื่อแฟ้มข้อมูลแบบจุดภาพเส้นขอบ
...	...
รหัสแอสกี	ชื่อแฟ้มข้อมูลแบบจุดภาพเส้นขอบ

รูปที่ 3.7 แสดงรูปแบบของข้อมูลที่เก็บอยู่ในแฟ้มกำกับ (script file)

การเก็บข้อมูลส่วนของแฟ้มข้อมูลของฟอนต์แบบเวกเตอร์นั้น จะทำการเก็บข้อมูลของฟอนต์ทั้งหมดเข้าด้วยกัน รูปแบบการเก็บข้อมูลแสดงได้ดังรูปที่ 3.8

ฟอนต์ที่ 1 - รหัสแอสกี จำนวนเส้น (N_1) จุดระบายพื้นแกน x จุดระบายพื้นแกน y
ข้อมูลเส้นเบซิเยร์เส้นที่ 1
ข้อมูลเส้นเบซิเยร์เส้นที่ 2
.
.
ข้อมูลเส้นเบซิเยร์เส้นที่ N_1

ฟอนต์ที่ 2 - รหัสแอสกี จำนวนเส้น (N_2) จุดระบายพื้นแกน x จุดระบายพื้น แกน y
ข้อมูลเส้นเบซิเยร์เส้นที่ 1
ข้อมูลเส้นเบซิเยร์เส้นที่ 2
.
.
ข้อมูลเส้นเบซิเยร์เส้นที่ N_2

.
.

ฟอนต์ที่ m - รหัสแอสกี จำนวนเส้น (N_m) จุดระบายพื้นแกน x จุดระบายพื้น แกน y
ข้อมูลเส้นเบซิเยร์เส้นที่ 1
ข้อมูลเส้นเบซิเยร์เส้นที่ 2
.
.
ข้อมูลเส้นเบซิเยร์เส้นที่ N_m

รูปที่ 3.8 แสดงรูปแบบของข้อมูลที่เก็บอยู่ในแฟ้มของฟอนต์แบบเวกเตอร์ (vector file)

โดยที่ รหัสแอสกีคือ รหัสของฟอนต์แบบเวกเตอร์นี้ซึ่งเราสามารถเรียกใช้งานได้โดยอ้างถึงรหัสแอสกีนี้ จำนวนเส้นคือจำนวนเส้นโค้งแบบเบซิเยร์ดีกรี 2 ที่ใช้ในวาดฟอนต์สัญลักษณ์นี้ จุดระบายพื้นแกน x และจุดระบายพื้นแกน y คือ พิกัดของจุดซึ่งใช้เริ่มต้นในการระบายพื้นที่ภายใน (flood fill) ของฟอนต์สัญลักษณ์นี้

อนึ่ง ฟอนต์แบบเวกเตอร์นี้ จะมีจำนวนฟอนต์บรรจุอยู่ในจำนวนไม่แน่นอน ขึ้นอยู่กับการเพิ่มเติมเข้าไปในแฟ้มควบคุมแต่ละครั้ง แต่จำนวนจะต้องไม่เกิน 256 ฟอนต์ตามข้อจำกัดของรหัสแอสกี

ในการแปลงฟอนต์แบบจุดภาพเส้นขอบเป็นฟอนต์แบบเวกเตอร์ เราจะแบ่งการทำงานภายในโปรแกรมออกเป็น 3 ขั้นตอนด้วยกัน คือ

3.4.1 <u>ขั้นตอนการตามรอย (trace) ฟอนต์แบบจุดภาพเส้นขอบ</u>

ในขั้นตอนนี้จะเป็นการตามรอยจุดปรากฏที่มีอยู่ เพื่อแยกออกเป็นเส้นที่ต่อเนื่องกัน โดยจะมีผลลัพธ์ที่ได้คือ

ก. ได้เส้นขอบ (outline) ที่ต่อเนื่องกัน ซึ่งอาจเป็นพื้นที่ปิดหรือไม่ก็ได้ เพื่อที่จะได้นำมาวิเคราะห์แบ่งเป็นเส้นย่อย ๆ ต่อไป

ข. แยกเส้นขอบทั้งหมดที่มีอยู่ออกมาจากกัน ซึ่งอาจมีจำนวนเส้นขอบมากกว่า 1 เส้นปรากฏอยู่ในฟอนต์สัญลักษณ์ 1 รูปก็ได้

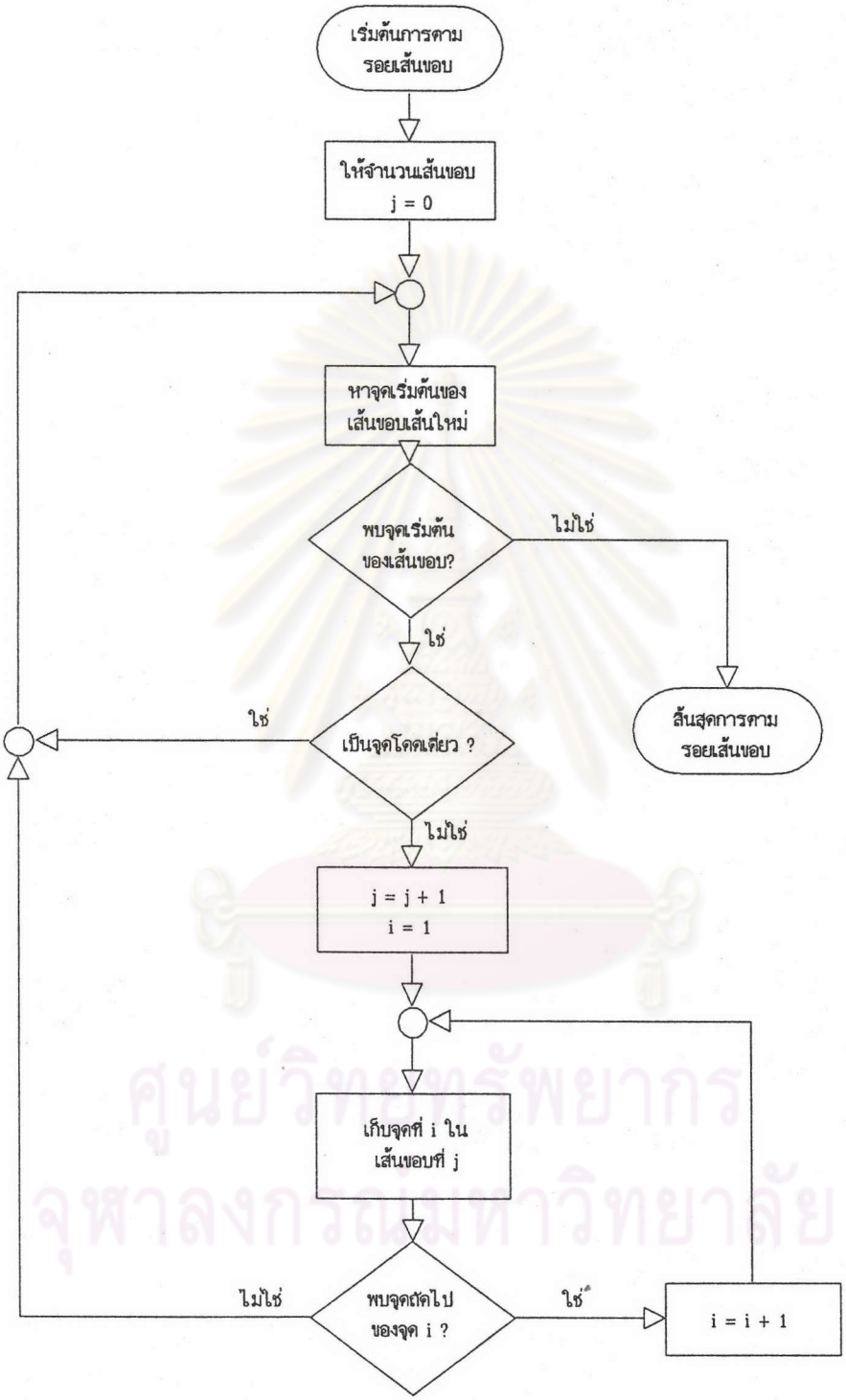
เทคนิคในการตามรอยเส้นขอบนี้ เราสามารถทำได้ตามหลักการในหัวข้อ 2.2.3 และมีขั้นตอนการทำงานของโปรแกรมแสดงได้ดังผังงานตามรูปที่ 3.9

3.4.2 <u>ขั้นตอนการตัดแบ่งฟอนต์แบบจุดภาพเส้นขอบเป็นเส้นโค้งย่อย</u>

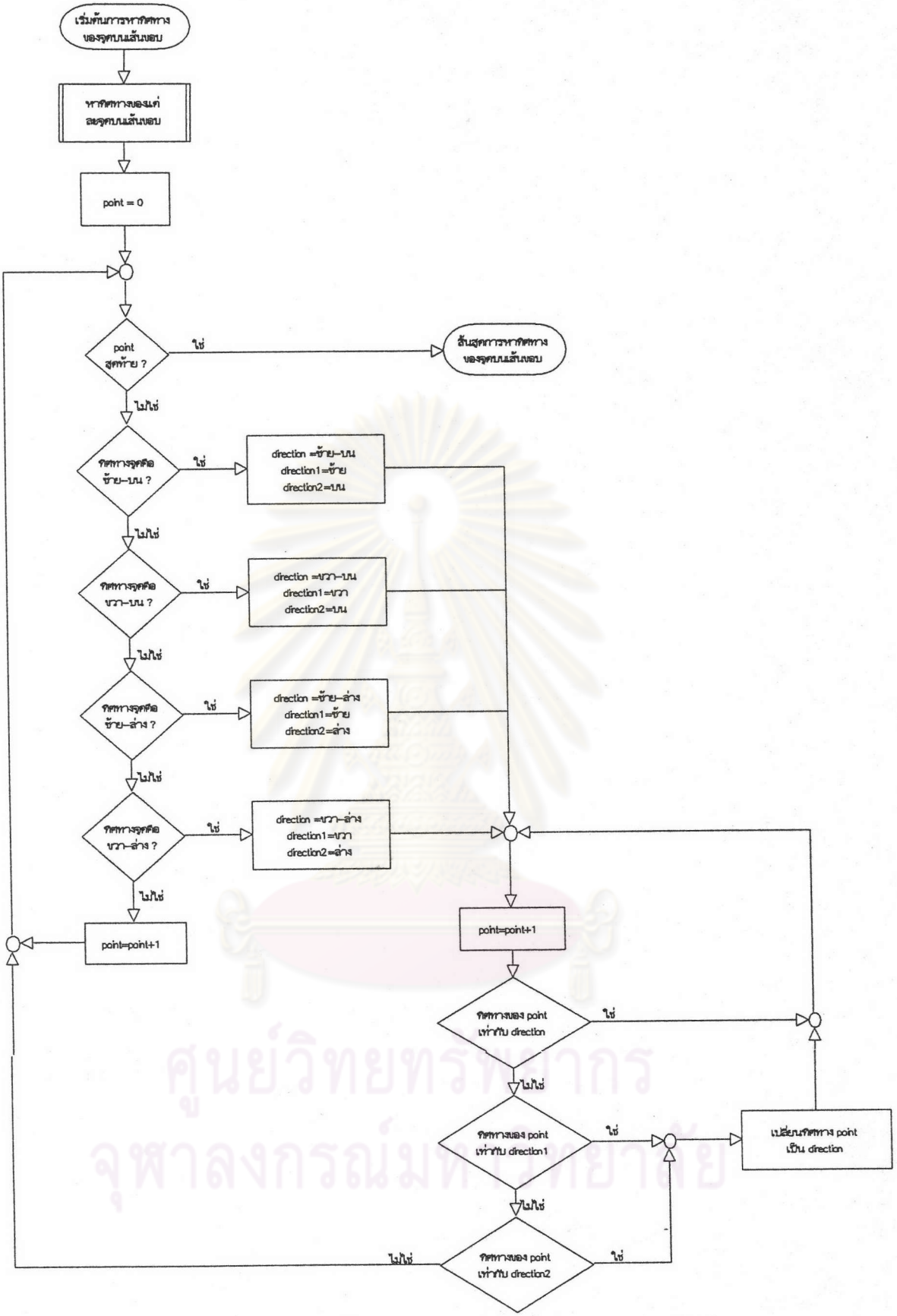
ในขั้นตอนนี้จะเป็นขั้นตอนในการตัดเส้นขอบ (outline) แต่ละเส้น ของฟอนต์แบบจุดภาพเส้นขอบ ที่ได้ตามรอยไว้แล้วออกเป็นส่วนโค้งย่อย (curve segment) โดยเราสามารถทำการตัดเส้นขอบได้ ตามขั้นตอนดังนี้

ก. ในขั้นตอนแรก เราจะทำการหาทิศทางของจุดแต่ละจุดของเส้นขอบก่อน เพื่อนำไปใช้ในการตัดสินใจตัดเส้นขอบต่อไป โดยในการหาทิศทางนั้นเราจะต้องทำการเปลี่ยนทิศทางของจุดย่อยให้เป็นไปตามทิศทางที่แท้จริงโดยรวม ตามเทคนิคในหัวข้อ 2.2.4 ซึ่งแสดงได้ดังผังงานตามรูป 3.10

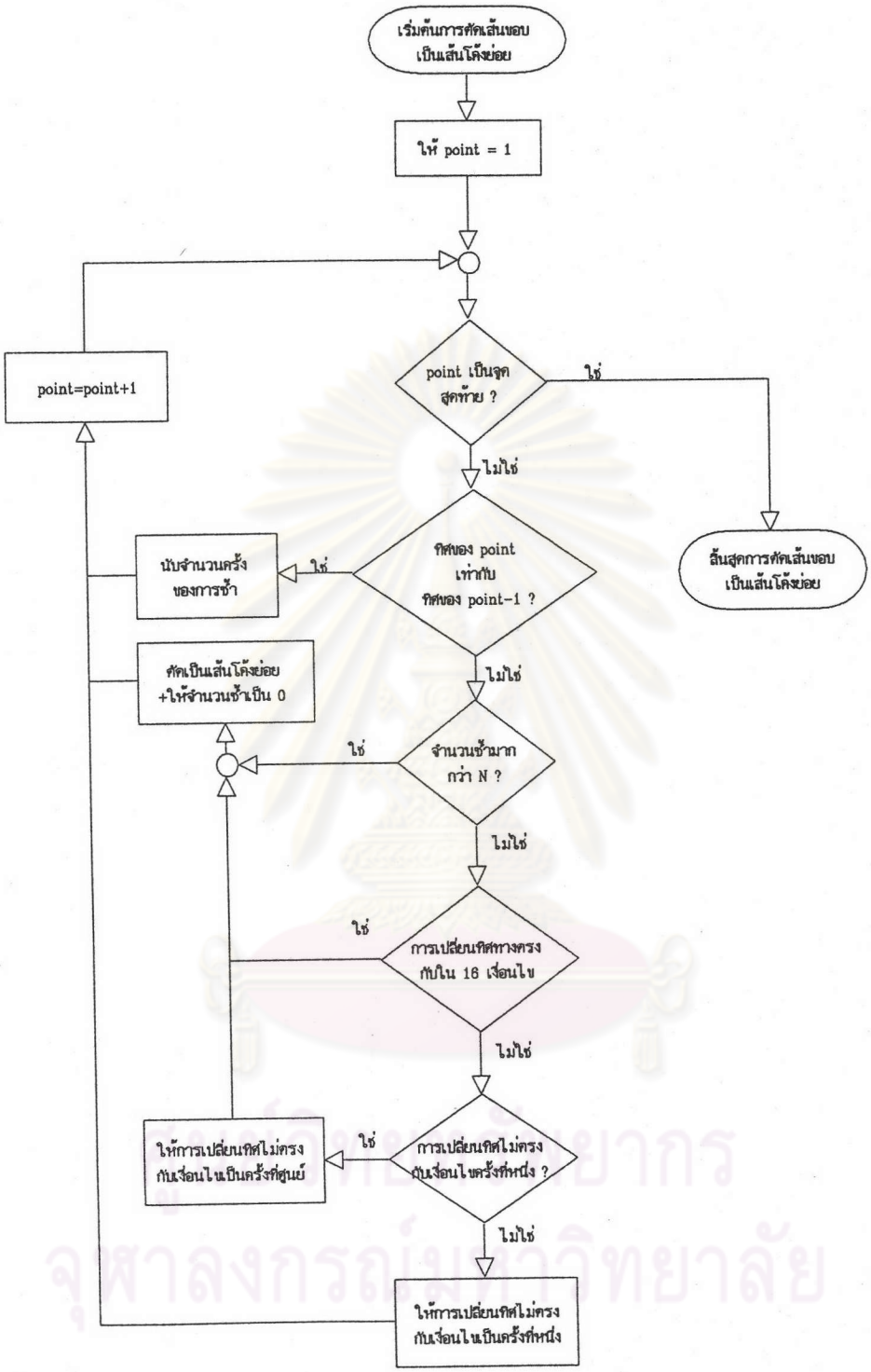
ข. ในขั้นตอนนี้เราจะทำการตัดเส้นขอบออกเป็นเส้นโค้งย่อย ๆ โดยใช้ทิศทางที่ได้จากขั้นตอนที่แล้ว และเทคนิควิธีตามหัวข้อที่ 2.2.5 ซึ่งจะทำให้เราได้เส้นโค้งย่อยและเส้นตรง ที่สามารถนำมาแทนได้ด้วยเส้นโค้งแบบเบซิเยร์ต่อไป โดยเราสามารถแสดงขั้นตอนการทำงานของการตัดเส้นขอบได้ตามรูปผังงานที่ 3.11



รูปที่ 3.9 ผังงานแสดงขั้นตอนการตามรอยเส้นขอบ



รูปที่ 3.10 ผังงานแสดงขั้นตอนการหาทิศทางของแต่ละจุดของเส้นขอบ



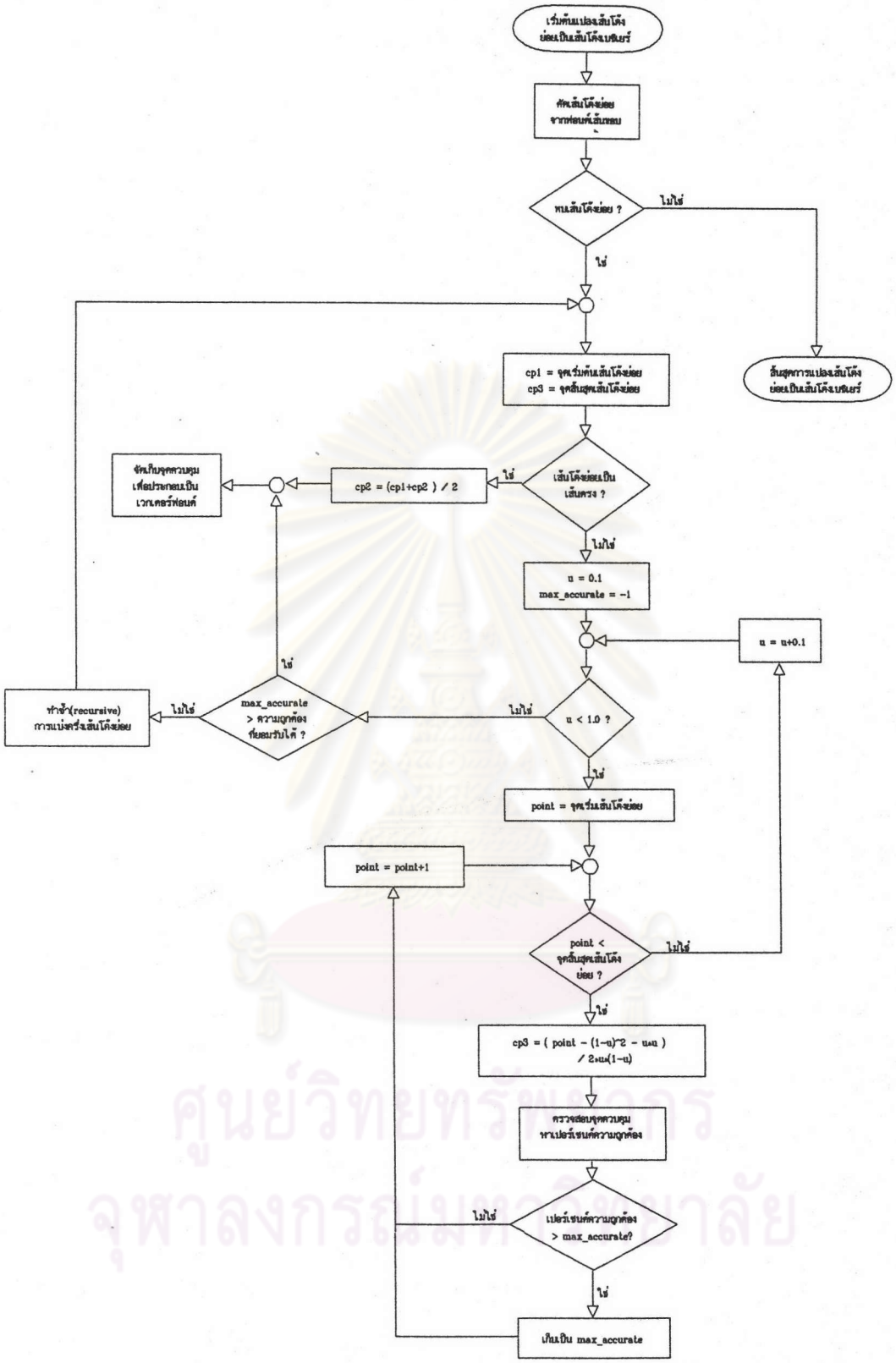
รูปที่ 3.11 ผังงานแสดงขั้นตอนการตัดเส้นขอบออกเป็นเส้นโค้งย่อย

3.4.3 ขั้นตอนการแปลงเส้นโค้งย่อยเป็นเส้นโค้งแบบเบซิเยร์

ขั้นตอนนี้จะเป็นการแปลงเส้นโค้งย่อย ที่เกิดจากการตัดแบ่งฟอนต์แบบจุดภาพเส้นขอบในขั้นตอนที่แล้ว เพื่อให้อยู่ในรูปแบบเวกเตอร์ คือเส้นโค้งเบซิเยร์ดีกรี 2 ซึ่งเทคนิคในการแปลงจะปรากฏตามหัวข้อ 2.2.6 แต่เนื่องจากการแปลงตามเทคนิควิธีนี้ จะเป็นการทดลองประมาณค่าจนกว่าจะได้ค่าที่เหมาะสม ดังนั้น เราจะควบคุมว่าค่าที่ได้มีความเหมาะสมหรือไม่จากตัวแปรระบุเปอร์เซนต์ความถูกต้องในการสร้างจุด นั่นคือ จุดควบคุมที่หาได้สามารถสร้างเส้นโค้งเบซิเยร์ ซึ่งมีจุดตรงกับจุดบนเส้นโค้งย่อยกี่เปอร์เซนต์ โดยที่ หากเรากำหนดให้เปอร์เซนต์ความถูกต้องที่ยอมรับได้ยิ่งสูง จุดควบคุมที่หาได้ก็จะยิ่งมีเส้นโค้งที่คล้ายคลึงกับเส้นโค้งย่อย ตัวอย่างเช่น หากเรากำหนด 100 เปอร์เซนต์ คือต้องสามารถสร้างจุดบนเส้นโค้งย่อยได้ครบทุกจุด

อย่างไรก็ดี มีข้อพึงระวังคือ ยิ่งเรากำหนดให้เปอร์เซนต์ความถูกต้องสูงเท่าใด โอกาสที่จะไม่มีจุดควบคุมที่มีคุณสมบัติตามนั้นก็จะยิ่งสูงขึ้น ซึ่งจะทำให้เกิดการเรียกซ้ำ (recursive) แบ่งเส้นโค้งย่อยนั้นให้ย่อยลงไปอีก เพื่อให้สามารถหาจุดที่ตรงตามคุณสมบัติที่ต้องการได้ อันจะทำให้เกิดเส้นโค้งย่อย ๆ มากขึ้นโดยไม่จำเป็น ซึ่งจะมีผลให้ขนาดและความเร็วในการแสดงผลเวกเตอร์ฟอนต์นั้นมีประสิทธิภาพลดลง และยังทำให้เวลาที่ใช้ในการแปลงเพิ่มมากขึ้นด้วย

เราสามารถแสดงขั้นตอนการแปลงเส้นโค้งย่อยเป็นเส้นโค้งแบบเบซิเยร์ได้ดังรูปผังงานที่ 3.12



รูปที่ 3.12 ผังงานแสดงขั้นตอนการแปลงเส้นโค้งย่อยเป็นเส้นโค้งแบบเบซิเยร์

3.5 โปรแกรมการดัดแปลงรูปร่างเวกเตอร์ฟอนต์ที่ได้ให้สวยงามตามต้องการ

เนื่องจากการแปลงฟอนต์แบบจุดภาพเป็นฟอนต์แบบเวกเตอร์ อาจให้ผลลัพธ์ที่ไม่ตรงกับที่เราต้องการทั้งหมด ไม่ว่าจะเป็นเพราะความคลาดเคลื่อนของตัวโปรแกรม หรือเป็นเพราะฟอนต์แบบจุดภาพต้นแบบมีขนาดที่เล็กเกินไปก็ตาม เราสามารถแก้ไขฟอนต์เวกเตอร์ที่ได้ให้มีความสวยงามตามที่ต้องการ โดยใช้โปรแกรมในส่วนของการดัดแปลงรูปร่างเวกเตอร์ฟอนต์นี้ ซึ่งจะมีคุณสมบัติ คือ

ก. สามารถดัดรูปร่างของเส้นโค้งให้เป็นไปตามที่ต้องการ โดยการเลื่อนจุดควบคุมทั้ง 3 จุดของเส้นโค้งแบบเบซิเยร์ดีกรี 2

ข. สามารถเพิ่มเติมเส้นโค้งเข้าไปในฟอนต์สัญลักษณ์ได้

ค. สามารถลบเส้นโค้งออกจากฟอนต์สัญลักษณ์ได้

ง. สามารถกำหนดและลบจุดเริ่มต้นระบายพื้น (flood fill point)

จ. สามารถเลื่อนฟอนต์สัญลักษณ์ให้ด้านซ้าย-ล่างมาเริ่มต้นที่จุดกำเนิด (0, 0) ได้

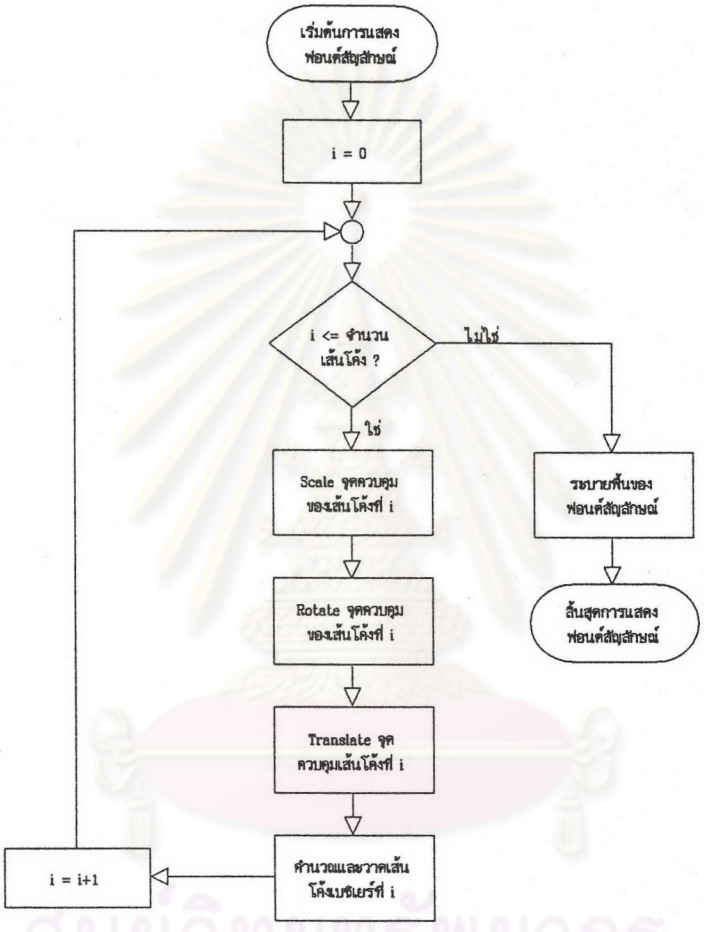
ฉ. สามารถขยายฟอนต์สัญลักษณ์ให้เต็มพิกัดจำลอง (1000x1000) ได้

เมื่อเราได้ดัดแปลงรูปร่างของเวกเตอร์ฟอนต์จนได้ตามที่ต้องการแล้ว เราก็สามารถจัดเก็บฟอนต์ที่แก้ไขแล้วนี้กลับลงไปยังแฟ้มของฟอนต์แบบเวกเตอร์ได้ จากนั้นจึงนำไปประยุกต์ใช้งานตามต้องการต่อไป

3.6 การเลื่อน การหมุน และการขยายเวกเตอร์ฟอนต์

ในการประยุกต์ใช้งานเวกเตอร์ฟอนต์ เราสามารถทำการแปลง (transform)ทางคณิตศาสตร์เพื่อเลื่อน หมุน และขยายเวกเตอร์ฟอนต์ได้ จากคุณสมบัติของเส้นโค้งแบบเบซิเยร์ข้อ 2.1.5.2.ฉ แสดงให้เห็นว่าเราสามารถทำการคำนวณเฉพาะจุดควบคุมของเวกเตอร์ฟอนต์นั้นเท่านั้น และเมื่อเราทำการแสดงผลเวกเตอร์ฟอนต์โดยใช้จุดควบคุมที่คำนวณได้ ภาพที่ได้ก็จะมีการแปลงทางคณิตศาสตร์ตามที่เรากำหนดไว้ทั้งภาพ

การแสดงเวกเตอร์ฟอนต์ในงานวิจัยนี้ เราจะต้องทำการกำหนดขนาด มุมของการหมุน และระยะทางในการเลื่อนของฟอนต์ที่จะแสดงก่อน จากนั้นเมื่อเราสั่งให้แสดงฟอนต์นั้นแล้ว ภาพที่ปรากฏก็จะเป็นไปตามที่เรากำหนดไว้ ซึ่งรูปที่ 3.13 จะแสดงผังงานในการทำงานของโปรแกรมในส่วนของการแสดงผลฟอนต์ ซึ่งจะทำการคำนวณจุดควบคุมก่อนตามข้อกำหนดขนาด มุม และการเลื่อนที่กำหนดไว้



รูปที่ 3.13 ผังงานการคำนวณจุดควบคุมเพื่อแสดงผลเวกเตอร์ฟอนต์