บทที่ 2

แนวทฤษฎีที่สำคัญที่ใช้พัฒนาโปรแกรมย่อยทางกราฟฟิกสำหรับงานวิศวกรรม

2.1 การจำลองแบบของปัญหา

ปัญหาของการสร้างโปรแกรมย่อยทางกราฟฟิก คือต้องแยกโปรแกรมของผู้ใช้ออกจากอุปกรณ์แสดงผลลัพธ์จริง เพราะเมื่อมีการเปลี่ยนแปลงอุปกรณ์แสดงผลลัพธ์ โปรแกรมของผู้ใช้ไม่จำเป็นต้องมีการเปลี่ยนแปลงตาม สาเหตุที่อุปกรณ์แสดงผลลัพธ์มีผลต่อโปรแกรมของผู้ใช้ เพราะโปรแกรมของผู้ใช้จะติดต่อกับอุปกรณ์แสดงผลลัพธ์โดยการส่งข้อมูล อุปกรณ์แสดงผลลัพธ์จะถูกโปรแกรมของผู้ใช้มองในลักษณะของโครงสร้างของข้อมูล เมื่อมีการเปลี่ยนแปลงอุปกรณ์แสดงผลลัพธ์ก็เท่ากับเป็นการเปลี่ยนแปลงโครงสร้างของข้อมูลไปด้วย และเมื่อโครงสร้างของข้อมูลเปลี่ยนไปจึงทำให้โปรแกรมของผู้ใช้เปลี่ยนตามไปด้วย

แนวทางการแก้ปัญหาคือ ทำการสร้างโครงสร้างของข้อมูลที่ทำหน้าที่เหมือนอุปกรณ์แสดงผลลัพธ์ หรืออีกนัยหนึ่งคือทำการสร้างอุปกรณ์แสดงผลลัพธ์จำลองขึ้นมาพร้อมกับสร้างโปรแกรมย่อยขึ้นมา 2 กลุ่ม คือ

- กลุ่มที่หนึ่ง ทำหน้าที่เชื่อมต่อโปรแกรมของผู้ใช้เข้ากับอุปกรณ์แสดงผลลัพธ์จำลอง
- กลุ่มที่สอง ทำหน้าที่เชื่อมต่ออุปกรณ์แสดงผลลัพธ์จำลองกับอุปกรณ์แสดงผลลัพธ์จริง

โครงสร้างข้อมูลที่จำลองแทนอุปกรณ์แสดงผลลัพธ์นี้ จะแบ่งเป็น 2 ส่วน คือ

- ส่วนที่รับข้อมูลเข้าออก ทำหน้าที่เก็บคำสั่งพลอตที่มาจากโปรแกรมผู้ใช้ ซึ่งส่งมายังอุปกรณ์แสดงผลลัพธ์จำลอง เพื่อเตรียมไว้ส่งไปยังอุปกรณ์แสดงผลลัพธ์จริงอีกทีหนึ่ง

- ส่วนที่เก็บข้อมูลที่กำหนดสภาวะของอุปกรณ์แสดงผลลัพธ์จำลอง เช่น
  + กำหนดขนาดของพื้นที่ที่จะทำการพลอตได้
  + กำหนดลักษณะของตัวอักษรที่มีให้พลอตในอุปกรณ์แสดงผลลัพธ์จำลอง
  + กำหนดอัตราการย่อขยายขนาดของภาพ
  + กำหนดจุดกำเนิดสัมพัทธ์
  + ตำแหน่งล่าสุดที่มีการพลอต
  + ฯ ล ฯ

จากที่กล่าวมาแล้วจะมีโปรแกรมย่อยที่เชื่อมต่อโครงสร้างของข้อมูลนี้อยู่ 2 กลุ่ม แต่เพื่อให้ช่วยผู้ใช้ได้มากขึ้นจึงมีโปรแกรมย่อยอีกกลุ่มหนึ่งที่ทำหน้าที่ทางกราฟฟิกเฉพาะอย่าง โดยในตัวโปรแกรมย่อยกลุ่มนี้จะทำการพลอตโดยการเรียกใช้ผ่านโปรแกรมย่อยในกลุ่มที่ 2

หลังจากที่ได้โครงสร้างของข้อมูลของอุปกรณ์แสดงผลลัพธ์จำลอง และหน้าที่ของโปรแกรมย่อยทั้ง 3 กลุ่มแล้ว ในหัวข้อต่อไปจะกล่าวถึงทฤษฎีที่นำมาใช้ในการสร้างโปรแกรมย่อยทั้ง 3 กลุ่ม (โปรแกรมย่อยทั้ง 3 กลุ่มนี้จะกล่าวถึงในบทต่อไป)

2.2 การสร้างเส้นตรง

ต้องทราบจุดเริ่มต้นและจุดปลาย ต้องมีวิธีการที่จะกำหนดจุดไหนบ้างระหว่างจุดทั้งลองที่จะต้องพลอต โดยมี 2 วิธีด้วยกันที่จะกล่าวถึง

- วิธีแรกเรียกว่า DIGITAL DIFFERENTIAL ANALYZER โดยอาศัยลักษณะที่คล้ายกับ NUMERICAL METHOD ที่ใช้แก้สมการ DIFFERENTIAL ของเส้นตรงแต่แทนที่เราจะพิมพ์ค่าตัวเลขออกมา เราจะนำมาใช้กำหนดจุดที่จะพลอตแทน จากจุดที่กำหนดในที่นี้ให้จุดเริ่มต้นคือ (X_1, Y_1) จุดปลายคือ (X_2, Y_2) เราจะได้สมการเส้นตรงในรูปพาราเมตริกซ์ คือ

$$X = X_1 + (X_2 - X_1)U$$
$$Y = Y_1 + (Y_2 - Y_1)U$$

จะเห็นได้ว่า X และ Y จะถูกกำหนดจาก U

เมื่อ $U = 0$; $X = X_1$; $Y = Y_1$
เมื่อ $U = 1$; $X = X_2$; $Y = Y_2$

เพราะฉะนั้นขั้นตอนที่จะเป็นจะอยู่ในรูปสมการพาราเมตริกซ์ ของค่า U แล้วค่อยๆเพิ่มค่า U จาก 0 ถึง 1 จะได้ตำแหน่งของที่จะพลอต ทุกๆครั้งที่เพิ่มค่า ถ้าเพิ่มค่า U ทีละมากๆภาพจะออกมาหยาบ แต่ถ้าเพิ่มทีละน้อยๆภาพจะละเอียดขึ้น

- วิธีที่สองเรียกว่า BRESENHAM'S ALGORITHM

พารามิเตอร์ที่ใช้คือ จุดสองจุด (จุดเริ่มต้น และจุดสิ้นสุด ของเส้นตรง) ในที่นี้แทนด้วย (Xs,Ys,) และ (Xe,Ye) ตามลำดับ การทำงานของโปรแกรมย่อยนี้เริ่มโดยแยกกรณีออกเป็น 9 กรณี คือ

กรณีที่ 1 พารามิเตอร์ที่รับเข้ามาเป็นจุด เพราะฉะนั้นจะทำการพลอตเพียงจุดจุดเดียว

Xs = Xe และ Ys = Ye

กรณีที่ 2 พารามิเตอร์ที่รับเข้ามาเป็นเส้นตรงขนานกับแกนแนวราบ (แกน-X) คือ

Xs <> Xe และ Ys = Ye

กรณีที่ 3 พารามิเตอร์ที่รับเข้ามาเป็นเส้นตรงขนานกับแกนแนวดิ่ง (แกน-Y) คือ

Xs = Xe และ Ys <> Ye

เพราะฉะนั้นจะทำการพลอตขนานกับแกนแนวดิ่ง โดยเริ่มที่ตำแหน่งแนวดิ่ง(ตำแหน่ง Y) ที่น้อยกว่าเพิ่มค่า Y ทีละหนึ่ง แล้วทำการพลอตโดยตำแหน่งแนวราบคงที่ (ตำแหน่ง X)

กรณีที่ 4 พารามิเตอร์ที่รับเข้ามาเป็นเส้นตรงที่มีความชันเท่ากับเท่ากับ 1 คือ

(Ye - Ys) / (Xe - Xs) = 1

เพราะฉะนั้นจะทำการพลอตโดยเริ่มจากตำแหน่งแนวดิ่งและแนวราบที่น้อยกว่า แล้วเพิ่มตำแหน่งแนวราบและแนวดิ่ง (ตำแหน่ง Y) ทีละหนึ่งแล้วทำการพลอตด้วยทุกครั้ง

กรณีที่ 5 พารามิเตอร์ที่รับเข้ามาเป็นเส้นตรงที่มีความชันเท่ากับ -1 คือ

(Ye - Ys) / (Xe - Xs) = -1

เพราะฉะนั้นจะทำการพลอตโดยเริ่มจากตำแหน่งแนวราบทีละหนึ่ง และลดตำแหน่งแนวดิ่งทีละหนึ่ง พร้อมทำการพลอตไปด้วย จนกระทั่งถึงตำแหน่งแนวราบที่มากกว่า และตำแหน่งแนวดิ่งที่น้อยกว่า

กรณีที่ 6 พารามิเตอร์ที่เข้ามาเป็นเส้นตรงที่มีความชันระหว่าง 0 กับ 1 คือ

0 < (Ys-Ye) / (Xs-Xe) <1 และ

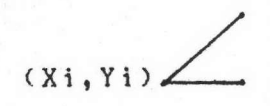
Xs <> Xe และ Ys <> Ye

กรณีที่ 7 พารามิเตอร์ที่เข้ามาเป็นเส้นตรงที่มีความชันระหว่าง 1 กับ inf.
(inf. means infinity)
1 < (Ys-Ye)/(Xs-Xe) < inf. และ
Xs <> Xe และ Ys <> Ye

กรณีที่ 8 พารามิเตอร์ที่เข้ามาเป็นเส้นตรงที่มีความชันระหว่าง -1 กับ 0
-1 < (Ys-Ye)/(Xs-Xe) < 0 และ
Xs <> Xe และ Ys <> Ye

กรณีที่ 9 พารามิเตอร์ที่เข้ามาเป็นเส้นตรงที่มีความชันระหว่าง -1 กับ -inf.
-inf. < (Ys-Ye)/(Xe-Xe) < -1 และ
Xs<> Xe และ Ys <> Ye

ทั้งสี่กรณีหลังนี้จะทำงานคล้ายกันจะกล่าวถึงกรณีที่ 6 เป็นหลักก่อน เพราะฉะนั้นในกรณีนี้จะใช้แนวราบ (แนว-X) เป็นแนวในการเพิ่มตำแหน่งแนวราบทีละหนึ่งหน่วยโดยเริ่มที่ตำแหน่งแนวราบที่น้อยกว่าก่อน (ต้องเปรียบเทียบ ระหว่าง Xs กับ Xe ค่าตัวไหนน้อยกว่าก็เลือกตัวนั้น) ตำแหน่งต่อไปที่จะต้องเลือกมี 2 ตำแหน่งดังรูปข้างล่างที่จะ



ทำให้เส้นตรงที่พลอตใกล้เคียงเส้นตรงที่ถูกกำหนดมามากที่สุด โดยตัดสินจากค่า (Ys-Ye) / (Xs-Xe) ซึ่งคืออัตราการเปลี่ยนแปลงของแนวดิ่งเมื่อแนวราบเปลี่ยนไปหนึ่งหน่วยความยาว เพราะฉะนั้นจะใช้ค่านี้เทียบว่าใกล้ศูนย์หรือใกล้หนึ่งมากกว่ากัน ถ้าใกล้ศูนย์มากกว่าก็จะเลือกตำแหน่ง (Xi +1,Yi) ถ้าใกล้หนึ่งมากกว่าก็จะเลือกตำแหน่ง (Xi +1,Yi +1) ความใกล้นี้วัดได้โดยการนำค่า (Ys-Ye)/(Xs-Xe) มาลบกับ ค่า 1/2 ถ้ามากกว่าศูนย์ก็คือใกล้ ตำแหน่ง (Xi+1/Yi+1) มากกว่า ถ้าน้อยกว่าศูนย์ก็คือใกล้ตำแหน่ง (Xi+1, Yi) มากกว่า และในกรณีใกล้ตำแหน่ง (Xi+1, Yi+1) จุด 2 จุด ต่อไปที่ต้องเลือกคือจุด (Xi+2, Yi+1) หรือจุด (Xi+2, Yi+2) ความใกล้ที่ใช้ตัดสินใจก็จะเป็นดังสมการข้างล่างนี้ คือ

NEAR = (2((Ye-Ys) / (Xe-Xs)) - 1/2 -1)

ค่า - 1 ตัวหลังเกิดการเพิ่มตำแหน่งแนวดิ่งอีก คือ Yi + 1 เพราะฉะนั้น ความใกล้จะอยู่ในรูปสมการดังนี้

1/2 คือค่าที่ตัดสินเลือกในครั้งแรก

ความใกล้ = n ((Ye-Ys) / (Xe-Xs)) - 1/2 -1 (p)

n คือ จำนวนครั้งที่ตำแหน่งแนวดิ่ง เพิ่มขึ้นหนึ่ง (แนว - ข)

แต่ทั้งนี้เพื่อลดเวลาที่ต้องเสียในการคูณ เราสามารถเปลี่ยนการทำงานจากการคูณเป็นการบวกแทน โดยอาศัยหลักที่ว่าการคูณคือการบวกหลายๆครั้ง

การทำงานในกรณีที่ 7 จะเหมือนกับกรณีที่ 6 เมื่อเรามองตำแหน่งแนวราบ (แนว-X) สลับกับตำแหน่งแนวดิ่ง (แนว-Y) เพราะฉะนั้น การทำงานก็จะสลับค่าทั้งลองเวลาพลอตจะใช้ตำแหน่งแนวราบเป็นแนวดิ่ง ตำแหน่งแนวดิ่งเป็นแนวราบ (Plot point (Y,X))

การทำงานในกรณีที่ 8 จะต่างกันกรณีที่ 6 คือแทนที่ตำแหน่งแนวดิ่งจะเพิ่มขึ้นทีละหนึ่งเมื่อมีความใกล้มากกว่าก็จะเป็นลดลงทีละหนึ่งแทน เพราะฉะนั้นการทำงานแทน เพราะฉะนั้น การทำงานแทนจะเปลี่ยนตำแหน่งแนวดิ่งโดยเพิ่มอีก 1 (Yi + 1) ก็จะเป็น (Yi-1) แทน ตำแหน่งแรกในการทำงานเป็นตำแหน่งมากกว่าระหว่าง Xs กับ Xe

การทำงานในกรณีที่ 9 จะต่างกับกรณีที่ 7 คือแทนที่ตำแหน่งแนวราบจะเพิ่มขึ้นทีละหนึ่งเมื่อมีความใกล้มากกว่า ก็จะเป็นลดลงทีละหนึ่งแทน เพราะฉะนั้นการพลอตแทนที่จะเปลี่ยนตำแหน่งแนวราบ โดยเพิ่มอีก 1 (Xi+1) ก็จะเป็น (Xi-1) (กรณีที่ 9 อาจเทียบกับได้กับกรณีที่ 8 โดยตำแหน่งแนวดิ่งกับตำแหน่งแนวราบต้องสลับกัน เวลามีการทำงาน) และตำแหน่งแรกในการทำงานเป็นตำแหน่งมากกว่าระหว่าง Xs กับ Xe

หอสมุดกลาง สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย

2.3 การสร้างเส้นโค้ง

เส้นโค้งสามารถแบ่งออกได้เป็นสองกลุ่มด้วยกันกล่าวคือ กลุ่มแรกเป็นเส้นโค้งที่มีรูปร่างลักษณะแน่นอนสามารถแทนได้ด้วยสมการทางคณิตศาสตร์ เช่น เส้นโค้งของวงกลม เส้นโค้งของวงรี เส้นโค้งพาราโบลา(PARABOLA) ฯลฯ กลุ่มที่สองเป็นเส้นโค้งที่มีลักษณะไม่แน่นอนไม่สามารถแทนได้ด้วยสมการทางคณิตศาสตร์ เพราะฉะนั้นจึงเป็นเหตุให้ต้องมีวิธีสร้างเส้นโค้งอยู่ 2 วิธีที่จะกล่าวถึง วิธีแรกใช้สำหรับการสร้างเส้นโค้งที่มีลักษณะแน่นอน โดยการลากเส้นตรงเล็กๆหลายๆเส้นมาต่อกัน อาศัยหลักการสร้างเส้นตรงวิธีที่หนึ่งที่กล่าวมาแล้ว ลักษณะคล้าย NUMERICAL METHOD ที่ใช้แก้สมการ DIFFERENTIAL ของเส้นโค้งโดยนำค่าที่เป็นค่าตัวเลขมาแทนตำแหน่งที่ใช้พลอต ถ้าแบ่งเป็นเส้นตรงเล็กๆมากภาพก็จะละเอียดมาก ถ้าแบ่งเป็นเส้นตรงเล็กๆน้อยภาพจะหยาบ ต่อไปนี้จะแสดงวิธีการเส้นโค้งของกลุ่มแรกโดยยกตัวอย่างการสร้างเส้นโค้งของวงกลม

กำหนดให้ สร้างเส้นโค้งที่มี (X_o, Y_o) เป็นจุดศูนย์กลางความโค้ง

R คือ รัศมีความโค้ง

(X_1, Y_1) เป็นจุดที่อยู่บนเส้นโค้ง

A คือ มุมที่เส้นรัศมีกวาดไป

สมการของวงกลมคือ $(X-X_o)^2 + (Y-Y_o)^2 = R^2$

จะได้สมการพาราเมตริกซ์คือ

$X = R\cos A + X_o$

$Y = R\sin A + Y_o$

จะเห็นได้ว่าพารามิเตอร์ คือ A และ สมการ DIFFERENTIAL คือ

$dX = -R\sin A\,dA$ จากสมการข้างบนจะได้ $dX = -(Y - Y_o)\,dA$

$dY = R\cos A\,dA$ จากสมการข้างบนจะได้ $dY = (X - X_o)\,dA$

ซึ่ง $dX = X_2 - X_1$ และ $dY = Y_2 - Y_1$

ทำให้ได้สมการที่ใช้กำหนดจุดแต่ละจุดบนเส้นโค้ง คือ

$$X_2 = X_1 - (Y_1 - Y_0)dA$$
$$Y_2 = Y_1 - (X_1 - X_0)dA$$

เมื่อได้ (X_2, Y_2) เป็นจุดที่อยู่บนเส้นโค้งจุดต่อไปก็จะนำไปใช้หาจุด (X_3, Y_3) เป็นจุดที่อยู่บนเส้นโค้งจุดต่อไปอีก ทำเช่นนี้ต่อไปจนกวาดมุมครบ 360 องศาก็จะได้ตำแหน่งที่จะพลอตเป็นวงกลมจนครบ ถ้าเปลี่ยนค่า A (คือมุมที่กวาดไปทีละครั้ง) ทีละค่ามากๆภาพที่ได้จะหยาบ ถ้าเปลี่ยนค่าทีละค่าน้อยๆภาพที่ได้จะละเอียด แต่ถ้าน้อยจนเกินไปอาจทำให้เสียเวลาโดยใช่เหตุ เนื่องจากมีการคำนวณตำแหน่งที่จะพลอตตกซ้ำตำแหน่งเดิมเพราะจำนวนจุดในจอภาพที่ใช้พลอตมีปริมาณจำกัด ส่วนในการสร้างเส้นโค้งของวงรีก็จะทำในลักษณะเดียวกัน โดยสมการของวงรีจะมีลักษณะดังนี้

สมการของวงรีคือ $$\frac{(X-X_0)^2}{M^2} + \frac{(Y-Y_0)^2}{N^2} = R^2$$

จะได้สมการพาราเมตริกซ์คือ

$$X = MR\cos A + X_0$$
$$Y = NR\sin A + Y_0$$

แล้วก็ใช้วิธีหาตำแหน่งที่จะทำการพลอตเช่นเดียวกับ การสร้างเส้นโค้งของวงกลม

ส่วนวิธีที่สองใช้สำหรับการสร้างเส้นโค้งที่มีลักษณะไม่แน่นอน โดยจะต้องมีตำแหน่งของจุดบนเส้นโค้งบางส่วนให้มา เพื่อมาใช้หาตำแหน่งของจุดบนเส้นโค้งที่ขาดไป วิธีนี้เรียกว่า อินเตอร์โปเลชัน (INTERPOLATION) โดยสร้างฟังก์ชันโพลิโนเมียล (POLYNOMIAL) ที่อยู่ในรูปแบบของพาราเมตริกซ์ ดังนี้

X = fx(u)

Y = fy(u)

สมมุติตำแหน่งของจุดบนเส้นโค้งบางส่วนให้มา คือ

(X_1, Y_1) , (X_2, Y_2) , _ _ _ , (Xn, Yn)

จากนี้จะทำการสร้างฟังก์ชันโพลิโนเมียลที่ใช้กำหนดตำแหน่งที่ขาดไป ซึ่งเกิดจากผลรวมของแต่ละเทอมที่มีตำแหน่งที่ให้มาเป็นพารามิเตอร์ คูณกับฟังก์ชันเบลนดิง(BLENDING FUNCTION) ดังสมการต่อไปนี้

$$fx(u) = \sum_{i=1}^{n} x_i B_i(u)$$

$$fy(u) = \sum_{i=1}^{n} y_i B_i(u)$$

โดยค่าของ u จะเป็นตัวกำหนดว่าแต่ละจุดที่ให้มาจะมีผลต่อการหาตำแหน่งของจุดที่ขาดไปมากน้อยเพียงใด เช่น ถ้าค่า u มีค่าใกล้ 1 มาก ตำแหน่งที่ขาดไปก็จะมีค่าใกล้ตำแหน่งของจุดที่ 1 ที่ให้มามากกว่า และ ใกล้ตำแหน่งของจุดที่ 2 รองลงไป และจุดอื่นๆ รองลงไปเรื่อยๆ และ ถ้าค่า u มีค่าเป็นจำนวนเต็มตำแหน่งที่ ก็คือตำแหน่งของจุดที่ 1 ที่ให้มา กล่าวอีกนัยหนึ่ง ถ้าจุดที่ขาดที่ต้องการจะหามีค่า u ใกล้จุดที่ให้มาจุดใดมาก ตำแหน่งของจุดที่ขาดไปที่ต้องการจะหาก็จะมีค่าใกล้เคียงกับตำแหน่งของจุดที่ให้มาจุดนั้นมากที่สุด

การลร้างฟังก์ชันเบลนดิง เพื่อความง่ายแทนที่จะให้ค่า u เท่ากับ 1 เป็นพารามิเตอร์ที่กำหนดว่า ตำแหน่งที่หาได้คือตำแหน่งที่ให้มาของจุดที่ 1 จะใช้ค่า u เท่ากับ -1 แทนลำหรับจุดที่ให้มาจุดที่ 1 ค่า u เท่ากับ 0 สำหรับจุดที่ให้มาจุดที่ 2 ค่า u เท่ากับ 1 สำหรับจุดที่ให้มาจุดที่ 3 และ ในทำนองเดียวกันลำหรับจุดต่อๆไป ฟังก์ชันเบลนดิงที่ได้จะมีลักษณะดังนี้

ลำหรับจุดที่ให้มา จุดที่ 1 ลำหรับค่า u เท่ากับ -1

$$B_1(u) = \frac{(u)(u-1)...(u-(n-2))}{(-1)(-2)(-3).....(1-n)}$$

ลำหรับจุดที่ให้มา จุดที่ i ลำหรับค่า u เท่ากับ i-2

$$Bi(u) = \frac{(u+1)(u)(u-1)...(u-(i-2))}{(i-1)(i-2)(i-3).....(i-n)}$$

สมมุติตำแหน่งของจุดบนเส้นโค้งบางส่วนให้มามีจำนวนจำกัดเพียงแค่ 4 จุด คือ

(X_1,Y_1) , (X_2,Y_2) , (X_3,Y_3) , (X_4,Y_4)

ฟังก์ชันเบลนดิงของทั้ง 4 จุด คือ

$$B_1(u) = \frac{u\,(u-1)(u-2)}{(-1)(-2)(-3)}$$

$$B_2(u) = \frac{(u+1)(u-1)(u-2)}{(1)(-1)(-2)}$$

$$B_3(u) = \frac{(u+1)u(u-2)}{(2)(1)(-1)}$$

$$B_4(u) = \frac{(u+1)u(u-1)}{(3)(2)(1)}$$

จากนี้สามารถสร้างสมการของเส้นโค้งระหว่างจุดที่ให้มา 4 จุดดังนี้

$$X = X_1B_1(u) + X_2B_2(u) + X_3B_3(u) + X_4B_4(u)$$

$$Y = Y_1B_1(u) + Y_2B_2(u) + Y_3B_3(u) + Y_4B_4(u)$$

ถ้าเปลี่ยนค่า u ทีละค่ามากๆเส้นโค้งที่ได้จะหยาบ ถ้าเปลี่ยนค่าทีละค่าน้อยๆเส้นโค้งที่ได้จะละเอียด แต่ถ้าน้อยจนเกินไปอาจทำให้เสียเวลาโดยใช่เหตุ เนื่องจากมีการคำนวณตำแหน่งที่จะพลอตตกซ้ำตำแหน่งเดิม เพราะจำนวนจุดในจอภาพที่ใช้พลอตมีปริมาณจำกัด สรุปแล้วในวิธีที่สองนี้ต้องทำการสร้างสมการทางคณิตศาสตร์ของเส้นโค้งที่มีรูปร่างไม่แน่นอนให้ได้ก่อนจึงทำการหาตำแหน่งของจุดของเส้นโค้งที่ขาดไปได้

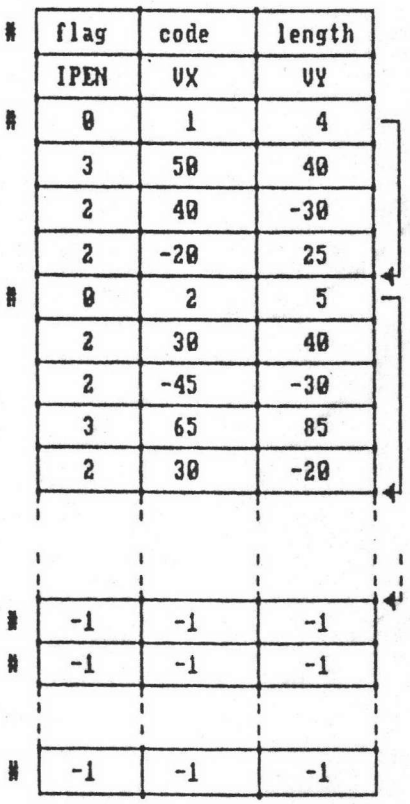
013814

2.4 การสร้างตัวอักษร

มีวิธีพื้นฐานอยู่สองวิธี คือ

- วิธีแรก แต่ละตัวอักษรจะมีข้อมูลประจำตัวที่กำหนดว่าจะต้องมีจุดไหนบ้างที่จะทำการพลอตเพื่อเป็นตัวอักษรตัวนี้(DOT-METRIX METHOD) โดยขนาดของข้อมูลของตัวอักษรแต่ละตัวจะอยู่ในลักษณะของตาราง เช่นตารางมีขนาด 5x7 แสดงว่าจำนวนจุดที่ต้องพลอตมี 35 จุดด้วยกันทุกตัวอักษรจะมีการกำหนดว่าใน 35 จุดนี้ จุดที่ต้องทำการพลอตจะมีค่าหนึ่ง และจุดที่ไม่ต้องทำการพลอตจะมีค่าศูนย์ โดยการกำหนดกับทั้ง 35 จุดของตัวอักษร ฯลฯ เวลาต้องการพลอตตัวอักษรตัวใดจะนำข้อมูลของตัวอักษรทั้ง 35 จุดที่มีอยู่ไปพลอตได้เลย การย่อขยายขนาด หรือเปลี่ยนตำแหน่ง หรือหมุนตัวอักษร ทำโดยการใช้วิธีทางคณิตศาสตร์ทำกับตำแหน่งแนวราบและตำแหน่งแนวดิ่งของทุกๆจุดของตัวอักษรนั้น ดังจะกล่าวในหัวข้อต่อไป

- วิธีที่สอง วิธีนี้แต่ละตัวอักษรจะประกอบด้วยเส้นตรงหลายๆเส้นต่อกัน (เรียกว่า STOKE METHOD) แต่ละตัวอักษรจะมีข้อมูลของตัวเองว่ามีเส้นตรงประกอบทั้งหมดกี่เส้น ซึ่งอยู่ในลักษณะของเวกเตอร์ของการพลอต เมื่อต้องการจะพลอตตัวอักษรตัวใดก็จะเรียกเวกเตอร์ของตัวอักษรตัวนั้นมาพลอต วิธีนี้ตัวอักษรสามารถหมุนหรือเคลื่อนย้ายหรือย่อขยายได้โดยง่าย เพราะจะทำในลักษณะเดียวกับเส้นตรง ลักษณะโครงสร้างข้อมูลที่เก็บเวกเตอร์ของตัวอักษรสามารถดูได้จากรูปที่ 2-1 ตารางที่เก็บเวกเตอร์ของตัวอักษรจะประกอบด้วยข้อมูลของแต่ละตัวอักษร ข้อมูลของแต่ละตัวอักษรจะแบ่งเป็นสองส่วน คือ ส่วนแรกจะมีแฟลก(flag) เป็นตัวบอกว่าความหมายของข้อมูลในช่วงต่อไปนี้ (ถ้ามีค่าเท่ากับ 0 หมายความว่าข้อมูลในช่วงต่อไปนี้จะมีข้อมูลของตัวอักษรอยู่ ถ้ามีค่าเท่ากับ -1 หมายความว่าข้อมูลในช่วงต่อไปนี้ยังเป็นที่ว่างอยู่ ถ้ามีค่าเท่ากับ 9999 หมายความว่าข้อมูลในช่วงต่อไปนี้เลิกใช้แล้ว) มีรหัส(code)ของตัวอักษรเก็บไว้ใช้ในการเรียก



#	flag	code	length
	IPEN	VX	VY
#	0	1	4
	3	50	40
	2	40	-30
	2	-20	25
#	0	2	5
	2	30	40
	2	-45	-30
	3	65	85
	2	30	-20
#	-1	-1	-1
#	-1	-1	-1
#	-1	-1	-1

flag = 0 :Character exist
flag = -1 :Empty
flag = 9999 :Mark delete

code :Character code
length :length of Character vectors

IPEN :Pen instruction
VX :Vector-X
VY :Vector-Y

รูปที่ 2-1 แสดงโครงสร้างข้อมูลของตัวแปรร่วม SYM

ออกมาพลอต มีความยาวข้อมูล(length)บอกจำนวนเวกเตอร์ของตัวอักษรจึงมีลักษณะแบบความยาวแปรผัน(variable length) ทำให้ใช้เนื้อที่ได้อย่างไม่สิ้นเปลือง ส่วนที่สอง จะเก็บเวกเตอร์ของตัวอักษรซึ่งประกอบด้วยคำสั่งลากเส้นหรือย้ายตำแหน่ง ตำแหน่งแนวราบ และตำแหน่งแนวดิ่ง

เมื่อการนำตัวอักษรมาพลอตจะต้องนำตำแหน่งแนวราบ และตำแหน่งแนวดิ่ง มาเปลี่ยนค่าตามอัตราส่วนที่กำหนดเพื่อให้อยู่ในขนาดที่ต้องการ และในกรณีที่มีการหมุนตัวอักษรจะกระทำกับตำแหน่งแนวราบและตำแหน่งแนวดิ่งของเวกเตอร์ทั้งหมดของตัวอักษร แล้วจึงนำไปพลอตเป็นตัวอักษรขึ้นมา (โปรดดูในหัวข้อที่ 4.3.8 ประกอบด้วย)

2.5 การเปลี่ยนขนาดของรูปภาพ

ภาพภาพหนึ่งจะประกอบด้วยเส้นหลายๆเส้นต่อเข้าด้วยกัน แต่ละเส้นจะถูกแทนด้วยเวกเตอร์ของการพลอตที่ประกอบด้วย คำสั่งลากเส้นหรือย้ายตำแหน่ง และ ตำแหน่งแนวราบตำแหน่งแนวดิ่ง ถ้าสามารถเปลี่ยนค่าที่ใช้แทนตำแหน่งทั้งลองนี้ เช่นให้มีค่าเพิ่มเป็นสองเท่าภาพที่ได้ก็จะเพิ่มเป็นสองเท่า ฯลฯ ในที่นี้จะอธิบายการเปลี่ยนขนาดของเวกเตอร์อันหนึ่งซึ่งเป็นส่วนประกอบของภาพ เมื่อต้องการขยายขนาดภาพก็จะทำกับทุกๆเวกเตอร์ของภาพในลักษณะเดียวกัน ลมมุติให้ P_1 แทนค่าของตำแหน่งแนวราบและตำแหน่งแนวดิ่งเดิมก่อนมีการเปลี่ยนแปลงขนาด และให้ P_2 แทนค่าของตำแหน่งแนวราบและตำแหน่งแนวดิ่งหลังมีการเปลี่ยนแปลงขนาด

กำหนดให้ P_1 เป็นมี มีตำแหน่งแนวราบตำแหน่งแนวดิ่งคือ (X_1, Y_1) หรืออยู่ในรูป MATRIX 1 x 2 คือ $[\ X_1\ Y_1\ ]$ เมื่อถูกคูณด้วย MATRIX 2 x 2 จะได้ผลลัพธ์เป็น MATRIX 1 x 2 ซึ่งยังใช้ได้ในความหมายเช่นเดิม

$$P_2 = [\ X_2\ Y_2\ ] = P_1\ T = [\ X_1\ Y_1\ ]T$$

MATRIX T (transformation matrix) จะเป็นตัวMAPPING ระหว่างจุด P_1 และ P_2 คือ $[\ X_2\ Y_2\ ]$

ถ้า MATRIX T $= \begin{matrix} 2 & 0 \\ 0 & 3 \end{matrix}$

$$[\ X_2\ Y_2\ ] = [\ X_1\ Y_1\ ] \begin{matrix} 2 & 0 \\ 0 & 3 \end{matrix} = [\ 2X_1\ 3Y_1\ ]$$

จะได้ว่า X_2 เป็น 2 เท่าของ X_1 และ

Y_2 เป็น 3 เท่าของ Y_1

เพราะฉะนั้นเมื่อต้องการเปลี่ยนขนาดของภาพก็จะทำโดยการนำเมตริกซ์ T ที่ใช้กำหนดขนาดของภาพมาคูณกับทุกจุดของภาพ ก็จะได้ภาพที่เปลี่ยนขนาดไป ในกรณีที่ขยายภาพมากๆภาพจะมีความหยาบมาก ก็จะแก้ไขโดยมีการลากเส้นระหว่างจุดที่ห่างเกินไป

หอสมุดกลาง สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย

2.6 การหมุนรูปภาพ

ภาพภาพหนึ่งจะประกอบด้วยเส้นหลายๆเส้นต่อเข้าด้วยกัน แต่ละเส้นจะถูกแทนด้วยเวกเตอร์ของการพลอตที่ประกอบด้วย คำสั่งลากเส้นหรือย้ายตำแหน่งและ ตำแหน่งแนวราบตำแหน่งแนวดิ่ง ถ้าลามารถเปลี่ยนค่าที่ใช้แทนตำแหน่งทั้งสองนี้ เช่นให้มีค่าหมุนไป 45 องศาภาพที่ได้ก็จะไป 45 องศา ฯลฯ ในที่นี้จะอธิบายการหมุนของเวกเตอร์อันหนึ่งซึ่งเป็นส่วนประกอบของภาพ เมื่อต้องการหมุนภาพก็จะทำกับทุกๆเวกเตอร์ของภาพในลักษณะเดียวกัน ลมมุติให้ P_1 แทนค่าของตำแหน่งแนวราบและตำแหน่งแนวดิ่งเดิมก่อนมีการหมุน และให้ P_2 แทนค่าของตำแหน่งแนวราบและตำแหน่งแนวดิ่งหลังมีการหมุนเวกเตอร์

รูปภาพจะถูกหมุนไปรอบจุดใดจุดหนึ่ง ในที่นี้จะทำการหมุนเฉพาะรอบจุดกำเนิดคือ [0 0] ซึ่งการหมุนจะทำได้โดยง่าย ส่วนการหมุนรอบจุดใดๆจะอธิบายภายหลัง

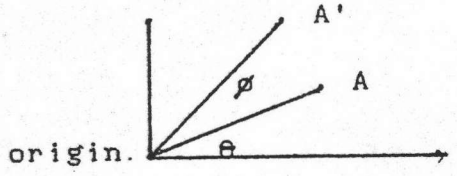
กำหนดให้จุด A คือ [X_1 Y_1] เป็นค่าของตำแหน่งแนวราบ และตำแหน่งแนวดิ่งของเวกเตอร์ เวกเตอร์จะหมุนรอบจุดกำเนิด [0 0] จากตำแหน่งเดิม A ไปยังตำแหน่งใหม่ A' คือ [X_2 Y_2]

L คือ ระยะจากจุดกำเนิดไปยังจุด A

$$L = (X_1^{\ 2} + Y_1^{\ 2})^{1/2}$$

$$\sin \quad = Y_1 / L$$

$$\cos \quad = X_1 / L$$



เมื่อรัศมี L กวาดไปเป็นมุม ø

$$\sin (\theta + \phi) = Y_2 / L = \cos \phi \sin \theta + \sin \phi \cos \theta$$

$$Y_2 = \cos \phi \sin \theta \, L + \sin \phi \cos \theta \, L$$

$$= \cos \phi \, Y_1 + \sin \phi \, X_1$$

ในทำนองเดียวกัน

$X_2 = \cos \phi X_1 - \sin \phi Y_1$

จะได้ MATRIX สำหรับการหมุนคือ $\sin \phi \cos \phi$

$-\sin \phi \cos \phi$

เพราะฉะนั้นเมื่อต้องการหมุนรูปภาพ ก็ทำโดยการคูณ MATRIX สำหรับการหมุนกับทุกเวกเตอร์ของภาพ ส่วนการหมุนรอบจุดใดๆ ก็ทำโดยการเคลื่อนย้ายภาพจากจุดนั้นมาที่จุดกำเนิดก่อน แล้วจึงทำการหมุน หลังจากหมุนแล้วก็เคลื่อนย้ายกลับที่เดิม ซึ่งรู้สึกยุ่งยากแต่เราทำโดยการคูณ MATRIX สำหรับการเคลื่อนย้าย กับ MATRIX สำหรับการหมุนเท่านั้นเอง

2.7 การเคลื่อนย้ายรูปภาพ

ภาพภาพหนึ่งจะประกอบด้วยเส้นหลายๆเส้นต่อเข้าด้วยกัน แต่ละเส้นจะถูกแทนด้วยเวกเตอร์ของการพลอตที่ประกอบด้วย คำสั่งลากเส้นหรือย้ายตำแหน่ง และ ตำแหน่งแนวราบตำแหน่งแนวดิ่ง ถ้าลามารถเปลี่ยนค่าที่ใช้แทนตำแหน่งทั้งลองนี้ เช่นให้มีตำแหน่งแนวราบเพิ่มอีกสองหน่วยความยาวภาพที่ได้ก็จะเคลื่อนย้ายไปตามแนวราบอีกลองหน่วยความยาว ฯลฯ ในที่นี้จะอธิบายการเคลื่อนย้ายของเวกเตอร์อันหนึ่งซึ่งเป็นส่วนประกอบของภาพ เมื่อต้องการเคลื่อนย้ายภาพก็จะทำกับทุกๆเวกเตอร์ของภาพในลักษณะเดียวกัน เช่นเดิมตำแหน่งของเวกเตอร์ทั้งหมดของภาพคือ $[\ X_1\ Y_1\ ]$, $[\ X_2\ Y_2\ ]$, _ _ _

ตำแหน่งของเวกเตอร์ทั้งหมดของภาพที่เคลื่อนย้ายคือ $[\ X_1 + k\ \ Y_1 + 1\ ]$, $[\ X_2 + k\ \ Y_2 + 1\ ]$,_ _ _

นั่นคือ MATRIX $[\ X'\ Y'\ ] = [\ X\ \ Y\ ] + [\ k\ \ 1\ ]$

แต่เทคนิคในการหมุนภาพหรือเปลี่ยนขนาดภาพใช้การคูณเมตริกซ์ (MATRIX) ทั้งนี้เพื่อให้วิธีการเป็นมาตรฐานเหมือนกัน จึงจำเป็นต้องลร้างเมตริกซ์ 1 x 3 สำหรับแทนตำแหน่งแนวราบตำแหน่งแนวดิ่ง และ เมตริกซ์ 3 x 3 สำหรับแทนเมตริกซ์ที่ใช้เปลี่ยนแปลงภาพ เพราะฉะนั้นจะได้ว่า

$$[\ X'\ \ Y'\ \ 1\ ] = [\ X\ \ Y\ \ 1\ ] \begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ k & 1 & 1 \end{matrix}$$

ซึ่งได้ผลเช่นเดียวกับสมการข้างบน

ส่วน เมตริกซ์ สำหรับการเปลี่ยนขนาดของภาพ คือ

$$\begin{matrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{matrix}$$

ส่วน เมตริกซ์ สำหรับการหมุนรูปภาพ คือ

$$\begin{bmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$