บทที่ 4
โครงสร้างแฟ้มข้อมูลกราฟิกแบบ PCX และ TIFF

ในปัจจุบันมีโครงสร้างข้อมูลกราฟิกที่เป็นภาพบิตแมพหลายรูปแบบ แต่ที่จัดได้ว่าเป็นมาตรฐานในปัจจุบันคือ PCX และ TIFF

โครงสร้างแฟ้มข้อมูลกราฟิกแบบ PCX

โครงสร้างแฟ้มข้อมูลกราฟิกแบบ PCX เป็นโครงสร้างแฟ้มข้อมูลกราฟิกที่พัฒนาโดยบริษัท Zsoft ซึ่งเป็นเจ้าของโปรแกรมวาดภาพ PC PAINTBRUSH โครงสร้างแฟ้มข้อมูลกราฟิกแบบ PCX จะขึ้นอยู่กับฮาร์ดแวร์ที่ใช้ในการสร้างรูปภาพ โดยจะแบ่งออก เป็น 3 ประเภทคือ

1) PCX สำหรับจอภาพสีเดียว (Monochrome PCX)

2) PCX สำหรับจอภาพ 16 สี (16-color PCX)

3) PCX สำหรับจอภาพ 256 สี (256-color PCX)

สำหรับในการวิจัยนี้จะใช้กับแฟ้มข้อมูล PCX สำหรับจอภาพสีเดียว และ PCX สำหรับจอภาพ 16 สี เท่านั้น

แฟ้มข้อมูล PCX ทุกประเภทจะมีการเข้ารหัสแบบ Run length Encoding (RLE) และมีส่วนหัวของแฟ้มข้อมูล (File Header) ยาว 128 ไบต์ โครงสร้างแฟ้มข้อมูลกราฟิกแบบ PCX จะแสดงได้ดังตารางที่ 4.1

ไบต์ที่	รายละเอียด	หมายเหตุ และค่าที่เป็นไปได้
0	บอกว่าเป็น PCX หรือไม่	ถ้าเป็นจะมีค่าเป็น 10
1	บอกเวอร์ชันของ PC Paint Brush ที่ใช้สร้างภาพ	0 = เวอร์ชัน 2.5 1 = เวอร์ชัน 2.8 3 = เวอร์ชัน 2.8 ที่ไม่มีรายละเอียดของสี (color pallete) 5 = เวอร์ชัน 3.0 ขึ้นไป
2	วิธีการเข้ารหัส	เป็น 1 เสมอ

ตารางที่ 4.1 โครงสร้างแฟ้มข้อมูลแบบ PCX

ไบต์ที่ 0	รายละเอียด	หมายเหตุ และค่าที่เป็นไปได้
3	บิต / เพลน / พิกเซล	ใช้ร่วมกับไบต์ที่ 65
4-5	XMIN	ดูหมายเหตุข้อ 2 โคออร์ดิเนตสำหรับกำหนดความกว้างของรูป
6-7	YMIN	
8-9	XMAX	
10-11	YMAX	
12-13	ความละเอียดตามแนวนอน	กำหนดความละเอียดของอุปกรณ์ที่ใช้สร้างรูป
14-15	ความละเอียดตามแนวตั้ง	
16-63	รายละเอียดของสี	สำหรับ 16 สี หรือน้อยกว่า
64	สำรอง (reserved)	มีค่าเป็น 0 เสมอ
65	จำนวนบิตเพลน	ใช้ร่วมกับไบต์ที่ 3 ดูหมายเหตุข้อ 2
66-67	ไบต์ / สแกนไลน์ / เพลน	แสดงจำนวนไบต์ทที่ใช้เก็บข้อมูลในแต่ละบรรทัด
68-69	ข้อมูลเกี่ยวกับสี (Pallete Information)	1 = สี หรือ ขาว-ดำ 2 = เกรย์สเกล (Gray Scale)
70-127		ไม่มีหน้าที่อะไร มีค่าเป็น 0
128	จุดเริ่มต้นของข้อมูล	
	ข้อมูลสำหรับสี 256 สี จะหาได้จากการนับย้อนหลังจาก ตำแหน่งสุดท้ายของแฟ้ม-ข้อมูลมา 769 ไบต์	ดูหมายเหตุข้อ 3

ตารางที่ 4.1 (ต่อ)

หมายเหตุ

1) ข้อมูลตั้งแต่ ไบต์ที่ 0 - 127 เป็นส่วนหัวของแฟ้มข้อมูล PCX

2) ไบต์ที่ 65 และไบต์ที่ 3 จะใช้ร่วมกัน โดยจะตรวจสอบที่ไบต์ที่ 65 ก่อน แล้วจึงตรวจสอบไบต์ที่ 3 ผลที่ได้แสดงใน ตารางที่ 4.2

3) ในการใช้งาน PCX 256 สี จะมีการใส่ข้อมูลของสี (Pallette Information) หลังจากจบส่วนของข้อมูลแล้ว การตรวจสอบว่าเป็นแฟ้มข้อมูล PCX 256 สี หรือไม่ทำได้โดยตรวจสอบ เลขเวอร์ชัน (ไบต์ที่ 1) ว่าเป็น 3 หรือไม่ จากนั้นจึงตรวจสอบว่าไบต์ที่ 3 เป็น 8 และ ไบต์ที่ 65 ว่าเป็น 1 หรือไม่ ถ้าเป็นแสดงว่าแฟ้มข้อมูลนี้อาจจะเป็นแฟ้มข้อมูล PCX 256 สี การแปลงค่าของสีจากค่าตัวเลข 0 - 255 ไปอยู่ในรูปแบบ RGB จะใช้ขนาดข้อมูล 3 ไบต์ สำหรับแต่ละสี (R 1 ไบต์ G 1 ไบต์ B 1 ไบต์) ดังนั้นจะใช้เนื้อที่เก็บข้อมูล 768 ไบต์ (3 x 256) สำหรับข้อมูลของ 256 สี เพื่อที่จะให้แน่ใจจริงๆ ว่าแฟ้มข้อมูลนี้เป็นแฟ้มข้อมูล PCX 256 สี ก็ให้อ่านข้อมูลย้อนหลัง จากจุด สิ้นสุดแฟ้มข้อมูลไป 769 ไบต์ ถ้าเจอ C0 ก็แสดงว่า เป็นแฟ้มข้อมูล PCX 256 สี

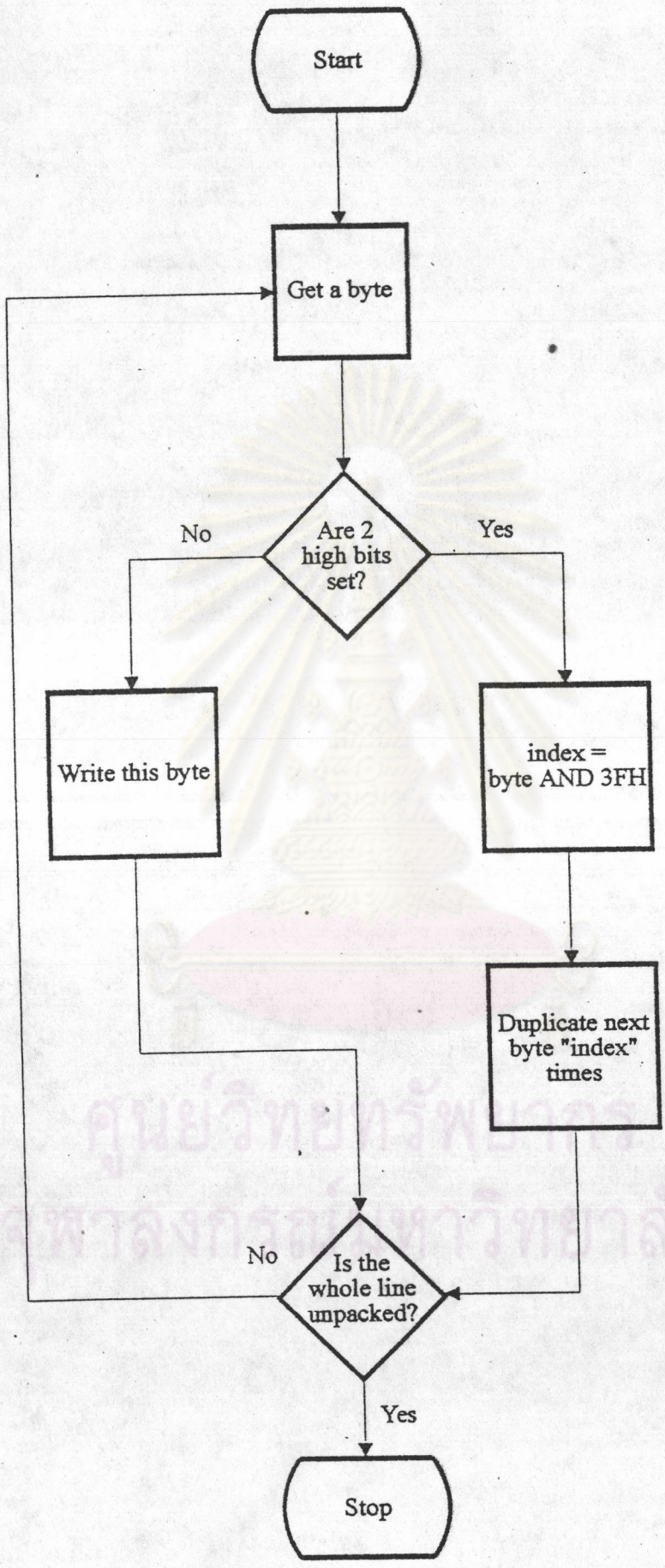
ไบต์ที่ 65	ไบต์ที่ 3	โหมดแสดงผล
1	1	Monochrome(CGA-mono, GA-mono, hercules)
1	2	4-color CGA
4	1	16-color or Gray EGA, VGA
1	8	256-color or Gray Extended

ตารางที่ 4.2 ตรวจสอบชนิดของจอภาพที่ใช้สร้างแฟ้มข้อมูลแบบ PCX

สำหรับส่วนที่สำคัญที่สุดในการเขียนโปรแกรมเพื่ออ่านรูปภาพ PCX คือ การถอดรหัสของการอัดข้อมูล (Data Compression) แบบ RLE ซึ่งเป็นวิธีการที่ใช้ในการอัดข้อมูลรูปภาพ PCX

ในการถอดรหัสของการอัดข้อมูลแบบ RLE เพื่ออ่านรูปภาพ PCX มีสิ่งที่ควรเข้าใจดังนี้ ข้อมูลที่อยู่ในแฟ้มข้อมูลแบบ PCX จะอยู่ในรูปแบบของคีย์ไบต์ (Key Byte) และดาต้าไบต์ (Data Byte) โดยถ้า 2 บิตแรกของคีย์ไบต์มีค่าเป็น 1 อีก 6 บิตที่เหลือของคีย์ไบต์จะใช้ ในการคำนวณว่าจะต้องอ่านดาต้าไบต์เป็นจำนวนกี่ครั้ง แต่ถ้า 2 บิตแรกของคีย์ไบต์ไม่ได้มีค่าเป็น 1 ก็ให้อ่านค่าข้อมูลจากคีย์ไบต์นั้นเลย

สำหรับผังงาน (Flow Chart) และโปรแกรมภาษา C สำหรับการอ่านแฟ้มข้อมูลภาพ PCX ขึ้นมา 1 สแกนไลน์ สามารถแสดงได้ดังรูปที่ 4.1 และรูปที่ 4.2 ตามลำดับ



รูปที่ 4.1 ผังงานของการอ่านแฟ้มข้อมูลภาพ PCX 1 สแกนไลน์

```
int readpcxline( char far *p, FILE *fp )
{
    int n = 0, c, i;

    /* null the buffer */
    memset( p, 0, bytes );   /* bytes global variable , it is number of
                                bytes to read per line  */
    do
    {
        /* get a key byte */
        c = fgetc( fp ) & 0xff;
        /* if it's a run of bytes field */
        if( ( c & 0xc0 ) == 0xc0 )
        {
            /* and off the high bits */
            i = c & 0x3f;
            /* get the run byte */
            c = fgetc( fp );
            /* run the byte */
            while( i-- ) p[n++] = c;
        }
        /* else just store it */
        else p[n++] = c;
    } while ( n < bytes )
}
```

รูปที่ 4.2 โปรแกรมภาษา C สำหรับการอ่านแฟ้มข้อมูลภาพ PCX 1 สแกนไลน์

โครงสร้างข้อมูลกราฟิกแบบ TIFF (Tagged Image File Format)

โครงสร้างข้อมูลกราฟิกแบบ TIFF ถูกพัฒนาขึ้นมาจากความร่วมมือของบริษัท Aldus และ Microsoft เพื่อเป็นรูปแบบพื้นฐานในการนำภาพจากการกราดตรวจ (Scanned Image) เข้ามายังโปรแกรมประเภทการจัดพิมพ์ด้วยคอมพิวเตอร์แบบตั้งโต๊ะ (Desktop Publishing) โดย TIFF ถูกออกแบบมาให้มีความยืดหยุ่นในการเก็บภาพแบบใดก็ได้ไม่ว่าจะเป็นภาพขาวดำ ภาพสี หรือภาพที่มีระดับของสี (Shade) รวมไป ถึงภาพที่มีความละเอียด และมีขนาดต่าง ๆ กัน นอกจากนี้ยังสามารถใช้งานกับเครื่องที่แตกต่างกัน เช่น PC หรือ MacIntosh ได้อีกด้วย แต่จากการที่ TIFF มีโครงสร้างข้อมูลที่ยืดหยุ่นนี้เองจึงทำให้ TIFF มีหลายรูปแบบ การเขียนโปรแกรมที่จะอ่านภาพ TIFF ขึ้นมาจึงเป็นเรื่องที่ยุ่งยาก และอาจกล่าวได้ว่าไม่มีโปรแกรมใดๆเลยที่สามารถอ่านภาพ TIFF ได้ทุกรูปแบบ สำหรับการวิจัยครั้งนี้จะยึดรูปแบบ TIFF มาตรฐานสำหรับภาพขาวดำเป็นหลัก สิ่งที่ทำให้ TIFF ต่างจากโครงสร้างแฟ้มข้อมูลแบบอื่น ๆ ก็คือแท็ก (Tag) แต่ละแท็กของแฟ้มข้อมูล TIFF จะเป็นส่วนหัว (Header) ของส่วนที่เก็บรายละเอียดของภาพ ซึ่งแท็ก นี้เองที่ทำให้ TIFF มีความยืดหยุ่น เช่น เราสามารถที่จะเพิ่มแท็กใหม่เข้าไปในโครงสร้างแฟ้มข้อมูล TIFF เช่น แท็กของการอัดข้อมูลด้วยวิธีการใหม่ แต่ถ้าซอฟท์แวร์ไม่รู้จักการอัดข้อมูลวิธีนี้ ก็จะไม่สามารถอ่าน ภาพ TIFF นั้นได้ อาจจะสรุปได้ว่า ส่วนประกอบหลักของ TIFF มี 3 ส่วนคือ

1) ส่วนหัวของ TIFF (TIFF Header)
2) สารบบของแฟ้มข้อมูลภาพ (Image File Directory (IFD))
3) แท็ก (TAG)

1) ส่วนหัวของ TIFF (TIFF Header)

รายละเอียดของส่วนหัวของ TIFF แสดงได้ดังตารางที่ 4.3

ไบต์ที่	รายละเอียด	หมายเหตุ และค่าที่เป็นไปได้
0-1	บอกการเรียงลำดับของไบต์ (Byte order)	II = Intel MM = Motorola ดูหมายเหตุ
2-3	เลขบอกรุ่น	มีค่าเป็น 42
4-7	ชี้ไปยัง IFD ตัวแรก	

ตารางที่ 4.3 ส่วนหัวของแฟ้มข้อมูล TIFF

หมายเหตุ

ถ้าไบต์ที่ 0 - 1 เป็น MM แสดงว่าใช้ โปรเซสเซอร์ตระกูล 68000 จะอ่านไบต์ที่มีนัยสำคัญสูงสุด (Most Significant Byte) ก่อนแล้วจึงอ่านไบต์ที่มีนัยสำคัญน้อยสุด (Least Significant Byte) แต่ถ้าเป็น II แสดงว่าใช้โปรเซสเซอร์ ตระกูล 80X86 จะอ่านไบต์ที่มีนัยสำคัญน้อยสุดก่อน แล้วจึงอ่านไบต์ที่มีนัยสำคัญสูงสุด

2) สารบบของแฟ้มข้อมูลภาพ

รายละเอียดของสารบบของแฟ้มข้อมูลภาพ แสดงได้ดังตารางที่ 4.4

ไบต์ที่	รายละเอียด	หมายเหตุ และค่าที่เป็นไปได้
0-1	จำนวนแท็ก	จำนวนแท็กทั้งหมดในไดเรกทอรี
2-13	แท็ก #0	รายละเอียดอยู่ในตารางที่ 4.5
14-25	แท็ก #1	
.	.	
.	.	
(12*จ.น.แท็ก)+2	ชี้ไปยัง IFD ถัดไป (ยาว 4 ไบต์)	เป็น NULL เมื่อเป็น IFD สุดท้าย

ตารางที่ 4.4 สารบบของแฟ้มข้อมูลภาพของแฟ้มข้อมูล TIFF

หมายเหตุ

เลขที่ของไบต์จะมีความสัมพัทธ์ (Relative) กับตำแหน่งของสารบบของแฟ้มข้อมูลภาพในแฟ้มข้อมูล

3) แท็ก

รายละเอียดของแท็กแสดงได้ดังตารางที่ 4.5

ไบต์ที่	รายละเอียด	หมายเหตุ และค่าที่เป็นไปได้
0-1	ชนิดของแท็ก	ตัวเลขที่บอก ชนิดของแท็ก (TAG Identification number)
2-3	ชนิดของข้อมูล	1 = Byte (8 bit unsigned integer) 2 = ASCII (NULL Terminated 8-bit string) 3 = Short (16 bit unsigned integer) 4 = Long (32 bit unsigned integer) 5 = Rational (2 Long Num/den)
4-7	ความยาวของข้อมูล	
8-11	ข้อมูล หรือพอยท์เตอร์	ถ้าข้อมูลมีขนาด 4 ไบต์ ก็เก็บลงส่วนนี้เลย ถ้าไม่ใช่ก็เก็บลงพอยท์เตอร์ไปยังข้อมูล

ตารางที่ 4.5. แท็กของแฟ้มข้อมูล TIFF

ในการอ่านรูปภาพ TIFF นั้น ส่วนที่สำคัญที่สุดก็คือ 2 ไบต์แรกที่อ่านจากส่วนหัวของ TIFF เนื่องจากมันจะเป็นตัวบอกโครงสร้างของข้อมูลที่เหลือในแฟ้มข้อมูลทั้งหมด ดังที่ได้อธิบายไปแล้วในตารางที่ 4.3 จากนั้นก็จะเป็นการอ่านข้อมูลที่เหลือในส่วนหัวของ TIFF ซึ่งได้แก่เลขที่บอกรุ่นของ TIFF และส่วนที่ สำคัญก็คือสารบบของแฟ้มข้อมูลรูปภาพ ซึ่งจะเป็นตัวบอกว่าในรูปภาพ TIFF นี้มีแท็กทั้งหมดกี่แท็ก จากนั้นจึงอ่านข้อมูลที่อยู่ในแท็กต่าง ๆ และสุดท้ายจึงอ่านข้อมูลของรูปภาพ ซึ่งตำแหน่งที่เก็บข้อมูลรูปภาพ จะอ่านได้จากไบต์ที่ 8 - 11 ของแท็กชนิดที่ใช้ในการบอกตำแหน่งของข้อมูล

สำหรับตัวอย่างโปรแกรมภาษา C ที่ใช้ในการอ่าน และแปลความหมายของแท็กชนิดต่างๆ สามารถแสดงได้ดังรูปที่ 4.3

```
int decodetag( FILE *fp )
{
    long length, offset;
    int tag, type;
    tag = fgetword( fp );
    type = fgetword( fp );
    if( type == tifflong )
    {
        length = fgetlong( fp );
        offset = fgetlong( fp );
    }
  else
    {
        length = ( unsigned long ) fgetword( fp );
        fgetword( fp );
        offset = ( unsigned long ) fgetword( fp );
        fgetword( fp );
    }
    switch( tag )
    {
         case subfiletype:
             tiffsubfile = ( unsigned int ) offset;
             break;
         case imagewidth:
             width = ( unsigned int ) offset;
             bytes = pixels2bytes( width );
             break;
        case imagelength:
             depth = ( unsigned int ) offset;
             break;
```

รูปที่ 4.3 ตัวอย่างโปรแกรมภาษา C สำหรับการอ่านข้อมูลชนิดของแท็ก

```
        case rowsperstrip:
            if( type == tifflong )
            tiffrowstrip = offset;
            else
                tiffrowstrip = offset & 0xffffL;
            break;
        case stripoffsets:
            if( type == tifflong )
            {
                tiffstripoff = offset; tiffstripcnt = length;
            }
            else
            {
                tiffstripoff = offset & 0xffffL;  tiffstripcnt = length & 0xffffL;
            }
            break;
        case stripbytecounts:
            if( type == tifflong )
                tiffbytecnt = offset;
            else
                tiffbytecnt = offset & 0xffffL;
            break;
        case samplesperpixel:
            tiffsamples = ( unsigned int ) offset;
            break;
        case bitspersample:
            break;
        case planarconfiguration:
            tiffplancfg = ( unsigned int ) offset;
            break;
        case compression:
            tiffcompress = ( unsigned int ) offset;
            break;
```

รูปที่ 4.3 (ต่อ)

```
case group3options:
    break;
case group4options:
    break;
case fillorder:
    break;
case threshholding:
    break;
case cellwidth:
    break;
case celllength:
    break;
case minsamplevalue:
    break;
case maxsamplevalue:
    break;
case photometricinterp:
    tiffphotmet = ( unsigned int ) offset;
    break;
case grayresponseunit:
    break;
case grayresponsecurve:
    break;
case colorresponseunit:
    break;
case colorresponsecurves:
    break;
case xresolution:
    break;
case yresolution:
    break;
case resolutionunit:
    break;
```

รูปที่ 4.3 (ต่อ)

```
        case orientation:
            break;
        case documentname:
            break;
        case pagename:
            break;
        case xposition:
            break;
        case yposition:
            break;
        case pagenumber:
            break;
        case imagedescription:
            break;
        case make:
            break;
        case model:
            break;
        case freeoffsets:
            break;
        case freebytecounts:
            break;
        case software:
            break;
        default:
            printf( "Unrecognizeed tag %d\n", tag );
            exit( 1 );
            break;
    }
    return( length );
}
```

รูปที่ 4.3 (ต่อ)