影像處理與機器人視覺 HW1

BMP格式圖檔的基本影像處理

教授 : 吳進義
助教 : 張哲睿、劉力元、賴亭諭

What is a BMP file?

- 由微軟所定義的圖像檔案格式，是bitmap的縮寫
- 點陣圖的一種
  - 其他類型的點陣圖格式：jpg、gif、png、tiff……等等。
- 共有黑白(1 bit)、灰階(4 bits or 8 bits)以及彩色(24 bits)這幾種。
- BMP檔案通常不壓縮，所以它們比同一張圖像的壓縮圖檔格式要大很多。

BMP的組成結構

BMP = **BMP File Header**

+ BMP Info Header

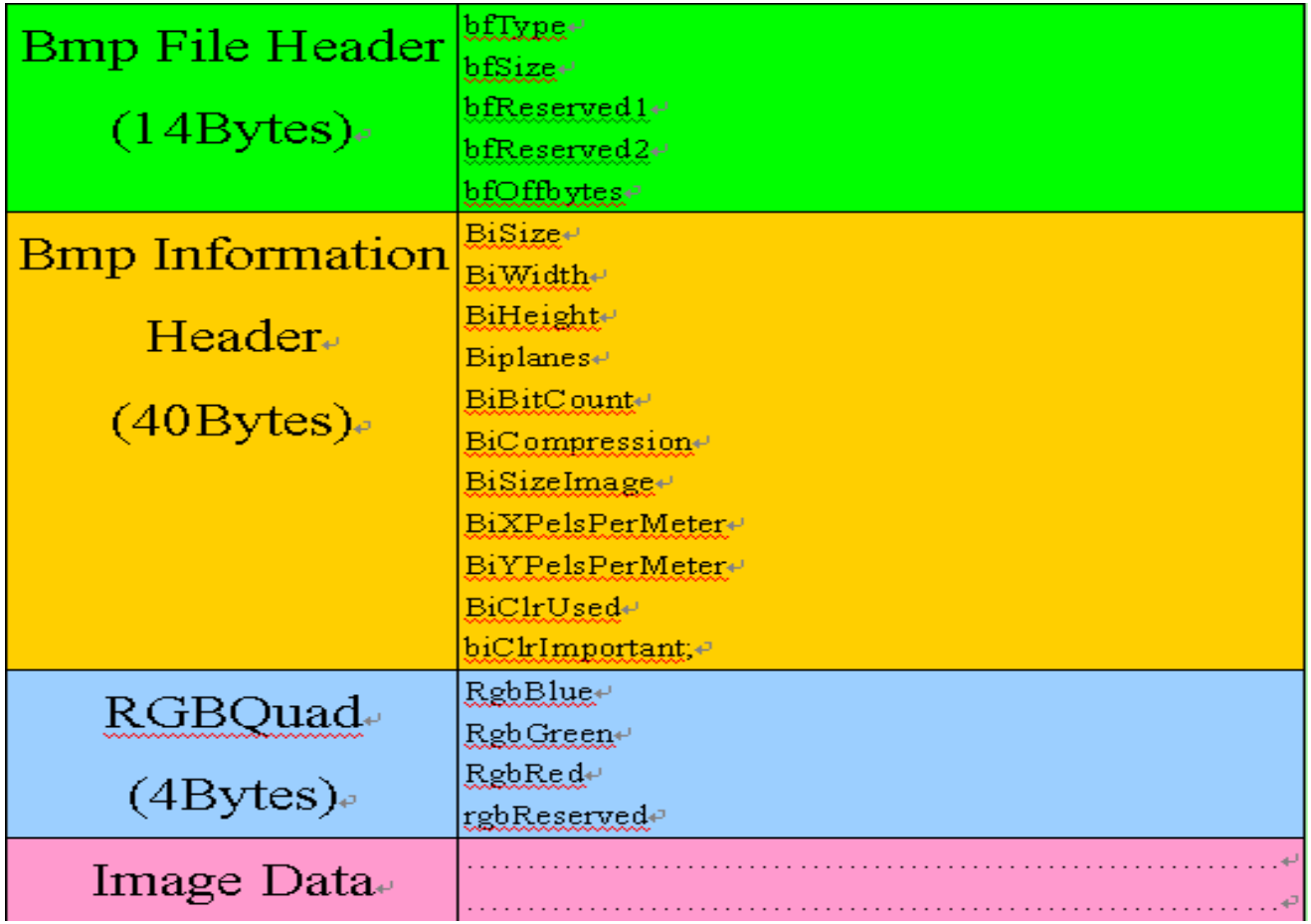
(色彩深度>=24bits) **+ Image Data Array**

= BMP File Header

+ BMP Info Header

+ RGBQUAD Array
(調色盤)

+ Image Data Array



BMP組成介紹

BMP文件由以下四部分組成：

1. BITMAPFILEHEADER:

   一共14個位元組，包含BMP圖像文件的類型、文件大小等訊息

2. BITMAPINFO:

   一共40個位元組，它包含有 BMP 圖像的寬、高、壓縮方法，以及定義顏色等訊息

3. RGBQUAD陣列

   可選，如使用索引來表示圖像，調色板就是索引與其對應的顏色的映射表

4. 點陣圖數據(bitmap data)

BMP的檔頭 - File Header

```
typedef struct tagBITMAPFILEHEADER {  /*(14bytes)*/

        WORD    bfType;         /* (2bytes)用來表示圖檔的檔案類型，一般
                                   為BM(0x42,0x4d) */
        DWORD  bfSize;          /* (4bytes) File size in bytes,整個檔案的大小 */
        WORD    bfReserved1; /* (2bytes) 保留欄位，always 0*/

        WORD    bfReserved2;  /* (2bytes) 保留欄位，always 0*/


        DWORD   bfOffbytes   /* (4bytes) 到image data的偏移量，這裡的值
                                表示檔頭和色盤總和的大小，會小於前面的
                                FileSize部份*/
} BITMAPFILEHEADER;
```

BMP的檔頭資訊 - Info Header

typedef struct tagBITMAPINFOHEADER{ (40bytes)

DWORD biSize; (4bytes) 在Windows 3.X之後這裡的值就恆為40，指的是說structure BITMAPINFOHEADER的size

LONG biWidth; (4bytes) 表示寬有多少的像素

LONG biHeight; (4bytes) 表示高有多少的像素

WORD biPlanes; (2bytes) 因為bitmap不像gif可以同許包含許多的圖，所以在這裡的值也永遠是1

WORD biBitCount (2bytes) 圖形的顏色，有1, 4, 8, 24, 32五種值

DWORD biCompression; (4bytes) 0:不經過壓縮，1:經過8-bit RLE壓縮，2:經過4-bit RLE壓縮（我們只考慮0的情況）

DWORD biSizeImage; (4bytes) 壓縮後的檔案大小，若沒有經過壓縮則此值不使用

LONG biXPelsPerMeter; (4bytes) 水平解析度（dots per meter）

LONG biYPelsPerMeter; (4bytes) 垂直解析度（dots per meter）

DWORD biClrUsed; (4bytes) 使用色盤中的色彩數，若為0則表示使用色盤裡全部的顏色

DWORD biClrImportant; (4bytes) 有幾個關鍵色，這個數字僅供參考。

} BITMAPINFOHEADER;

BMP的調色盤 - RGBQUAD

```
typedef struct tagRGBQUAD{        //(4bytes)
    BYTE    rgbBlue;              //(1bytes)
    BYTE    rgbGreen;             //(1bytes)
    BYTE    rgbRed;               //(1bytes)
    BYTE    rgbReserved;          //(1bytes)
} RGBQUAD;
```

BMP的內容 – Image Data

- 所謂的**ImageData**就是指一個巨大的矩陣，在**BMP**中有一個特點，就是每一列的長度剛剛好都是**4**的倍數，假若不恰好為**4**的倍數那就會補**0**

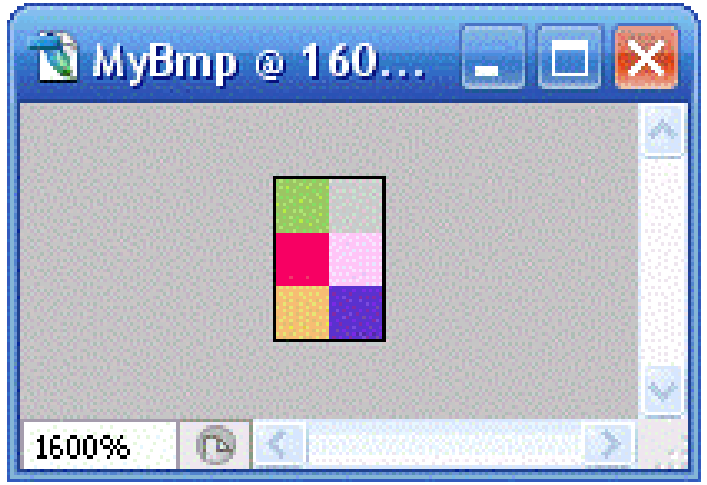
- **ImageData**的排列順序是從**image**的最後一行開始紀錄，到最後紀錄的是第一行的資料。

BMP的檔頭資訊與內容

testgray.bmp

```
00000000h: 42 4D 76 A0 00 00 00 00 00 00 36 04 00 00 28 00 ; BMv?.....6...(.
00000010h: 00 00 C8 00 00 00 C8 00 00 00 01 00 08 00 00 00 ; ..?..?........
00000020h: 00 00 40 9C 00 00 13 0B 00 00 13 0B 00 00 00 00 ; ..@?...........
00000030h: 00 00 00 00 00 00 00 00 00 00 01 01 01 00 02 02 ; ................
00000040h: 02 00 03 03 03 00 04 04 04 00 05 05 05 00 06 06 ; ................
00000050h: 06 00 07 07 07 00 08 08 08 00 09 09 09 00 0A 0A ; ................
00000060h: 0A 00 0B 0B 0B 00 0C 0C 0C 00 0D 0D 0D 00 0E 0E ; ................
00000070h: 0E 00 0F 0F 0F 00 10 10 10 00 11 11 11 00 12 12 ; ................
00000080h: 12 00 13 13 13 00 14 14 14 00 15 15 15 00 16 16 ; ................
00000090h: 16 00 17 17 17 00 18 18 18 00 19 19 19 00 1A 1A ; ................
000000a0h: 1A 00 1B 1B 1B 00 1C 1C 1C 00 1D 1D 1D 00 1E 1E ; ................
000000b0h: 1E 00 1F 1F 1F 00 20 20 20 00 21 21 21 00 22 22 ; ......   .!!!."
000000c0h: 22 00 23 23 23 00 24 24 24 00 25 25 25 00 26 26 ; ".###.$$$.%%%.&&
000000d0h: 26 00 27 27 27 00 28 28 28 00 29 29 29 00 2A 2A ; &.'''.(((.))).**
000000e0h: 2A 00 2B 2B 2B 00 2C 2C 2C 00 2D 2D 2D 00 2E 2E ; *.+++.,,,.---...
000000f0h: 2E 00 2F 2F 2F 00 30 30 30 00 31 31 31 00 32 32 ; ../// .000.111.22
00000100h: 32 00 33 33 33 00 34 34 34 00 35 35 35 00 36 36 ; 2.333.444.555.66
00000110h: 36 00 37 37 37 00 38 38 38 00 39 39 39 00 3A 3A ; 6.777.888.999.::
00000120h: 3A 00 3B 3B 3B 00 3C 3C 3C 00 3D 3D 3D 00 3E 3E ; :.;;;.<<<.===.>>
00000130h: 3E 00 3F 3F 3F 00 40 40 40 00 41 41 41 00 42 42 ; >.???.@@@.AAA.BB
00000140h: 42 00 43 43 43 00 44 44 44 00 45 45 45 00 46 46 ; B.CCC.DDD.EEE.FF
00000150h: 46 00 47 47 47 00 48 48 48 00 49 49 49 00 4A 4A ; F.GGG.HHH.III.JJ
00000160h: 4A 00 4B 4B 4B 00 4C 4C 4C 00 4D 4D 4D 00 4E 4E ; J.KKK.LLL.MMM.NN
00000170h: 4E 00 4F 4F 4F 00 50 50 50 00 51 51 51 00 52 52 ; N.OOO.PPP.QQQ.RR
```

BMP的格式與檔頭資訊




```
00000000h: 42 4D 50 00 00 00 00 00 00 00 36 00 00 00 28 00 ; BMP.......6...(.
00000010h: 00 00 02 00 00 00 03 00 00 00 01 00 18 00 00 00 ; ................
00000020h: 00 00 1A 00 00 00 12 0B 00 00 12 0B 00 00 00 00 ; ................
00000030h: 00 00 00 00 00 00 72 C6 F0 CC 33 66 00 00 66 00 ; ......r起?f..f.
00000040h: FF FF CC FF 00 00 66 CC 99 CC CC CC 00 00 00 00 ;     ?..f謊燙?...
```

作業要求

產生灰階漸層圖(10%)

讀BMP圖檔(15%)

將讀進來的BMP檔頭資訊show出來(10%)

影像處理(50%)

- 點波源干涉(25%)
  - background(+10%)(加分題:自己產生作為背景的點波源干涉圖)
  - normalization(10%)
  - combine(15%)
- 顏色修改(25%)

將處理結果寫成BMP檔(15%)

有限制不能使用Python的第三方套件喔
ex : Numpy、OpenCV…
請各位試著自己做做看~

Note: 產生灰階漸層圖

- 產生一張寬是 255、高是100的灰階變化圖(8bit灰階)，然後存成 bmp
- x位置是多少，該直行的灰階值就是多少

↑像這樣子的圖

Note:點波源干涉

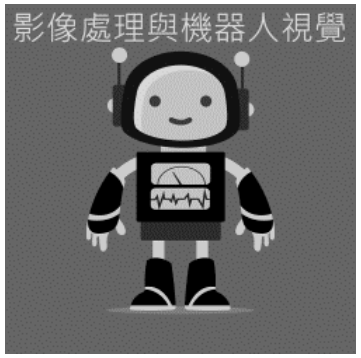
- 背景(會附上，有自己做出來有加分)
  - 第一個波源影響值(-1~1)
  - 第二個波源影響值(-1~1)
  - 第一個+第二個波源影響值(-2~2)
  - 第一個+第二個波源影響 Normalization(0~255)
- 與原前景圖結合
  - 背景+Weight*前景(<0, >255)
  - Normalization(0~255)

Note :顏色修改

- 透過**修改RGBQUAD調色盤**將灰階影像修改成彩色影像
  - 檔案大小不會改變

原始影像

Inverse



黃色系

綠色系

紅色系