

# Digital Image Processing Homework 1 Report

109511207 蔡宗儒

## 1. BMP Format

BMP 為點陣圖 Bitmap 的縮寫，也稱為 DIB，是一種儲存點陣式數位圖像的檔案格式。BMP 檔案包含 4 個部分: bitmap file header(14 bytes)、bitmap information header(40 bytes)、palette(4 \* Used Colors bytes)和 bitmap array。

### Bitmap File Header

Name	Size (bytes)	Start Address	Description
Identifier (ID)	2	0x0000	Identify type of bitmap. It is usually 'BM'
File Size	4	0x0002	Total size of file (unit: byte)
Reserved	4	0x0006	Reserved
Bitmap Data Offset	4	0x000A	The offset of the beginning of the bitmap array. It is always 54 bytes in this homework.

### Bitmap Info Header

Name	Size (bytes)	Start Address	Description
Header Size	4	0x000E	Total size of bitmap info header. It is always 40 bytes in this homework.
Width	4	0x0012	Width of bitmap array in pixels
Height	4	0x0016	Height of bitmap array in pixels
Planes	2	0x001A	Number of planes of bitmap array. It is always = 1.
Bits Per Pixel	2	0x001C	Pixel size 1: 1-bit image(使用 2 色調色盤) 4: 4-bit image(使用 16 色調色盤) 8: 8-bit image(使用 256 色調色盤) 16: 16-bit image(不一定使用調色盤) 24: 24-bit image(不使用調色盤) 32: 32-bit image(不一定使用調色盤)
Compression	4	0x001E	Type of compression 0: uncompressed, without using a palette 1 : RLE 8-bit/pixel 2 : RLE 4-bit/pixel 3 : bit fields
Bitmap Data Size	4	0x0022	Size of bitmap array (unit: byte). Valid to set this equal 0 if Compression equal 0
H Resolution	4	0x0026	Horizontal resolution
V Resolution	4	0x002A	Vertical resolution
Used Colors	4	0x002E	Palette colors used
Important Colors	4	0x0032	Important color count

## Palette

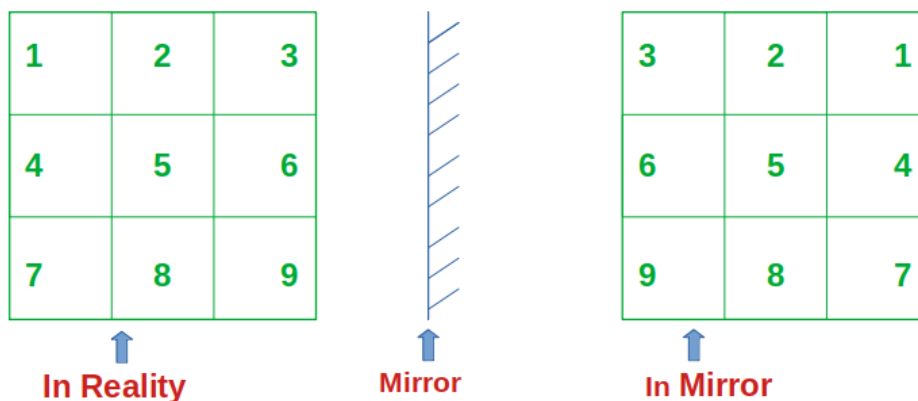
Name	Size (bytes)	Start Address	Description
Palette	$N * 4$	0x0036	Palette data

## Bitmap Array

Name	Size (bytes)	Start Address	Description
Bitmap Data			Image data

## 2. Flip

原本的 BMP 格式是由左下的 pixel 往右上讀取，所以要水平翻轉整張圖的話，只要改成由右下往左上寫入檔案即可，即將 bitmap array 每一個 row 反轉，如下圖。



## Result

input1.bmp	output1_flip.bmp	input2.bmp	output2_flip.bmp
			

## 3. Resolution







對圖像進行 quantization 的話，可以減少一張圖像中所使用的顏色數量。而對於高頻的圖像來說，對比度較高，如果透過 quantization 減少 bit 數的話，人的肉眼較不擅長區別高頻率亮度變化的確切強度，也較不易感受到差異或是失真。所以我們可以透過 quantization 適度減少圖像所需的儲存空間。

而我透過以下的程式碼來實作，我將每個 pixel 往右 shift 要量化的 bit 數(ex. 原先 8bits 變 6bits, factor 即為  $8-6=2$ )，再將其往左 shift 回來，就可以簡單地量化為新的 bit 數。

```
// quantization resolution
for(int i=0;i<_height * _width * _channel;i++)
    _out_image[i] = (_out_image[i] >> factor) << factor;
```

## Result

由結果可觀察到，人的肉眼其實感受不太到 8bits 和 6bits 圖像的差別，仔細觀察 6bits 的 input2 的天空的話，可以發現有些許失真。這是因為 input2 較為低頻，並沒有快速或劇烈的變化，人眼也就較容易辨識這些損失，而 input1 較為高頻，可以看到每朵花之間的色彩鮮艷、變化迅速，因此人眼也就較難辨識高頻率亮度的確切強度。若是再觀察兩個的 4bits 圖，便可發現 input2 已產生肉眼明顯可見的失真了，然而 input1 失真效果較不明顯。這個結果也證實了人眼對於圖像中的高頻成分較不敏感的事實。

Resolution	8bits	6bits	4bits	2bits
input1				
input2				

## 4. Scaling

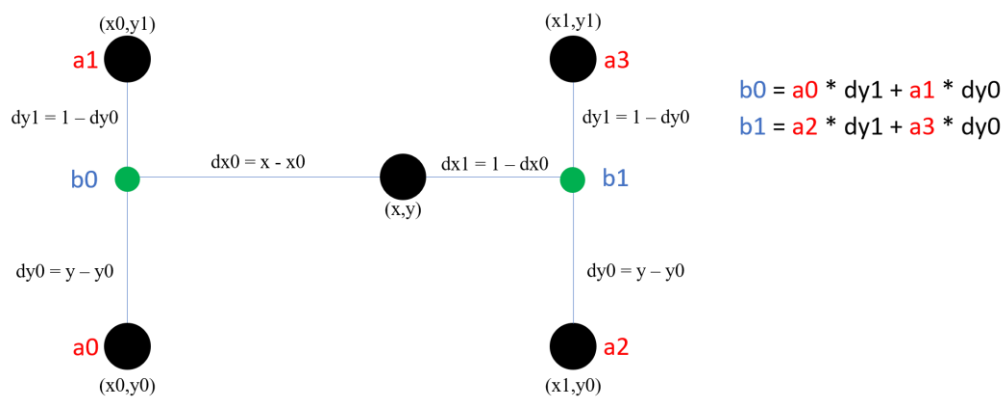
在做 Bilinear Interpolation 前，要先計算出 output image 的 size，並且因為這次作業是使用 BMP Format，所以要將 width 變成 4 的倍數。最後要把原本 output image 的位置轉換回去原本 input pixel 的位置(x,y)，為了對齊和避免超界，我在計算 inverse ratio 的時候將 input/output 的 width/height 都 -1.0，最後用 floor()和 ceil()找出離此點最近的四個點，如下圖。

```
// get inverse ratio, - 1.0 to avoid x0, x1, y0 and y1 from exceeding the boundaries
xRatio = (_height - 1.0) / ((*int*) &_out_header[22]) - 1.0;
yRatio = (_width - 1.0) / ((*int*) &_out_header[18]) - 1.0;







// get the original position where each new pixel point should be on the original image
x = i * xRatio;
y = j * yRatio;

// get 4 nearest pixels of the original position
int x0 = floor(x);
int x1 = ceil(x);
int y0 = floor(y);
int y1 = ceil(y);
```

Bilinear Interpolation 如下圖，先在 y 方向用插值法將 b0 和 b1 求出來，再對 x 方向做一次插值法，插值出(x,y)的 data，即  $b0 * dx1 + b1 * dx0$ 。



## Result

Scaling	Down Scaling ( $\div 1.5$ )	Original	Up Scaling ( $\times 1.5$ )
input1			
input2			

## 5. Reference

<https://zh.wikipedia.org/zh-tw/BMP#%E6%96%87%E4%BB%B6%E6%A0%BC%E5%BC%8F>

<https://blog.lusw.dev/posts/bitmap-file-structure.html>

<https://www.796t.com/content/1549504280.html>

<https://www.geeksforgeeks.org/check-if-the-given-two-matrices-are-mirror-images-of-one-another/>

<https://charlottehong.blogspot.com/2017/11/bilinear.html>