



第九章 影像壓縮

內 容

- 9.1 前 言
- 9.2 霍夫曼編碼
- 9.3 向量量化法
- 9.4 靜態影像壓縮
- 9.5 作 業

9.1 前言

- JPEG 內含數種壓縮技巧混合而成的系統。我們將針對其中的霍夫曼編碼 (Huffman Coding)，向量量化法 (Vector Quantization)，和 JPEG 基本架構做介紹。

9.2 霍夫曼編碼

■ 霍夫曼樹

給一符號其 $S = \langle S_1, S_2, \dots, S_8 \rangle$ ，而其
對應的頻率為 $S = \langle 14, 13, 5, 3, 3, 2, 1, 1 \rangle$ ，

$\langle S_1, S_2, \dots, S_8 \rangle$ 的碼可編成

$\langle C_1, C_2, \dots, C_8 \rangle$

$= \langle 11, 10, 010, 001, 000, 0111, 01101, 01100 \rangle$

這些碼的長度為

$\langle l_1, l_2, \dots, l_8 \rangle = \langle 2, 2, 3, 3, 3, 4, 5, 5 \rangle$

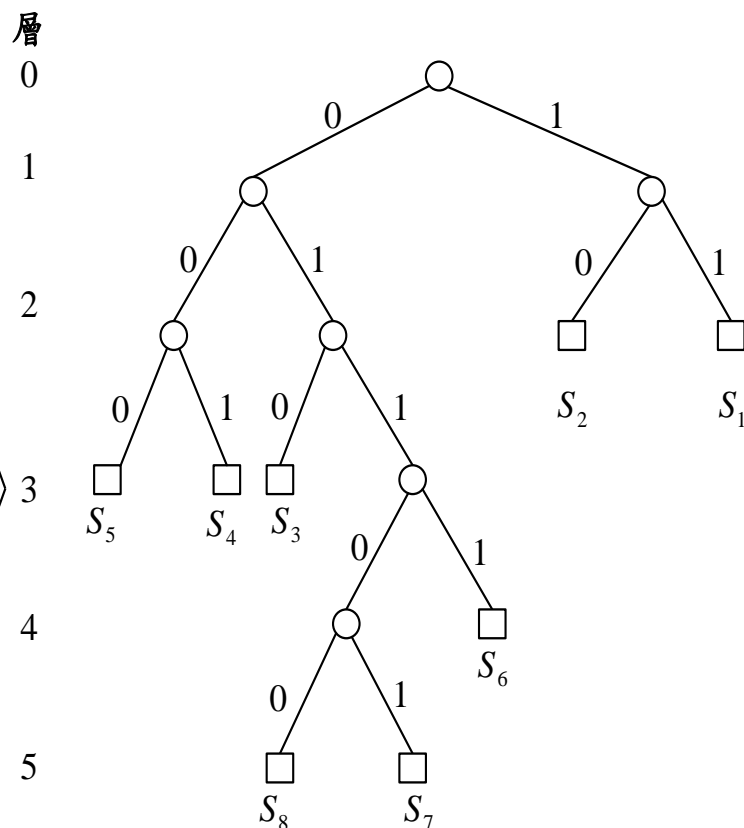


圖 9.2.1 霍夫曼樹

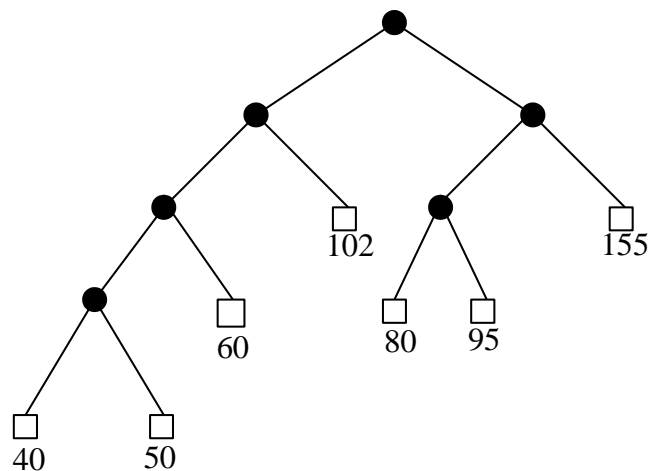
範例 9.2.1：

在影像處理中，霍夫曼編碼可用於不失真壓縮上，現有一 4×4 灰階影像如下所示，假設符號集 S 為灰階值，而頻率集 W 為每個灰階值所對應的出現頻率，利用霍夫曼編碼，請實作出本張影像所代表的霍夫曼樹，並寫出灰階值為 50 的像素之霍夫曼碼長。

60	102	80	95
95	40	155	60
102	155	102	155
50	80	155	95

解答：

符號集 $S = \langle 40, 50, 60, 80, 95, 102, 155 \rangle$ 對應的頻率集為 $W = \langle 1, 1, 2, 2, 3, 3, 4 \rangle$ ，建出的霍夫曼樹如下所示：



灰階值 50 的霍夫曼碼長為 4，而其對應的碼為 0001。

■ 單邊成長 (Single-side Growing) 霍夫曼樹

令 $C'_1 = 11\dots 1$ ，且 $|C'_1| = l_1$ 。 $C'_2 = (C'_1 \times 2^{l_2 - l_1}) - 1 = 11\dots 10$ ，

$C'_i = (C'_i \times 2^{l_i - l_{i-1}}) - 1$ ，可見出圖 9.2.2 單邊成長霍夫曼樹。

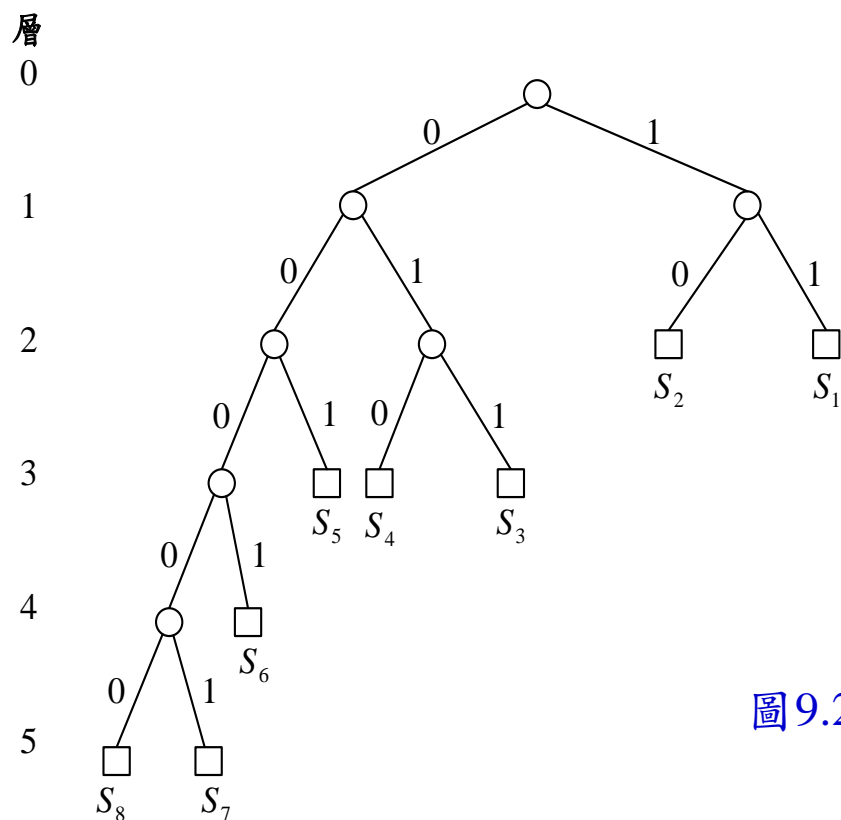


圖9.2.2 單邊成長霍夫曼樹

$$\langle C'_1, C'_2, \dots, C'_8 \rangle = \langle 11, 10, 011, 010, 001, 0001, 00001, 00000 \rangle$$

f_i 代表第*i*層的葉子數， I_i 代表第*i*層的內部節點樹

$$\langle f_1, f_2, f_3, f_4, f_5 \rangle = \langle 0, 2, 3, 1, 2 \rangle$$

$$\langle I_0, I_1, I_2, I_3, I_4 \rangle = \langle 1, 2, 2, 1, 1 \rangle$$

$$A[0..7] = [S_2, S_1, S_5, S_4, S_3, S_6, S_8, S_7]$$

$$\text{給 } H = 001 \Rightarrow H[1] = 0, f_1 = 0$$

$$\Rightarrow H[1..2] = 00, f_2 = 2 \text{ 需跳過 } A[0..1] \text{ 中的兩個樹葉}$$

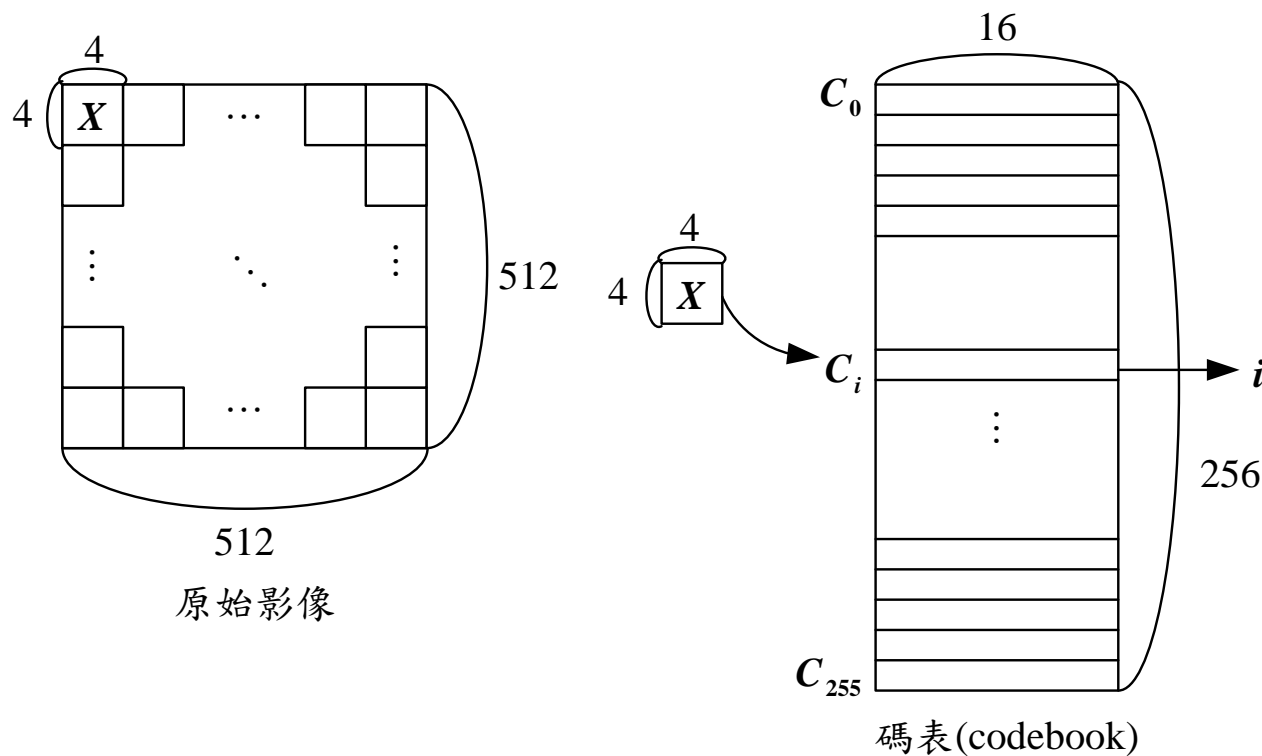
$$\Rightarrow H[1..3] = 001, f_3 = 3 \Rightarrow A[2] = S_5$$

9.3 向量量化法

令碼表中的碼為 $\{C_i \mid 0 \leq i \leq 255\}$ 而待搜尋的區塊向量為 X ，找到 C_i

$$\text{使得 } d^2(X, C_i) = \min_j \sum_{n=1}^{16} (X_n - C_{jn})^2$$

這裡 $X = (X_1, X_2, \dots, X_{16})$ 而 $C_j = (C_{j1}, C_{j2}, \dots, C_{j16})$



■ 金字塔式向量搜尋法

若每四個元素縮成一個平均值

$$\begin{aligned} \rightarrow & 4^q d^2(f_q(X), f_q(C_i)) \\ & \leq 4^{q-1} d^2(f_{q-1}(X), f_{q-1}(C_i)) \\ & \vdots \\ & \leq 4d^2(f_1(X), f_1(C_i)) \\ & \leq d^2(X, C_i) \end{aligned}$$

使用的資料結構為金字塔， q 可被看成為金字塔的高度。在上面的不等式中， $f_1(X)$ 為 X 縮小 $1/4$ 後的上一層之向量，而 $f_1(C_i)$ 為 C_i 的上一層之向量，這裡 X 和 C_i 皆可視為最底層的向量。

若 $4^q d^2(f_q(X), f_q(C_i))$ 的值比目前暫時的最小值都來的大時，則 C_i 就不必再往金字塔的下層考慮了。

9.4 靜態影像壓縮

JPEG 首先將輸入的影像切割成 8×8 的子影像集。將輸入全彩影像中每一像素的 R、G 和 B 值轉換為 Y、Cb 和 Cr 值。

- 每一像素皆先減去 128，以下列的計算完成 DCT

$$F(u,v) = C(u)C(v) \sum_{x=0}^7 \sum_{y=0}^7 f(x,y) \cos \frac{(2x+1)u\pi}{16} \cos \frac{(2y+1)v\pi}{16}$$

圖9.4.1

經 DCT 作用後的結果

79	75	79	82	82	86	94	94
76	78	76	82	83	86	85	94
72	75	67	78	80	78	74	82
74	76	75	75	86	80	81	79
73	70	75	67	78	78	79	85
69	63	68	69	75	78	82	80
76	76	71	71	67	79	80	83
72	77	78	69	75	75	78	78

(a) 8×8 子影像

619	-29	8	2	1	-3	0	1
22	-6	4	0	7	0	-2	-3
11	0	5	-4	-3	4	0	-3
2	-10	5	0	0	7	3	2
6	2	-1	-1	-3	0	0	8
1	2	1	2	0	2	-2	-2
-8	-2	-4	1	2	1	-1	1
-3	1	5	-2	1	-1	1	3

(b) 8×8 係數矩陣

圖9.4.2

量化表與量化後的結果

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

(a) 8 × 8 量化表

39	-3	1	0	0	0	0	0
2	-1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	-1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(b) 8 × 8 量化後DCT係數矩陣

圖 9.4.2(b) 的 DCT 係數矩陣經 IDCT(Inverse DCT)

$$f(x, y) = \sum_{u=0}^7 \sum_{v=0}^7 C(u)C(v)F(u, v) \cos \frac{(2x+1)u\pi}{16} \cos \frac{(2y+1)v\pi}{16}$$

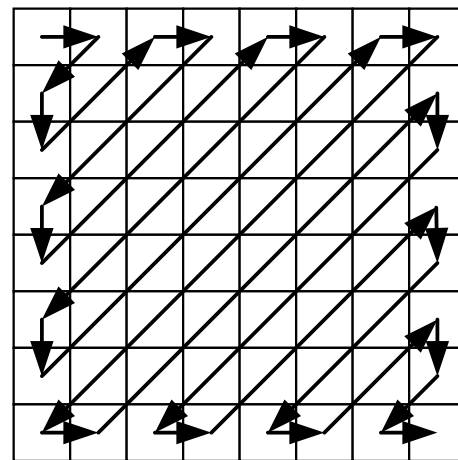
作用後，可得解壓後的影像，如圖 9.4.3 所示。

74	75	77	80	85	91	95	98
77	77	78	79	82	86	89	91
78	77	77	77	78	81	83	84
74	74	74	74	76	78	81	82
69	69	70	72	75	78	82	84
68	68	69	71	75	79	82	85
73	73	72	73	75	77	80	81
78	77	76	75	74	75	76	77

圖9.4.3 8 × 8解壓後影像

- 依據 Zig-Zag 的掃描次序，得到圖 9.4.2(b) 的向量型式

$(39, -3, 2, 1, -1, 1, 0, 0, 0, 0, 0, -1, 0, 0, 0, \dots, 0, 0, 0)$ 。



- 進行 Run-length 編碼

可編碼為 $(0, -3)(0, 2)(0, 1)(0, -1)(0, 1)(5, -1)$ EOB 圖9.4.4 Zig-Zag掃描次序

在 Run-Length 編碼的格式 (x, y) 中， x 通常採用固定長度編碼，而 y 則依照事先建好的圖 9.4.5 表進行變動長度編碼。上述的向量型式進一步編成

$(0, 2)(00)(0, 2)(10)(0, 1)(1)(0, 1)(0)(0, 1)(1)(5, 1)(0)$

位元數	y 的範圍
0	0
1	-1, 1
2	-3, -2, 2, 3
3	-7, ..., -4, 4, ..., 7
4	-15, ..., -8, 8, ..., 15
:	:
:	:

圖9.4.5 y 的編碼對照表

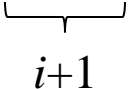
- 進行 DPCM (Differential Pulse Code Modulation) 和霍夫曼編碼 (Huffman Encoding)

範例 9.4.1：

試問如何對 y 的範圍編碼？

解答：

根據圖 9.4.5 中位元數的機率分佈，若 DCT 係數掉在位元數 i 所對應的 y 範圍內，則該 DCT 係數可編碼為 $0 \dots 01$ 。



例如：某一 DCT 係數為 -5，則很容易找到其對應的位元數為 3，我們可將 DCT 係數 -5 編碼成 0001。

範例 9.4.2：

給予下面 AC 值所編的 Run-Length 編碼：

(0,3) (011) (0,2) (11) (0,1) (1) (0,2) (10) (0,2) (01) (0,1) (1) (1,1) (0)
(0,1) (1) (8,1) (1) 配合下圖的 y 編碼對照表

位元組	y 的範圍
0	0
1	-1,1
2	-3,-2,2,3
3	-7,...,-4,4,...,7
4	-15,...,-8,8,...,15
...	...

求出影像經量化後的 DCT 係數矩陣。（假設 DC 值為 34，區塊大小為 8）

解答：

可得出 AC 值在矩陣中的位置及大小，所得出的值如下：

(0,-4) (0,3) (0,1) (0,2) (0,-2) (0,1) (1,-1) (0,1) (8,1) 根據上面的向量，最後所得到的矩陣如下：

34	-4	-2	1	0	0	0	0
3	2	0	0	0	0	0	0
1	-1	0	0	0	0	0	0
1	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

9.5 作 業

- 作業一：寫一 C 程式以完成霍夫曼編碼的實作。
- 作業二：依本章介紹的 JPEG 系統架構，實作出簡易版的 JPEG 系統。