Journal of Science and Engineering Technology, Vol. 4, No. 1, pp. 1-9 (2008)

利用動態樹狀結構及區塊預測技術的動量絕對值區塊截短編碼研究

陳文儉　曾裕洲　黃文聰
大葉大學資訊工程學系
彰化縣大村鄉山腳路 112 號

摘　要

一般而言，以空間域爲主的影像壓縮技術，相鄰的影像區塊間通常具有非常高的空間相似性，所以區塊預測技術便可利用來降低影像壓縮的位元率。此外，有許多以區塊爲主的壓縮技術根據運用場合的不同，藉著互相結合運用以達到更好的效能。因此本篇論文提出了一個植基於區塊截短編碼的影像壓縮技術，此方法結合了原本運用在無失真向量量化索引值壓縮上的動態樹狀結構編碼法觀念及一種利用量化表技術的改良式區塊截短編碼壓縮法，不但能降低影像壓縮位元率，且有不錯的重建影像品質。

根據實驗結果顯示，我們的方法所需要的位元率非常低且重建品質相當不錯，以灰階 512×512 像素的 Lena 影像而言，可以達到 PSNR 值接近 32dB、壓縮位元率 0.6 bits/pixel 左右的效果。特別的是，相較於其他方法，我們提出的方法是以更簡單的方法達到非常高的效率，計算複雜度相當的小。

關鍵詞：區塊預測技術，動量絕對值區塊截短編碼，向量量化，動態樹狀結構編碼法

An AMBTC Coding Scheme Using the Dynamic Tree and Block Prediction Techniques

WEN-JAN CHEN, YU-GHOU TSENG and WEN-TSUNG HUANG
Department of Computer Science and Information Engineering, Da-Yeh University
No. 112, Shanjiao Rd., Dacun, Changhua, Taiwan 51591, R.O.C.

ABSTRACT

In spatial domain compression schemes, there is usually a high correlation between the neighboring blocks. Therefore, the Block Prediction Technique can be employed to further reduce the bit rate of a compressed image. Moreover, many block-dominated compression schemes have been combined to achieve good performance according to various applications.

Thus, we proposed a method based on Absolute Moment Block Truncation Coding (AMBTC), which combines the concepts of the Dynamic Tree Coding Scheme of Vector Quantization and Block Code Modified-BTC. The experimental results show that the proposed algorithm achieves a significant reduction in the bit rate and yields a reconstructed image of very acceptable quality. For

a 512 × 512 pixel image, “lean” with 256 gray levels, the quality of the resulting reconstructed image is about 32 dB at 0.6 bits/pixel. More importantly, our proposed method achieves compression efficiency while maintaining a low computational complexity.

***Key Words*:** block prediction technique, absolute moment block truncation coding, vector quantization, dynamic tree coding scheme

一、緒論

區塊截短編碼（block truncation coding , BTC）[3] 這個影像壓縮技術，是由 Delp 與 Mitchell 於 1979 年所提出的，此壓縮技術並不需要複雜的數學計算，壓縮及解壓縮的速度快，此外壓縮之後可以的到不錯的影像重建品質。然而「太低的壓縮率」則是 BTC 的主要缺點，相較於近年來熱門的壓縮法，BTC 顯得相形見絀。爲了提升 BTC 的效能，到目前爲止已有相當多的研究來探討它。這些技術除了較早提出由 Lema 與 Mitchell 於 1984 年所提出的動量絕對值區塊截短編碼（absolute moment BTC, AMBTC）[5] 之外，有針對位元圖的取樣來編碼，以原有的 16 位元的位元圖（區塊大小爲 4×4）只取其中 8 個位元當作樣本做編碼 [9]，位元圖以類似向量量化（vector quantization, VQ）[4, 6, 8] 的編碼簿（codebook）的概念，找出最具有代表性的位元圖來編碼 [9, 11]。在重建階量化方面，有根據區塊特性來採取不同的編碼策略，例如較平滑的區塊用區塊平均值來編碼，較複雜的區塊使用查表或原來的區塊截短編碼方式來編碼 [7]。另外，植基於 BTC 的技術應用在較特別用途的有 Rozinaj 等人於 2002 年提出的 Block Code Modified-BTC（MBTC）[10] 使用量化表的觀念針對衛星影像壓縮編碼，能保有不錯重建品質且位元率也降低一些。

向量量化的特點有二，高壓縮率和相當簡單的解碼端。在傳統 VQ，會將影像切成不重疊的區塊（通常爲 4×4），然後再從事先訓練過的編碼簿裡找出與它最匹配的區塊並送出該區塊在編碼簿裡的索引值（index）給解碼端，解碼端根據接收到的索引值後，只需經過簡單的查表法（table look-up）便可很容易的重建影像。然而雖然 VQ 有上述的優點，在編碼端方面仍是相當費時，造成編碼端與解碼端的不對稱。針對 BTC 與 VQ 兩種技術而言，VQ 的高壓縮率正是 BTC 所欠缺的，相對來講 BTC 的簡單性也優於 VQ，所以有許多新提出的壓縮方法便常常結合兩種壓縮技術來達到更好的效能 [1]。

在 VQ 中，通常也因爲區塊的大小切成非常小，因此具有相當大的空間相似性。也就是說，相鄰的區塊有可能被量化成相同的索引值並且索引值之間的差距也可能非常的小，因此無失真的變動長度編碼技術（variable length coding, VLC）便可應用來對 VQ 編碼端索引值再做進一步的無失真壓縮。近來，Chen 等人 [2] 提出了一種利用從周圍鄰近區塊獲取空間相似性的資訊，再以此空間相似性資訊選擇不同的編碼樹對索引值編碼，將壓縮位元率又更進一步的降低且不會造成任何失真。實際上，近來的無失真向量量化索引值編碼法（lossless index-coding）都是利用空間相似性來預測目前要編碼的索引值與可能會與周圍索引值相同的機率來做編碼。同樣的道理，許多區塊預測技術也是利用此原理來預測區塊。因此在本篇論文中，我們將利用動態樹狀結構編碼（dynamic tree coding scheme, DTCS）觀念來作區塊預測，並結合了原本運用在衛星影像的 Block Code Modified-BTC 提出一個新的植基於區塊截短編碼的影像壓縮技術。我們根據對於目前要編碼的區塊，在它的左方和上方區塊來判斷選擇不同的編碼方法。如果它的上方區塊或者左方區塊與本身具有相當大的空間相似性，我們選擇使用以 DTCS 的觀念來作區塊預測編碼。反之如果空間相似性非常小時，我們使用 MBTC 編碼。

根據實驗結果顯示，我們提出的方法不但能有效的降低位元率，平均壓縮比可達 10 倍以上，且重建後的影像品質亦相當不錯，在計算量方面與其它技術相比仍相當具有效率。本篇論文其他章節架構如下：第二節我們將回顧相關的技術，包括 AMBTC、MBTC 和 DTCS。我們所提出的方法將詳細的在第三節中說明。第四節裡，將對我們的所提出的方法和 AMBTC、MBTC 和傳統 VQ 作相關實驗比較及分析結果。本篇論文的結論列於第五節。

二、相關技術回顧

在這一章節裡，我們首先在第一節裡介紹絕對值動量區塊截短編碼 [5]。另外由於我們提出的方法是結合一個植基於動態樹狀結構編碼法的無失真向量量化索引值壓縮法 [2]

裡所用的觀念和一個改良的區塊截短編碼（modified block truncation coding, MBTC）技術 [10]，因此我們分別在第二節及第三節裡面回顧這兩個演算法。

（一）動量絕對值區塊截短編碼（Absolute Moment BTC, AMBTC）

Lema 與 Mitchell 於 1984 年由所提出的動量絕對值區塊截短編碼 [5]，是先將原影像切割成不重疊的 n×n 區塊，令 $k=n\times n$，x_1，x_2，x_3...x_k 爲原影像中一個區塊 X 的個別像素值。首先要計算出區塊的平均值 $\bar{x}$ ，利用公式（1）算出區塊平均值 $\bar{x}$。另外，將動量絕對值區塊截短碼的保留動量絕對值 α，以公式（2）計算。

$$\overline{X}=\frac{1}{k}\sum_{i=1}^{k}x_i \qquad (1)$$

$$\alpha=\frac{1}{k}\sum_{i=1}^{k}\left|x_i-\overline{X}\right| \qquad (2)$$

在 AMBTC 中，一般來說是把區塊平均值 $\overline{X}$ 定義爲區塊門檻值。計算出區塊平均值之後，如果像素值大於或等於區塊平均值 $\bar{x}$ 則位元圖上該像素的對應位元值設爲 1；像素值小於區塊平均值 $\bar{x}$ 則位元圖上該像素的對應位元值設爲 0。整個區塊依照位置順序比較所有像素值，就可以產生該區塊位元圖。而區塊中所有像素的對應位元值決定了之後，也將影像區塊中的像素分成兩群：像素值大於或等於平均值 $\overline{X}$ ，位元值 1 的爲一群；像素值小於平均值 $\overline{X}$ ，位元值 0 的爲另一群。接著可以運用這兩群的資料計算出其對應的重建階。兩個重建階 a 與 b 的計算公式（3）及（4）如下：

$$a=\overline{X}-\frac{k\alpha}{2(k-q)} \qquad (3)$$

$$b=\overline{X}+\frac{k\alpha}{2q} \qquad (4)$$

其中 $k=n\times n$，同前面爲區塊中包含的像素個數，q 爲方塊中像素值大於或等於 $\bar{x}$ 的數目。一個經過 AMBTC 技術壓縮的影像區塊其區塊壓縮碼會以三欄的資料（a, b, B）表示，其中 B 是爲位元圖，或也可以另一個三欄的資料（$\bar{x}$, α, B）來傳送給解碼端。

在解壓縮的程序中，解碼端會依照壓縮端所傳送的資料（a, b, B）依序的將影像區塊重建起來。解碼端依照位元圖的位元值，若 位元圖中位元值是 1 時，會以較高的重建階 b 來取代；若位元圖中位元值爲 0 時，則以較低重建階 a 來取代。每一個區塊依照這樣的重建方式來重建影像。

（二）改良式區塊截短編碼（Modified Block Truncation Coding, MBTC）

現在我們介紹原本用於衛星影像用途的 Block Code Modified-BTC（MBTC）[10]，將此方法改爲應用於自然影像上的理想狀況，可以把壓縮位元率降低到 1.5 bits/pixel 左右。

在 MBTC 中，主要改進的是編碼端在編碼第一個區塊時，按照 AMBTC 的編碼規則，送出一個 16 位元的位元圖及二個 8 位元的重建階。但是從第二個區塊開始，雖依舊送出位元圖但不再送出二個重建階，而改送該區塊重建階與前一個區塊重建階的差（difference）。一般來說，基於相鄰區塊的空間相似性關係，重建階的差值將遠遠小於原來的重建階。因此在 MBTC 方法中，原本屬於高重建階（upperlevel）和低重建階（lowerlevel）的差只需用 8 位元來表示，此 8 位元稱之爲差值的位元組（difference byte），其結構如圖 1 所示。

在差值的位元組中，前四個位元代表低重建階，其中 X 爲代表正負號的位元，DDD 則爲低重建階的差值。同樣的，後四個位元代表高重建階，其中 Y 爲代表正負號的位元，HHH 則爲高重建階的差值。在差值的位元組的設計中，扣除表示正負號的位元之外，最多只能表示由二進位的 000 到 111 的訊息。換算成十進位數值表示來說，只能表示 0 到 7 的數值。因此爲了有效的增進壓縮效率，所以採取了量化表的設計，如表 1 所示。如此一來，每一重建階有 7 階的量化，和 1 個表示溢位（overflow）的情形。當低重建階（或高重建階）出現溢位時，編碼端必需重送原始的低重建階（或高重建階）的值。

圖 1. 差值的位元組（difference byte）結構示意圖

表 1. 最佳化 difference 量化表

Difference Code	0	1	2	3	4	5	6	7
Code	0	4	8	16	24	40	56	overflow

當解碼端接收到編碼端的訊息之後，第一個區塊依照 AMBTC 解碼方式重建影像。從第 2 個區塊開始，根據編碼端送出的差值的位元組，計算出本區塊的兩個重建階值，再配合位元圖進行影像區塊的重建。由作者提出的最佳化量化表量化後的結果，可大為減低溢位出現的機會，因此位元率可以有效的降低，甚至幾乎達到 MBTC 設計的理想值 1.5bits/pixel。但是也因為量化的影響，使得影像重建品質（peak signal to noise ratio, PSNR）會些許的降低。雖說如此，MBTC 確實能夠增進 AMBTC 的壓縮效能。

（三）植基於動態樹狀結構編碼法的無失真向量量化索引值壓縮法（Dynamic Tree Coding Scheme, DTCS）

在 2003 年，Chen 等人提出了以動態樹狀結構編碼的觀念 [2]，針對向量量化編碼端輸出的索引值再做進一步的無失真壓縮（dynamic tree-coding scheme, DTCS）。在第一節裡面我們提到，因為區塊之間有極大的空間相似性，這意味著區塊之間的量化索引值可能會相同或者差距相當小。圖 2 表示部分 VQ 編碼索引位址表，從圖裡我們可以很明顯發現到許多區塊的量化索引值都可在它周圍附近找到相同索引值。為了表示方便，我們令 B_c 及 I_c 分別代表目前將要編碼的區塊和它的量化索引值。在 DTCS 裡可分為兩大步驟：1. 找出 B_c 與周圍區塊之間關聯性，及 2. 根據區塊之間的相似性動態選擇不同編碼樹對 I_c 編碼。以下我們將簡潔的介紹有關 DTCS 的編碼步驟。

1. 找出 B_c 與周圍區塊之間關聯性

由於需要從周圍區塊間獲得有助於對 B_c 編碼的資訊，我們令 B_1 代表 B_c 左方的區塊 ，B_2 代表 B_c 左上方的區塊，B_3 代表 B_c 上方的區塊，和 B_4 代表 B_c 右上方的區塊，如圖 3 所示。每個編碼區塊中 B_i 都記錄著 4 個判斷區塊之間是否

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	134	134	150	3	134	11	19	11	11	126	174	24	40	150	82	81
1	134	3	134	11	11	76	11	11	11	134	174	94	94	150	93	81
2	3	19	3	68	11	19	76	11	52	126	102	94	102	4	117	81
3	3	11	3	11	11	11	11	11	134	174	102	110	118	68	117	81
4	11	68	11	3	67	11	28	134	158	110	158	158	150	35	53	89
5	11	11	3	3	3	11	11	134	174	158	158	150	134	35	77	97
6	68	11	3	3	3	3	3	158	174	110	126	134	134	19	77	177
7	3	3	3	3	134	134	158	110	102	150	134	3	134	19	77	81
8	3	3	3	134	134	126	110	110	110	1520	134	3	134	19	93	81
9	134	134	126	134	158	110	118	118	158	134	3	134	134	19	117	185
10	134	134	134	126	174	185	134	134	134	134	3	134	134	19	77	193
11	150	150	134	174	110	3	10	44	10	134	150	134	134	19	77	81
12	134	134	134	102	166	83	138	138	50	126	150	134	150	19	117	209
13	3	134	174	110	19	146	149	61	27	126	150	134	150	68	117	185
14	134	158	40	150	83	149	17	5	27	158	158	126	150	19	77	209
15	126	40	110	19	93	177	197	13	27	158	158	158	126	19	93	20

圖 2. 部分 VQ 編碼索引位址表

	B_2	B_3	B_4
	B_1	B_c	

圖 3. 區塊相對位置示意圖

相似的關係位元（relation bits），分別為 LE_i、LU_i、UP_i 和 RU_i，將在下列作一說明。

Left relation（LE）：如果目前即將編碼的區塊 B_c ，它的量化索引值 I_c 與 B_c 水平左方區塊 B_1 的量化索引值 I_1 相同，則令關係位元 LE_i=1，否則 LE_i=0。同樣的，LU_i、UP_i 和 RU_i 都可根據此規則來找出他們的關係位元值，因為規則與 LE_i 相當相似且簡單，所以我們在此省略。

2. 動態選擇不同編碼樹對 I_c 編碼

如圖 3 所示，區塊 B_1 是位於區塊 B_c 左方的區塊，所以我們只要考慮 B_1 所提供的水平左方的空間相似性資訊，也就是說在 B_1 記錄著四個關係位元分別為 LE_1、LU_1、UP_1 和 RU_1，但當我們要針對 I_c 編碼時，只要考慮 LE_1 這個關係位元就可。同樣的道理，根據所屬相對位置關係，對於 B_c 而言，我們只需考慮記錄在 B_2 的 LU_2，B_3 的 UP_3 和 B_4 的 RU_4。根據這 4 個關係位元，DTCS 定義了七種不同的編碼樹如表 2 所示。

整個 DTCS 編碼的演算法步驟如下：

步驟一：輸入即將編碼索引值 I_c。

步驟二：從 B_c 周圍區塊，分別找出 LE_1、LU_2、UP_3 和 RU_4 關係位元值。

規則一：如果 LE_1=1 和 UP_3=1，選擇用第一種編碼樹來對 I_c 編碼。

表 2. 七種不同編碼樹及它們的編碼字

Index / Tree No.	I1	I2	I3	I4	otherwise
Tree-1	01	0000	1	0001	“001”+the original indx I_c
Tree-2	1	0000	001	0001	“01”+the original indx I_c
Tree-3	001	0000	1	0001	“01”+the original indx I_c
Tree-4	0000	1	0001	01	“001”+the original indx I_c
Tree-5	0001	1	001	0000	“01”+the original indx I_c
Tree-6	0000	0001	001	1	“01”+the original indx I_c
Tree-7	001	0000	01	0001	“1”+the original indx I_c

規則二：如果 LE_1=1 和 UP_3=0，選擇用第二種編碼樹來對 I_c 編碼。

規則三：如果 LE_1=0 和 UP_3=1，選擇用第三種編碼樹來對 I_c 編碼。

規則四：如果 LE_1=0 和 UP_3=0，及 LU_2=1 和 RU_4=1，選擇用第四種編碼樹來對 I_c 編碼。

規則五：如果 LE_1=0 和 UP_3=0，及 LU_2=1 和 RU_4=0，選擇用第五種編碼樹來對 I_c 編碼。

規則六：如果 LE_1=0 和 UP_3=0，及 LU_2=0 和 RU_4=1，選擇用第六種編碼樹來對 I_c 編碼。

規則七：如果 LE_1=0 和 UP_3=0，及 LU_2=0 和 RU_4=0，選擇用第七種編碼樹來對 I_c 編碼。

步驟三：輸出編碼結果。

步驟四：重複步驟一到四，直到所有來源索引值都被編碼完成。

舉例來講，假如當 LE_1=0 和 UP_3=1 時，我們選擇用第三種編碼樹來對 I_c 編碼，當 $I_c = I_3$ 時，我們輸出 “1” 來替代目前編碼索引值 I_c，假如 $I_c = I_1$ 則輸出 “001”，假如 $I_c = I_2$ 則輸出 “0000”，假如 $I_c = I_4$ 則輸出 “0001”，如果 I_2、I_3 和 I_4 都不與 I_c 相等，則我們先送出 “01” 加上原來的索引值。此外在解碼端方面，由於解碼的演算法非常簡單且與編碼演算法相似，所以我們在此省略。

三、我們提出的影像編碼法

在前一節裡提到的動態樹狀結構編碼法中，我們發現到它具有計算複雜度低和只需要用到少量的記憶體的特性，如果能將動態樹狀編碼觀念應用在區塊預測技術上，不但可以有效的更進一步降低位元率，並且不會增加額外的龐大計算量。基於這個想法，我們提出了一個結合動態樹狀編碼法（DTCS）觀念和改良式區塊截短編碼的之影像壓縮技術。

一般而言，植基於 BTC 的區塊編碼程序是從左至右、從上到下，在我們所提出的技術亦然。如圖 2 所示，根據實驗結果發現，對於目前要編碼的區塊 B_c 而言，左方區塊 B_1 *和*上方區塊 B_3 所提供空間相似性的資訊貢獻會比左上方區塊 B_2 *和*右上方區塊 B_4 多。這代表說，我們可以利用在編碼處理過程中用不同的方法來對這兩種特徵作預測。因此我們提出的演算法可以分爲兩大部分：使用動態樹狀編碼法（DTCS）對 B_c 編碼和使用 Block Code Modified-BTC（MBTC）演算法對 B_c 編碼。以下將詳細的介紹我們新提出的區塊預測技術。

（一）使用動態樹狀編碼法（DTCS）觀念對 B_c 編碼

如之前所提到，對於 B_c 而言，B_1 和 B_3 提供的貢獻會比 B_2 和 B_4 多，因此我們每個區塊 B_i 只紀錄了兩個關係位元 LE_i 和 UP_i。然而在 DTCS 中，LE_i 和 UP_i 的值是根據相對應區塊的量化索引值決定，這跟區塊截短編碼壓縮技術有很大不同，因此我們應用了歐基里德距離的概念，我們分別利用公式（5）-（8）計算目前區塊 B_c 與 B_1、B_2、B_3 和 B_4 之間的歐基里德距離，及事先定義一個門檻值 TH_{RB}。因此應用於此區塊預測技術的關係位元 LE_i 和 UP_i，重新定義如下：

Left relatio（LE）：如果區塊 B_c 與 B_1 之間的歐基里德距離小於門檻值 TH_{RB}，則令 LE_i=1，否則爲 0。

Upper relation（UP）：如果區塊 B_c 與 B_3 之間的歐基里德距離小於門檻值 TH_{RB}，則令 UP_i=1，否則爲 0。

$$d(B_c, B_1) = (\sum_{i=1}^{k} B_{c\ i} - \sum_{i=1}^{k} B_{1\ i})^2 \qquad (5)$$

$$d(B_c, B_2) = (\sum_{i=1}^{k} B_{c\ i} - \sum_{i=1}^{k} B_{2\ i})^2 \qquad (6)$$

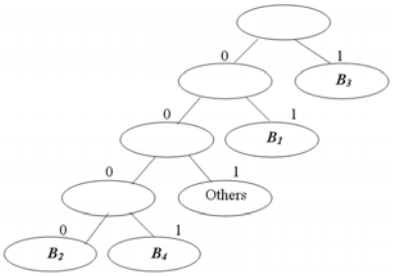
$$d(B_c, B_3) = (\sum_{i=1}^{k} B_{c\ i} - \sum_{i=1}^{k} B_{3\ i})^2 \qquad (7)$$

$$d(B_c, B_4) = (\sum_{i=1}^{k} B_{c\ i} - \sum_{i=1}^{k} B_{4\ i})^2 \qquad (8)$$

很明顯的，當 LE_1=1 或者 UP_3=1 時，表示目前即將編碼的區塊 B_c 和 B_1 或 B_3 具有極大的空間相似性關係，根據這兩個關係位元，我們只使用 DTCS 中的第一到第三種編碼樹。另外同樣的，我們也重新定義了第一到第三種編碼樹中的編碼對應關係如圖 4 到 6 所示，詳細說明如下：

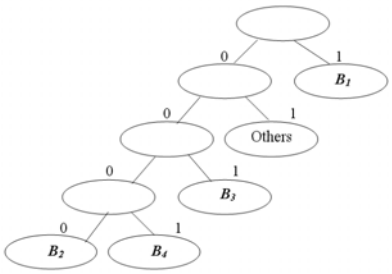
選擇任何一種編碼樹之後（以下以進入第一棵樹爲例），接著分別計算目前區塊 B_c 與 B_1、B_2、B_3 和 B_4 之間的歐基里德距離，判斷是否小於門檻值。若大於門檻值的距離關係則直接排除不考慮，其餘比較出小於門檻值之區塊中與 B_c 之距離爲最小之區塊，並以下列規則來編碼：

1. 如果 B_3 和 B_c 之間距離最小，用 1 來編碼 B_c。
2. 如果 B_1 和 B_c 之間距離最小，用 01 來編碼 B_c。
3. 如果 B_c 與 B_1、B_2、B_3 和 B_4 之間的歐基里德距離都大於門檻值，則用 001+MBTC 所送出差值的位元組和位元圖



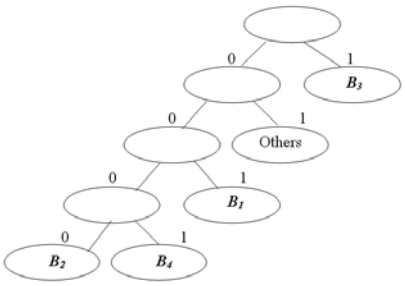
Current block	Codeword
B_3	1
B_1	01
otherwise	"001"+MBTC
B_2	0000
B_4	0001

圖 4. 重新定義的第一種編碼樹（LE_1=1 和 UP_3=1）和它相對應的編碼字



Current block	Codeword
B_1	1
otherwise	"01"+MBTC
B_3	001
B_2	0000
B_4	0001

圖 5. 重新定義的第二種編碼樹（LE_1=1 和 UP_3=0）和它相對應的編碼字



Current block	Codeword
B_3	1
otherwise	"01"+MBTC
B_1	001
B_2	0000
B_4	0001

圖 6. 重新定義的第三種編碼樹（LE_1=0 和 UP_3=1）和它相對應的編碼字

來編碼 B_c。

4. 如果 B_2 和 B_c 之間距離最小，用 0000 來編碼 B_c。
5. 如果 B_4 和 B_c 之間距離最小，用 0001 來編碼 B_c。

（二）使用 MBTC 對 B_c 編碼（LE_1=0 和 UP_3=0）

根據我們的實驗結果顯示，對於複雜影像而言，利用 DTCS 的第四到第七種編碼樹來做區塊預測的影像品質會較差，因此在這裡不使用它們。也就是說，對於每個編碼區塊 B_i 不需紀錄也不需判斷 LU_i 和 RU_i 兩個關係位元，我們改採用 2.2 節所提到的 MBTC 演算法來對 B_c 進行編碼，送出位元圖和差值的位元組。

圖 7 爲整個編碼處理的流程圖，而整個影像壓縮演算法步驟如下：

步驟一：輸入即將編碼的區塊 B_c。

步驟二：從 B_c 周圍區塊，分別找出 LE_1 和 UP_3 關係位元值。

規則一：如果 LE_1=1 和 UP_3=1，選擇用第一種編碼樹來對 B_c 編碼。

規則二：如果 LE_1=1 和 UP_3=0，選擇用第二種編碼樹來對 B_c 編碼。

規則三：如果 LE_1=0 和 UP_3=1，選擇用第三種編碼樹來對 B_c 編碼。

規則四：如果 LE_1=0 和 UP_3=0，選擇用 MBTC 來對 B_c 編碼。

步驟三：輸出編碼結果。

步驟四：重複步驟一到四，直到所有來源區塊都編碼完成。

舉個例子，當關係位元爲 LE_1=0 和 UP_3=0 時，即是利用 MBTC 來編碼 B_c。因爲此情形不屬於任一種樹，於是編碼端送出現正處理區塊的位元圖，以及該區塊中的兩個重建階與前一個區塊中的兩個重建階的差，並使用表 1 所示的量化表以減少溢位的發生。但是當出現溢位的時候，編碼端將隨即送出重建階的原始值。另外進入任何一種樹後，若發生相鄰區塊的歐基里得距離都大於門檻值，亦採用 MBTC 來處理。

在解碼端方面，解碼端接收到編碼端送出的編碼訊息之後，同樣地先求出該區塊的判斷關係位元 LE_1 和 UP_3 的值，以決定進入選擇哪種解碼方式，再根據該編碼樹結構的編碼原則進行解碼的工作。解碼時若找到對應的區塊，即用該區塊來取代目前處理的影像區塊。因爲是把對應到的區塊完全複製到目前處理的區塊位置上，因此編碼端不需另外送出位元圖，所以位元率會大幅降低，最少可用 1 位元即可重建整

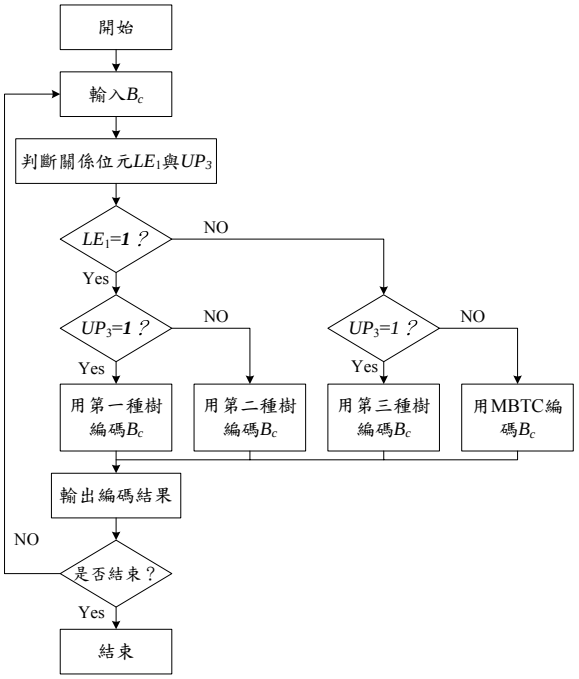


圖 7. 編碼流程圖

個區塊。

至於運用 MBTC 的解碼工作（LE_1 和 UP_3=0），解碼端接收到編碼端使用 MBTC 量化的訊息之後，根據編碼端所傳送目前處理區塊與前一區塊兩個重建階的差值，還原出原來兩個重建階的值，接著配合編碼端所送的位元圖，即可進行影像重建的工作；如果接收到的是溢位訊息，解碼端就會利用接著收到重建階的原始值，依據區塊截短編碼的影像重建程序進行影像重建的工作。

我們提出的這個方法，結合動態樹狀結構編碼的觀念及採用 MBTC 的方式，不但可以大幅降低編碼位元率，並且整個編碼、解碼程序非常簡單，並無牽涉到複雜的運算，乃具有不錯的整體效能表現。

四、實驗結果與分析

實驗用了五張標準的灰階測試影像如圖 8 所示，每一張影像原圖大小皆爲 512x512 像素，每張影像切割爲不重疊的區塊，區塊的大小爲 4x4。另外，也實作了 AMBTC、MBTC 和傳統 VQ 與我們提出的方法作一比較。

很明顯的，門檻值的設定將會影響我們方法重建影像的品質及壓縮的位元率，越高的門檻值位元率會越低，然而這可能會造成較低的重建影像品質。根據實驗結果顯示，從

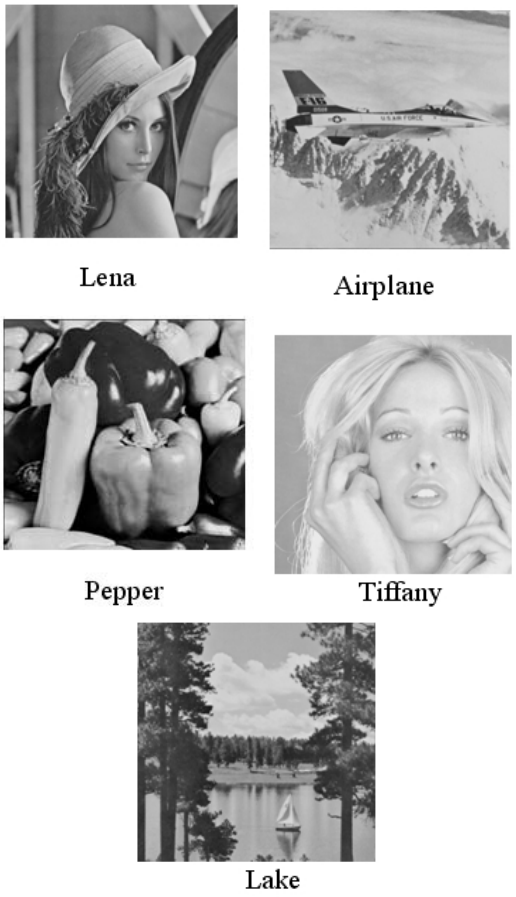


圖 8. 實驗用的五張測試影像

表 3 中我們可以發現到門檻值 TH_{RB}=4000，所有的測試影像幾乎都有 31dB 左右的影像品質，而平均位元率也有 0.716bits/pixe 的水準，雖然更高的門檻值 TH_{RB}=7000 時，仍有平均 30dB 以上的影像品質、平均位元率 0.619 bits/pixel 的表現，但基於須適應於多種不同複雜度影像前提之下，因此我們決定使用 TH_{RB}=4000 做爲區塊預測技術的門檻值。

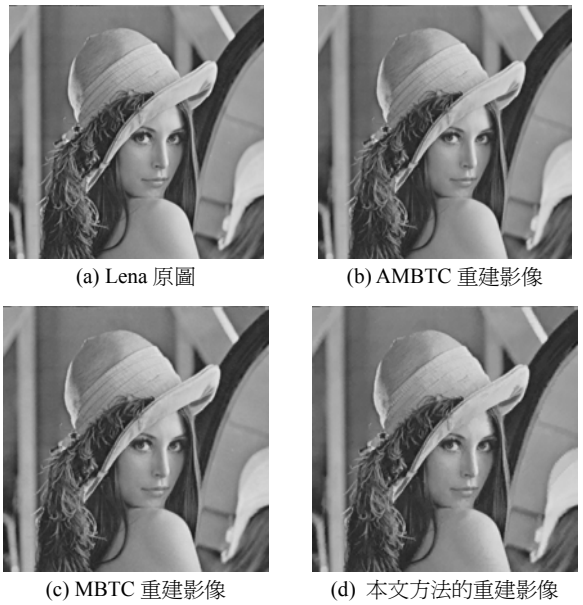
表 4 爲我們提出的方法與 AMBTC、MBTC、傳統 VQ 與 VQ 結合 DTCS [2] 比較，在此 VQ 的編碼簿大小爲 256，VQ 編碼後其索引值經過 DTCS 再做進一步的無失真壓縮其位元率可以從 0.5 降至平均 0.327。由表中可得知，雖然我們的方法（TH_{RB}=4000）比 AMBTC 和 MBTC 的重建品質平均值稍微低些，但我們的方法在平均位元率方面的比較則是大幅的降低。而且如圖 9 所示，我們將測試影像 Lena 原圖及分別經由 AMBTC、MBTC 和我們的方法重建過後的影像作比較，其中圖 9(b) AMBTC 重建影像的 PSNR=33.22dB，圖 9(c) MBTC 重建影像的 PSNR=32.69dB，而我們的方法重建影像的 PSNR=31.59dB 如圖 9(d)，可以發現雖有些微的重

表 3. 在不同的門檻值下重建品質與位元率關係

Images / TH_{RB}	Lena		Tiffany		Airplane		Lake		Pepper		Averaged	
	PSNR	BR	PSNR	BR	PSNR	BR	PSNR	BR	PSNR	BR	PSNR	BR
1000	32.33	0.938	31.80	1.097	31.41	0.872	29.38	1.158	32.45	0.998	31.47	1.013
1500	32.22	0.848	31.70	1.000	31.06	0.801	29.27	1.058	32.00	0.888	31.25	0.919
2000	32.00	0.733	31.50	0.933	31.06	0.751	29.21	0.996	31.90	0.816	31.13	0.846
2500	31.90	0.774	31.41	0.876	30.97	0.715	29.15	0.949	31.80	0.76	31.04	0.815
3000	31.80	0.700	31.32	0.839	30.89	0.691	29.1	0.915	31.80	0.725	30.98	0.774
3500	31.70	0.667	31.23	0.798	30.89	0.667	29.05	0.882	31.70	0.687	30.91	0.740
4000	31.60	0.642	31.06	0.769	30.81	0.652	28.99	0.856	31.60	0.659	30.81	0.716
4500	31.41	0.621	30.97	0.742	30.73	0.637	28.94	0.834	31.50	0.635	30.71	0.694
5000	31.41	0.605	30.89	0.720	30.65	0.622	28.89	0.818	31.41	0.618	30.65	0.677
5500	31.23	0.586	30.81	0.694	30.65	0.609	28.84	0.818	31.32	0.598	30.56	0.661
6000	31.14	0.573	30.65	0.676	30.57	0.600	28.79	0.784	31.32	0.583	30.49	0.643
6500	31.06	0.559	30.57	0.66	30.50	0.587	28.69	0.769	31.23	0.571	30.40	0.629
7000	30.97	0.547	30.5	0.647	30.42	0.578	28.64	0.769	31.14	0.558	30.33	0.619

表 4. 我們提出的方法與 AMBTC、MBTC 和傳統 VQ 比較

Method / Images	Proposed (TH_{RD}=4000)		AMBTC		MBTC		VQ (Cookbook size=256)		VQ+DTCS [2]	
	PSNR	BR	PSNR	BR	PSNR	BR	PSNR	BR	PSNR	BR
Lena	31.59	0.642	33.22	2.000	32.69	1.517	30.91	0.500	30.91	0.363
Tiffany	31.06	0.769	32.57	2.000	31.83	1.524	30.13	0.500	30.13	0.242
Airplane	30.81	0.652	32.00	2.000	31.50	1.556	29.24	0.500	29.24	0.313
Lake	28.99	0.856	29.93	2.000	29.56	1.567	27.97	0.500	27.97	0.366
Pepper	31.59	0.659	33.51	2.000	32.81	1.537	30.12	0.500	30.12	0.353
Averaged	30.81	0.716	32.64	2.000	31.67	1.540	29.67	0.500	29.67	0.327



(a) Lena 原圖　(b) AMBTC 重建影像

(c) MBTC 重建影像　(d) 本文方法的重建影像

圖 9. 本文方法與其他方法的重建影像比較

建品質降低但是並不容易被察覺出來，且位元率降到 0.642 bits/pixel。另外當我們的方法將 TH_{RB} 設爲 1000 時（如表 3 所示），平均重建品質和位元率分別爲 31.47dB 及 1.013 bits/pixel，位元率仍小於 AMBTC 和 MBTC。另一方面，雖然我們方法的平均位元率稍微高於傳統 VQ，然而對於重建品質和複雜度而言，我們的方法仍優於傳統 VQ 許多。由以上實驗結果顯示，我們提出的方法不但能大幅降低位元率，而且仍能保有不錯的重建影像品質，相對於其他壓縮方法的計算量，更顯得具有相當不錯的效率。

五、結論

在本篇論文中，我們提出了一個植基於區塊截短編碼的影像壓縮技術。我們利用了原本用在無失真向量量化索引值編碼上的動態樹狀結構編碼法（DTCS）的觀念，將它應用在區塊預測技術上，及結合了一種改良的區塊截短編碼

Block Code Modified-BTC（MBTC）技術，不但達成了相當高的壓縮率效果外，重建品質亦相當不錯，由於我們掌握了區塊與區塊之間的空間相似性，巧妙的運用動態樹狀編碼的觀念來做區塊之間的預測，且根據目前要編碼的區塊它的左方區塊和上方區塊所提供空間相似性的資訊來判斷選擇不同的編碼方法，因此相較於其他影像壓縮方法下，具有相當簡單的特色和較高的效率。

參考文獻

1. Alcaim A. and L. V. Oliverira (1992) Vector quantization of the side information in BTC image coding. Singapore International Conference on Communication Systems, Singapore.
2. Chen, P. Y. and R. D. Chen (2003) An index coding algorithm for image vector quantization. *IEEE Transactions on Consumer Electronics*, 49(4), 1513-1520.
3. Delp, E. J. and O. R. Mitchell (1979) Image coding using block truncation coding. *IEEE Transactions on Communications*, 27(9), 1335- 1342.
4. Gray, R. M. and A. Gersho (1992) *Vector Quantization and Signal Compression*, Kluwer, Norwell, MA.
5. Lema, M. D. and O. R. Mitchell (1984) Absolute moment block truncation coding and its application to color image. *IEEE Transactions on Communications*, 32(10), 1148-1157.
6. Linde, Y., A. Buzo and R. M. Gray (1980) An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1), 84-95.
7. Nasiopoulos, P., R. K. Ward and D. J. Morse (1991) Adaptive compression coding. *IEEE Transactions on Communications*, 39(8), 1245-1254.
8. Nasrabadi, N. M. and R. B. King (1988) Image coding using vector quantization: A review. *IEEE Transactions on Communications*, 36(8), 957-971.
9. Ramana Rao, Y. V. and C. Eswaran (1995) A new algorithm for BTC bit plane coding. *IEEE Transactions on Communications*, 43(6), 2010-2011.
10. Rozinaj, G., S. Herreraand and J. Mikula (2002) Modified BTC algorithm for satellite image coding. 4th EURASIP-IEEE Region 8 International Symposium on Video/Image Processing and Multimedia Communications, Zadar, Croatia.
11. Udpikar, V. R. and J. P. Raina (1987) BTC image coding using vector quantization. *IEEE Transactions on Communications*, 35(3), 352 -356.

收件：96.04.03　修正：96.05.10　接受：96.07.17