一個基於空間與時間關聯性的 H.264 模式選擇演算法

梁廷宇
國立高雄應用科技大學電機工程學系
lty@mail.ee.kuas.edu.tw

黃竟閔
國立高雄應用科技大學電機工程學系
diohwang@hpds.ee.kuas.edu.tw

摘要

本篇論文設計與實作了一個基於空間與時間關聯性的快速 H.264 模式選擇演算法，以減少編碼器選擇切割模式的時間。基本上，此方法原理乃是從過去的模式選擇樣本中，探勘出巨集方塊之模式選擇在時間上與空間上的關聯性，藉以預測目標巨集方塊可能的切割模式，而不對所有可能的模式進行評估。為了提升預測的準確度，此方法會動態地調整模式選擇的關聯性規則，並且定期清除過時的樣本與考慮物件移動的區域性。另一方面，當預測模式之信心指數不夠時，此方法會從預測模式以及與目標方塊相鄰之方塊的切割模式中選出一個最佳的解以降低預測錯誤的機率。我們已經將此演算法應用於實際的 H.264 編碼器並且進行實際的效能測試。實驗結果顯示我們所提出的演算法和完全搜尋方法比較起來，確實能有效大幅減少模式選擇的時間，並且維持相似畫面品質。

關鍵詞：H.264、模式選擇、空間與時間的關聯法則、移動的區域性

1. 簡介

近來國際視訊組織已針對不同環境的需求訂定出各種技術標準，例如：MPEG-1[2]、MPEG-2[3]、MPEG-4[4]、MPEG-7[5]、H.261[6]、H.263[7]、H.263+[8]、H.264[9][15]等，而其中以 H.264 最受矚目。和目前最常用的 MPEG-4 比較起來，H.264 不僅具有較優異的壓縮性能，同時也具有良好的網路親和性，而且編碼後的位元數也較 MPEG-4 少了一半左右。由此可見，H.264 無論在視訊品質與網路的可靠性上都具有相當的優勢。然而，雖然 H.264 具有眾多優勢，但其複雜度卻比 MPEG-4 複雜 10 幾倍。主要是因為 H.264 在編碼過程中需對為每一巨集方塊選擇出最佳的切割模式，而其過程需對所有可選擇的模式進行動作估計與補償，加上可選擇的切割模式又有很多種，所以整個編碼過程所需執行的計算量變得相當龐大。故對於 H.264 編碼器的效能而言，如何有效降低模式選擇的時間代價是一個非常重要的研究課題。

針對這個問題，本研究將切割模式分為十六種(如圖一所示)，並使用完全搜尋的方法對不同的視訊包括 container 和 football 進行編碼，然後紀錄每個畫面裡每個巨集方塊所選擇的模式，並且去統計每個巨集方塊與其左邊相鄰巨集方塊之模式配對樣本(pattern of mode pair) 出現次數，以及目標巨集方塊與其在前一個畫面的模式配對樣本出現次數。

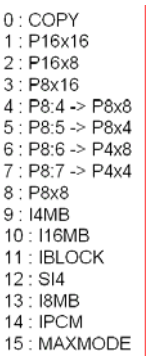

圖 1 切割模式的種類

以 container 為例，影像中唯一運動的物體只有兩艘貨輪在跑動，背景部份在畫面中大部份呈現的是為靜態的狀態，故其結果如圖(2)和圖(3)所示顯示其選擇的切割尺寸大小模式為 SKIP 模式較為眾多。

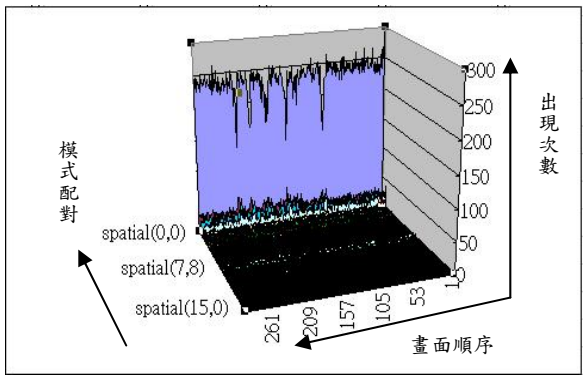

圖 2. Container 之模式選擇於空間上的關聯性

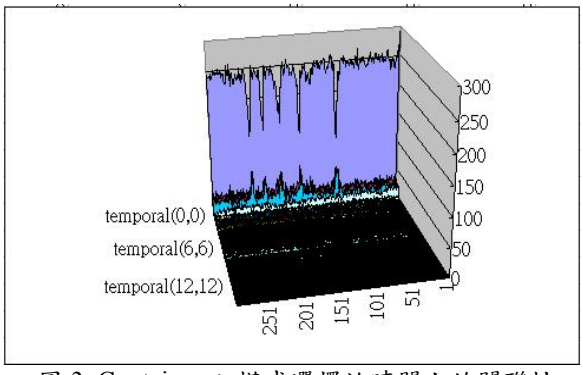

圖 3. Container 之模式選擇於時間上的關聯性

相對地，影像 football 則是呈現眾多美式足球員四處跑動並爭搶橄欖球的畫面，大部份選擇的切割尺寸是比較小，所以圖(4)、圖(5)顯示配對 spatial(8,8)與 temporal(8,8)在每一畫面都比其他配對的多。

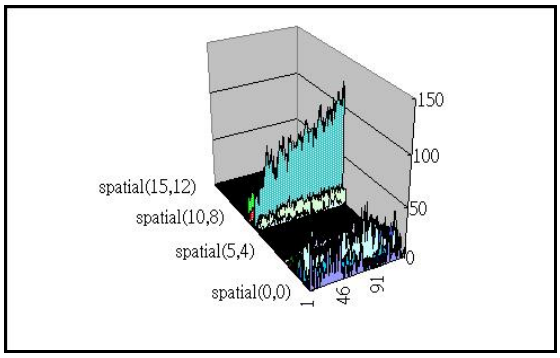

圖 4. Football 之模式選擇於空間上的關聯性

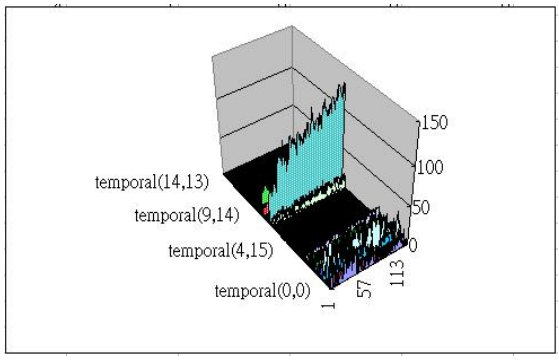

圖 5. Football 之模式選擇於空間上的關聯性

從上述的實驗中，可觀察出不管是影像是靜態還是動態，其模式配對樣本的出現次數都集中在少數幾種，而且選擇相同模式的機率很高，因此可以證明切割模式的選擇在空間和時間上確實都具有很大的關聯性。如果可以從過去的模式選擇樣本中找出可信的關聯規則，那麼就不用對每個模式進行評估，可以很快地選擇出合適切割模式，進而提升編碼的速度。而要達到這個目的，可以運算資料探勘（Data Mining）的關聯性法則（Association Rules）[14]。

簡單地說，關聯規則即是在 A 事件發生底下同時 B 事件亦會發生的條件機率。此規則過去經常用於市場分析與決策之中，其探勘過程會使用到支持度和信心度兩個數值，其定義如下：

(1)支持度(Support): 為所有交易記錄(transaction)之中，出現項目 A（item）百分比，其計算公式如下:

$$s = \frac{N_A}{N} \qquad (1)$$

式(1)中 N代表交易紀錄的總數，而 N_A代表出現項目A的交易數目。

(2)信心度(Confidence): 為所有出現項目 A 的交易中，也同時出現項目 B 的百分比機率，其計算公式如下:

$$c = \frac{s_{A\cap B}}{s_A} = \frac{N_{A\cap B}}{N_A} \qquad (2)$$

式(2)中 $N_{A\cap B}$ 代表交易記錄中，同時出現項目 A 與項目 B 的交易數目。

至於關聯性規則的探勘步驟，則如下所述：首先針對單一項目的候選樣式（candidate pattern）進行支持度的統計，然後篩選出支持度超過臨界值的樣式（亦稱為經常樣式，frequent pattern），接著將篩選出的經常樣式配對出長度為 2(即包含兩個項目)的候選樣式，然後再計算這些長度為 2 之候選樣式的支持度，並再次篩選出支持度超過臨界值的經常樣式，接著從這些樣式中繼續配對產生長度為 3 的候選樣式，重複相同的步驟直至無法再產生長度更長的候選樣式為止。接著，即可利用篩選出的經常樣式去推算出在經常樣式 A 出現下經常 B 樣式也會跟著出現的信心度。例如：在所有的交易中，有 80%買光碟機，有 60%會同時買光碟機與光碟片，則買了光機後也會買光碟片的機率即為 75%。如果此信心度已超過設定的臨界值，即可推斷"買光碟機後必買光碟片"的關聯規則。為了解決 H.264 的模式選擇問題，可將前面實驗之模式配對的數目視為交易的數目，而將目標巨集方塊的模式與其左邊相鄰巨集方塊的模式，或者將目標巨集方塊的模式與其在前一個畫面的模式視為交易的項目，透過相同的資料探勘過程即可挖掘出有用的空間關聯規則與時間關聯規則，以作為模式選擇的依據，進而達到縮短選擇切割模式的時間。

根據前面所述，本篇論文之目的即在研究開發一個基於空間與時間關聯性規則（Spatial and Temporal Association Rules，STAR）之 H.264 模式選擇演算法。為了提升關聯規則的準確度，此演算法會一直動態更新樣式的支持度，並且週期性地將過時統計結果清除與重新統計樣式的出現次數。另一方面，由於考慮物體在空間上移動的區域性，此演算法將畫面分成四個區域（sections），並且分別針對各個區域進行模式選擇樣本的收集與關聯規則的探勘，然後用各個區域的關聯規則去對屬於該區域的巨集方塊進行模式選擇。

在簡單描述本研究之動機與目的後，第二章將討論 H.264 模式選擇的相關研究。第三章則介紹我們所提出的 STAR 模式選擇演算法。第四章則是 STAR 演算法的效能評估與討論。最後，第五章則為本論文的結論與未來工作方向。

2. 相關研究

近年來，許多研究學者提出不少快速的方塊切割模式決策演算法。在[10]論文中，作者採用了完全零係數方塊(All Zero Coefficients Block, AZCB)來快速去除不必要選擇的切割模式，然後對剩餘的模式進行 RD cost 的評估，並選擇 RD cost 最小的模式作目標巨集方塊（target macro-blocks）。實驗結果顯示此演算法所需的時間只有完全搜尋法的一

半，而且 PSNR 值相差只有 0.011~0.02dB 左右。

在[11]的研究所提出的演算法則是先用 RD cost 來評估所有的外部模式（inter-modes），並令 RD cost 最小者為最佳外部模式。然後計算出用最佳外部模式對目標巨集方塊編碼之運動補償的殘餘資料產生所需編碼的平均位元率（Average Rate，AR），以及目標巨集方塊與相鄰巨方塊的平均邊界像素錯誤值（Average Boundary Error，ABE）。如果 ABE 大於 AR 則最佳外部模式即為最佳方塊切割模式。否則就必須再對所有的內部模式（intra-modes）進行評估，然後再從內部模式中選擇最佳的切割模式。經由測試結果顯示此演算法的執行時間比完全搜尋演算法少了 30%，且 PSNR 品質卻只有相差 0.03dB。

在文獻[12]的演算法則依據巨集方塊在連續畫面之間的平均絕對差值(Mean Absolute Difference，MAD)以及平均絕對畫面差值（Mean Absolute Frame Difference，MAFD）為準則來快速選擇合適的切割模式。如果絕對差值很小的話，表示這巨方塊為一平坦巨方塊，因此只要選擇較大切割模式進行運動估計即可。反之，則表示巨方塊具有激烈的運動特徵，所以需要多選擇更小尺寸的切割模式達到更佳的位元失真補償。實驗數據顯示此方法節省的時間最高可達到 48%。

在[13]研究中，作者所提之切割模式選擇演算法類似二分搜尋法。首先，分別計算 SKIP 以及 P16x16 的 RD cost。如果前者較小則選擇 SKIP 模式，不然就再評估 P8x8 的 RD cost。相同地，如果 P16x16 比 P8x8 低且小於臨界值就選擇 P16x16，不然就再評估 P8x16 及 P16x8，且重選 RD cost 較小的模式。相反地，如果 P16x16 比 P8x8 大，就必須再估計 P4x4。如果 P4x4 的 RD cost 比 P8x8 大就選擇 P8x8，不然就是評估 P8x4 及 P4x8，並將 RD cost 最小的模式當作最後的選擇。整體演算法訂出了一臨界值以提早與減少選擇的次數。另外，此演算法在運動估計方面使用了鑽石搜尋法。實驗結果此演算法比完全搜尋法快了 4 倍之多。

上述的演算法中大都會設立一些條件以去除不必要的模式評估次數，而評估次數通常會隨著畫面物體與背景移動的激烈程度而增加。相對地，本研究所提出的演算法是從模式選擇樣本去探勘出可信的空間與時間的關聯性規則，來作為選擇切割模式的依據，故評估次數比較不受畫面動態程度的影響。

3. STAR 模式選擇演算法

3.1 考量因素

在關聯性規則的探勘前，必須累積一定數量樣本。其實作方法可在連續幾個畫面，使用完全搜尋來進行切割模式的選擇並將結果紀錄與統計，接著即可依左邊相鄰（或前一畫面相同位置）之巨集方塊的模式來找出信心度最高的關聯性規則，以作為未來決定目標巨集方塊之切割模式的參考。

由於關聯性規則的正確性程度取決於所採樣的樣本是否具有代表性。故運用關聯性規則於 H.264 切割模式選擇時必須考慮兩個因素。一是樣本採樣的週期，二是畫面分區。

第一個考量因素是基於時間上的區域性。當畫面的背景或前景內容有所改變時，之前所紀錄的樣本已經過時。故先前所挖掘的關聯性規則亦將不符合未來預測切割模式的需求。解決此一問題必須定期地將過時的樣本統計予以清除歸零，並重新採樣統計。意即在連續的幾個畫面中再度使用完全搜尋方法來選擇切割模式，並統計選擇樣式出現次數。由於使用完全模式的代價很高，若經常重新採樣則會危及到編碼的速度。反之，若重新採樣的週期過長，則模式選擇樣式之統計結果將無法反映出最近畫面的特性。故必須適當地決定一個適當的樣本採樣週期。

至於第二個考慮因素則是基於空間上的區域性。當畫面中物體只有在某一特定區域中運動，而其他的區域則是屬於靜態的背景居多時（例如：container）。如果持續的使用整張畫面的模式採樣統計結果來預測目標巨集方塊的模式時，則會選擇出不恰當的切割模式。在圖（6）中，紅色橢圓形部份表示巨方塊中運動較激烈的部份，藍色則表示平坦且運動較為小的部份。因此在進行巨集方塊 B 與 D 的尺寸大小切割模式選擇的時候，如果分別利用巨集方塊 A 與 C 的採樣結果來進行預測則會容易發生預測錯誤的情形。以 B 巨集方塊來說，如果利用巨集方塊 A 採樣結果來進行預測，則很有可能會預測出 16X8、8X16、或 8X8 幾種可能的切割模式出現。但其實巨集方塊 B 只需在 SKIP 或 16X16 幾種可能的切割模式中選擇即可。為了避免這種錯誤的情況發生，對影像畫面進行分區取樣與預測是有其必要性的。然而，分區的數目實為一個需要小心決定的問題。若分區太少，則無法反映出物體在空間位置的區域性。反之，若畫面切割的越細，則必須付出更大的記憶體空間的代價來存放樣本統計的結果，而且越容易發生一個物體橫跨不同分區的情形。

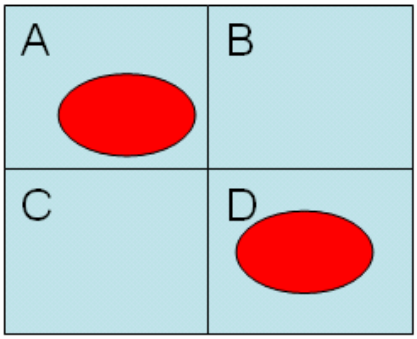

圖 6. 巨方塊的區域性

3.2 模式選擇演算法

圖 7.為運用 STAR 模式選擇演算法的編碼流程。首先，在前兩個畫面裡使用完全搜尋演算法來決定切割模式，並按照先前實驗的 16 種模式分類將選擇樣式予以統計並記錄於支持表中。例如：假設目標巨集方塊的模式為 P8x8 (MODE 8) ,與之左邊相鄰的巨集方塊為 P16x16(MODE 1)，目標方塊在前一畫面的模式為 SKIP（MODE 0)，則空間樣式支持表[1][8]這個欄位、時間樣式支持表[0][8]以及空間與時間樣式支持表[1][0][8]都要加 1。

從第三個畫面開始，即開始使用關聯性規則來預測目標巨集方塊的切割模式。每個方塊的模式決定後，會依照其結果將前述的三個表格予以更新，並繼續下一個方塊模式的預測與選擇直至此畫面的最後一個方塊編碼為止。當要繼續進行下個畫面的編碼時，會先檢查是否已經超過了畫面數目限制 ξ。如果是，則將前述三個支持表的統計結果予以清除，然後在接下來的兩個畫面內再度使用完全搜尋法來選擇切割模式，並對模式選擇樣式進行統計。否則就繼續使用關聯規則來預測方塊的切割模式。

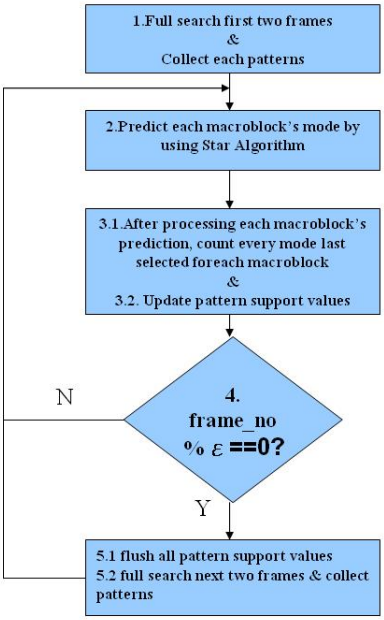

圖 7. 使用 STAR 演算法之編碼流程圖

另外，根據3.1節所述，STAR演算法採用了2X2分區大小進行分區採樣與預測。這是因為對於QCIF影像來說，採用4X4的分區模式會太細。相對地，如此的分區模式也意味著前述的三個支持表會有四組。

STAR模式選擇演算法如圖(8)。首先，利用左邊巨集方塊的模式（left mode）在空間支持表中找出支持度最高的模式配對樣式[left mode][target mode]，並檢查其支持度與信心度是否同時高過臨界值 α 和 β。如果是，則目標巨集方塊的模式即預測為此配對樣式之target mode。若是支持度超過 α 而信心度沒超過 β，則從空間與時間支持表中查出支持度最高的模式配對樣式[left mode][previous mode][target mode]，並檢查此樣式的支持度與信心度是否超過臨界值 γ 和 δ。如果是，則目標巨集方塊的模式即預測為此樣式的target mode。如果支持度超過 γ 而信心度沒超過 δ 則預測為left mode。相對地，如果在空間與時間支持表中找不到支持度超過 γ 的配對樣式，即預測目標方塊的模式為P8x8。

另一方面，如果一開始在空間支持表中就找不到支持度超過α的配對樣式，就使用目標巨集方塊在前一畫面的模式（previous mode）去搜尋時間支持表裡支持度最高的模式配對樣式[previous mode][target mode]，如果其支持度和信心度超過μ和ν，目標巨集方塊的模式即預測為此樣式的target mode。如果支持度超過μ但信心度沒超過ν，目標方塊就預測為previous mode。最後，如果兩者都沒超過臨界值，就預測為目標方塊的模式為P8x8。

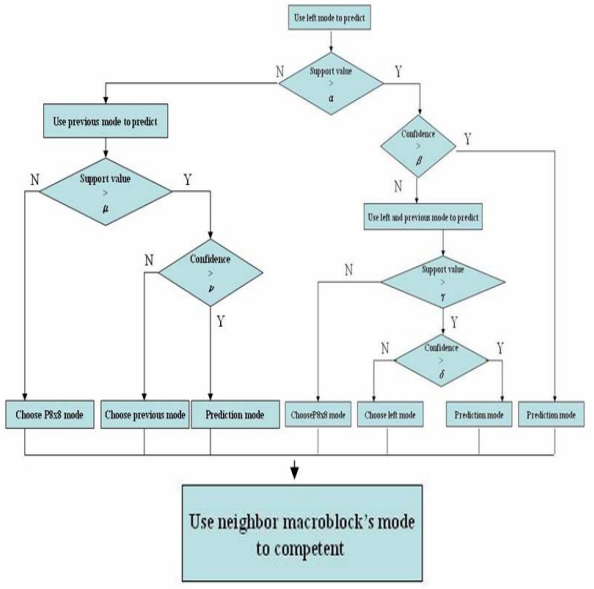

圖8. STAR演算法流程圖

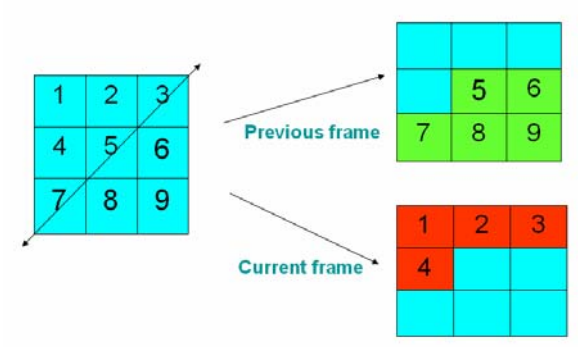

圖9. 使用相鄰巨方塊選擇的切割模式進行補償

當使用關聯規則得到一個預測模式（predicted mode）後，如果其信心度夠高的話，則此預測模式為最佳的可能性很高。然而，其預測錯誤的機會還是有，而且一旦預測錯誤則可能造成後面連續的巨集方塊都會預測錯誤，直至另一個區域的邊界為

止。為了避免這種情形發生，進行其他模式的評估是有其必要的。根據[1]的研究以及我們實驗結果顯示目標巨集方塊與相鄰巨集方塊具有相同切割模式的機率是很高的。因此，STAR 演算法會在空間與時間上參考相鄰巨集方塊的模式選擇進行補償，如圖(9)所示。編號 1～4 方塊為目前畫面的巨集方塊，編號 5～9 為前一畫面的巨集方塊。前者是取空間上的關聯性，後者則是取時間的關聯性。如果對此九個方塊的模式一一地評估將會耗費很多時間，故 STAR 演算法只取此九個方塊中出現次數最多的兩種來評估。加上前面的預測模式，STAR 演算法最少需要二次，最多需要三次 RD cost 評估即可決定目標巨集方塊的模式。

4. 效能評估

本篇論文，我們已經將 STAR 演算法應用於實際的 H.264 編碼器[16]，並且進行實際的效能評估。其測試環境與編碼器之參數則分別如表一與表二所示。

表 1 測試平台環境

Test bed	
CPU	Intel Pentium Ⅲ 1.066G
Memory	256 RAM
OS	Windows XP Home Edition
Compiler	Visual C++

表 2 編碼器參數表

Parameter Name	Value
Frame Rate	30 frame/sec
Quantization Parameter	28
Search Range	32
Reference Frame Number	5
RD-Optimization	Used
Entropy Coding	CAVLC
Fast Motion Estimation	Used
GOP Type	IPPP
Hadamard transform	Used

表 3 實驗測試影像樣本

測試樣本 (Image sequence)	解析度 (Frame size)	張數 (Length)
container	352x288	300
foreman	352x288	300
tempete	352x288	260
football	352x240	125
salesman	176x144	300
stefan	176x144	300

在效能評估中，一共對表三所列之 6 個不同的視訊進行測試模式選擇的測試。在實驗過程中，分別使用完全搜尋演算法、適應性的方塊切割模式決策演算法(**AMD**) [10]以及 STAR 演算法，然後比較這不同三種演算法的效能。其評估的標準包含：PSNR 值、壓縮時間、以及 bit-rate。關於 STAR 演算法所使用的參數值則如表 4 所設定。

表 4 演算法參數

Parameter	Parameter value
α	0.25
β	0.8
γ	0.5
δ	0.6
μ	0.25
ν	0.8
ε	120

4.1 STAR 演算法的有效性

就時間而言，STAR 演算法和完全搜尋法比較起來，節省了 40~65%的時間，如圖(5)所示。和在 AMD 比較起來，也快了 18~640 秒。其主要原因在於本論文所採用的方法在選擇切割模式時最多只做了二到三次切割模式的選擇，而 AMD 演算法會因為巨集方塊運動激烈程度高低而會多測試其他更小尺寸的切割模式，因此 STAR 演算法執行速度比其他兩種快。只有 **salesman** 是例外，時間較 AMD 演算法多了 13 秒之多。其原因在於 STAR 演算法選擇了較多 P8X8 的大小切割模式，所以執行時間較 AMD 演算法來的長。

就位元編碼來說，AMD 花費在所需夾帶的編碼位元率上都超出完全搜尋法太多。雖然 AMD 演算法能夠有效減少選擇切割模式所花的時間，但在傳送壓縮後的視訊編碼位元組時，其所需的傳輸時間必會大大增加。相反地，STAR 演算法所產生的編碼位元組較 AMD 少，尤其對 container 和 saleman 最為明顯。

最後就 PSNR 值而言，使用 AMD 演算法選擇與編碼後的視訊畫面品質最佳，比完全搜尋的方法要好上 0.25~0.53dB。雖然完全搜尋法會選 RD-cost 最小的切割模式，但是所 RD-cost 包含失真代價和位元率代價，故所選擇的模式不見得對畫面品質是最好的。相反地，AMD 經常選擇內部模式（intra modes）所以獲得畫面品質要比完全搜尋好，不過也付出了更多的位元率代價。相對地，STAR 演算法在 PSNR 值幾乎和完全搜尋法表現的一樣。

綜合以上的結果，可以看出本研究所提出的 STAR 演算法雖然較完全搜尋法付出多一些的位元率代價，卻可以大幅度地降低編碼的時間，並維持幾乎相同的畫面品質。和 AMD 演算法比較，STAR 演

算法不僅使得編碼速度加快，而且也降低了編碼所需的位元數。

表 5 三種切割模式選擇演算法效能之比較

(a)

	PSNR			DPSNR	
	Full	AMD	STAR	AMD	STAR
container	35.645	36.178	35.552	0.533	-0.093
fb	33.401	33.732	33.41	0.331	0.009
foreman	36.276	36.747	36.156	0.471	-0.12
salesman	35.522	35.837	35.505	0.315	-0.017
stefan	34.046	34.298	34.045	0.252	-0.001
tempete	34.869	35.304	34.842	0.435	-0.027

(b)

	Bit-rate			DBR(%)	
	Full	AMD	STAR	AMD	STAR
container	168154	221667	178680	31.8238	6.259738
fb	1818142	1865441	1859150	2.601502	2.255489
foreman	412049	466523	452856	13.22027	9.903434
salesman	46099	50942	48216	10.50565	4.592291
stefan	564545	574210	574547	1.711998	1.771692
tempete	1111672	1194782	1157839	7.476126	4.152934

(c)

	Time			DT(%)	
	Full	AMD	STAR	AMD	STAR
container	1645.551	894.704	565.971	45.62891	65.60599
fb	1372.029	993.255	821.008	27.60685	40.16103
foreman	2556.69	1572.824	930.355	38.48202	63.61096
salesman	394.399	184.163	203.5	53.30541	48.40251
stefan	934.963	668.576	565.738	28.49172	39.49087
tempete	2482.18	1743.626	1308.818	29.75425	47.27143

表 6. 採樣週期對 STAR 演算法的影響

(a)

	PSNR		
	30 frame length	60 frame length	120 frame length
container	35.647	34.695	33.829
fb	33.451	33.45	33.454
foreman	36.484	36.493	36.491
salesman	34.731	33.552	32.082
stefan	34.068	34.053	34.055
tempete	34.981	34.991	34.991

(b)

	Bit-rate		
	30 frame length	60 frame length	120 frame length
container	337700	315018	274298
fb	1944943	1947327	1952959
foreman	606958	615123	617394
salesman	74504	67545	59449
stefan	592459	592291	593255
tempete	1274482	1281467	1284220

(c)

	Time		
	30 frame length	60 frame length	120 frame length
container	964.76	867.907	747.199
fb	761.25	742.119	729.219
foreman	1578.646	1538.665	1525.228
salesman	237.168	211.775	189.47
stefan	552.079	548.012	532.966
tempete	1459.112	1423.133	1406.559

4.2 採樣週期對 STAR 演算法的影響

為了得知採樣週期對使用關聯規則預測切割模式的影響。本研究在編碼過程中分別採用 30 張、60 張、120 張之採樣週期，搭配 STAR 演算法之模式預測部分（不進行預測後的補償）來對所有視訊樣本進行測試。表 6 的結果顯示，採樣週期越短可得到較高的畫面品質。相對地所需花費的運算時間就越長。這是因為週期越短，採到的模式選擇樣本越具有代表性，所挖掘出的關聯性規則也就越準確。然而，執行完全搜尋的畫面數目越多，故編碼的時間也就越長。至於位元率，通常隨著畫面品質的好壞而呈反向的增加或減少。

前面的實驗中，所選擇的採樣週期為 120 張畫面。其原因在於採用 30 張畫面的週期雖然可得到的畫面品質極為優良，但執行時間太長。由於本研究以節省編碼時間為首要目標，所以採用了 120 畫面的採樣週期。至於，畫面的品質則由預測後的補償機制來對影像畫面品質做改善。

4.3 畫面分區對 STAR 演算法的影響

除了採樣週期外，本研究亦將畫面分區對關聯性規則的影響作了評估，其結果如表 7 所示。在 container 和 salesman 中，由於物體的移動都具有相當程度的空間區域性，故分區採樣與預測確實可以有效地改善畫面品質。

相反地，football、foreman 以及 tempete，使用畫面分區方法所獲得的畫面品質卻比不分區的方法。其原因在於這些視訊裡的物體其移動激烈且不規則，移動的區域經常改變，而且改變的速度比演算法學習的速度還快很多，因此採用分區的關聯性規則來預測，反而會選到不合適的切割模式。在位元率方面，則與 4.2 的實驗一樣，畫面品質呈現反向的變化，亦即畫質好所需的位元數也高。至於執行時間上，由於對於分區邊界的巨集方塊都會採用完全搜尋，故分區的執行時間在所有的視訊中都要比不分區來得長。

表 7. 畫面分區對 STAR 演算法的影響

(a)

	PSNR		DPSNR
	Reigon	No reigon	Reigon - No reigon
container	33.829	31.589	2.24
fb	33.454	33.462	-0.008
foreman	36.491	36.504	-0.013
salesman	32.082	31.55	0.532
stefan	34.055	34.136	-0.081
tempete	34.991	34.991	0

(b)

	Bit-rate		DBR
	Reigon	No reigon	Reigon - No reigon
container	274298	250726	23572
fb	1952959	1953631	-672
foreman	617394	617962	-568
salesman	59449	56896	2553
stefan	593255	593698	-443
tempete	1284220	1284220	0

(c)

	Time		DT
	Reigon	No reigon	Reigon - No reigon
container	747.199	658.169	89.03
fb	729.219	706.268	22.951
foreman	1525.228	1467.552	57.676
salesman	189.47	175.022	14.448
stefan	532.966	517.405	15.561
tempete	1406.559	1352.524	54.035

4.4 模式補償對 STAR 演算法的影響

最後，本研究對模式補償對 STAR 演算法的影響作一評估。表 8 是只使用模式補償（不含圖 8. 的預測部分）來決定目標巨集方塊模式的效能。在執行時間上，只使用模式補償（VMS）來選擇切割模式要比完整的 STAR 演算法與完全搜尋演算法節省了更多時間，因為它每次只選擇兩個大小切割模式進行運算。在畫面品質上，與完全搜尋法相差不遠。但其所需編碼位元率比起有搭配預測部份的 STAR 演算法還來得高，因此說明了 STAR 演算法中預測部份的重要性。另外，如果只單純利用 STAR 之預測部份來選擇切割模式所得到的結果如表 8 的 Region 部份，其畫面品質跟完全搜尋法相比差距最多高達將近 3.4dB 左右，但經過模式補償後可減少至最多只差 0.12dB，表示模式補償也有其存在的必要性。

至於執行時間上，只使用模式補償要比預測來得快的原因，在於模式預測必須根據採樣的統計結果，而採樣時必須整個畫面使用完全搜尋。相對地，模式補償只有在邊界的巨集方塊會使用完全搜尋，其他的巨集方塊只需參考相鄰方塊的模式，故不需要採樣而且執行完全搜尋的次數也較少。

綜合以上結果，顯示 STAR 演算法的預測與補償兩部份是缺一不可的。不論在時間、畫質與位元率，兩者結合所產生的效能都要比各自獨立時要來得好。

表 8 模式補償之效能

(a)

	PSNR			DPSNR	
	full	Reigon	VMS	Reigon	VMS
container	35.645	33.829	35.488	-1.816	-0.157
fb	33.401	33.454	33.062	0.053	-0.339
foreman	36.276	36.491	36.025	0.215	-0.251
salesman	35.522	32.082	35.283	-3.44	-0.239
stefan	34.046	34.055	33.768	0.009	-0.278
tempete	34.869	34.991	34.56	0.122	-0.309

(b)

	Bit-rate			DBR(%)	
	full	Reigon	VMS	Reigon	VMS
container	168154	274298	183784	63.12308955	9.29505
fb	1818142	1952959	2145301	7.415097391	17.9941
foreman	412049	617394	471792	49.83509243	14.499
salesman	46099	59449	57010	28.95941344	23.6686
stefan	564545	593255	607585	5.085511341	7.62384
tempete	1111672	1284220	1229711	15.52148475	10.6181

(c)

	Time			DT(%)	
	full	Reigon	VMS	Reigon	VMS
container	1645.55	747.199	419.634	-54.59277774	-74.499
fb	1372.03	729.219	303.365	-46.85105052	-77.889
foreman	2556.69	1525.228	598.729	-40.34364745	-76.582
salesman	394.399	189.47	120.753	-51.95981734	-69.383
stefan	934.963	532.966	285.025	-42.996033	-69.515
tempete	2482.18	1406.559	563.217	-43.33372278	-77.31

5. 結論與未來工作

在本研究中，我們提出一個快速的H.264切割模式選擇演算法稱為STAR。此演算法的特點是運用模式選擇於空間與時間上的關聯性，來決定巨集方塊的切割尺寸。除此之外，此演算法亦考慮物體移動的空間於時間的區域性，因此採用分區採樣與預測以及定期地清除過時的樣本並重新收集。另外還在空間與時間上參考相鄰方塊的模式選擇，以避免錯誤之預測模式所造成RD cost的損失。經過效能評估後，發現本研究所提出的演算法確實能夠快速的選擇到適當的切割模式，並且保有良好的畫面品質與較低的位元率。另一方面，實驗的結果顯示，在模式選擇中考慮物體移動空間與時間區域性，對STAR演算法是有其必要性，而且模式預測與補償兩者缺一不可。

雖然 STAR 演算法具有良好效果，但是為了保

有較佳視訊品質而採用了補償機制，多少還是影響了整體執行運算時間，平均多 1~2 次 RD-cost 的計算。如何能更準確的預測與適當的補償來降低選擇與運算的時間將是可以繼續改善的地方。另一方面，考量現今尚未有很不錯的 H.264 硬體壓縮晶片量產，所以接下來本實驗室也將目標著重在研發一個即時的平行 H.264 編碼器，一方面從軟體上的平行技術進行效率的改善，再來則將 H.264 中各個部份功能逐一使用嵌入式系統的技術逐一實作於晶片中。以上是我們未來要進行的工作研究。

參考文獻

[1] 周敬利， 向東， 餘勝生， 陳加忠 "*A Predicting Algorithm of Macroblock Coding Mode Based on Spatio-Temporal Correlation for H.264*"，數字化期刊，電腦科學，pp.118-120, 2005

[2] Shlien, S. "*Guide to MPEG-1 audio standard*", Broadcasting, IEEE Transactions on ,Volume 40, Issue 4 , pp. 206 –218, Dec. 1994

[3] M. Bosi. et al, "*ISO/IEC MPEG-2 Advanced Audio Coding*", Journal of the Audio Engineering Society, Volume: 45, NO.10, pp. 789-813, October 1997

[4] W. Li. ,"*Overview of fine granularity scalability in MPEG-4 video standard.*", Circuits and Systems for Video Technology, IEEE Transactions on, Volume:11 Issue: 3 ,pp. 301–317, March 2001.

[5] Shih-Fu Chang Sikora, T. Purl, A. ," *Overview of the MPEG-7 standard"*, Circuits and Systems for Video Technology, IEEE Transactions on, Volume: 11, Issue 6,pp. 688-695, Jun 2001

[6] T. Turletti, "*H.261 Software Codec for Videoconferencing over the Internet*", research report n. 1834, INRIA, January 1993

[7] F. Fitzek and M. Reisslein, "*MPEG-4 and H.263 video traces for network performance evaluation*", Network IEEE, Volume 15, Issue 6, pp. 40-54 ,Nov.-Dec. 2001

[8] G. Cote, B. Erol, M. Gallant, and F. Kossentini, "*H.263+: video coding at low bit rates*", Circuits and Systems for Video Technology, IEEE Transactions on, Volume 8, Issue 7, pp. 849–866, November 1998.

[9] TW, GJ. S, G. Bjontegaard, and A. Luthra, "*Overview of the H.264/AVC video coding* ", Circuits and System for Video Technology, IEEE Trans. on, Volume 13, Issue 7, pp. 560-576, July 2003.

[10] Y.-H. Kim, J.-W. Yoo, S.-W. Lee, J. Shin, J. Paik, and H.-K. Jung, "*Adaptive Mode Decision for H.264 Encoder*," Electronics Letters, vol. 40, no. 19, pp. 1172-1173, Sept. 2004.

[11] B. Jeon and J. Lee, "*Fast mode decision for H.264*", in ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6, JVT 10th Meeting, Waikoloa, Hawaii, December 2003.

[12] X. Jing and L.-P. Chau, "*Fast approach for h.264 inter mode decision*", in Electronic Letters, Volume 40, Issue 17, pp. 1050–1052, 19 Aug. 2004.

[13] Jianjun Guo Kui Dai Yun Cheng Zhiying Wang ,"*Research on Fast Block Partition Mode Selection Algorithm in H.264*", Computer and Information Science, 2005. Fourth Annual ACIS International Conference on**,pp.** 111- 113 , 2005.

[14] Han, J., and Kamber, M., "*Data mining: concepts and techniques*," Morgan Kaufmann, San Francisco. 2000.

[15] Iain E.G. Richardon, "*H.264 and MPEG4 video compression: video coding for next generation multimedia*", John Wiley & Sons, Ltd. ISBN: 0-470-84837-5, 2003.

[16] Joint Video Team Reference Software JM10.2 (2006 Aug.). [Online] Available: http://iphome.hhi.de/suehring/tml/download/