利用時空關聯性搜尋之快速 HEVC 編碼器

陳翰彥、李政庭、王周珍、董棋偉
義守大學 電子工程學系

摘要 —本論文是利用時間與空間的關聯性來提出一個 HEVC 快速編碼演算法。新一代的視訊編解碼標準 HEVC 提供可調式結構編碼單位，雖能大幅提升編碼效能，但運算複雜度卻非常高，以至於無法達到即時的視訊應用。為了有效降低運算的複雜度，本論文藉由參考鄰近周圍 CTU 樹形(空間關聯性)及相對位置 CTU 樹形(時間關聯性)搭配預設臨界值，來進行 CTU 時空關聯性搜尋以降低 HEVC 編碼時間。論文所提的 HEVC 快速編碼器與原始的 HEVC 架構在 HM 8.1 測試平台下，從實驗結果顯示，在犧牲可接受的位元率下，論文所提的方法平均降低約 80%的編碼時間，且視訊品質幾乎不變。

一、 簡介

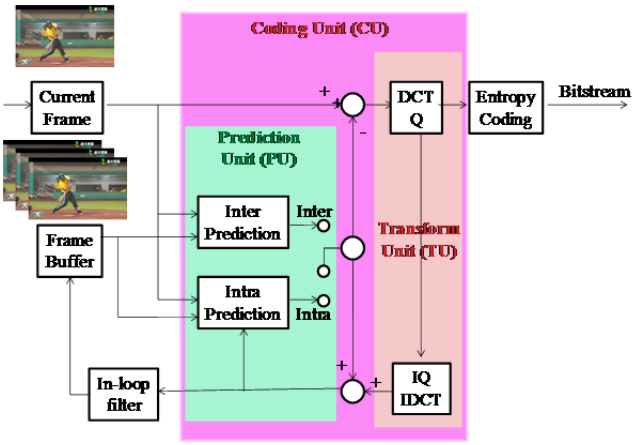
隨著數位多媒體技術的蓬勃發展，人們對於影像解析度的要求越來越高，然而面對最新的超高解析度(UHD: 4k×2k)影像，以往的 H.264 已經不敷使用。因此最新一代的視訊編碼標準：高效率視訊編碼(high efficiency video coding: HEVC)在這樣的環境之下被提出，第一版的 HEVC 已經在 2013 年 4 月被接受為國際電信聯盟(ITU-T)的正式標準 [1]-[4]。

HEVC 提供了許多新的工具，在編碼架構中分成了三個基本單位，分別是編碼單位(coding unit: CU)、預測單位(predict unit: PU)和轉換單位(transform unit: TU) [1]，並且在預測的部分有更多模式可以選擇，能比以往的 H.264 預測得更加準確，但也是由於諸多新增的編碼工具，使得編碼時的運算複雜度大大增加，編碼時間也大幅提升。因此為了減少 CU 模組龐大的運算量，本論文利用時間及空間之關聯性(temporal- spatial correlation)提出了一快速演算法，利用鄰近已編碼之 CTU(coding tree unit)樹形搭配我們所提出的臨界值公式來選擇相近的 CTU 樹形來當作當前待編碼 CTU 樹形之預測，在保有品質幾乎不變和犧牲可接受範圍內的位元率(bitrate)下，大幅度降低 HEVC 的編碼時間，以期能達到即時(real-time)的視訊應用。

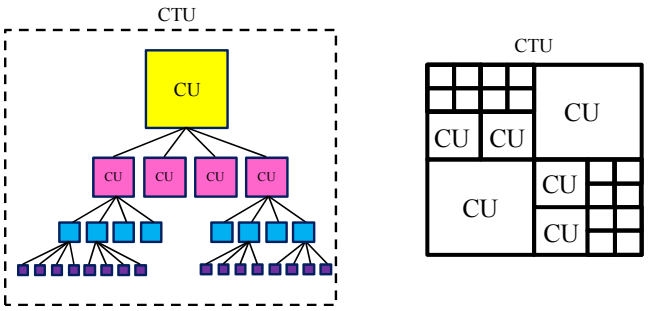
二、 HEVC 視訊編碼標準介紹

圖一為 HEVC 視訊編碼架構圖，大致上和 H.264 相似，主要包含 畫框間預測(intra prediction)、畫面間預測(inter prediction)、轉換(transform)和量化(quantization)等模組。但 HEVC 編碼架構中還增加 CU、PU 和 TU 三種編碼方式[5]，以下將對這三種單位做簡單的介紹。

[1] 本研究由國科會贊助，計畫編號 NSC 102-2815-C-214-020-E。
[2] 通訊作者: 王周珍 E-mail: chchwang@isu.edu.tw



圖一：HEVC 視訊編碼架構圖



圖二：四分樹劃分與最佳 CTU 樹形示意圖

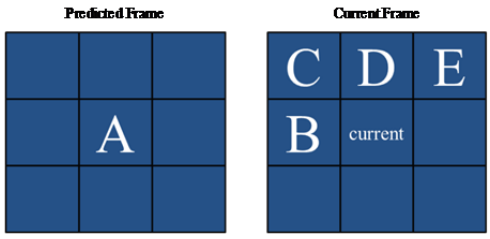
在 HEVC 中，CTU(64×64)就像是 H.264 的 MB 一樣，包含了所有的編碼程序，而 CTU 利用四分樹劃分(quad tree partition)可以切割成一個以上的 CU，而 CU 的大小可以從 64×64 到 8×8，並且以深度(depth)表示其大小。每個 CU 都必須經過 PU 以及 TU 去算出其 RDcost，而 RDcost 的計算公式如下：

$$J_{mode} = SSE + \lambda_{mode} \times B_{mode} \quad (1)$$

其中 SSE 為當前區塊與預測區塊的差值平方和，λ_{mode} 為一個 Lagrange 函數，B_{mode} 為編碼時所需要用到的位元數。最後再利用 RDcost 選擇出最適當的 CU，所有 CU 做完便完成了 CTU 樹形之切割，如圖二所示。

PU 是預測的基本單位，包含 Intra 預測及 Inter 預測。PU 的大小可從 64×64 到 4×4，其尺寸會隨著 CU 尺寸及所選的預測模式而改變。其中 2N×2N 相當於當前 CU 的尺寸，另外在 Inter 預測時提供非對稱類型的預測，使預測能夠更加準確。TU 是 HEVC 中做轉換、量化和熵編碼(entropy coding)的最基本單位，TU 的大小從 32×32 到

Corresponding author:chchwang@isu.edu.tw
DOI:10.6159/IJSE.2014.(4-2).79



圖三：參考 CTU 之位置

表 I、 當前 CTU 與鄰近 CTU 樹形相同之機率分布

	A(%)	B(%)	C(%)	D(%)	E(%)	All(%)
Vidyo1	77.15	73.03	56.01	61.39	55.63	88.70
Vidyo3	76.07	70.09	55.59	61.59	55.02	84.43
Vidyo4	72.44	67.34	53.06	58.76	52.82	85.45
Kimono	33.71	30.55	22.49	27.51	22.13	60.56
ParkScene	35.80	36.81	29.67	34.60	29.10	55.98
BasketballDrive	46.01	49.42	39.68	43.38	40.55	67.45
Cactus	52.69	48.57	39.69	45.35	40.43	65.95
BQTerrace	45.85	47.57	35.42	40.45	35.79	67.49
Average	54.97	52.92	41.45	46.63	41.43	72.00

4×4，和 CU 有類似的四分樹結構，稱為差值四分樹 (residual quad tree: RQT)。HEVC 編碼時，首先會將畫面切成數個 CTU，再以四分樹結構開始做 Intra 及 Inter 預測。從 CU 64×64 做到 CU8×8，每個 CU 都必須利用預測後的 RD cost 找到最佳尺寸以及所選擇的預測模式，再經由熵編碼完成視訊壓縮編碼。

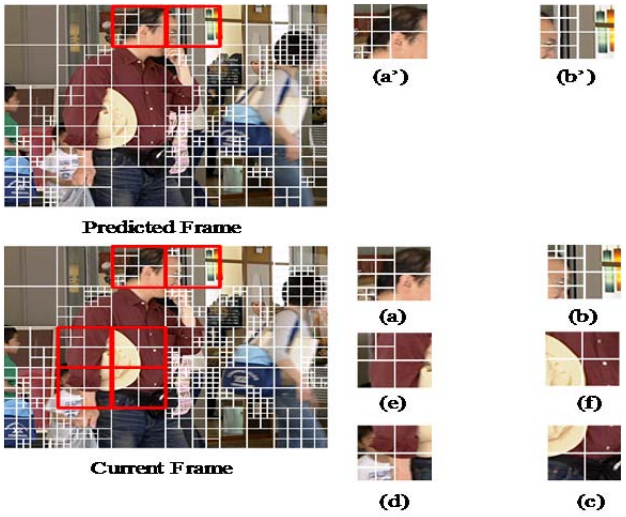
由於在決定 CU 的過程需要利用 PU 去計算每一種模式的 RD cost，因此 CU 模組的運算量非常龐大且耗時。因為 CU 包含了 PU 和 TU，所以 CU 約佔了 HEVC 所有模組編碼時間的 96%，為了加速編碼時間，論文提出一考慮時空關聯性的搜尋方法，來大幅降低 CU 模組所需的龐大運算量及時間。

三、 HEVC 快速編碼演算法

由於自然視訊存在很高的時空關聯性，因此從HEVC的編碼結果，可以發現當前CTU與鄰近已編碼CTU之樹形相同或相似之機率非常高。本章針對這些CTU樹形的機率分佈進行分析，並訂定一臨界值來進行CTU樹形預測，並提出利用時空關聯性(temporal-spatial correlation)搜尋預測的HEVC快速編碼演算法。

3.1 當前CTU與鄰近已編碼CTU之樹形分布觀察

如先前所介紹，HEVC支援的CU大小從64×64到8×8，且每個CU又會切成各種形狀的PU和TU，為了找出最佳的CU，會計算每個CU尺寸、預測模式和TU組合，再從中找最小的RDcost選為最佳的切割模式。完成一個CTU非常的複雜及耗時。由於HEVC主要是針對高解析度影像，而相較於低解析度影像，高解析度影像畫框間(inter frame)的關聯性會更高，加上畫面內(intra frame)的關聯性也很高，因此我們利用時間及空間的關聯性來進行CTU的預測，我們選擇當前CTU前一張畫面的相對位置A(co-located)和當前CTU之左邊B、左上C、上方D、右上等五個位置來進行相同CTU樹形機率分佈統計，如圖三所示。



圖四：當前 CTU 的樹形跟鄰近 CTU 的樹形的時空關聯性

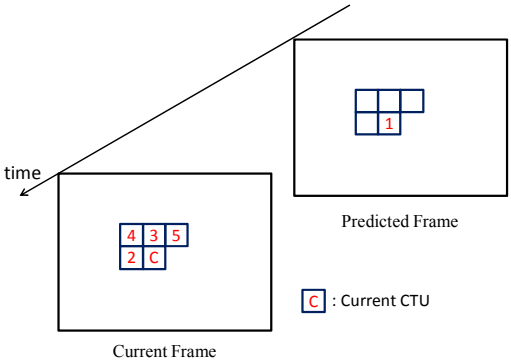
表I為利用HEVC的測試平台HM8.1各種解析度影像的編碼結果之相同CTU樹形機率分佈統計，其中All代表的是五個位置的聯集，也就是有其中一個相同的機率。從表中可以發現，平均相同的機率有40%~55%，而相鄰時空位置相同的CTU樹形機率大小排序分別為A、B、D、C和E。

另外從編碼結果也可以發現，當前CTU的樹形跟鄰近CTU的樹形雖然很多並不一定完全相同但卻十分相似。圖四為已編碼後的兩張連續畫面的最佳CTU結構圖，其中圖四(a)及(a')為在相對位置(A)有完全相同的CTU樹形，(b)及(b')則為相似的CTU樹形。另圖四(d)及(e)為與圖四(c)相似樹形的空間CTU，而圖四(f)為與圖四(c)完全相同樹形的空間CTU。從HM8.1的實驗結果，我們發現到當前CTU的樹形跟鄰近CTU的樹形具有很高的時空關聯性，如圖四所示。若能利用CU之RDcost去設定一最佳的臨界值來進行樹形預測，在犧牲些許的預測準確度，將相似的CTU樹形作為預測CTU之樹形，將可大幅提高HEVC視訊編碼效率。

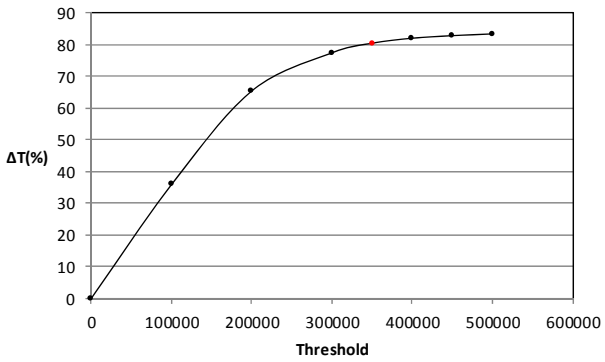
3.2 時空關聯性搜尋演算法

根據 3.1 節的 CTU 樹形統計分析與觀察，我們提出一利用時空關聯性(temporal-spatial correlation)搜尋預測的 HEVC 快速編碼演算法，圖五為根據 CTU 樹形機率大小排序之預測搜尋順序，所提演算法的步驟如下：

Step1. 根據設定之 QP 計算出預設 RDcost 臨界值(Thr_{Qp})。
Step2. 計算和檢查前一張畫面相對位置(A)之 CTU 樹形 RDcost 是否小於Thr_{Qp}，若是則跳至 Step5，否則進入 Step3。
Step3. 計算和檢查畫面內相鄰位置(B)之 CTU 樹形 RDcost 是否小於Thr_{Qp}，若是則跳至 Step5，否則回到 Step3 按圖五之搜尋順序依序搜尋相鄰位置之 CTU，直到最後一個鄰近位置結束，進入 Step4。
Step4. 進行原始 HEVC 的 CTU 之 RDO 模組，進行完整四分樹修剪(quad tree pruning)以得到最佳的 CTU 樹形。
Step5. 紀錄 CTU 樹形及相關的編碼參數。



圖五：時空關聯性搜尋順序示意圖



圖六：QP = 32之Δ*Time*與臨界值曲線圖，設定 $Thr_{Qp} = 350000$

四、 實驗結果與討論

本論文採用 HEVC 編碼參考軟體 HM 8.1[5]，以 Visual++. 2008 作為編譯環境，完成 PC 上的軟體模擬測試後，對數據進行統計與分析，主要的實驗條件如下：

1. 影像序列：Kimono 、 ParkScene 、BasketballDrive 和 Cactus ，解析度均為 1920×1080
2. 編碼規範：Low delay P (IPPPP…)
3. 畫面張數：100
4. 量化參數：24、28、32、36 及 25、30、35、40
5. 參考畫面：1
6. 畫面率(fps)：25、50
7. CPU: Intel(R) Core i5 3.20GHz.

為了進一步評估所提方法與 HM 8.1 的編碼效能，我們定義分別 Δ*PSNRY*、Δ*Bitrate* 和 Δ*Time* 如下：

$$\Delta PSNRY = PSNRY_{proposed} - PSNRY_{HM} \quad (2)$$

$$\Delta Bitrate = \frac{Bitrate_{HM} - Bitrate_{proposed}}{Bitrate_{HMl}} \times 100\% \quad (3)$$

$$\Delta Time = \frac{Time_{HM} - Time_{proposedl}}{Time_{HM}} \times 100\% \quad (4)$$

4.1 臨界值的設定

演算法中臨界值的設定是非常重要的參數，我們知道當臨界值過大時，雖然畫面編碼時間會大幅降低，但是編碼準確率卻會因此下降，導致畫面品質降低以及位元率的提升，當臨界值設定過小時則反之。因此，我們以QP = 32來進行四組影像序列的 HEVC 編碼，從不同臨界值(threshold)的實驗結果，可以發現雖然 Δ*Bitrate* 和 Δ*PSNR* 均下降，但都在可接受範圍內，相較之下，Δ*Time* 改善率卻大幅提高(50%以上)。因此，本論文以 Δ*Time* 作為選擇臨界值的主要考量。圖六為 Δ*Time* 與不同臨界值曲線圖，和一般 RD 曲線圖一樣的特性，從圖中我們可以發現 Δ*Time*(ΔT)之增加量隨著臨界值提高而減少。因此，我們選擇 ΔT 增加量約剩下 3%且 ΔT 約為 80%時之 $Thr_{Qp} = 350{,}000$作為臨界值。

因為 HEVC 之 RDcost 會隨著 λ 而改變，而 λ 也會因為 QP 而有所不同，兩者之間的關係如下：

$$\lambda = 0.4845 \times 2^{(QP-12)/3} \quad (5)$$

為了訂定出一臨界值的通式，以適用於任何不同的 QP。在相同的實驗條件下，我們也進行 QP = 24、28 和 36 之 HEVC 視訊編碼，並選擇出最好臨界值，分別為 130000、220000 和 600000。最後利用這四組不同 QP 所選出的臨界值及 QP 之間的關係，再利用線性回歸(linear regression)找出一條適用於任何 QP 之臨界值的線性方程式。本論文以 QP = 32 為基準下，推導出 Thr_{Qp}與 QP 之線性關係如下：

$$Thr_{QP} = (\lambda_{QP} - \lambda_{32}) \times 4350 + 350000 \quad (6)$$

因此在進行快速 HEVC 編碼時，對任何 QP 均可以利用(6)來計算出所對應的臨界值。

4.2 實驗結果

表 II 和表 III 分別為兩組不同 QP 下之編碼效能比較，其中表 II 為參與線性回歸線推導之 QP，表 III 為任意選擇之 QP。從表中我們可以發現，利用(6)所求出的臨界值，在任何 QP 下平均 ΔT 改善率維持在 80%左右，加速編碼效能十分顯著。另外從表中我們也發現，在 QP 值較高時，像是 BasketballDrive 或是 Cactus 畫面運動較劇烈的影像序列，Bitrate 會上升將近 15%，但因現今網路頻寬相當寬且便宜，所以在犧牲可接受的位元率下，換取視訊品質幾乎不變的快速編碼，已應用在即時的視訊應用，是相當值得的。

結論

HEVC 為了達到更好的壓縮率，使用了比以往更多的編碼工具，使其預測更加精準，達到相同品質下可以有更低的位元率，雖然編碼效率提升，但其增加的複雜度也讓 HEVC 在編碼時更加耗時。而本論文利用了 CTU 時間及空間之關聯性，配合臨界值公式做預測，改善原本 HM8.1 裡的 CU 模組。從實驗結果來看，改善後的 HEVC 快速編碼器的 Bitrate 平均增加了 8%，PSNR 平

均降低了 0.12 dB，但時間平均降低了 82%，大幅縮短了 CU 模組的時間，達到加速 HEVC 的編碼效率。

參考文獻

[1] G. J. Sullivan, J. R. Ohm, W. J. Han and T. Wiegand,"Overview of the high-efficiency video coding standard," *IEEE trans. on circuits and systems for video technology*, vol. 22, no. 12, pp. 1649-1667, Dec. 2012.

[2] B. Bross, W. J. Han, G. J. Sullivan, J.-R. Ohm, and T. Wiegand, "High efficiency video coding (HEVC) text specification draft 8," ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC) document JCTVC-J1003, July 2012.

[3] C. J. Lian, S. Y. Chien, C. P. Lin, P. C. Tseng, and L. G. Chen, "Power-aware multimedia: concepts and design perspectives", *IEEE Circuits and Systems Magazine*, vol. 7, pp. 26-34, 2007.

[4] J. R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the Coding Efficiency of Video Coding Standards – Including High Efficiency Video Coding (HEVC)", *IEEE trans. on circuits and systems for video technology*, vol. 22, no. 12, pp. 1668-1684, Dec. 2012.

[5] Reference software HM8.1, https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/branches/

[6] H. S. Lee, K. Y. Kim, T. R. Kim and G. H. Park, "Fast encoding algorithm based on depth of coding-unit for high efficiency video coding", *Opt. Eng.* 51(6), 067402, Jun 08, 2012. ;

表 II、各種 QP 之編碼效能

QP	Kimono1			ParkScene			BasketballDrive			Cactus			Average		
	ΔB(%)	ΔP(dB)	ΔT(%)	ΔB(%)	ΔP(dB)	ΔT(%)	ΔB(%)	ΔP(dB)	ΔT(%)	ΔB(%)	ΔP(dB)	ΔT(%)	ΔB(%)	ΔP(dB)	ΔT(%)
24	-4.64	-0.03	88.11	-3.92	-0.05	84.69	-6.76	-0.06	83.93	-3.95	-0.02	84.09	-4.81	-0.04	85.20
28	-4.98	-0.05	87.13	-4.69	-0.10	80.01	-8.51	-0.10	82.47	-5.42	-0.05	79.41	-5.90	-0.08	82.26
32	-5.81	-0.08	85.59	-5.71	-0.15	77.51	-9.40	-0.20	80.86	-6.83	-0.11	77.89	-6.94	-0.14	80.46
36	-6.11	-0.11	86.08	-7.56	-0.19	80.96	-11.47	-0.28	82.83	-9.45	-0.16	81.16	-8.65	-0.19	82.76

表 III、各種 QP 之編碼效能

QP	Kimono1			ParkScene			BasketballDrive			Cactus			Average		
	ΔB(%)	ΔP(dB)	ΔT(%)	ΔB(%)	ΔP(dB)	ΔT(%)	ΔB(%)	ΔP(dB)	ΔT(%)	ΔB(%)	ΔP(dB)	ΔT(%)	ΔB(%)	ΔP(dB)	ΔT(%)
25	-4.17	-0.03	87.39	-4.30	-0.07	83.11	-7.61	-0.06	83.34	-4.49	-0.03	81.70	-5.14	-0.05	83.88
30	-4.91	-0.05	85.83	-4.62	-0.13	78.65	-8.41	-0.14	79.44	-5.93	-0.07	76.93	-5.97	-0.10	80.21
35	-5.29	-0.11	85.36	-7.72	-0.18	79.41	-11.72	-0.26	82.10	-8.97	-0.15	79.17	-8.42	-0.17	81.51
40	-7.26	-0.13	85.88	-11.69	-0.18	83.21	-16.89	-0.38	84.88	-14.66	-0.25	82.62	-12.63	-0.23	84.15