國 立 交 通 大 學

電 機 與 控 制 工 程 研 究 所

碩 士 論 文

變動取樣率之低功率移動估測設計

On Power-Saving Motion Estimation using Variable Subsample Ratios

指導教授：董蘭榮　博士

研究生：葉松樹

中華民國九十四年七月

On Power-Saving Motion Estimation Using Variable Subsample Ratios

Advisor: Dr. Lan-Rong Dung

Graduate Student: Sung-Shu Yeh

June 2005

Graduate Institute of Electrical and Control Engineering

National Chiao Tung University

Hsinchu, Taiwan, ROC

On Power-Saving Motion Estimation Using Variable Subsample Ratios

Graduate Student: Sung-Shu Yeh Advisor: Lan-Rong Dung

Department of Electrical and Control Engineering
National Chiao Tung University

Abstract

In modern video standard, such as MPEG-1, MPEG-2, MPEG-4 and H.264/MPEG-4 AVC, motion estimation requires the heaviest computational load and hence dominates main power requirement in video compression. Lots of published papers have presented efficient algorithms for motion estimation. But they don't consider the influence of the video content. An adaptive motion estimation algorithm with variable subsample ratios has been presented. This proposed algorithm can adaptively select the compatible subsample ratio for each current group of picture (GOP). This proposed algorithm has been successful implemented in H.264/MPEG-4 AVC with software model JM9.2. Experimental results have shown that the proposed algorithm can not only adaptively select the suitable subsample ratio to various video sequences but also maintain ΔPSNRY of 0.36 dB at most to save about 69.6% power consumption of motion estimation in a fixed bit rate control on average. The proposed algorithm can also easily combine with other fast algorithms which reduce the computational complexity of FSBM. For FME in JM 9.2, we save 73.6% power consumption and keep quality degradation under 0.33 dB. Hence the proposed algorithm is suitable for real-time implementation of high quality and power-saving video applications using a powerful CPU.

變動取樣率之低功率移動估測設計

學生：葉松樹　　　　　　指導教授：董蘭榮博士

國立交通大學

電機與控制工程學系研究所

摘要

最新的影像壓縮規格中，如MPEG-1，MPEG-2，MPEG-4 and H.264/MPEG-4 AVC，移動位移估測需要龐大的計算量與能量消耗。因此，移動位移估測主導了在影像壓縮中的計算量與能量需求。針對位移估測，很多論文已經提出了不同的快速演算法，可是他們並沒有考慮到影像內容的影響。所以，我們提出了一種新的快速演算法，稱為“變動取樣率之低功率位移估測演算法”，這個演算法可以針對不同性質的影像內容，選取最適合的取樣率來完成位移估測，以達到節省能量且畫質衰退在可以容許的範圍之內。針對不同性質的影像內容，動態的選擇四組不同的取樣率，分別為16：16、16：8、16：4、16：2。我們提出的這個演算法已經成功的實現在H.264/MPEG-4 AVC的軟體模型JM9.2中，實驗結果顯示這個演算法不只可以動態的依照不同的影像內容來選擇適合的取樣率，而且可以維持最多0.36 dB的畫質衰退，在固定的傳輸頻率下，平均可以節省69.9%的能量損失。這個演算法的另一個好處是可以輕易的與其他位移估測快速演算法結合，而達到進一步減少所需的計算量，我們結合JM9.2中的FME快速演算法，可以維持最多0.33 dB的畫質衰退且節省73.6%的能量損失。因此，對於我們所提出的“變動取樣率之低功率位移估測演算法”，是適合實現在高畫質的即時影像壓縮應用上。

誌 謝

能夠完成這篇碩士論文，我心中的充滿無限的感謝與激動，然而它對我的意義，並不止於表面上的這些文字。

首先，我要感謝在董蘭榮老師的多方指導，使學生在這兩年的研究生生涯中，獲益良多，從針對問題的態度、解決問題的技巧、尋找問題的原委，給了我相當大的啟發，在此，我獻上至高的謝意。

感謝整個實驗室研究團隊互助合作、無私的付出。感謝顯文學長、盟淳學長的指導，樹德、明峰、育聖、芳彥的幫助，以及學弟岳璋、泰佑的協助，在我研究上需要幫助或是精神上受到挫折的時候，大家總是支持我、幫助我，讓我一直堅持到最後。

最後，最深的感謝，要獻給養育我、栽培我二十幾年的父母，你們的支持是我在研究上最堅強的後盾，所有的喜悅與榮耀，都希望能與你們一同分享，謝謝你們！我最敬愛的爸媽！

僅以此篇論文獻給所有愛我及我所愛的人，再次獻上我由衷的感激，謝謝大家！

松樹 謹誌

國立交通大學 系統晶片實驗室

民國九十四年七月

Contents

List of Figures

List of Tables

Chapter 1 Introduction

In modern video standard, such as MPEG-1 [1], MPEG-2 [2], MPEG-4 [3] and H.264/MPEG-4 AVC [4], motion estimation requires the heaviest computational load and hence dominates main power requirement in video compression. Lots of published papers [4]-[17] and [39]-[47] have presented efficient algorithms for motion estimation. But they don't consider the influence of the video content. In our observation, the video content effects on the performance of motion estimation. So we base on the video content to select the suitable motion estimation in order to achieve the best tradeoff between the power and quality. We think that we must use the different motion estimation in the different degree of the video content. Among these fast algorithms [4]-[17], the subsample algorithms [11]-[17] and [39]-[47] can not only easily combine with other approaches mentioned above but also reduce the number of matching points with flexibly changing subsample ratio.

The subsample algorithm, also called the pixel decimation algorithm, can be, in general, classified into two categories. One is fixed patterns [11]-[15], and the other is adaptive patterns [16] [17]. Bierling used an orthogonal sampling lattice with a 4:1 subsample [11]. Liu and Zaccarin implemented pixel decimation that is similar to Bierling's approach with four alternating subsample patterns selected for each step so that all the pixels in the current block are visited [12]. T.Chiang et al presented an *N*-queen decimation approach to address the spatial homogeneity and directional coverage [14], [15]. The pixel decimation can be adapted based on the spatial luminance variation within a picture [16], [17]. The content-based subsample algorithm is proposed in [39]-[47]. Adaptive techniques can achieve better coding efficiency as compared to the uniform subsample schemes with an overhead in deciding which pattern is more representative. These presented subsample algorithms can successfully reduce the computational complexity of motion estimation to save much power dissipation.

The reason why we choose the adaptive subsample ratios is because we believe that the subsample ratios should be varying with the video content. Although the subsample algorithms [16], [17] use the adaptive subsample patterns based on the spatial luminance variation within a picture, they all don't mention the temporal variation. They result in serious aliasing problems in high frequency band to degrade the visual quality without considering the temporal variation. The temporal variation in the video means the degree of object-moving. The degree of object-moving is faster, and the temporal variation is stronger. Although the low subsample ratio cause aliasing in high frequency band, the degree of temporal variation will affect the degree of quality degradation. If the temporal variation is strong, aliasing problems will degrade the validity of motion vector (MV) and result in visual quality degradation to video sequences obviously. On the contrary, if the temporal variation is weak, aliasing problems will not degrade the validity of motion vector (MV) although the low subsample ratio still cause aliasing in high frequency band. That is because we do not need the high frequency band information to find the motion vector when the degree of object-moving is slow. Hence, using higher subsample ratio to reduce the prediction residual is necessary when temporal variation is stronger. In DSP theory [18] the subsample process will induce the aliasing in high frequency band. The aliasing problem affects the variance of the prediction residual under a fixed bit-rate constraint. The variance of the prediction residual affects the compression quality. The quality degradation of 0.5 dB is empirically reasonable for the perceptual tolerance of decompressed visual quality in video coding community. Therefore, in order to efficiently alleviate the aliasing problem to satisfy the visual quality under the quality threshold of 0.5 dB for general video sequences, adaptively selecting the suitable subsample ratio according to the degree of temporal variation in the content is imperative.

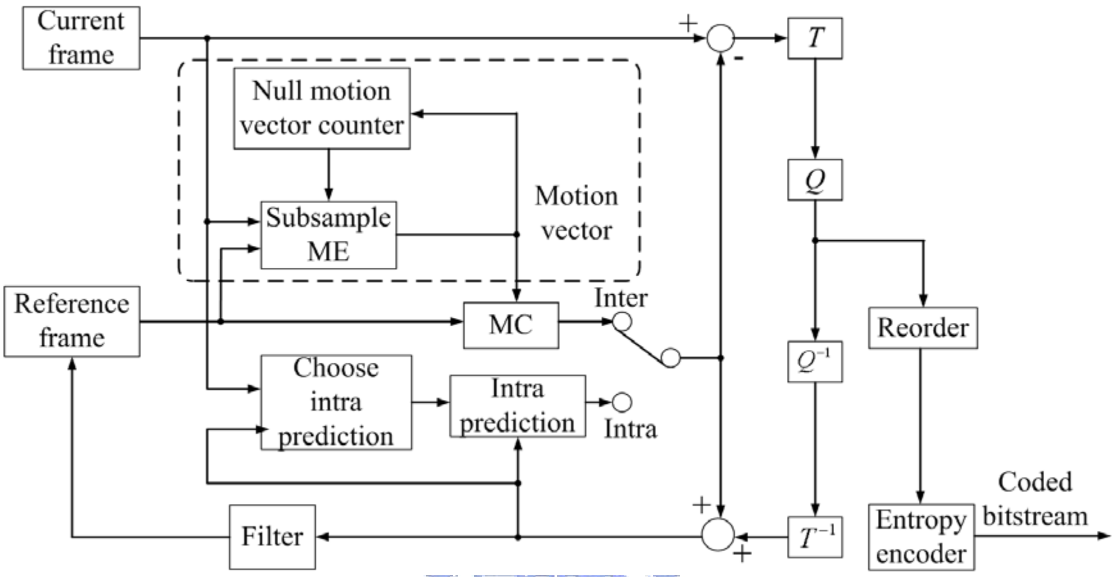


Fig 1.1: The proposed system diagram in H.264/MPEG-4 AVC encoder

In this thesis, we develop an adaptive motion estimation algorithm with variable subsample ratios and this proposed algorithm can adaptively select the compatible subsample ratio for each current group of picture (GOP). The proposed algorithm is first to analyze the degree of the object-moving between the first P-frame and I-frame for the current GOP and then adaptively selects the suitable subsample ratio to the current GOP according to analysis result. The proposed algorithm also has been successfully implemented in the encoder model of H.264/MPEG-4 AVC [4] reference software JM9.2 [36] and the proposed system diagram is shown in Fig 1.1. The dash-lined region is the proposed motion estimation algorithm and the proposed algorithm offers four kinds of subsample ratios to adaptively switch. We use the statistics science to analyze every GOP degradation and null motion vector count (NMVC). And we get several different threshold values to experiment with video sequences. The experimental results have been shown that the proposed algorithm with the optimal threshold value can not only adaptively maintain visual quality under the quality degradation of 0.36 dB in a fixed bit-rate control for general video sequences but also meaningfully achieve the target of power-saving.

The rest of the thesis is organized as follows. We introduce the study background in chapter 2. In chapter 3, we introduce the generic subsample algorithm in detail and describe the aliasing problem in the subsample algorithm. Chapter 4 describes the proposed algorithm. Chapter 5 shows the experimental performance of the proposed algorithm in H.264/MPEG-4 AVC [4] software model JM9.2 [36]. Finally, Chapter 6 concludes our contribution and merits of this work.

Chapter 2 Background

In this chapter, technical overview of H.264/MPEG-4 AVC [4] will be introduced [19] [20]. The feature of H.264/MPEG-4 AVC [4] unlike MPEG-4 [3] will be pointed out [19] [20]. The variable block size motion estimation is the main different place in H.264/MPEG-4 AVC [4]. The paper [21] proposed the new one-dimensional (1-D) very large-scale integration architecture for full-search VBSME (FSVBSME). About the full-search algorithm, it is particularly attractive to ones who require extremely high quality. However, it requires a huge number of arithmetic operations and results in highly computational load and power dissipation. In order to reduce the computational complexity of the FSBM, lots of published papers [4]-[17] and [39]-[47] have presented efficient algorithms for motion estimation. Among these fast algorithms [4]-[17] and [39]-[47], the subsample algorithms [11]-[17] and [39]-[47] can not only easily combine with other approaches mentioned above but also reduce the number of matching points with flexibly changing subsample ratio. In general, the subsample algorithm, also called the pixel decimation algorithm, can be classified into two categories. One is fixed patterns [11]-[15], and the other is adaptive patterns [16] [17] and [39]-[47]. Adaptive techniques can achieve better coding efficiency as compared to the uniform subsample schemes with an overhead in deciding which pattern is more representative. These presented subsample algorithms [11]-[17] and [39]-[47] can successfully reduce the computational complexity of motion estimation to save much power dissipation.

2.1 H.264/MPEG-4 AVC Video Coding System

H.264/MPEG-4 AVC [4] provides ultra high coding efficiency and network friendly functionalities. It has been a hot candidate for future's video streaming and communications. Fig 2.1 [22] shows that rate-distortion curve comparison of H.264/MPEG-4 AVC [4] with previous video coding standards. Under medium bit-rate, its PSNR quality outperforms MPEG-4 [3] simple profile by more than 3dB. Fig 2.2 shows H.264 baseline subjective view comparison with MPEG-4 advanced simple profile at the specification of QCIF and bit-rate 112Kbps.

H.264/MPEG-4 AVC [4] has such high performance because it adopts several novel coding tools in its algorithm design. For example, variable block size motion estimation, multiple reference frame motion estimation, and intra frame prediction are used in its prediction algorithm. In-loop deblocking filter offers good subjective view. The 6-tap filter is incorporated to do the quarter pixel interpolation. CAVLC (Context-Adaptive Variable Length Coding) and CABAC (Context-Adaptive Binary Arithmetic Coding) are adopted in its entropy coding design. H.264/MPEG-4 AVC [4] is the first video coding standard that adopts the arithmetic coding into its entropy design. The block diagram of H.264/MPEG-4 AVC encoder is shown in Fig 2.3. Video frames are captured into intra prediction and inter prediction parts. If the frame type is intra, the inter prediction part will be disabled. Multiple reference frames and variable block size motion estimation is used for inter prediction. The best mode among these prediction modes is chosen in the mode selection block. The input frame is then subtracted from the prediction and forms the residue block. The residue blocks are transformed by 4x4 integer DCT for luminance and 2x2 transform for chrominance DC coefficient. Scan and quantization procedures are then applied to the coefficients. The entropy coder receives these quantized coefficients and generates codeword. The mode information is also transformed by the mode tables and fed into the entropy coder. The reconstruction loop includes the dequantization, inverse transform and deblocking filter. Finally, the reconstruct frame is written to the frame buffer for motion estimation.

There are three kinds of profile for H.264/MPEG-4 AVC standard [4]: baseline profile is for real-time communication, main profile is for digital storage application, and x-profile is for network streaming application. In the baseline profile, B-frame is not used and CAVLC is adopted in entropy coding. In the main profile, B-frame coding is used and CABAC is adopted for entropy coding. And X-profile has all the features of baseline profile while B-frame coding, SI-frame coding, and SP-frame [23] coding are included. Although the coding performance of H.264 is good, more than four times of the algorithm complexity compared to MEG-4 simple profile prevents its practical implementation. Several previous papers and documents have addressed the coding complexity of this new state of art video coding algorithm.

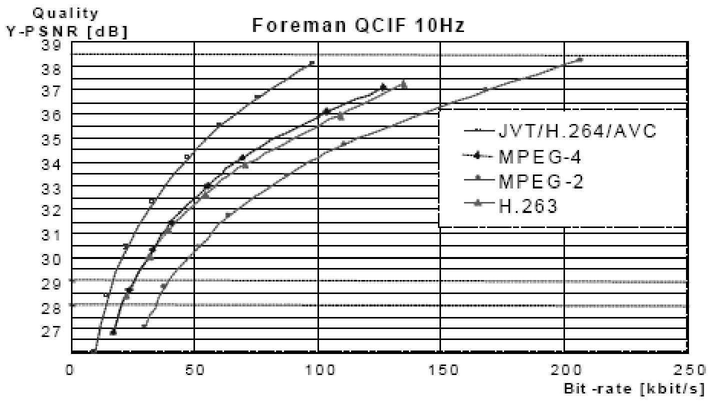


Fig 2.1: Rate-distortion curve comparison of H.264/MPEG-4 AVC with previous standards. (Excerpted from [22])

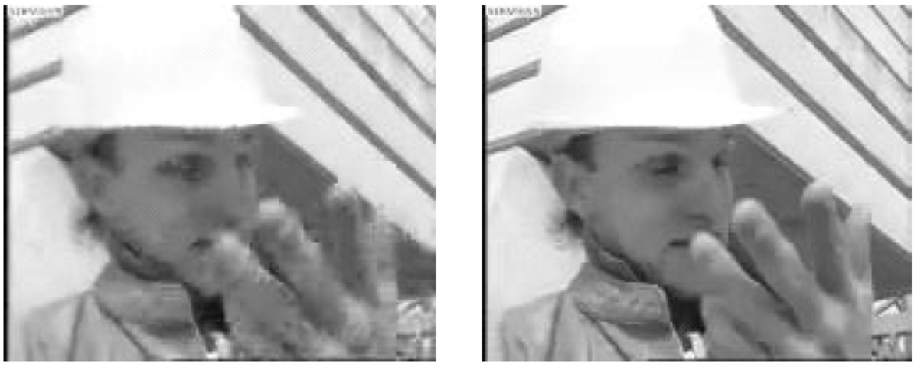


Fig 2.2: Subjective view comparison of MPEG-4 ASP (left) and H.264/MPEG-4 AVC baseline (right) at bit-rate 112Kbps. (Excerpted from [22])

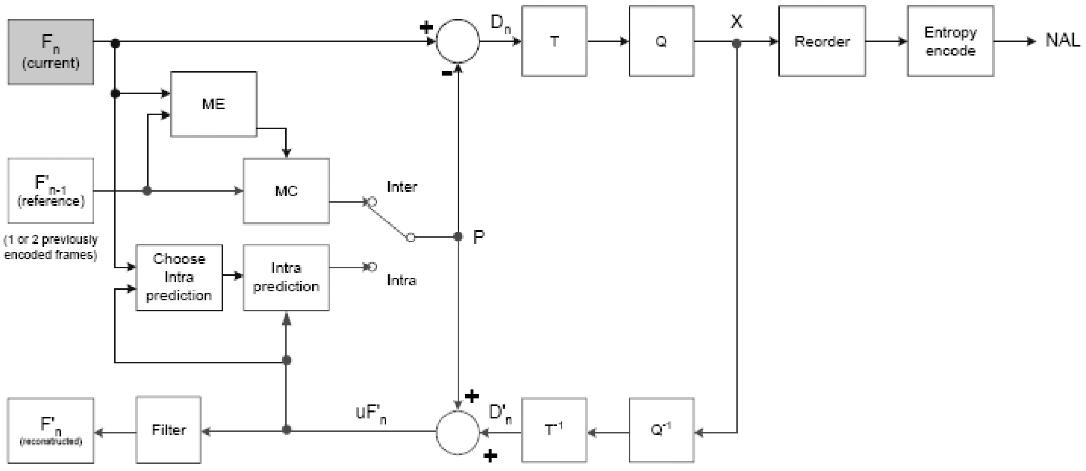


Fig 2.3: Block diagram of H.264/MPEG-4 AVC encoder.

Next section, we will discuss the architecture of motion estimation. And we will focus on the full-search variable block size motion estimation architecture.

2.2 Full-search Variable Block Size Motion estimation Architecture (FS-VBSME)

The computational requirements for motion estimation are heavy and a real-time video application usually requires a direct mapped hardware architecture. Direct mapped architectures also have important advantages in terms of reduced power dissipation. Full-search algorithms, typically, can be implemented using regular 1-D or 2-D systolic or systolic-like architectures as described by the paper [24]. 1-D systems offer a number of attractive features over their full 2-D counterparts, in particular much less complex data scheduling and simpler structures. These architectures are therefore attractive for portable devices because of their lower silicon area. The paper [25] has also demonstrated that flexible 1-D systems can be used to implement other fast matching algorithms, such as a three-step search (TSS) and pixel subsample.

To date, conventional VLSI architectures for computing variable block size motion estimation (VBSME) have been based on 2-D processor systems. For example, the architecture in the paper [26] uses a 2-D array with appropriate through masking of process elements (PEs). However, this results in low processor utilization. The architecture in the paper [27] uses a smaller 2-D array with partial-sum the sum of absolute difference (SAD) calculations performed sequentially using the smallest block size, 8×8. However, these architectures do not incorporate the capability to process all the variable block sizes (VBSs).

In H.264/MPEG-4 AVC [4], a macroblock is further segmented with the smallest block size being 4×4, as shown in Fig 2.4. This has two modes, the macroblock mode and the 8×8 mode. They are illustrated in Fig 2.5(a) and (b), respectively. VBSs must be accommodated, namely 4×4, 4×8, 8×4, 8×8, 16×8, 8×16, and 16×16. Referring to Fig 2.5(b), it will be noted that there are four quarter-blocks in a macroblock, each of which contains nine block patterns i.e., a total of 36 block patterns. However, observed in Fig 2.5(a), each macroblock contains another nine block patterns, with four of the 8×8 blocks common with the equivalent 8×8 blocks in Fig 2.5(b). Therefore, the total number of block patterns, to be processed is 36+9-4=41 i.e., a total of 41 motion vectors.

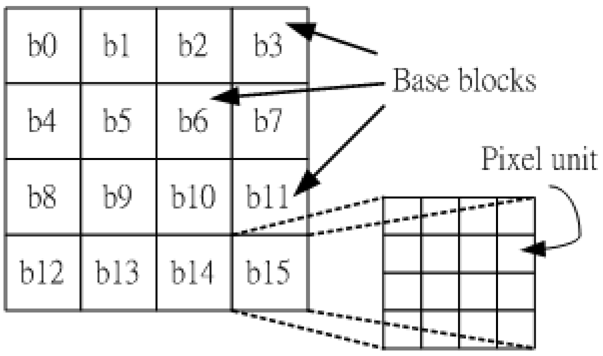


Fig 2.4: Segmented macroblock: Base block is 4x4. (Excerpted from [21])

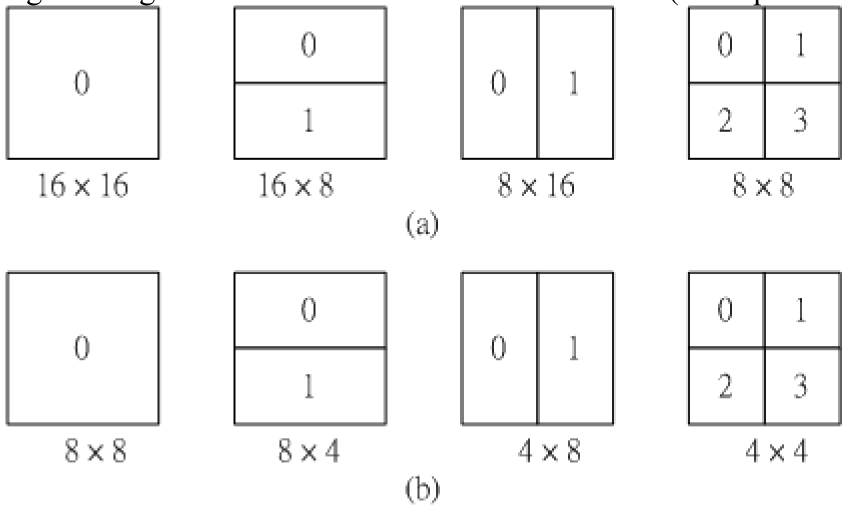


Fig 2.5: Variable block sizes in H.264/MPEG-4 AVC [4] (a) Macroblock mode. (b) 8×8 mode. (Excerpted from [21])

The architecture presented in the paper [21] is based on 1-D array processor, in this case containing 16 PEs, in general, N for an N×N macroblock. This is summarized in Fig 2.6. A key aspect of the approach proposed is that it incorporates within the basic PE the means to accumulate the partial SAD values through shuffling. The scheduling of the current macroblock data (CMD) and search region data (SRD) is similar to a conventional 1-D architecture [28] with the CMD arranged in a raster scan sequence and the SRD arranged in a dual raster scan sequence. They apply this approach to the macroblock shown in Fig 2.4 and result in 16 SADs being computed, each with block size 4×4. The stored SADs are then re-used to compute SAD values for other block sizes. This is done by shuffling and combining the computed sub-block SAD values appropriately to derive SADs for each of the other larger blocks sizes. For example, the results of two 4×4 sub-block computations can be combined to derive results for a 4×8 or 8×4 computation, and so on. This avoids the need to compute each of these from scratch and allows the overall computational requirements to be significantly reduced by avoiding the need to derive sub-block computation values that already have been established. As discussed below, this allows up to 41 VBS SAD values to be processed in a single processor.

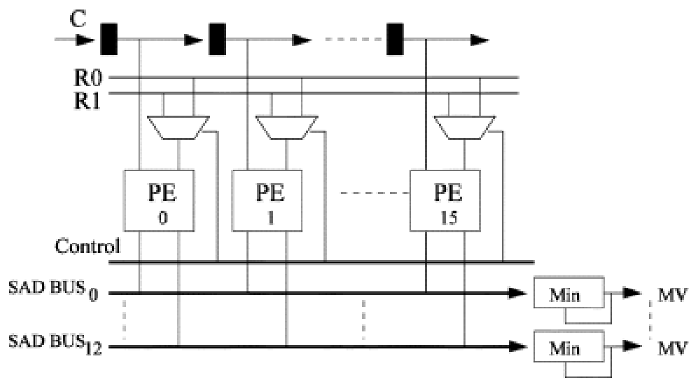


Fig 2.6: One-dimensional array VBSME architecture in the paper [21]. (Excerpted from [21])

These computations of VBS's SAD are performed using the internal PE circuitry. Details of which are shown in Fig 2.7. This uses a three stage process, provides 100% PE efficiency and allows SAD value computation to be choreographed directly with the data flow within the image. The first stage in the PE contains hardware to derive absolute difference values between the CMD and the SRD. These values are then latched to a second stage where they are multiplexed appropriately and stored in one of eight registers. The function of the registers and Mux C is to ensure that once computations have been performed these are stored and fed back in the correct order to compute the overall AD values for each of the sub-blocks 4×4. The function of the second stage of the array is twofold. The first is simply to pass, on successive cycles, the values 4×4 downwards through the PE cell. The second is to combine these values appropriately to compute results of larger block size such as 8×4, 4×8 etc. This is done in a similar manner to stage 1 i.e., shuffling and combining results using a combination of multiplexing and adder circuitry, with results and intermediate results, in this case, being assigned to one of six registers, and so on. The third stage in the PE has a similar function to the second stage, but in this case feeding back the SAD values stored in the stage 2 registers via Mux D and Mux F. The net result is that by clock cycle 261 (256 cycles plus 5 cycles internal cell latency) all 41 candidate MVs are available from each PE.

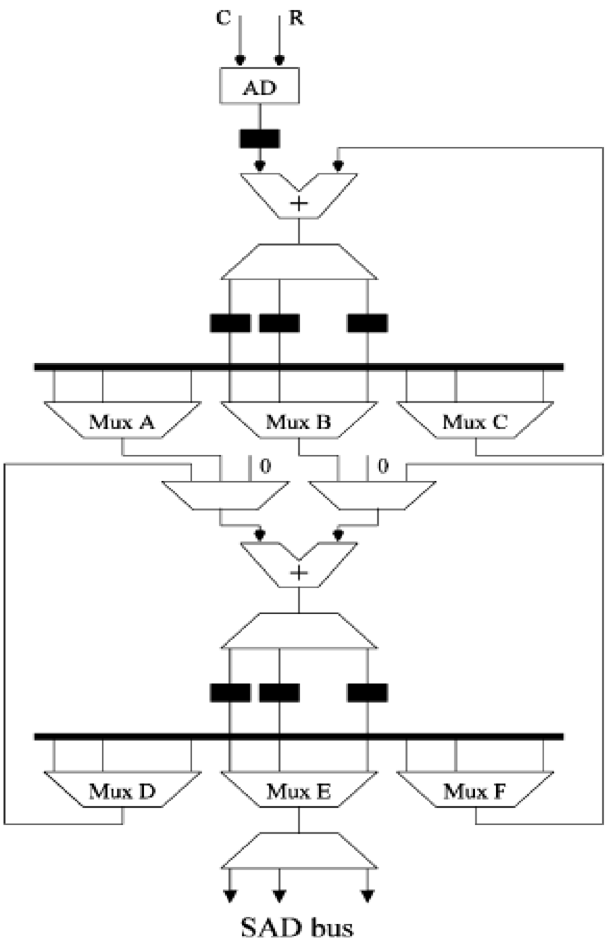


Fig 2.7: Process element of the FS-VBSME architecture in [21] (Excerpted from [21])

Once all the values from an image block have been input then the data from a new block can immediately be input to each PE. This thus provides a continuous streaming process that directly synchronizes with a constant flow of image data and means that each PE is 100% utilized. With 16 PEs working concurrently, the architecture described allows a total of 256 candidate MVs (16 16 search region) per sub-block to be processed in parallel with each PE producing all the information needed for a full search every 256 clock cycles— the same as existing architectures. However, in this case, this is done through the derivation of 41 MVs rather than one for each macroblock. Repeating this further 16 times means that up to 4096 clock cycles are required to complete a full search.

2.3 The Subsample Algorithm Using Fixed Pattern

The subsample algorithms [11]-[17] can not only easily combine with other approaches mentioned above but also reduce the number of matching points with flexibly changing subsample rate. For the fixed pattern, we can be sure that the power dissipation will be down by subample scale. But different patterns will case the different degreed of quality degradations.

Bierling used an orthogonal sampling lattice with a 4:1 subsample [11]. The pattern they used is the quarter pattern, shown in Fig 2.8 (b) [14]. The quarter pattern can save the power dissipation for 75%. And the paper [12] uses four different quarter pattern to the different search area. They are based on motion-field and pixel subsample. They first determine a subsample motion field by estimating the motion vectors for a fraction of the blocks. The motion vectors for these blocks are determined by using only a fraction of the pixels at any searched location and by alternating the pixel subsample patterns with the searched locations. They then interpolate the subsample motion field so that a motion vector is determined for each block of pixels. Fig 2.9 (a) shows a block of 8×8 pixels with each pixel labeled a, b, *c*, or d in a regular pattern. We call pattern A the subsample pattern that consists of all the "a" pixels, as the quarter pattern. Similarly, patterns B, C, and D are the subsample patterns that consist of all the "b", "c", and "d" pixels, respectively. If only the pixels of pattern "A" are used for block matching, then the computation is reduced by a factor of 4. However, since 3/4 of the pixels do not enter into the matching computation, the use of this subsample pattern alone can seriously affect the accuracy of the motion vectors. To reduce this drawback, they proposes using all four quarter patterns, but only one at each location of the search area and in a specific alternating manner. Fig 2.9 (b) shows some pixels forming part of the search region in the previous frame. The pixels are labeled 1, 2, 3, and 4 in a regular pattern. The labeling of the pixels refers to which of the four quarter patterns of Fig 2.9 (a) is to be used for computing the matching at that location. That is, when computing the match at locations labeled 1 (i.e., when the upper-left pixel of the block to match falls on those locations), pattern A is used. Similarly, pattern B, C, or D is used when computing the match at locations labeled 2, 3, or 4.

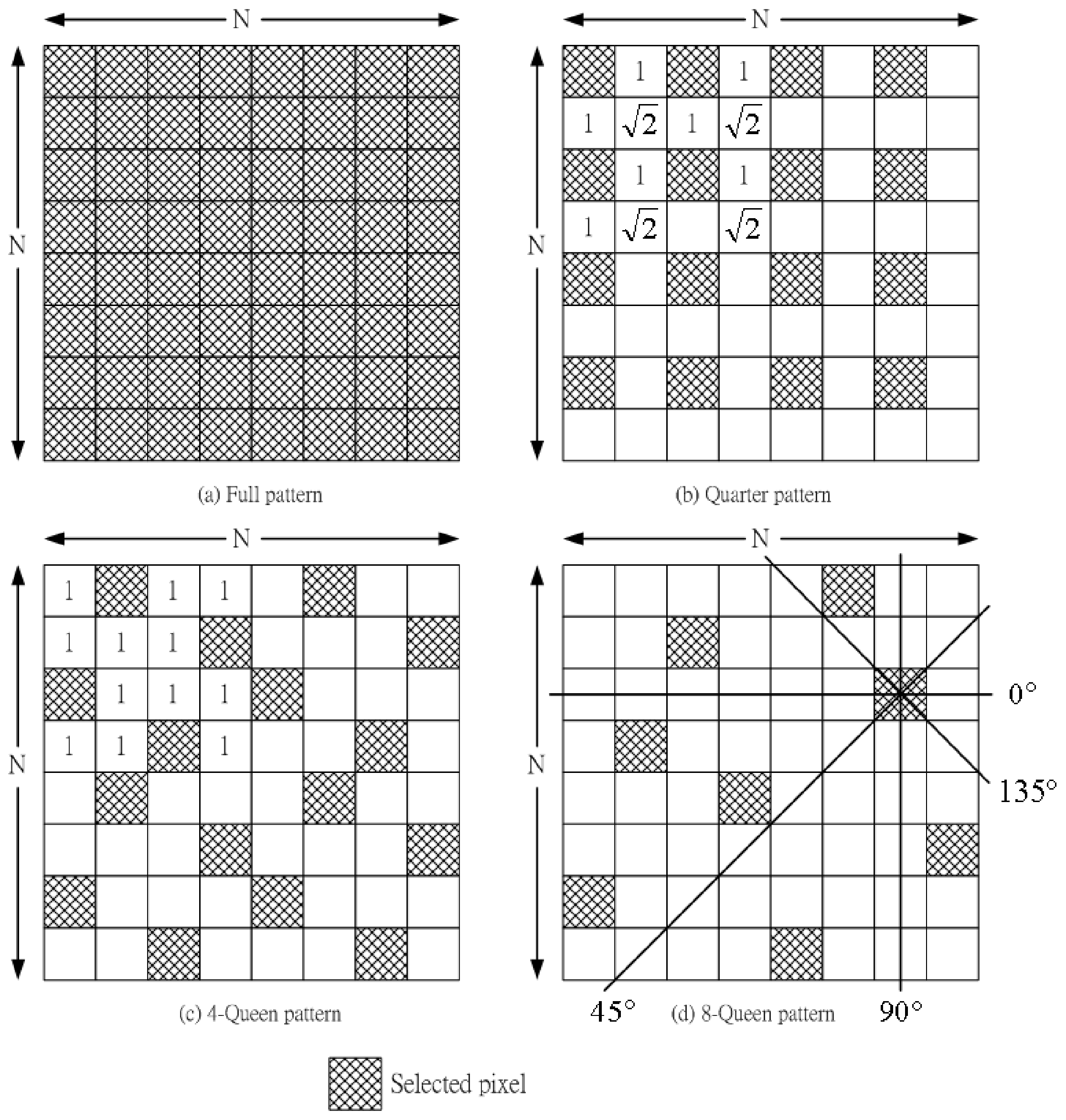
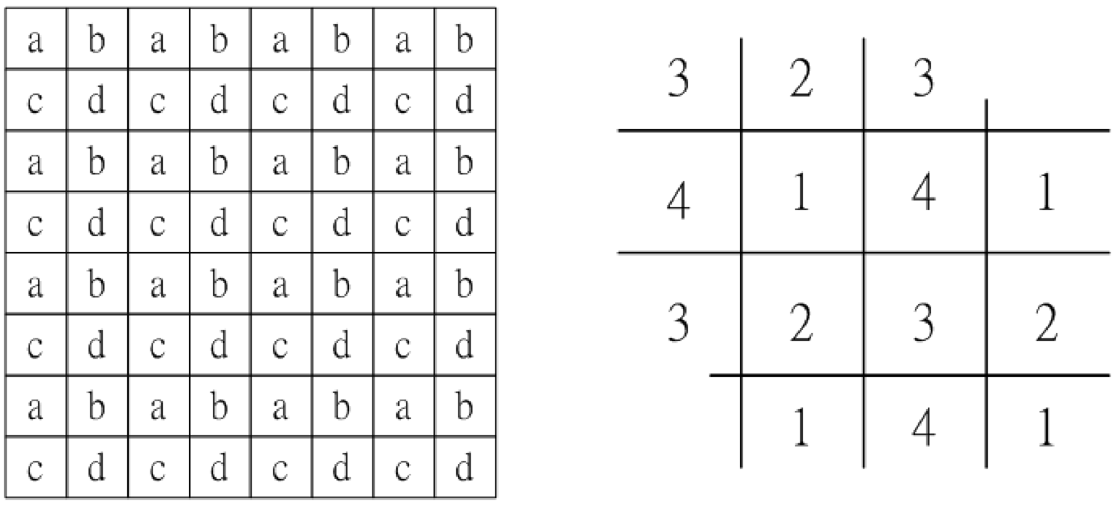


Fig 2.8: Pixel patterns for decimation. (a) Full pattern with N×N pixels selected. (b) Quarter pattern uses 4:1 subsampling. (c) Four-queen pattern is tiled with four identical patterns. (d) Eight-queen pattern. (c) and (d) are derived from the N-queen approach with N = 4 and N = 8, respectively. (Excerpted from [14])



(a) (b)

Fig 2.9: (a) Patterns of pixels used for computing the matching criterion with a 4 to 1 subsample ratio. (b) Alternating schedule of the four pixel subsample patterns over the search area (Excerpted from [12])

We can analyze the subsample pattern with the spatial homogeneity and directional coverage [14]. The spatial homogeneity is measured by the average and variance of spatial distances from each skipped pixel to its nearest selected pixel where N is the dimension of the block, and $S(x,y)$ indicates the coordinates of the selected pixel nearest to the pixel at the position (x,y). K is the number of the selected pixels. Smaller μ_d and σ_d^2 indicate a more spatially homogeneous sampling lattice. An edge is defined as a line passing through the sampling grids in any of $0°, 45°, 90°$ and $135°$ directions in Fig 2.8 (d). The directional coverage is measured as the percentage of edges that at least one of the selected pixels exists on an edge. Table I shows that the quarter pattern has less spatial homogeneity and lacks half of the coverage in the specified directions. To address the issues of spatial homogeneity and directional coverage, the paper [14] construct a new N-queen sampling lattice Fig 2.8 (c) and (d).

$$\mu_d = \frac{1}{(N^2 - K)} \sum_{x=1,y=1}^{N} \|(x,y) - S(x,y)\|$$

$$\sigma_d^2 = \frac{1}{(N^2 - K)} \sum_{x=1,y=1}^{N} \left(\|(x,y) - S(x,y)\| - \mu_d\right)^2$$

(Excerpted from [14])

In the paper [14], to fully represent the spatial information of a N×N block, it is required that at least one pixel should be selected for each row, column, and diagonal. To satisfy such a constraint, the solution is identical to the problem of placing queens on a chessboard, which is referred to as -queen pattern. For a N×N block, as shown in Fig 2.8 (c) and (d), every pixel of the N-queen pattern occupies a dominant position, which is located at the center. All the other pixels located on the four lines in the vertical, horizontal and diagonal directions are removed from the list of the selected pixels. With such elimination process, there is exactly one pixel selected for each row, column, and (not necessarily main) diagonal of the block. Thus, the N-queen patterns present a subsample lattice that can provide N times of speedup improvement. Despite the randomized lattice, the paper [15] designed compact data storage architecture for efficient memory access and simple hardware implementation for the N-queen patterns.

Table 1
Comparison of the sampling lattices an 8×8 block in measuring the directional coverage, four orientation described in Fig 2.8 (d) are used for horizontal, vertical and diagonal directions, there are eight, eight, and 15 possible edges, respectively, while for the diagonal directions, there are 15 possible edges. (Excerpted from [14])

Pattern	**Spatial homogeneity**			**Directional coverage (θ)**			
	μ_d	σ_d^2	σ_d / μ_d	0°	45°	90°	135°
Full	0	0		8/8	8/8	15/15	15/15
Quarter [11]	1.14	0.04	17.16%	4/8	4/8	7/15	7/15
Hexagonal [13]	1.03	0.11	11.07%	4/8	8/8	12/15	12/15
4-Queen [14]	1	0		8/8	8/8	10/15	10/15
8-Queen [14]	1.32	0.14	28.77%	8/8	8/8	8/15	8/15

2.4 The Subsample Algorithm using adaptive Pattern

The approach using the fixed patterns could possibly be able to obtain a good estimation of motion when the intensity of the block is nearly uniform. However, in the case of high activity blocks, some details may be neglected. Thus, it probably would introduce excessive prediction error. The paper [16] is based on the fact that high activity in spatial domain such as edges and texture mainly contributes to the MAD criterion. We can vary the number of selected pixels based on the image details. In other words, we can use fewer pixels when the block has uniform intensity. But in the high activity block, more pixels can be employed for the MAD matching criterion. This adaptive approach [16] can reduce the prediction error compared with standard pixel decimation [11]-[15]. In the algorithm [16], they used the relationship between a pixel and its neighbors to select the most representative pixels. For example in 8×8 block size, initially, nine pixels are selected as shown in Fig 2.10 (a). The 8 x 8 pixel block is divided into nine regions, depicted in Fig 2.10(b), and each region has its corresponding central pixel. In each region, the difference D_k is defined the difference between central pixel and one of its neighbor pixels. If the difference D_k is greater than threshold, this pixel is selected. We have used block size of 8 x 8 as an example for the description of the proposed algorithm in the paper [16], however, the extension of the proposed scheme to a large block size, say 16 x 16, is straightforward.

$D_k(h,k) = |I_k(h,k) - I_K|$, where (h, k) is the location of the neighbor pixel in region K, with (h, k) as the displacements from the central pixel I_k.

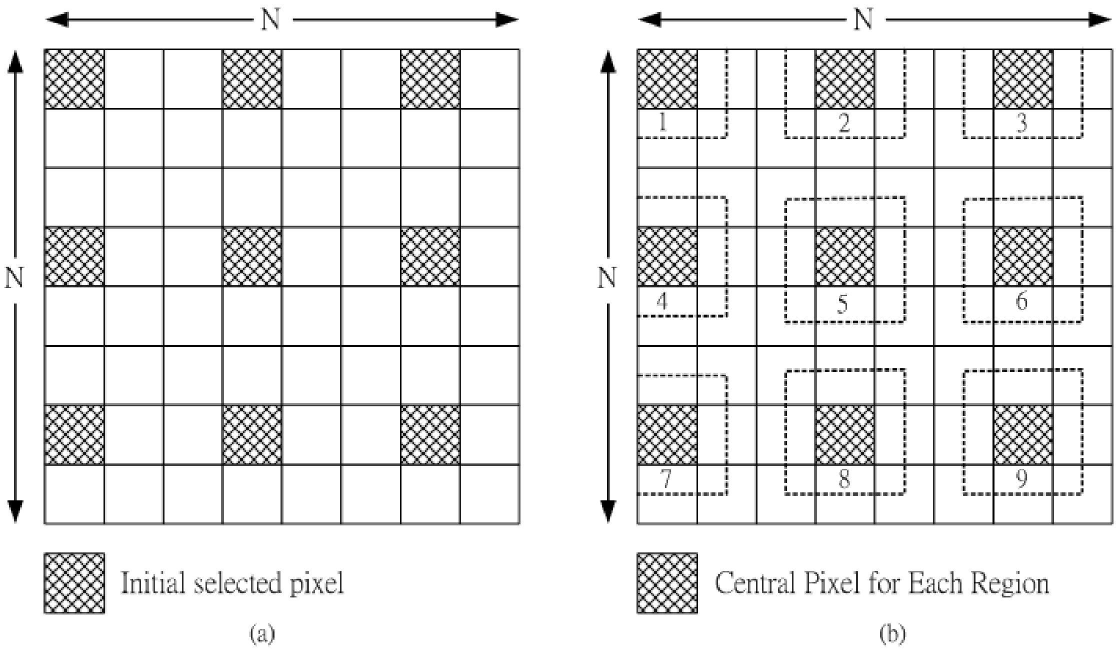


Fig 2.10: Adaptive pixel selection (a) nine selected pixels. (b) The selected pixels in (a) are considered as the central pixel for each region, the dotted lines indicate the neighbor pixels of respective central pixels in each region. (Excerpted from [16])

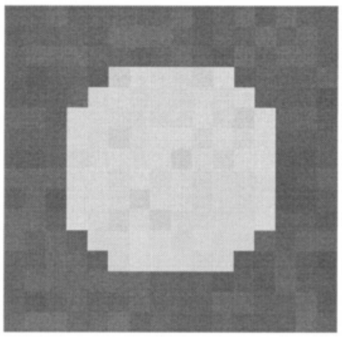


Fig 2.11: An edge in a 16×6 block for testing the subsample algorithm [17]

About the paper [16], their scheme still requires an initial uniform division of a block, and therefore the pattern is locally adaptive. The pixel-decimation algorithm proposed in the paper [17] also utilizes edge information. Compared to Chan's method [16], it extends the adaptivity from local to global. To realize global adaptivity, the algorithm [17] looks directly for edge pixels instead of requiring an initial uniform division of a block. This task [17] is made easier in a 1-D space with the help of Hilbert scan [29]. The Hilbert scan was named after the great German mathematician Hilbert, who found the simplest family of curves (Hilbert curves) that pass through all the grid points only once in a 2-D space [29]. The Hilbert scan, defined as a scan of a 2-D image through one of its Hilbert curves, is equivalent to a depth-first scanning of a quad-tree representation of the 2-D image. Some interesting features of this scan method used in previous applications include: 1) it is easier to extract clusters in an image with a Hilbert scan than other scan methods, e.g., row scan, row-prime scan, Morton scan, etc., and 2) it preserves 2-D coherence [30]–[35]. In addition, Kamata has shown that edge information in a 2-D image is preserved in its 1-D Hilbert-scan sequence, and has demonstrated an effective compression of 2-D images by compressing their 1-D sequences using the edge information [34]. The compressed images have a similar visual quality to that of the JPEG images at a high compression rate.

To illustrate how edges are detected in a 1-D Hilbert sequence, Fig 2.11 shows a 2-D block with a closed circular edge, and Fig 2.12 (a) is the 1-D Hilbert sequence converted from the block in Fig 2.11. If edge pixels are defined at where pixel intensity changes the most, 22 edge pixels can be located in Fig 2.12 (a). All of the 22 pixels, when mapped back to 2-D, appear evenly distributed on the circular edge as shown in Fig 2.12 (b). For comparison, Fig 2.12 (c) is the 1-D row sequence converted from the same block in Fig 2.11. Although the row sequence contains 20 edge pixels, they all appear at the left and right vertical portion of the circular edge, and none appear on the upper and lower horizontal edges, as shown in Fig 2.12 (d). In general, the Hilbert scan not only provides edge information with little directional preference, but also preserves pixel coherence more effectively than other scan methods. In contrast, row scan, typical of many other scan methods, may miss edges due to its scan direction. Based on edge information in 1-D Hilbert sequences [29], the algorithm [16] selects pixels at which the matching criterion is evaluated.

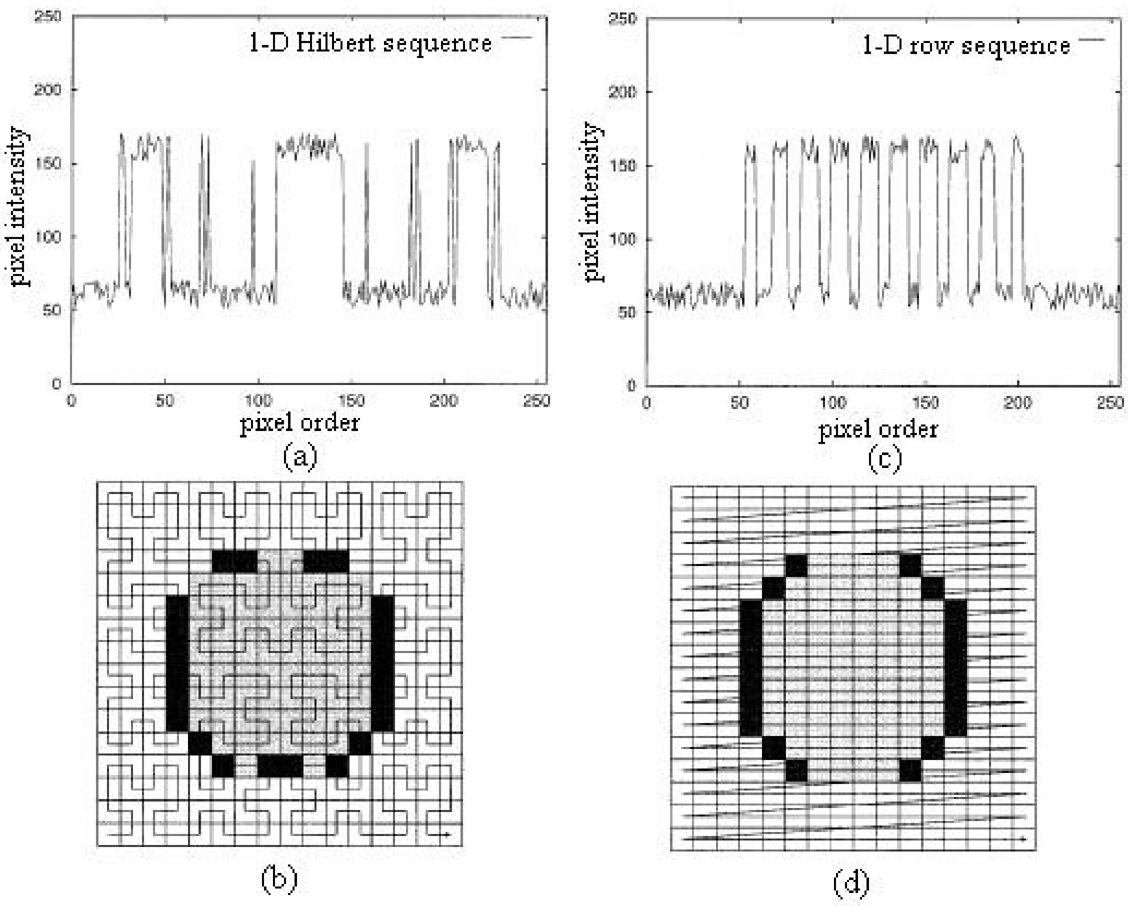


Fig 2.12: (a) 1-D Hilbert sequence converted from Fig 2.11. (b) Edge pixels detected from 1-D Hilbert sequence. (c) 1-D row sequence converted from Fig 2.11. (d) Edge pixels detected from 1-D row sequence (Excerpted from [17])

The paper [39]-[47] proposed that the general subsample algorithm has aliasing problem when it is in high subsample rate. The aliasing problem leads to considerable quality degradation because the high frequency band is messed up. To alleviate the problem, he uses edge extraction techniques to separate the edge pixels from a macro-block and then perform subsampling to the remaining pixels.

Although the subsample algorithms [11]-[17] and [39]-[47] reduce the number of matching points with flexibly changing subsample rate to save the power dissipation, they will cause the aliasing problem in high frequency band. Next, we will explain aliasing problems in subsample algorithms.

Chapter 3 Aliasing Problems in Subsample Algorithm

In this chapter, we present a generic subsample algorithm in which the subsample ratio ranges from 16-to-2 to 1-to-1. We use the fixed subsample ratio to test the video sequences and observe that the quality degradation is dependent on the video content. That is because the subsample process will induce the aliasing in high frequency band [18]. The video with high motion have the high quality degradation. On the contrary, the video with low motion have the low quality degradation. We discuss aliasing problems in subsample algorithms in this chapter. From DSP theory [18], we will kwon the reason why aliasing problems happen. We also can find this situation in every GOP of the video sequence. We assume that the frames in the GOP are very correlative. So we can see the GOP as a control unit to adaptively select the best subsample ratio. Aliasing problems can be solved more accurate. And the video sequence "Table" is the example to explain the relationship between the quality degradation and the video content in aliasing problems in high frequency band.

3.1 Generic Subsample Algorithm

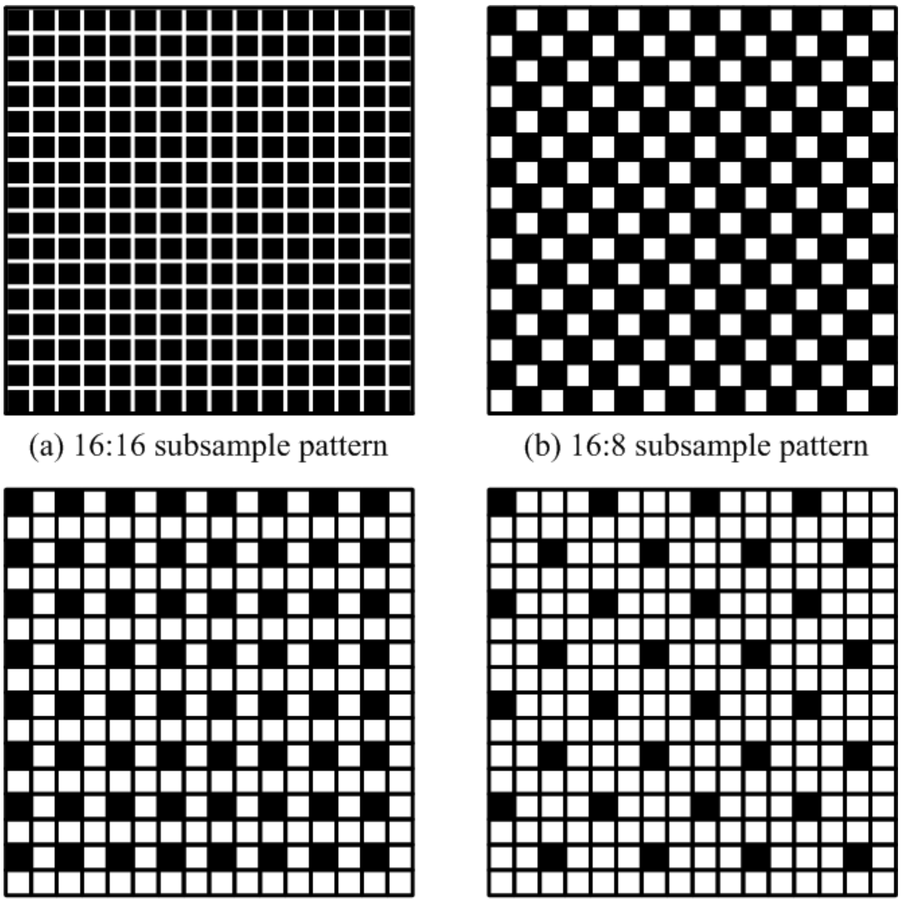
Here, we present a generic subsample algorithm in which the subsample ratio ranges from 16-to-2 to 1-to-1. The basic operation of the generic subsample algorithm is to find the best motion estimation with less SAD computation. The generic subsample algorithm uses Eq.1 as a matching criterion, called as subsample sum of absolute difference (SSAD), where the macro-block size is N-by-N, $R(i, j)$ is the luminance value at (i, j) of the current macro-block (CMB). The $S(i+u, j+v)$ is the luminance value at (i, j) of the reference macro-block (RMB) which offsets (u, v) from the CMB in the searching area *2p*-by-*2p*. $SM_{16:2m}$ is the subsample mask for the subsample ratio *16*-to-*2m* as shown in Eq.2. The subsample mask $SM_{16:2m}$ is generated from basic mask as shown in Eq.3. Fig 3.1 shows the subsample patterns of 16:16, 16:8, 16:4 and 16:2 generated by the generic subsample algorithm respectively. The other subsample patterns are omitted because they also can be generated from the generic subsample algorithm and the subsample patterns of 16:16, 16:8, 16:4 and 16:2 shown in Fig 3.1 are symmetry and their scale is power of two.

$$SSAD_{SM_{16:2m}}(u,v) = \sum_{i=0}^{N-1}\sum_{i=0}^{N-1}\left| SM_{16:2m}(i,j)\cdot\left[S(i+u, j+v) - R(i,j)\right]\right|, \quad for \quad -p \le u,v \le p-1 \qquad (1)$$

$$SM_{16:2m}(i,j) = BM_{16:2n}(i \ \ \text{mod} \ \ 4, j \ \ \text{mod} \ \ 4), \quad m = 1,2,3,4,5,6,7,8 \qquad (2)$$

$$BM_{16:2m} = \begin{bmatrix} u(m-1) & u(m-5) & u(m-2) & u(m-6) \\ u(m-7) & u(m-3) & u(m-8) & u(m-4) \\ u(m-2) & u(m-5) & u(m-1) & u(m-6) \\ u(m-7) & u(m-3) & u(m-8) & u(m-4) \end{bmatrix} \qquad (3)$$

$$when \quad u(n) \quad is \quad a \quad step \quad function: \quad that \quad is, \quad u(n) = \begin{cases} 1, & for \quad n \ge 0 \\ 0, & for \quad n < 0 \end{cases}$$



(a) 16:16 subsample pattern (b) 16:8 subsample pattern

(c) 16:4 subsample pattern (d) 16:2 subsample pattern

Fig 3.1: The subsample patterns with 16:16, 16:8, 16:4 and 16:2 respectively

Given a subsample mask generated from Eq.3, the computational cost of SSAD can be lower than that of SAD. Hence, the generic subsample algorithm can achieve the target of power-saving with flexibly changing subsample ratio. However, the generic subsample algorithm suffers form aliasing problems in high frequency band. Aliasing problems will degrade the validity of motion vector (MV) and result in visual quality degradation to video sequences obviously.

We use the fixed subsample ratio from 16:2 to 1:1 to experiment the twelve video sequences [37] in H.264/MPEG-4 AVC [4] coder with JM9.2 [36]. Here, we define one group of picture (GOP) is fifteen frames, frame rate is 30 frames/s, the bit rate is 450k bits/s and initial Qp is 34. We can observe the quality degradation of the video sequence in the Fig 3.2 and Fig 3.3. In Fig 3.2, the video sequence "dancer" has the most quality degradation. The most quality degradation of “dancer” is 0.93 dB using the fixed subsample pattern of 16:2. And the other sequences in the Fig 3.2 have the excess of the 0.3dB quality degradation. We call those video sequences as high or normal motion video sequences. They have the high degree of high-frequency characteristic. We have to choose higher subsample ratio to keep the quality degradation acceptable. The quality degradation of 0.5 dB is empirically reasonable for the perceptual tolerance of decompressed visual quality in video coding community. Therefore, we can conclude that we must use the difference subsample ratio to keep the quality degradation acceptable. It is not enough to only use the fixed subsample pattern for all video sequences. From Fig 3.3, the most quality degradation is not over the 0.35 dB. We call those video sequences as low motion video sequences. They have the low degree of temporal variation. The temporal variation in the video means the degree of object-moving. The degree of object-moving is faster, and the temporal variation is stronger. Although the low subsample ratio cause aliasing in high frequency band, the degree of temporal variation will affect the degree of quality degradation. If the temporal variation is strong, aliasing problems will degrade the validity of motion vector (MV) and result in visual quality degradation to video sequences obviously. On the contrary, if the temporal variation is weak, aliasing problems will not degrade the validity of motion vector (MV) although the low subsample ratio still cause aliasing in high frequency band. That is because we do not need the high frequency band information to find the motion vector when the degree of object-moving is slow. Hence, using higher subsample ratio to reduce the prediction residual is necessary when temporal variation is stronger. For those video sequences, unlike the video sequences in the Fig 3.2, we can use the lowest the subsample ratio to save the largest power dissipation. That is because the quality degradation is acceptable.

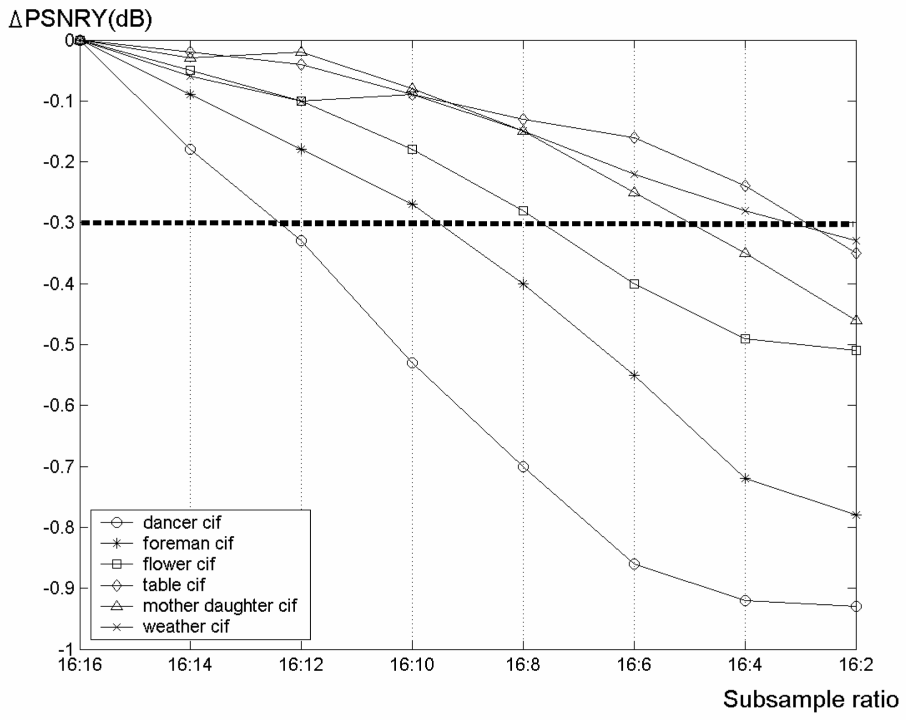


Fig 3.2: The results ΔPSNRY of testing sequences “Dancer“, “Foreman“, “Flower“, “Table“, “Mother Daughter“ and “Weather“

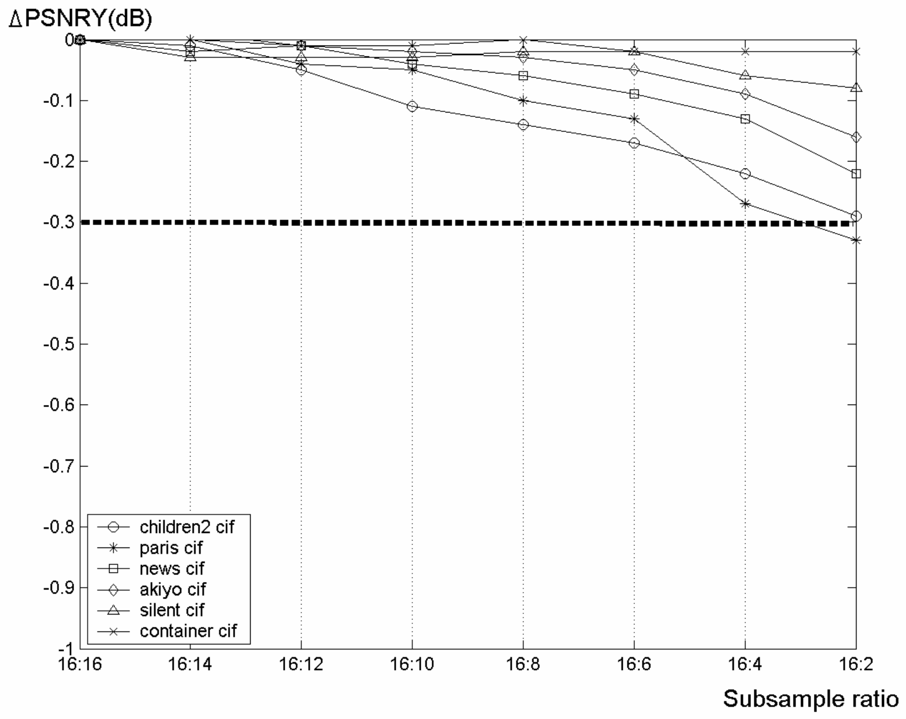


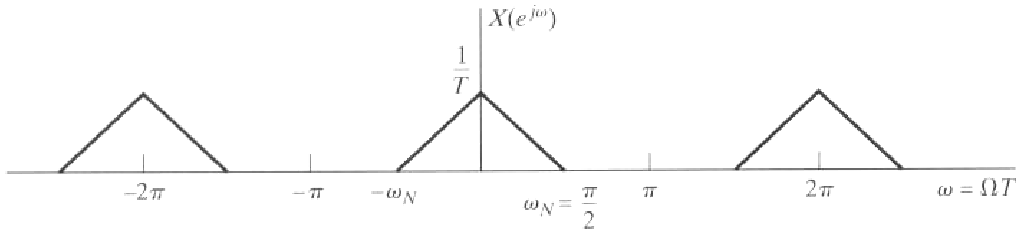
Fig 3.3: The results ΔPSNRY of testing sequences “Children“, “Paris“, “News“, “Akiyo“, “Silent“ and “Container“

If we use the fixed subsample ratio for all the twelve video sequences and keep the quality degradation acceptable, we have to choose the 16:12 subsample ratio. We will have the large power dissipation. We will waste the power in the low motion video sequences. Hence, the adaptive subsample ratios have to be used on power-saving and the acceptable quality degradation.

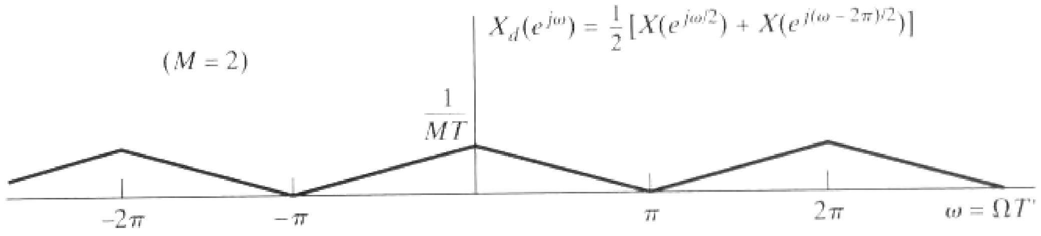
3.2 Aliasing Problems in High Frequency Band

As mentioned above, the generic subsample algorithm has aliasing problems for low subsample ratio and leads to considerable quality degradation because the high frequency band is messed up.

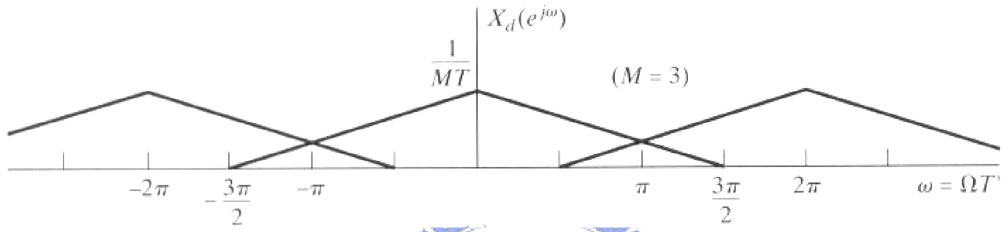
The subsample process is like the down-sampling process in DSP theory [18]. In general, the operation of reducing the sampling ratio will be called down-sampling. Down-sampling is illustrated in Fig 3.4. We assume that the Fig 3.4 (a) is the conceptual spectrum of a macroblock in a frame of a video sequence. If this macroblock is down-sampling by 2, then his new conceptual spectrum will be Fig 3.4 (b). Because the original conceptual spectrum is low bandwidth and the down-sampling ratio is high, the aliasing don’t happen in this case. If the down-sampling ratio becomes 3, the aliasing will happen shown in Fig 3.4 (c). The aliasing in the high frequency band will case the motion estimation is no accurate. Aliasing problems affect the variance of the prediction residual under a fixed bit-rate constraint. The variance of the prediction residual affects the compression quality. Therefore, in order to efficiently alleviate aliasing problems to satisfy the visual quality under the quality threshold of 0.5 dB for general video sequences, adaptively selecting the suitable subsample ratio according to the degree of temporal variation in the content is imperative.



(a) The original conceptual spectrum


$X_d(e^{j\omega}) = \frac{1}{2}[X(e^{j\omega/2}) + X(e^{j(\omega-2\pi)/2})]$
$(M = 2)$
$\frac{1}{MT}$
-2π
$-\pi$
π
2π
$\omega = \Omega T'$

(b) Down-sampling by 2



(c) Down-sampling by 3 (with aliasing problem)

Fig 3.4: Frequency-domain illustration of down-sampling (Excerpted from [18])

In all the twelve video sequences, the video sequences with low motion have no aliasing problems, like “Children“, “Paris“, “News“, “Akiyo“, “Silent“ and “Container“. About these video sequences, we can use the low subsample ratio in order to save the large power dissipation without the high quality degradation. That is because these video sequences have low frequency distribution and weak temporal variation. Although we down-sample the video sequences by large number, the high frequency don’t be messed up. On the other hand, the video sequences with high and normal motion have aliasing problems. If we down-sample those video sequences by large number, the quality degradation will be large. For example, the video sequence “Dancer” will case 0.93 dB quality degradation using the 16:2 subsample ratio. That quality degradation is very serious and can’t be acceptable.

In the video sequence, temporal variation is in proportion to the degree of object-moving and the degree of object-moving means moving motion vector count (MMVC) between adjacent frames. For a fast motion video sequence, the degree of temporal variation is stronger than a normal motion or slow motion video sequence. Hence, the higher subsample ratio can more efficiently reduce the prediction residual to maintain the visual quality in a bit rate control. On the contrary, the lower subsample ratio results in more noticeable aliasing because of increasing the inaccurate moving motion vector count (MMVC) and furthermore increases prediction residual to degrade the visual quality in a bit rate control. Therefore, the high subsample ratio is necessary for a fast motion video. And the low subsample ratio is also suitable for a normal or slow motion video.

We define the null motion vector count to reflect the spectrum in frequency deomain. We use the lower of subsample ratio for the larger of NMVC. On the other hand, we use the higher of subsample ratio for the smaller of NMVC. The overhand of the point for the spectum is only an extra counter for implementation. But in the encode system, we get the NMVC value after encoding. But we have to decide the subsample ratio before encoding. In order to solve this problem, we use the GOP as a processing unit. For any video sequence, the degree of temporal variation in this video sequence is not the same. We also can find this situation in every GOP in the video sequence. The GOP is like the small size of video sequence, about 0.5 second. Because the number of frames in the GOP is smaller than that in the video sequence, we assume that the frames in the GOP are very correlative. So we can see the GOP as a control unit to adaptively select the best subsample ratio. If we do that, we can get the degree of temporal variation more accurate. Aliasing problems can be solved more accurate. So We get the NMVC of the first P-frame in the GOP as the point to select the suitable subsample ratio for this GOP. The more detail will be discuss in next chapter.

We take the video sequence “Table” for example. To particularly analyze the results of visual quality degradation with different subsample ratios for a video, the normal motion video sequence “table” is simulated in H.264/MPEG-4 AVC [4] coder with JM9.2 [36]. Here, we define one group of picture (GOP) is fifteen frames, frame rate is 30 frames/s, the bit rate is 450k bits/s and initial Qp is 34. Subsample ratios are 16:8, 16:4 and 16:2 respectively and can be generated from Eq.3. Fig 3.5 shows quality degradation results versus these subsample ratios. Fig 3.6 shows the NMVC value of the first P-frame in each GOP. The average quality degradation of ith GOP ($\Delta Q_{ith\ \ GOP}$) is defined as shown in Eq.4, where $PSNRY_{i\ \ FSME}$ is the average PSNRY of ith GOP using the full-search block-matching (FSBM). $PSNRY_{i\ \ SSR}$ is the average PSNRY of ith GOP with specific subsample ratio (SSR). From Fig 3.5, there exists the stronger temporal variation between third GOP and seventh GOP, hence, the lower subsample ratio leads to more obviously aliasing problems and results in higher quality degradation. We can see the NMVC values of these GOPs are larger in Fig 3.6. This can reflect the the aliasing problem. Furthermore, the tenth GOP has maximum quality degradation because of scene change. Although lower subsample ratio leads to more obviously aliasing problems in high frequency band, Fig 3.5 also shows that quality degradation is unobvious between eleventh GOP and twenty GOP because of the weaker temporal variation. In Fig 3.6, the NMVC values of these GOPs are small. So we can use the NMVC to select the suitable subsample ratio for the GOP.

$$\Delta Q_{ith\ \ GOP} = \left(PSNRY_{i\ \ FSME} - PSNRY_{i\ \ SSR}\right) \qquad (4)$$

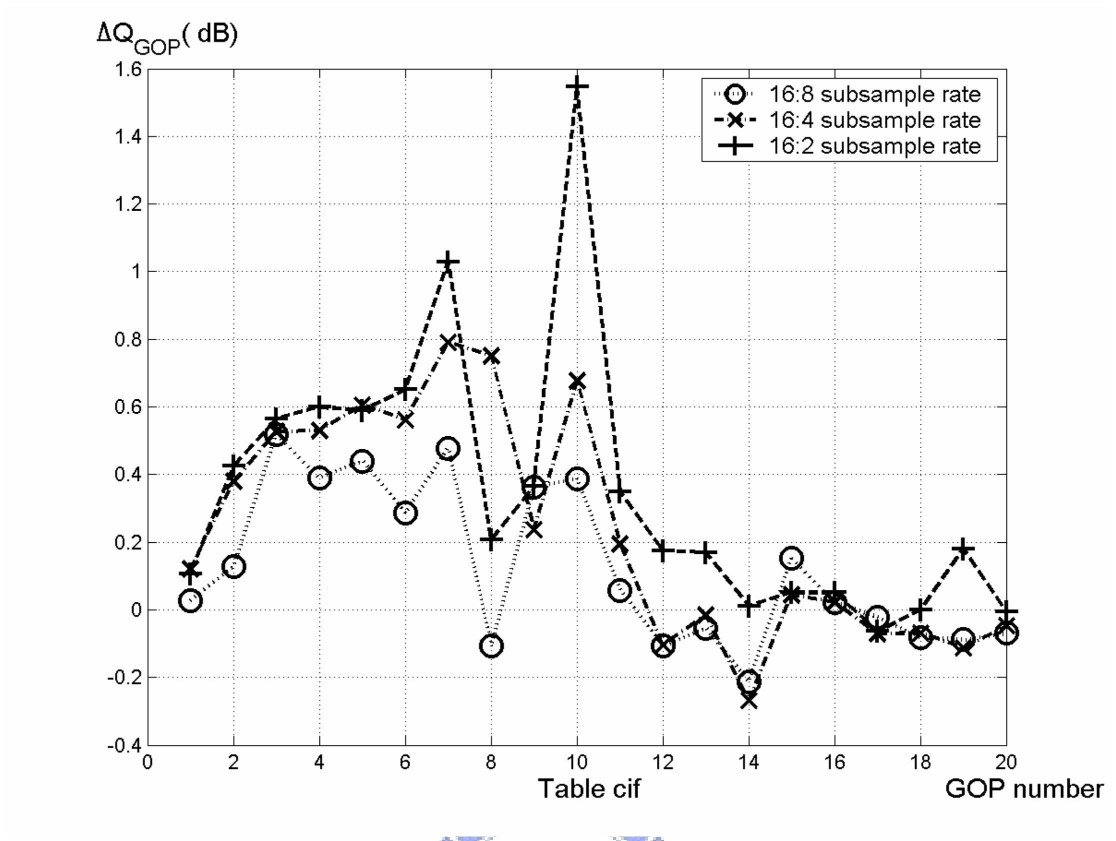


Fig 3.5: The diagram of ΔQ with 16:8, 16:4, 16:2 subsample ratios for table sequence.

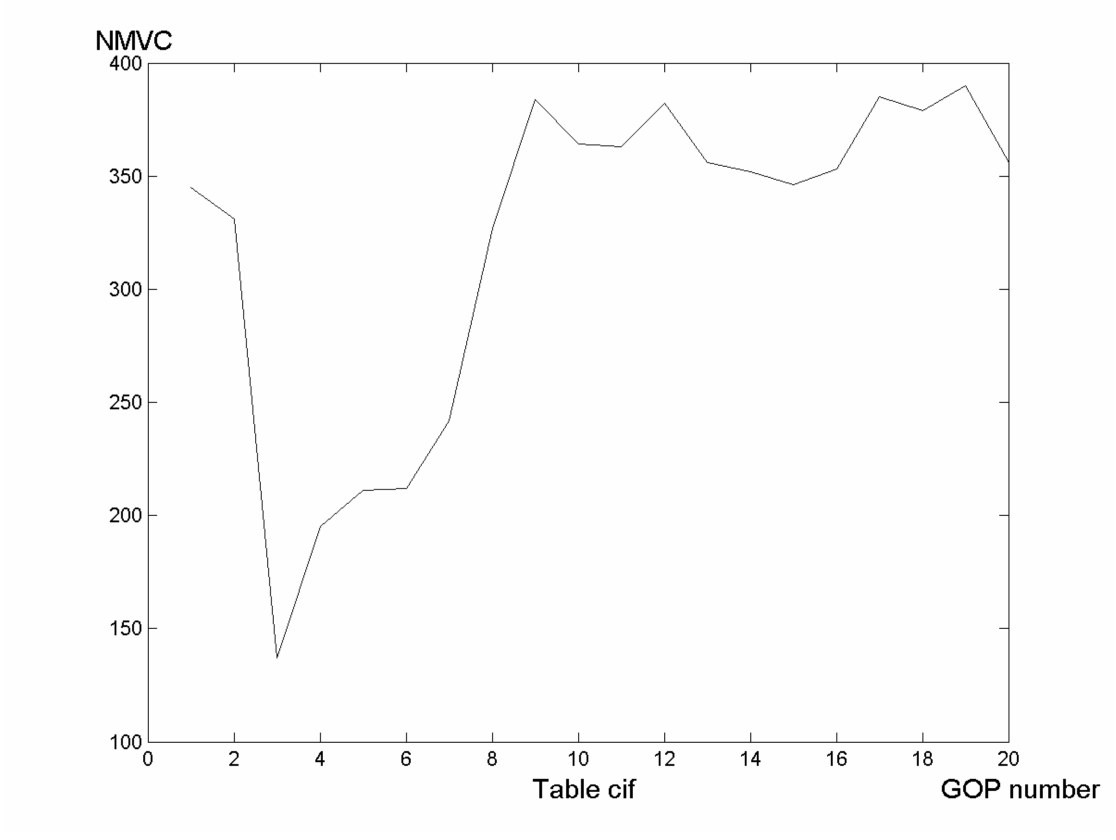


Fig 3.6: The NMVC of the first P-frame in each GOP for table sequence.

From Fig 3.5 and Fig 3.6, the various degrees of temporal variation are distributed over GOPs even though the “table” video sequence is regarded as a normal motion video. Therefore, in order to efficiently alleviate the aliasing problems, we need develop an adaptive motion estimation scheme and this scheme can adaptively supply the suitable sample ratio to each GOP according to the degree of NMVC in order to maintain the better visual quality in a bit rate control.

Chapter 4 Adaptive Motion Estimation with Variable Subsample Ratios

In this chapter, we describe the proposed algorithm in detail. We use a GOP as a process unit. We get the null motion vector count (NMVC) from the first P-frame in the current GOP. In order to make sure the correct of NMVC, we set the first P-frame in the GOP to use the full search motion estimation. That is 16:16 subsample ratio. According the value of NMVC, we select the suitable subsample ratio for the next 13 P-frames in the current GOP. Then the flowchart of the proposed algorithm is developed. Next, we provide four subsample ratios of 16:16, 16:8, 16:4 and 16:2 in order to let the proposed algorithm having better adaptive ability. The reason why to choose those subsample ratios is because they are symmetry and their scale is power of two. Final, we propose an adaptive subsample ratio threshold decision to set the compatible threshold values and get the optimal result. The static science is adopted in the adaptive subsample ratio threshold decision. We test the percentage of 90%-65% in the static data of the quality degradation versus NMVC to get the different threshold value. From the result of twelve testing video sequence, we take the 70% result as the optimal threshold value.

4.1 Proposed Algorithm Development

To efficiently alleviate the aliasing problems in subsample algorithm to maintain the visual quality under the threshold of 0.3 dB for general video sequences, we propose an adaptive motion estimation algorithm using variable subsample ratios and the proposed algorithm is based on the observation in Fig 3.4. From Fig 3.4, the temporal variation in a frame is in proportion to moving motion vector count (MMVC), meaning that it is in inverse proportion to null motion vector count (NMVC). Therefore, we use one GOP as a processing unit and calculate the NMVC of the first P-frame in a processing unit. Next, we compare NMVC with threshold values to determine the suitable subsample ratio for the current GOP. We recursively execute these steps above, and we can adaptively supply the suitable subsample ratio to each GOP in one video sequence and also achieve the target of power saving.

A flowchart of the proposed algorithm is shown Fig 4.1 and the realization procedure of the adaptive motion estimation algorithm using variable subsample ratios is as follows.

Step 1: Setting initial value

Set i=1.

We set the initial value in this proposed algorithm. And the proposed algorithm is ready to start

Step 2: Starting

When starting the proposed algorithm, the *i*th GOP of current video sequence is picked out and the first frame of the *i*th GOP goes to Step 3.

In this step, we check the number of the frames in the GOP. We have to realize which frame is the first frame in the GOP and start our proposed algorithm from beginning. For the arrangement of a GOP, the first frame is coded using intra-prediction and the others are coded using inter-prediction. So, every GOP can be seen as a small size video sequence, about 0.5 second. But the advantage of this GOP is the high correlation between the frames in the GOP. That is why we choose the GOP as the process unit to adaptively select the suitable subsample ratio.

Step 3: Determining the current frame whether an I-frame or not

If the current frame is an I-frame, the proposed algorithm executes intra-frame coding to encode the current I-frame; otherwise, the current frame is a P-frame and then goes to Step 4.

We can recognize the I-frame in the GOP in this step. We don't change the inter-predication in the proposed algorithm. Hence, the proposed algorithm uses the same inter-prediction like H.264/MPEG-4 AVC for the I-frame.

Step 4: Determining the current frame whether a first P-frame or not

If the current frame is a first P-frame, the proposed algorithm executes inter-frame coding for the current P-frame using 16:16 subsample ratio and then calculates the null motion estimation count (NMVC) of the current frame; otherwise, the current P-frame goes to Step 5.

The reason why the first P-frame in the GOP use the 16:16 subsample ratio is that we want the accurate NMVC. If the NMVC is not correct, the subsample ratio for the rest 13 P-frames will not be suitable for this GOP. It will cause the large quality degradation or waste the power dissipation for low motion GOP. Therefore, in order to get the correct NMVC, we consume the power to using the 16:16 subsample ratio.

Step 5: Adaptively selecting the suitable subsample ratio to the current P-frame

The proposed algorithm compares NMVC of the first P-frame with optimal threshold values to adaptively select a suitable subsample ratio and then uses this selected subsample ratio to execute inter-frame coding for the current P-frame and then the current P-frame goes to Step 6.

In this step, we process the rest 13 P-frames in the GOP. These frames will be code using the selected subsample ratio. The selected subsample ratio is according to the NMVC of the first P-frame in the GOP. We assume that the frames in the current GOP have the high correlation between each other. According to the NMVC of the first P-frame, we recognize the current GOP as high, normal or low motion GOP. And we compare this NMVC with the threshold value to decide the suitable subsample ratio for the current GOP. The inter-prediction of the rest 13 P-frames uses this suitable subsample ratio.

Step 6: Determining the current P-frame whether a last P-frame or not

If the current P-frame is a last frame, the procedure goes to Step 7; otherwise, the next frame goes to Step 3.

When the current GOP is end, we have to start the proposed algorithm again and to control the next GOP. Otherwise, the frame in the current GOP will be coded according to the situation in the current GOP.

Step 7: Ending

If all GOPs in the current video sequence are encoded, the proposed algorithm finishes; otherwise, the procedure sets i=i+1 and goes to step 2;

This video is end and all frames in the video sequence have been coded using the proposed algorithm in the H.264/MPEG-4 AVC [4].

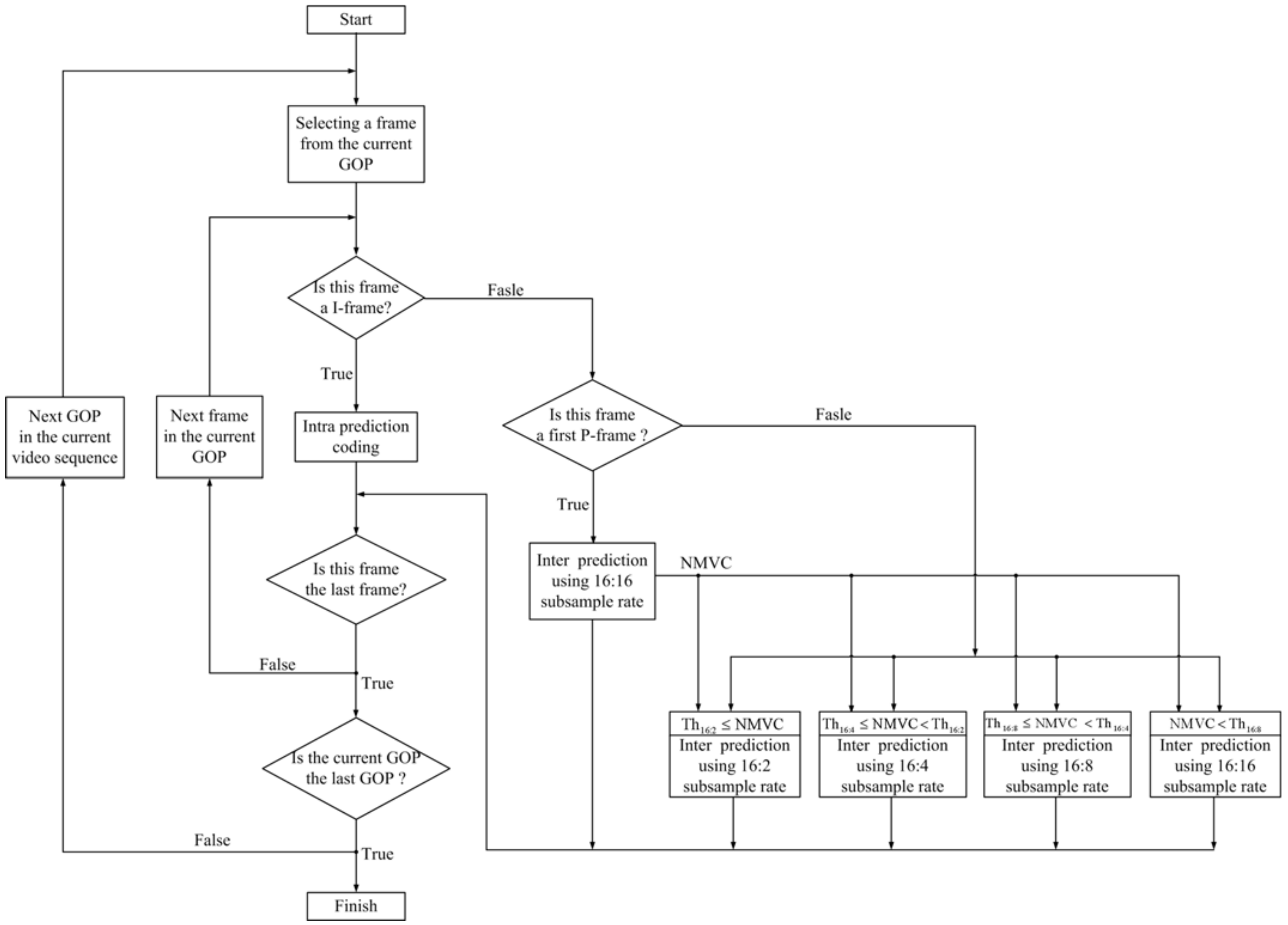

Start
Selecting a frame from the current GOP
Is this frame a I-frame?
Fasle
True
Intra prediction coding
Is this frame a first P-frame ?
Fasle
True
Inter prediction using 16:16 subsample rate
NMVC
Next GOP in the current video sequence
Next frame in the current GOP
Is this frame the last frame?
False
True
Is the current GOP the last GOP ?
False
True
Finish
$Th_{16:2} \leq NMVC$
Inter prediction using 16:2 subsample rate
$Th_{16:4} \leq NMVC < Th_{16:2}$
Inter prediction using 16:4 subsample rate
$Th_{16:8} \leq NMVC < Th_{16:4}$
Inter prediction using 16:8 subsample rate
$NMVC < Th_{16:8}$
Inter prediction using 16:16 subsample rate

Fig 4.1: The flowchart of the proposed algorithm (Th16:2 is the threshold between 16:2 subsample ratio and 16:4 subsample ratio, Th16:4 is the threshold between 16:4 subsample ratio and 16:8 subsample ratio and Th16:8 is the threshold between 16:8 subsample ratio and 16:16 subsample ratio).

4.2 Subsample Patterns of the Proposed Algorithm

To demonstrate that the proposed algorithm has better adaptive ability, the proposed algorithm provides four subsample ratios and adaptively selects the suitable subsample ratio from these subsample patterns to the current GOP. These subsample ratios are fixed at powers of two in spatial distribution and are 16:16, 16:8, 16:4 and 16:2 respectively. These subsample masks can be generated in a 16-by-16 macro-block using Eq.3 and are shown in Fig 3.1. The reason why to choose those subsample ratios is because they are symmetry and their scale is power of two. We select four subsample ratios in our proposed algorithm. There are four levels of 16:16, 16:8, 16:4 and 16:2. All P-frames expect the first P-frame in the GOP will be classified into four levels according to the NMVC of the first P-frame in the GOP. Every GOP have only one subsample ratio. According to these subsample ratios, the proposed algorithm can adaptively select the suitable subsample ratio to the current GOP. For example, the proposed algorithm can provide the 16:16 subsample ratio for the current GOP which has the stronger degree of temporal variation or provide the 16:2 subsample ratio for the current GOP which has the weaker degree of temporal variation. The temporal variation in the video means the degree of object-moving. The degree of object-moving is faster, and the temporal variation is stronger. Although the low subsample ratio cause aliasing in high frequency band, the degree of temporal variation will affect the degree of quality degradation. If the temporal variation is strong, aliasing problems will degrade the validity of motion vector (MV) and result in visual quality degradation to video sequences obviously. On the contrary, if the temporal variation is weak, aliasing problems will not degrade the validity of motion vector (MV) although the low subsample ratio still cause aliasing in high frequency band. That is because we do not need the high frequency band information to find the motion vector when the degree of object-moving is slow. According to the NMVC of the first P-frame, we recognize the current GOP as high, normal or low motion GOP. High motion means the high degree of high frequency. On the contrary, Low motion means the low degree of high frequency. And we compare this NMVC with the threshold value to decide the suitable subsample ratio for the current GOP. The inter-prediction of the rest 13 P-frames uses this suitable subsample ratio. Therefore, a threshold decision for variable subsample ratios is necessary to set the compatible threshold values in order to adaptively choosing the suitable subsample ratio to the current GOP. Next, a threshold decision for variable subsample ratios will be presented in chapter 4.3.

4.3 Threshold Decision for Variable Subsample Ratios

To support a suitable subsample ratio to other P-frames of current GOP, except the first P-frame of the current GOP, an adaptive subsample ratio threshold decision is necessary. Therefore, we use 16:2, 16:4, 16:8 and 16:16 subsample ratios respectively to calculate the statistical distribution of ΔQ_{GOP} versus NMVC for twelve video sequences (Fig 4.2) [37].

The statistical results are shown as in Fig 4.3 and each coordinate means ΔQ_{GOP} versus NMVC using a specific subsample ratio. From Fig 4.3, we can observe that the statistical distribution of ΔQ_{GOP} versus NMVC focus on the right side. This situation means the most video sequence must have a part of background region. The background region means the MV is null. There is not video sequence without the background region except for scene change. For scene change case, there is no algorithm can be solved success.

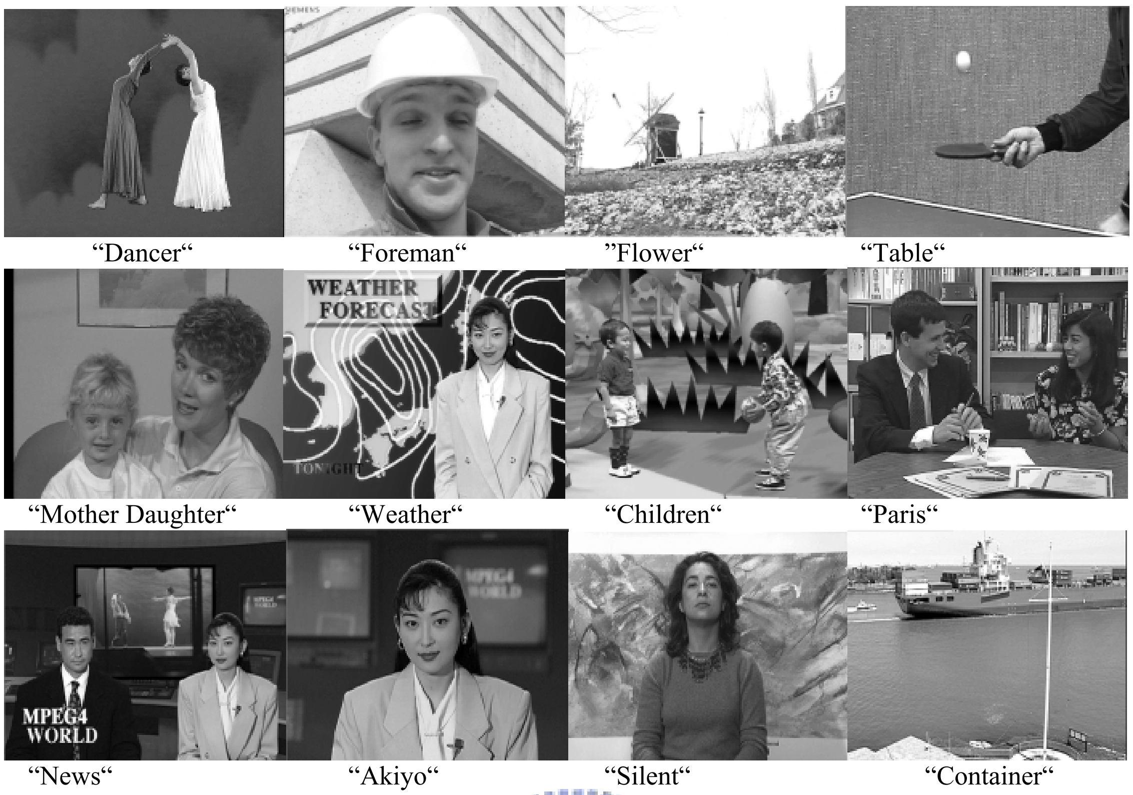


Fig 4.2: The twelve testing video sequences

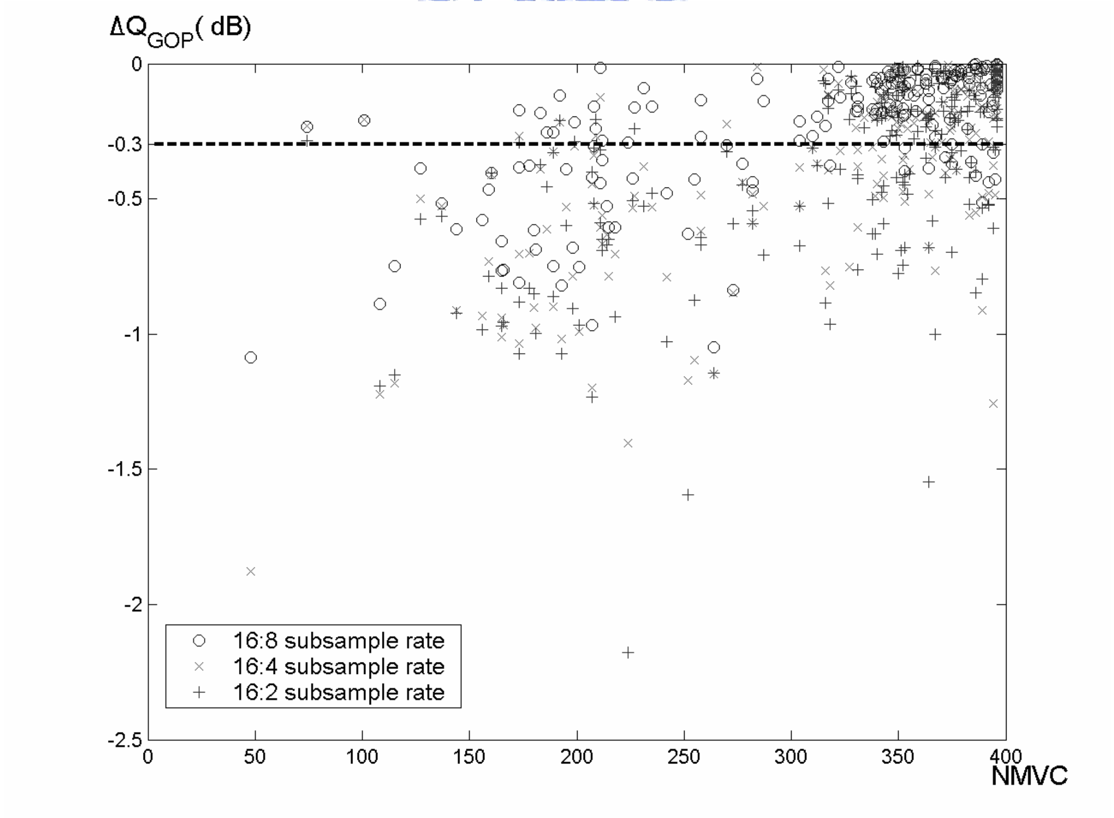


Fig 4.3: The statistical distribution of ΔQ_{GOP} versus NMVC for twelve video sequences

In order to efficiently use the statistical distribution to get the threshold values between these subsample ratios, we propose an adaptive subsample ratio threshold decision to decide the threshold values. The method is to statistically calculate the maximum distributed range of NMVC in which the number of ΔQ_{GOP} under a desired threshold of quality degradation are smaller than or equal to a fixed percentage of total using a selected subsample ratio and this method is proposed as Fig 4.4.

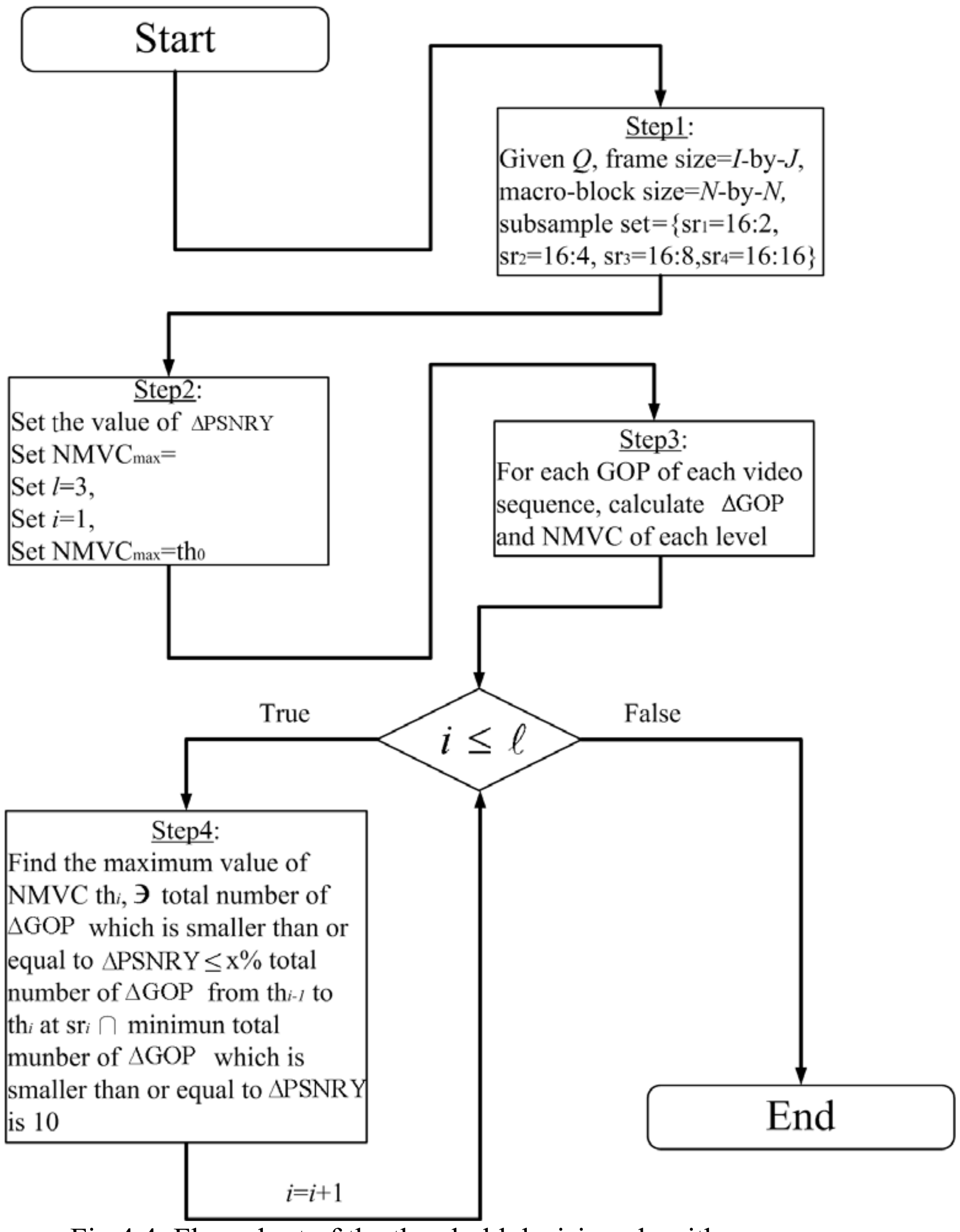

Start
Step1:
Given Q, frame size=I-by-J, macro-block size=N-by-N, subsample set={sr1=16:2, sr2=16:4, sr3=16:8,sr4=16:16}
Step2:
Set the value of ΔPSNRY
Set NMVCmax=
Set l=3,
Set i=1,
Set NMVCmax=th0
Step3:
For each GOP of each video sequence, calculate ΔGOP and NMVC of each level
i ≤ ℓ
True
False
Step4:
Find the maximum value of NMVC thi, ∋ total number of ΔGOP which is smaller than or equal to ΔPSNRY ≤ x% total number of ΔGOP from thi-1 to thi at sri ∩ minimun total munber of ΔGOP which is smaller than or equal to ΔPSNRY is 10
i=i+1
End

Fig 4.4: Flow chart of the threshold decision algorithm

We first set the quality degradation threshold is 0.3 dB. We define the first region is the selected 16:2 subsample ratio. The second region is the selected 16:4 subsample ratio. The selected 16:8 subsample ratio is the third region. The last region, fourth region, is the selected 16:16 subsample ratio. In the first region, we calculate the percentage of the number of point with the quality degradation under 0.3dB using the 16:2 subsample ratio in this region. For the second region, the percentage of the number of point with the quality degradation under 0.3dB using the 16:2 subsample ratio change to 16:4 subsample ratio. And third region is for 16:8 subsample ratio. So we set the percentage threshold from 90% to 60%, every decreasing for 5%. We will get the seven forms of the adaptive subsample ratio threshold value.

To get the threshold values between these subsample ratios, we use the threshold decisions mentioned above to calculate the threshold values and the distribution of threshold values is shown as Fig 4.5. We can observe the size of every region directly in Fig 4.5.

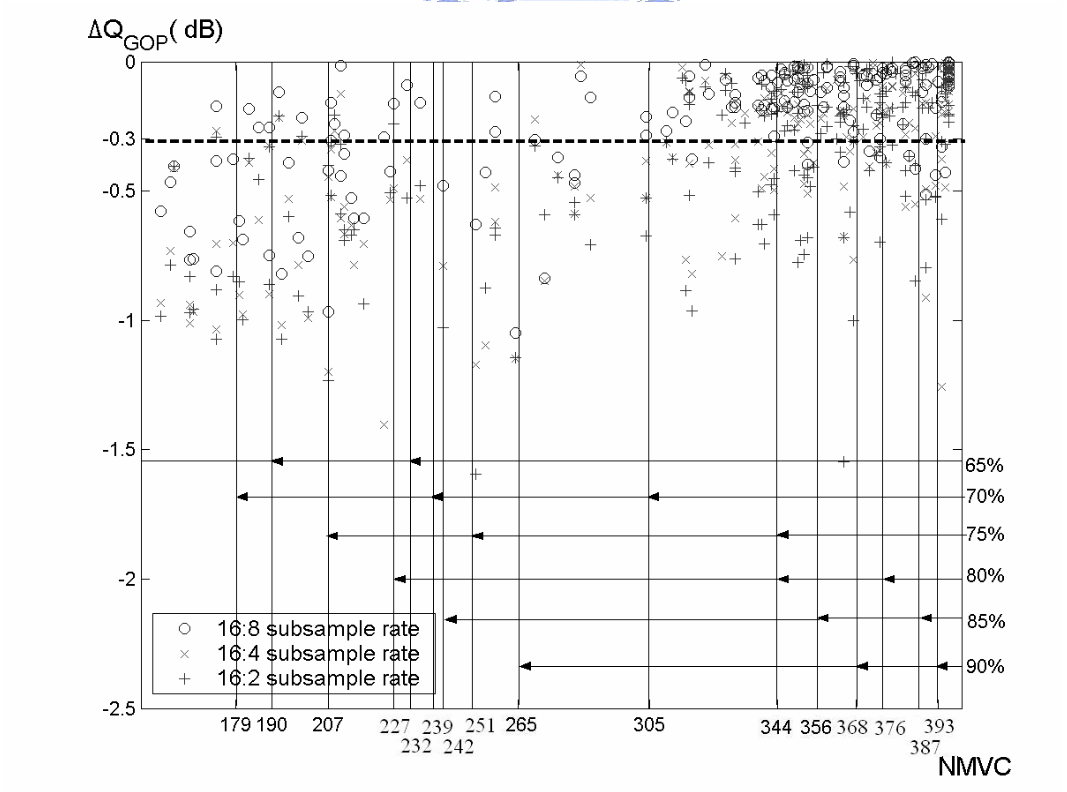


Fig 4.5: The distribution of threshold values

Table 2
Threshold Setting of the adaptive subsample ratio threshold decision

	The adaptive subsample ratio threshold decision						
	90%	85%	80%	75%	70%	65%	60%
Threshold of 16:2 ($TH_{16:2}$)	393	387	376	344	305	232	190
Threshold of 16:4 ($TH_{16:4}$)	368	356	344	251	239	190	49
Threshold of 16:8 ($TH_{16:8}$)	265	242	227	207	179	49	X

Table 2 shows the summary of threshold values using different adaptive subsample ratio decision. From the Table 2, we observe the adaptive subsample ratio threshold value of 60% is not complete. There is not the $TH_{16:8}$ value. That is because the second region is too big and the rest region can make the percentage down to 60%. About the 65%, the $TH_{16:8}$ value is too small so that the 16:16 subsample ratio is hardly selected. For the 90%, 85%, 80%, 75% and 70%, the first region increases. That means we can tolerate more change of the quality degradation over 0.3 dB. The same situation is happened in the second and third regions.

Chapter 5 Experimental Result

In our simulation, the proposed algorithm is simulated in H.264/MPEG-4 AVC [4] with software model JM9.2 [36] using AMD 2.0G Hz and the distortion measure is sum of absolute difference (SAD) which is computed for a 16-by-16 macro-block. We use twelve famous video sequences [37] to be tested and the simulation environment in JM9.2 is shown as in Table 3. From Table 3, the file format of these video sequences is CIF (352 × 288 pixels) and the search range is ±16 in both horizontal and vertical directions for a 16-16 macro-block. The bit-rate control is turned on to maintain a fixed bit rate of 450k bits/s under displaying 30 frames / s. In Chapter 4, we proposed an adaptive subsample ratio decision to pick the suitable subsample ratio and the adaptive subsample ratio threshold decision support six different threshold values between 16:16, 16:8, 16:4 and 16:2, which are shown as in Table 2. To choose the optimal threshold values from Table 2, we simulate these tested video sequences using these subsample ratio decisions respectively in the same simulation condition and then analysis to decide the optimal threshold values from these decisions based on two factors: average quality degradation (ΔPSNRY) and average subsample ratio. The PSNRY is defined as Eq.5 where the frame size is N × M, $I_Y(x,y)$ and $\hat{I}_Y(x,y)$ denote the Y components of original frame and reconstructed frame at (x; y). The ΔPSNRY is defined as Eq.6 and it means the difference of PSNRY which is calculated by a chosen algorithm and PSNRY which is calculated by using full-search block-matching algorithm (FSBM).

$$PSNRY = 10\log_{10} \times \frac{255^2}{\left(1/(N \times M)\right)\sum\sum\left(I_Y(x,y) - \hat{I}_Y(x,y)\right)^2} \qquad (5)$$

$$\Delta PSNRY = PSNRY_{Chosen\quad a\lg orithm} - PSNRY_{FSME} \qquad (6)$$

The average subsample ratio is also defined as Eq.7 and it averagely estimates what subsample ratio can be used to execute the motion estimation for a video sequence.

$$Average \quad Subsample \quad Rate$$
$$= 16 : [(\# \quad of \quad P-frames_{16:16}) \times 16 + (\# \quad of \quad P-frames_{16:8}) \times 8 + (\# \quad of \quad P-frames_{16:4}) \times 4$$
$$+ (\# \quad of \quad P-frames_{16:2}) \times 2] \Big/ (Total \quad \# \quad of \quad P-frames) \qquad (7)$$

Table 3
Testing Video Sequences and Simulation Conditions

	Video Sequence	Number of Frames	Format	Frame Rate (frames/s)	Bit Rate (bits/s)	Initail Q_p	Search Range	GOP Unit	Video Type
Fast Motion	Dancer	250	CIF (352 × 288)	30	450k	34	±16	15 frames	IPPP… IPPP…
	Foreman	300							
	Flower	250							
Normal Motion	Table	300							
	Mother Daughter	300							
	Weather	300							
	Children	300							
	Paris	300							
Slow Motion	News	300							
	Akiyo	300							
	Silent	300							
	Container	300							

To demonstrate that the proposed algorithm can adaptively select the suitable subsample ratio to each GOP for a tested video sequence, we analysis the average quality degradation of each GOP using Eq.4 for the video sequence “table” and the results is shown as in Fig 5.1. This case is the same with the Fig 3.4 in chapter 3. But the Fig 5.1 adds the distribution of the proposed algorithm to demonstrate the performance of the proposed algorithm. From Fig 5.1, there exists the stronger temporal variation between third GOP and seventh GOP, the proposed algorithm can adaptively support higher subsample ratio to efficiently reduce the ΔGOP. Besides, the proposed algorithm can adaptively support lower subsample ratio to save power dissipation without affecting the ΔGOP between eleventh GOP and twenty GOP because of the weaker temporal variation.

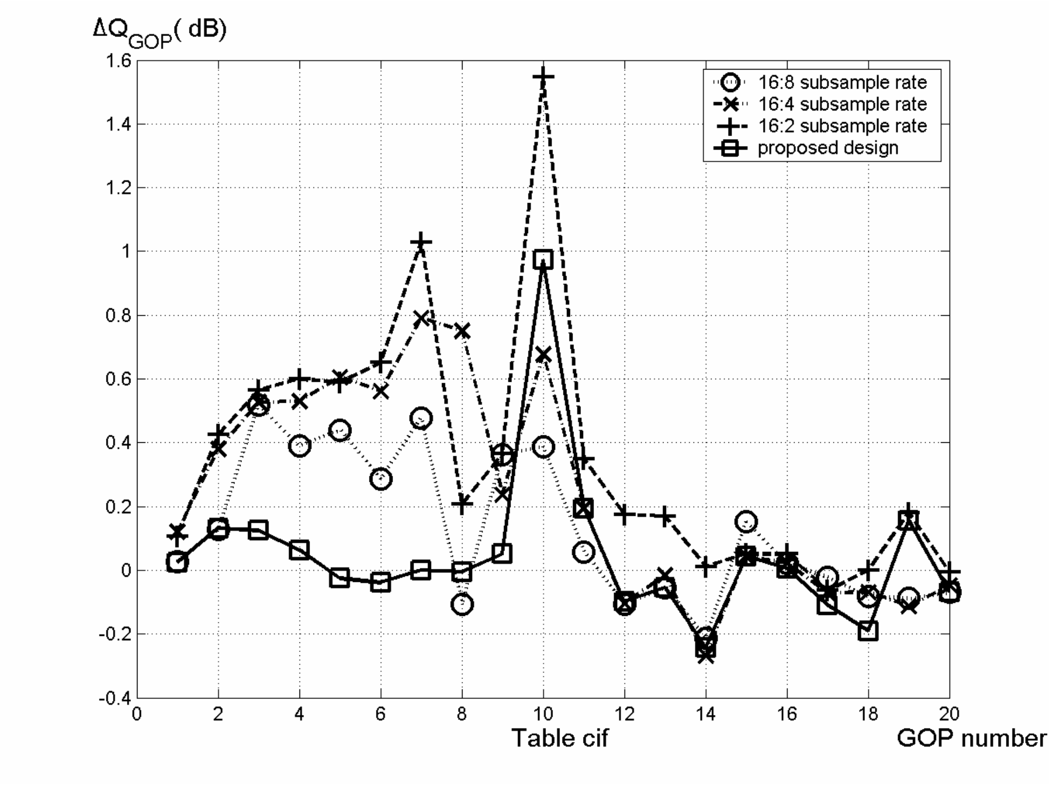


Fig 5.1: Te average quality degradation of each GOP for the video sequence "table"

Table 4 shows the simulation results of PSNRY and ΔPSNRY for these tested video sequences using this threshold decision method. Table 5 shows the simulation results of average subsample ratio and overall average subsample ratio for these tested video sequences using this threshold decision method. Because threshold values of Table 2 can be calculated according to the target of average quality degradation of 0.3 dB, the average quality degradation of 0.3 dB is an important index for all tested video sequences. From Table 4 and Table 5, 90%, 85% and 80% statistics of threshold decision method can satisfy all tested video sequences under the average quality degradation of 0.3 dB, however their overall average subsample ratio are higher than 75% and 70% statistics of threshold decision method. Among 75% and 70% statistics of threshold decision method, 70% statistics causes average quality degradation exceed 0.3 dB for sequences “Dancer”, “Foreman”, “Mother Daughter”, “Weather” and “Paris”, but these average quality degradations are very close to the target of 0.3 dB. For the video sequences “Dancer” and “Mother Daughter”, their quality degradations in this 70% method are the same and are equal to 0.36 dB. And the quality degradation in this 70% method of video sequence “Foreman” is equal to 0.33 dB. For “Paris”, it is 0.35 dB. And 0.33 dB is for “Weather”. Although the overall average subsample ratio of 65% statistics of threshold decision method is the lowest, the average quality degradation of it exceeds 0.3 dB too much. For example, the average quality degradations of the sequences “Dancer” and “Foreman” are 0.77dB and 0.59 dB. These quality degradations are not acceptable. For the 70% statistics, we can observe the video sequences of fast motion have the maximum acceptable quality degradation for near 0.3 dB. In this quality degradation, the power consumption is the maximum. We can save the power efficiently. For the other threshold values, they can also keep the quality degradation acceptable. But they waste the power to gain the better quality degradation under 0.3 dB. For the low motion video sequences, the algorithm using the threshold value of 70% statistics can select adaptively the minimum power consumption to save power efficiently. The minimum power consumption is the average subsample ratio 16:3. That contains the number of the first P-frame in the GOP using 16:16 subsample ratio and all the rest P-frame using 16:2 subsample ratio. Therefore, in order to minimize the power consumption of motion estimation and maintain the average visual quality about 0.3dB, threshold values of 70% statistics is the optimal choice for adaptively selecting the suitable subsample ratio.And we save 69.6% power consumption and keep quality degradation under 0.36

dB.

Table 4
Analysis of quality degradation using adaptive subsample ratio decision

	Video Sequence	Full Search PSNRY	The adaptive subsample ratio threshold decision											
			90%		85%		80%		75%		70%		65%	
			PSNRY	Δ PSNRY	PSNRY	Δ PSNRY	PSNRY	Δ PSNRY	PSNRY	Δ PSNRY	PSNRY	Δ PSNRY	PSNRY	Δ PSNRY
Fast Motion	Dancer	33.42	33.4	-0.02	33.4	-0.02	33.4	-0.02	33.33	-0.09	33.06	-0.36	32.65	-0.77
	Foreman	29.51	29.42	-0.09	29.36	-0.15	29.35	-0.16	29.2	-0.31	29.18	-0.33	28.92	-0.59
	Flower	19.58	19.58	0	19.54	-0.04	19.54	-0.04	19.43	-0.15	19.31	-0.27	19.14	-0.44
Normal Motion	Table	31.04	30.99	-0.05	30.98	-0.06	30.93	-0.11	30.85	-0.19	30.78	-0.26	30.7	-0.34
	Mother Daughter	39.34	39.14	-0.2	39.12	-0.22	39.11	-0.23	39.01	-0.33	38.98	-0.36	38.89	-0.45
	Weather	32.26	32.06	-0.2	32.04	-0.22	32.01	-0.25	31.97	-0.29	31.93	-0.33	31.93	-0.33
	Children	29	28.87	-0.13	28.84	-0.16	28.81	-0.19	28.72	-0.28	28.71	-0.29	28.71	-0.29
	Paris	30.67	30.5	-0.17	30.45	-0.22	30.46	-0.21	30.36	-0.31	30.32	-0.35	30.32	-0.35
Slow Motion	News	37.27	37.19	-0.08	37.17	-0.1	37.15	-0.12	37.12	-0.15	37.07	-0.2	37.07	-0.2
	Akiyo	42.36	42.27	-0.09	42.24	-0.12	42.24	-0.12	42.21	-0.15	42.21	-0.15	42.21	-0.15
	Silent	34.62	34.56	-0.06	34.57	-0.05	34.58	-0.04	34.56	-0.06	34.53	-0.09	34.53	-0.09
	Container	35.47	35.45	-0.02	35.45	-0.02	35.45	-0.02	35.45	-0.02	35.45	-0.02	35.45	-0.02

Table 5
The simulation results of average subsample ratio and overall average subsample ratio

	Video Sequence	Threshold Decision					
		90%	85%	80%	75%	70%	65%
		Average Subsample ratio	Average Subsample ratio	Average Subsample ratio	Average Subsample ratio	Average Subsample ratio	Average Subsample ratio
Fast Motion	Dancer	16:15.55	16:15.55	16:15.55	16:14.43	16:11.75	16: 6.91
	Foreman	16:14.32	16:13.31	16:12.93	16:10.61	16:10.24	16: 6.06
	Flower	16:16.00	16:15.10	16:15.10	16:11.98	16: 8.80	16: 5.12
Normal Motion	Table	16: 9.50	16: 9.03	16: 7.17	16: 5.32	16: 4.67	16:3.55
	Mother Daughter	16: 7.08	16: 6.43	16: 6.34	16: 3.92	16: 3.55	16: 3.00
	Weather	16: 5.87	16: 5.32	16: 4.39	16: 3.18	16: 3.00	16: 3.00
	Children	16: 7.82	16: 7.27	16: 6.43	16: 3.83	16: 3.27	16: 3.00
	Paris	16: 6.52	16: 6.25	16: 5.22	16: 3.46	16: 3.00	16: 3.00
Slow Motion	News	16: 7.45	16: 6.71	16: 4.95	16: 3.09	16: 3.00	16: 3.00
	Akiyo	16: 4.76	16:3.83	16: 3.46	16: 3.00	16: 3.00	16: 3.00
	Silent	16: 7.27	16: 7.08	16: 6.34	16: 3.92	16: 3.00	16: 3.00
	Container	16: 3.18	16: 3.00	16: 3.00	16: 3.00	16: 3.00	16: 3.00
Overall Average Subsample ratio		16: 8.58	16: 8.04	16: 7.35	16: 5.60	16: 4.87	16: 3.74

After choosing the optimal threshold values between 16:16, 16:8, 16:4 and 16:2, we compare the proposed algorithm using the optimal threshold value with generic subsample ratio algorithms. The PSNRY and ΔPSNRY of the proposed algorithm and generic subsample ratio algorithm are shown in Table 6 and Table 7. Fig 5.2 and Fig 5.3 are similar with Fig 3.2 and Fig 3.3 respectively. Fig 5.2 and Fig 5.3 add the location of the proposed algorithm with the optimal threshold value. We can easily observe the relation between the generic subsample ratio algorithm and the proposed algorithm with the optimal threshold value. For Fig 5.2, the quality degradations of these testing sequences using generic subsample ratio algorithms are strong. The maximum quality degradation is 0.93 dB. It happens in "Dancer" sequence using the 16:2 generic subsample ratio. From Fig 5.3, the proposed algorithm can adaptively maintain ΔPSNRY under the threshold of about 0.3 dB and has lower subsample ratio to substantially save power dissipation than the generic subsample ratio algorithm under the same ΔPSNRY for tested video sequences. For Fig 5.3, the quality degradations of these testing sequences using generic subsample ratio algorithms are light. The maximum quality degradation is 0.33 dB and it is acceptable. It happens in "Weather" and "Paris" sequences using the 16:2 generic subsample ratio. From Fig 5.3, therefore, the proposed algorithm can select the lowest subsample ratio and maintain ΔPSNRY under the threshold of about 0.3 dB. We can determine the performance of the proposed algorithm with different threshold value from Fig 5.2 and Fig 5.3. The optimal threshold value can make the quality degradation of high and low motion video sequence keep near by 0.3 dB. That will save the maximum power consumption. And for low motion video sequence, the selected subsample ratio is the lowest one, 16:2. The power consumption is the minimum, for 16:3. If the other threshold value is used in the proposed algorithm, the location will be away from 0.3 dB and have no the minimum power consumption case.

Table 6

The PSNRY of the proposed algorithm and generic subsample ratio algorithm

Full Search Block Matching

	video Sequence	Generic 16:16 Subsample ratio	Generic 16:14 Subsample ratio	Generic 16:12 Subsample ratio	Generic 16:10 Subsample ratio	Generic 16:8 Subsample ratio	Generic 16:6 Subsample ratio	Generic 16:4 Subsample ratio	Generic 16:2 Subsample ratio	Proposed Algorithm Method (70%)
		PSNRY	PSNRY	PSNRY	PSNRY	PSNRY	PSNRY	PSNRY	PSNRY	PSNRY
Fast Motion	Dancer	33.42	33.24	33.09	32.89	32.72	32.56	32.5	32.49	33.06
	Foreman	29.51	29.42	29.33	29.24	29.11	28.96	28.79	28.73	29.18
	Flower	19.58	19.53	19.48	19.4	19.3	19.18	19.09	19.07	19.31
Normal Motion	Table	31.04	31.02	31	30.95	30.91	30.88	30.8	30.69	30.78
	Mother Daughter	39.34	39.31	39.32	39.26	39.19	39.09	38.99	38.88	38.98
	Weather	32.26	32.2	32.16	32.17	32.11	32.04	31.98	31.93	31.93
	Children	29	28.99	28.95	28.89	28.86	28.83	28.78	28.71	28.71
	Paris	30.67	30.67	30.63	30.62	30.57	30.54	30.4	30.34	30.32
Slow Motion	News	37.27	37.25	37.26	37.23	37.21	37.18	37.14	37.05	37.07
	Akiyo	42.36	42.37	42.35	42.34	42.33	42.11	42.27	42.2	42.21
	Silent	34.62	34.59	34.59	34.59	34.6	34.6	34.56	34.54	34.53
	Container	35.47	35.47	35.46	35.46	35.47	35.45	35.45	35.45	35.45

Table 7

The ΔPSNRY of the proposed algorithm and generic subsample ratio algorithm

Full Search Block Matching

	video Sequence	Generic 16:14 Subsample ratio	Generic 16:12 Subsample ratio	Generic 16:10 Subsample ratio	Generic 16:8 Subsample ratio	Generic 16:6 Subsample ratio	Generic 16:4 Subsample ratio	Generic 16:2 Subsample ratio	Proposed Algorithm Method1 (70%)
		ΔPSNRY	ΔPSNRY	ΔPSNRY	ΔPSNRY	ΔPSNRY	ΔPSNRY	ΔPSNRY	ΔPSNRY
Fast Motion	Dancer	-0.18	-0.33	-0.53	-0.7	-0.86	-0.92	-0.93	-0.36
	Foreman	-0.09	-0.18	-0.27	-0.4	-0.55	-0.72	-0.78	-0.33
	Flower	-0.05	-0.1	-0.18	-0.28	-0.4	-0.49	-0.51	-0.27
Normal Motion	Table	-0.02	-0.04	-0.09	-0.13	-0.16	-0.24	-0.35	-0.26
	Mother Daughter	-0.03	-0.02	-0.08	-0.15	-0.25	-0.35	-0.46	-0.36
	Weather	-0.06	-0.1	-0.09	-0.15	-0.22	-0.28	-0.33	-0.33
	Children	-0.01	-0.05	-0.11	-0.14	-0.17	-0.22	-0.29	-0.29
	Paris	0	-0.04	-0.05	-0.1	-0.13	-0.27	-0.33	-0.35
Slow Motion	News	-0.02	-0.01	-0.04	-0.06	-0.09	-0.13	-0.22	-0.2
	Akiyo	0.01	-0.01	-0.02	-0.03	-0.25	-0.09	-0.16	-0.15
	Silent	-0.03	-0.03	-0.03	-0.02	-0.02	-0.06	-0.08	-0.09
	Container	0	-0.01	-0.01	0	-0.02	-0.02	-0.02	-0.02

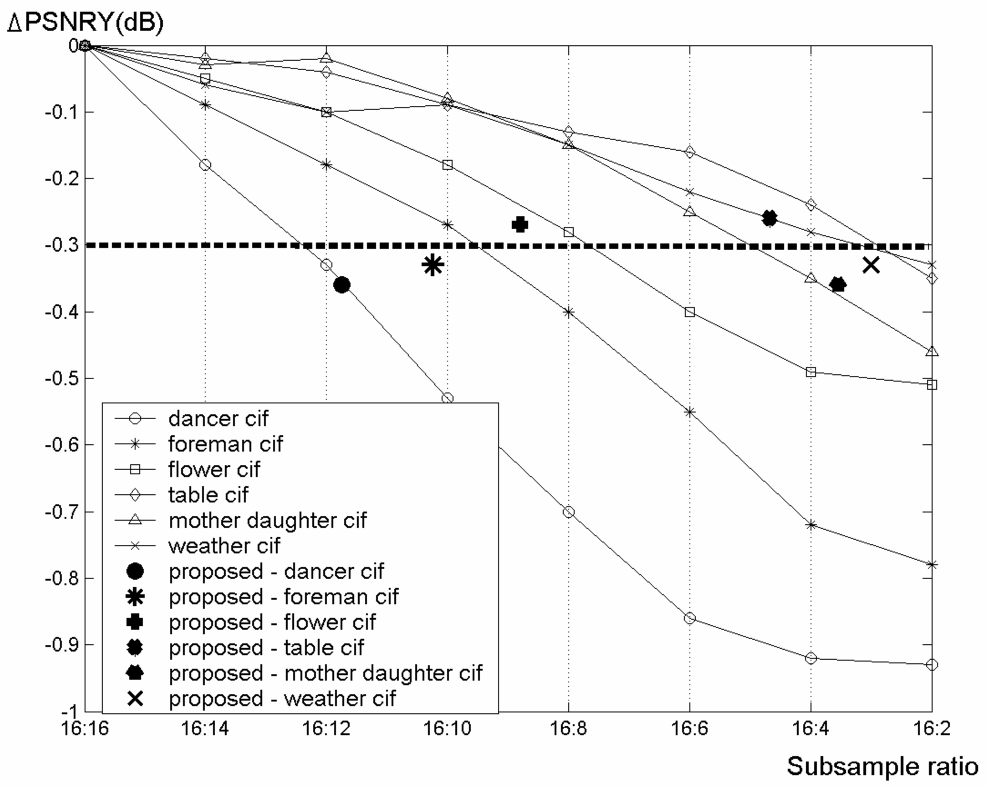


Fig 5.2: The results ΔPSNRY of testing sequences "Dancer", "Foreman", "Flower", "Table", "Mother Daughter" and "Weather" and the proposed algorithm results location

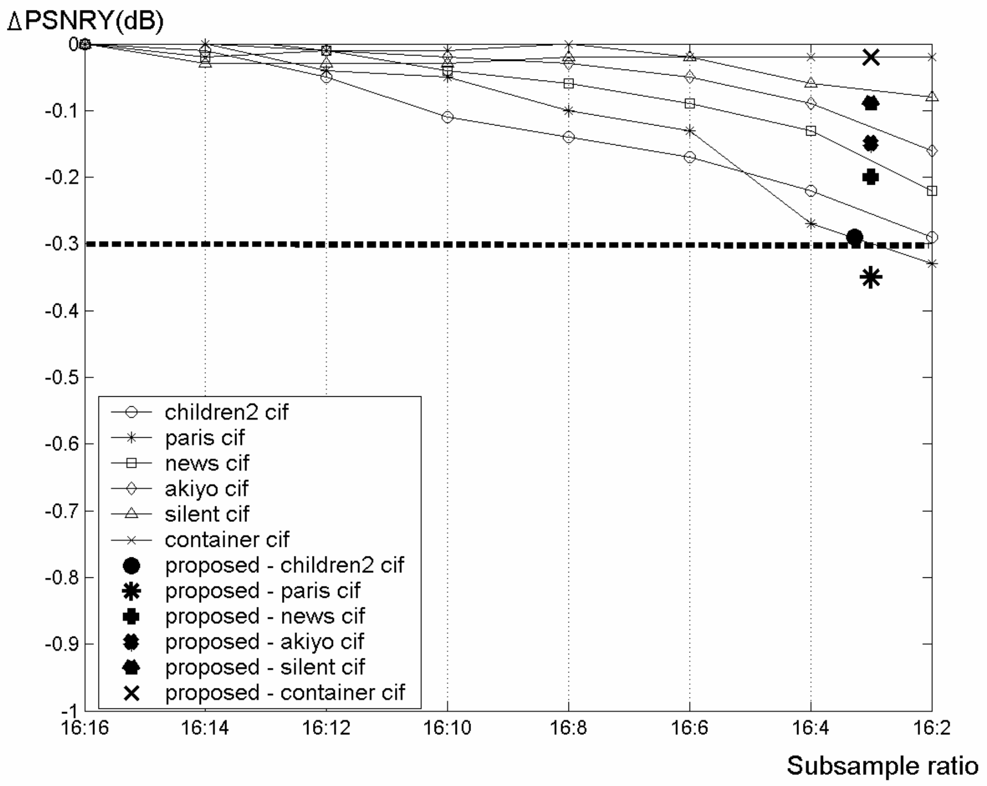


Fig 5.3: The results ΔPSNRY of testing sequences "Children", "Paris", "News", "Akiyo", "Silent" and "Container" and "Weather" and the proposed algorithm results location

The subsample algorithm, also called the pixel decimation algorithm, can be, in general, classified into two categories. One is fixed patterns [11]-[15], and the other is adaptive patterns [16] [17]. For the subsample algorithm using fixed patterns [11]-[15], they have to choose the only subsample pattern. In our Experimental Result, it is obvious that the only subsample pattern is not suitable for every video sequence.

Although the subsample algorithm using the fixed pattern make sure the power consumption is low, they can not keep the quality degradation of all video sequence near 0.3 dB. If we want to keep the quality degradation of all video sequence near 0.3 dB using the fixed pattern, we have to choose the subsample ratio of 16:12. Because the worse case is the "Dancer" video sequence shown in Fig 5.2. In order to make the quality degradation of "Dancer" near 0.3 dB, we choose the 16:12 fixed subsample ratio. But it is waste the power consumption to using the 16:12 fixed subsample ratio in the low motion video sequence. Therefore, we have to using the adaptive subsample ratios in all video sequences. In our proposed algorithm with the optimal threshold value, it is achieved the best tradeoff between the quality degradation and the power consumption. It can keep the quality degradation near 0.3 dB, and save the maximum power consumption at the same time.

In order to save more power consumption, we also can combine our algorithm with some fast algorithms, like [4]-[17]. The quality is also near 0.3 dB. We can make the video sequences keep their quality near 0.3 dB. At the same time, the power comsuption can be redued more. We simulate that our algorithm is combined with FME mode [38] in JM9.2. We compare the proposed algorithm in FME mode [38] with generic subsample ratio algorithms in FME mode [38]. The PSNRY and ΔPSNRY of the proposed algorithm in FME mode [38] and generic subsample ratio algorithm in FME mode [38] are shown in Table 8, Table 9 and Table 10. Fig 5.4 and Fig 5.5 are similar with Fig 5.2 and Fig 5.3 respectively. Fig 5.2 and 5.3 are result of full search mode and Fig 5.4 and 5.5 are result of FME mode [38]. We can easily observe the relation between the generic subsample ratio algorithm in FME mode [38] and the proposed algorithm in FME mode [38]. In table 8, we can observe the quality and average subsample ratio of our proposed algorithm in FME mode [38]. The quality of our proposed algorithm in FME mode [38] is still near 0.3 dB. This can be acceptable. And the power consumption can be got from the average subsample ratio. We can save the power consumption up to 73.6% in FME mode [38]. For Fig 5.4, the quality degradations of these testing sequences using generic subsample ratio algorithms in FME mode [38] are strong. The maximum quality degradation is 1.05 dB. It happens in "Dancer" sequence using the 16:2 generic subsample ratio in FME mode [38]. From Fig 5.4, the proposed algorithm can adaptively maintain ΔPSNRY under the threshold of about 0.3 dB and has lower subsample ratio to substantially save power dissipation than the generic subsample ratio algorithm under the same ΔPSNRY for tested video sequences. For Fig 5.5, the quality degradations of these testing sequences using generic subsample ratio algorithms in FME mode [38] are light. The maximum quality degradation is 0.3 dB and it is acceptable. It happens in "Childern" sequence using the 16:2 generic subsample ratio in FME mode [38]. From Fig 5.5, therefore, the proposed algorithm can select the lowest subsample ratio and maintain ΔPSNRY under the threshold of about 0.3 dB. The situation in FME mode [38] is the same in full search mode. Therefore, we can know that our algorithm can be combined with FME [38] and the result is similar with in full search mode. We can save more power consumption using this method, combined with some fast algorithms.

Table 8

The result of the proposed algorithm in FME mode [38]

	video Sequence	Full search in FME mode [38]	Proposed Algorithm Method in FME mode [38]		
		PSNRY	PSNRY	ΔPSNRY	Average Subsample Rate
Fast Motion	Dancer	33.48	33.23	-0.25	16:12.35
	Foreman	29.63	29.31	-0.32	16: 6.23
	Flower	19.64	19.35	-0.29	16: 4.24
Normal Motion	Table	31.07	30.84	-0.23	16: 3.00
	Mother_Daughter	39.44	39.2	-0.24	16: 3.55
	Weather	32.34	32.08	-0.26	16: 3.00
	Children	29.12	28.85	-0.27	16: 3.27
	Paris	30.75	30.5	-0.25	16: 3.00
Slow Motion	News	37.37	37.25	-0.12	16: 3.00
	Akiyo	42.43	42.32	-0.11	16: 3.00
	Silent	34.7	34.64	-0.06	16: 3.00
	Container	35.52	35.48	-0.04	16: 3.00
Overall Average Subsample Rate					16: 4.22

Table 9

The PSNRY of the proposed algorithm and generic subsample ratio algorithm in FME mode [38]

FME Search Block Matching										
	video Sequence	Generic 16:16 Subsample ratio	Generic 16:14 Subsample ratio	Generic 16:12 Subsample ratio	Generic 16:10 Subsample ratio	Generic 16:8 Subsample ratio	Generic 16:6 Subsample ratio	Generic 16:4 Subsample ratio	Generic 16:2 Subsample ratio	Proposed Algorithm Method (70%)
		PSNRY	PSNRY	PSNRY	PSNRY	PSNRY	PSNRY	PSNRY	PSNRY	PSNRY
Fast Motion	Dancer	33.48	33.31	33.17	33.01	32.85	32.64	32.47	32.43	33.23
	Foreman	29.63	29.57	29.52	29.46	29.42	29.34	29.18	28.94	29.31
	Flower	19.64	19.63	19.61	19.58	19.56	19.49	19.39	19.16	19.35
Normal Motion	Table	31.07	31.05	31.04	31.01	31	30.96	30.9	30.82	30.84
	Mother Daughter	39.44	39.42	39.4	39.37	39.33	39.29	39.25	39.12	39.2
	Weather	32.34	32.33	32.32	32.29	32.25	32.27	32.21	32.07	32.08
	Children	29.12	29.06	29.04	29.01	28.97	28.96	28.89	28.82	28.85
	Paris	30.75	30.73	30.71	30.7	30.68	30.65	30.58	30.48	30.5
Slow Motion	News	37.37	37.35	37.34	37.32	37.3	37.28	37.25	37.24	37.25
	Akiyo	42.43	42.42	42.41	42.41	42.4	42.37	42.36	42.31	42.32
	Silent	34.7	34.68	34.66	34.65	34.64	34.64	34.63	34.63	34.64
	Container	35.52	35.51	35.51	35.51	35.5	35.5	35.49	35.47	35.48

Table 10
The ΔPSNRY of the proposed algorithm and generic subsample ratio algorithm in FME mode [38]

FME Search Block Matching									
	video Sequence	Generic 16:14 Subsample ratio	Generic 16:12 Subsample ratio	Generic 16:10 Subsample ratio	Generic 16:8 Subsample ratio	Generic 16:6 Subsample ratio	Generic 16:4 Subsample ratio	Generic 16:2 Subsample ratio	Proposed Algorithm Method1 (70%)
		ΔPSNRY	ΔPSNRY	ΔPSNRY	ΔPSNRY	ΔPSNRY	ΔPSNRY	ΔPSNRY	ΔPSNRY
Fast Motion	Dancer	-0.17	-0.31	-0.47	-0.63	-0.84	-1.01	-1.05	-0.25
	Foreman	-0.06	-0.11	-0.17	-0.21	-0.29	-0.45	-0.69	-0.32
	Flower	-0.01	-0.03	-0.06	-0.08	-0.15	-0.25	-0.48	-0.29
Normal Motion	Table	-0.02	-0.03	-0.06	-0.07	-0.11	-0.17	-0.25	-0.23
	Mother Daughter	-0.02	-0.04	-0.07	-0.11	-0.15	-0.19	-0.32	-0.24
	Weather	-0.01	-0.02	-0.05	-0.09	-0.07	-0.13	-0.27	-0.26
	Children	-0.06	-0.08	-0.11	-0.15	-0.16	-0.23	-0.3	-0.27
	Paris	-0.02	-0.04	-0.05	-0.07	-0.1	-0.17	-0.27	-0.25
Slow Motion	News	-0.02	-0.03	-0.05	-0.07	-0.09	-0.12	-0.13	-0.12
	Akiyo	-0.01	-0.02	-0.02	-0.03	-0.06	-0.07	-0.12	-0.11
	Silent	-0.02	-0.04	-0.05	-0.06	-0.06	-0.07	-0.07	-0.06
	Container	-0.01	-0.01	-0.01	-0.02	-0.02	-0.03	-0.05	-0.04

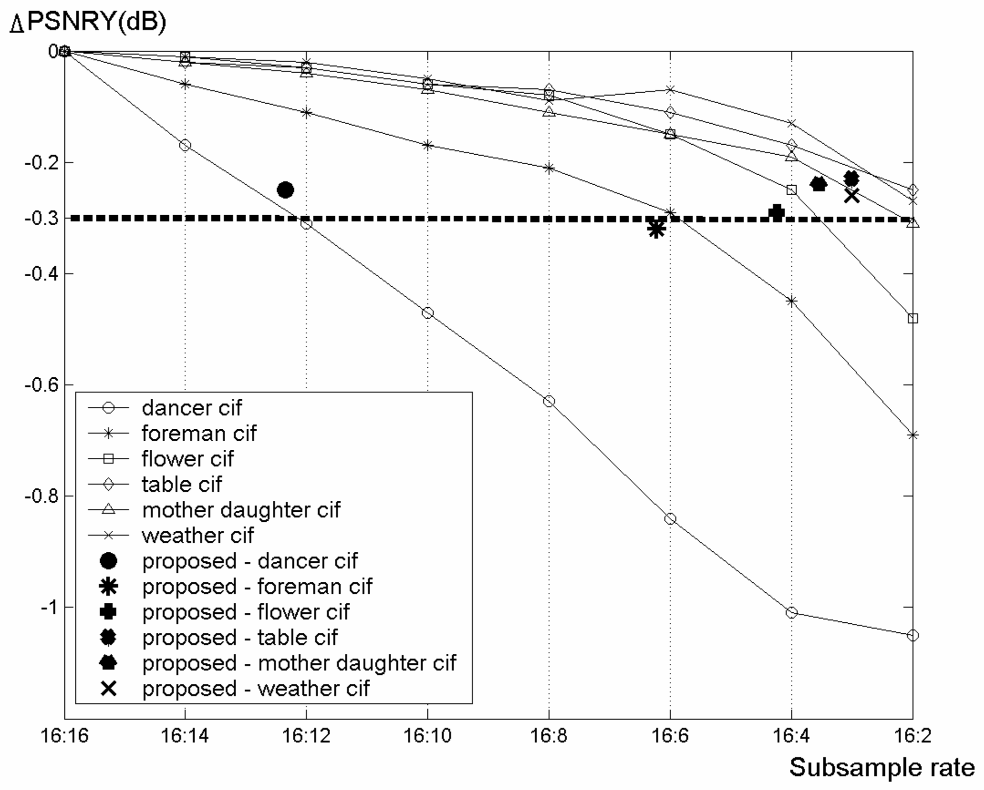


Fig 5.4: The results ΔPSNRY of testing sequences "Dancer", "Foreman", "Flower", "Table", "Mother Daughter" and "Weather" and the proposed algorithm results location in FME mode [38]

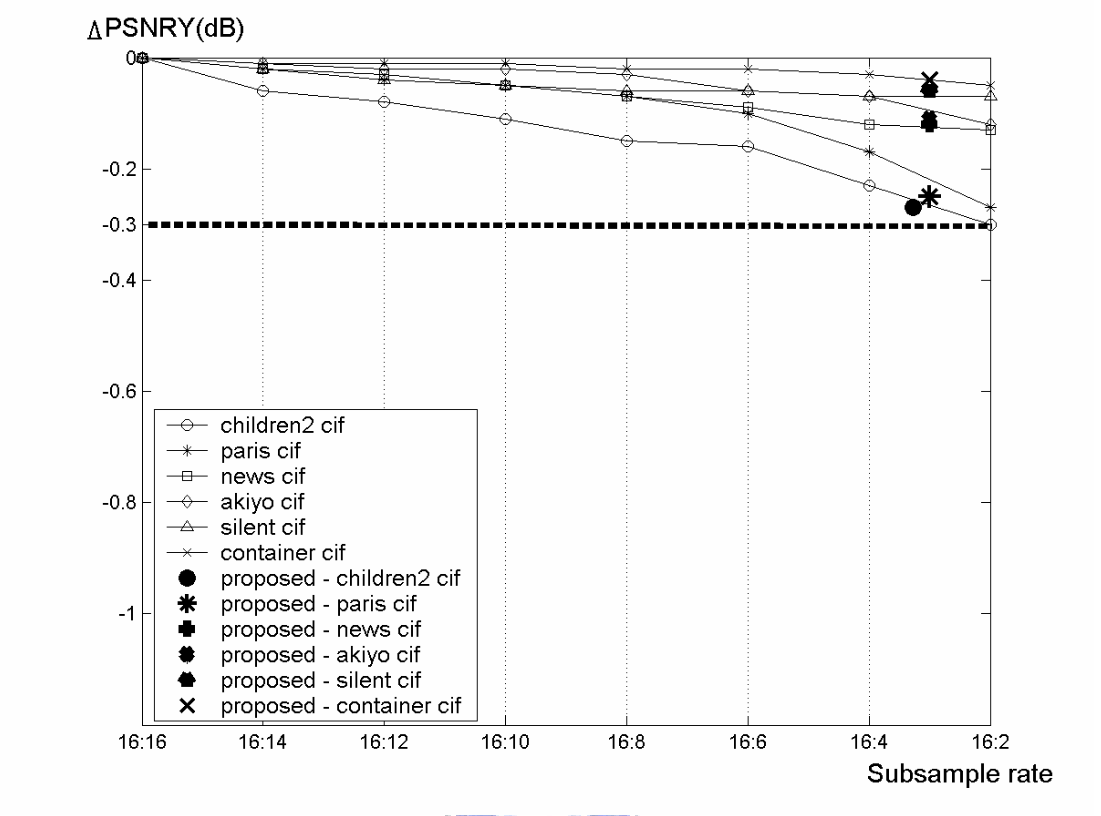


Fig 5.5: The results ΔPSNRY of testing sequences “Children“, “Paris“, “News“, “Akiyo“, “Silent“ and “Container“ and “Weather“ and the proposed algorithm results location in FME mode [38]

We simulate our algorithm in H.264 software model JM 9.2[36] for two situations, full search and FME [38]. We can keep the quality degradation near 0.3 dB and save the maxium power consumption. For full search in JM 9.2, we save 69.6% power consumption and keep quality degradation under 0.36 dB. For FME in JM 9.2, we save 73.6% power consumption and keep quality degradation under 0.33 dB. Therefore, the proposed algorithm can steady the video quality in power-saving situation.

Chapter 6 Conclusion

In modern video standard, such as MPEG-1 [1], MPEG-2 [2], MPEG-4 [3] and H.264/MPEG-4 AVC [4], motion estimation requires the heaviest computational load and hence dominates main power requirement in video compression. Lots of published papers [4]-[17] have presented efficient algorithms for motion estimation. But they don't consider the influence of the video content. Among these fast algorithms [4]-[17], the subsample algorithm [11]-[17] can not only easily combine with other approaches mentioned above but also reduce the number of matching points with flexibly changing subsample ratio. The reason why we choose the adaptive subsample ratios is because we believe that the subsample ratios should be varying with the video content.

An adaptive motion estimation algorithm with variable subsample ratios has been presented. This proposed algorithm can adaptively select the compatible subsample ratio for each current group of picture (GOP). The proposed algorithm is first to analyze the degree of the object-moving between the first P-frame and I-frame for the current GOP and then adaptively selects the suitable subsample ratio to the current GOP according to analysis result. This proposed algorithm has been successful implemented in H.264 with software model JM9.2. An adaptive subsample ratio threshold decision is used to set the compatible threshold values and get the optimal result. The static science is adopted in the adaptive subsample ratio threshold decision. Experimental results has shown that the proposed algorithm can not only adaptively select the suitable subsample ratio to various video sequences but also maintain ΔPSNRY of 0.36 dB at most to save about 69.6% power consumption of motion estimation in a fixed bit rate control on average. The proposed algorithm can also easily combine with other fast algorithms which reduce the computational complexity of FSBM. For FME in JM 9.2, we save 73.6% power consumption and keep quality degradation under 0.33 dB. Hence the proposed algorithm is suitable for real-time implementation of high quality and power-saving video applications using a powerful CPU.

Bibliography

[1] ISO/IEC CD 11172-2 (MPEG-1 Video), "Information technology—Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbits/s: Video," 1993.

[2] ISO/IEC CD 13818-2—ITU-T H.262 (MPEG-2 Video), "Information technology—Generic coding of moving pictures and associated audio information: Video," 1995.

[3] "Information Technology—Generic Coding of Audio-Visual Objects"Part 2: Visual, ISO/IEC 14 496-2 (MPEG-4 Video), 1999.

[4] T. Wiegand, G.J. Sullivan, and A. Luthra, "Draft ITU-T Recommendation H.264 and Final Draft International Standard 14496-10 AVC,.VT of ISO/IEC JTC1/SC29/WG11 and ITU-T SG16/Q" 6, Doc. JVT-G050r1, Geneva, Switzerland, May 2003.

[3] J.R. Jain and A.K. Jan, "Displacement measurement and its application in interframe image coding," IEEE Trrans. Commun., vol.COM-29, pp.1799-1808, Dec. 1981.

[4] M.J. Chen, L.G. Chen, and T.D. Chiueh, "One-Dimensional Full Search Motion Estimation Algorithm For Video Coding," IEEE Trans. Circuits Syst. Video Technol., vol.4, no.5, pp.504-509, Oct. 1994.

[5] R. Li, B. Zeng, and M.L. Liou, "A New Three-Step Search Algorithm for Block Motion Estimation" IEEE Trans. Circuits Syst. Video Technol., vol.4, no.4, pp.438-442, Aug. 1994.

[6] J.Y. Tham, S. Ranganath, M. Ranganath, and A.A. Kassim, "A Novel Unrestricted Center-Biased Dimond Search Algorithm for Block Motion

Estimation," IEEE Trans. Circuits Syst. Video Technol., vol.8, no.4, pp.369-377, Aug. 1998.

[7] C. Zhu, X. Lin, and L.P. Chau, "Hexagon-Based Search Pattern for Fast Block Motion Estimation," IEEE Trans. Circuits Syst. Video Technol., vol.12, no.5, pp.349-355, May 2002.

[8] W. Li and E. Salari, "Successive Elimination Algorithm for Motion Estimation," IEEE Trans. Image Processing, vol.4, no.1, pp.105-107, Jan. 1995.

[9] V.L. Do and K.Y. Yun, "A Low-Power VLSI Architecture for Full-Search Block-Matching Motion Estimation," IEEE Trans. Circuits Syst. Video Technol., vol.8, no.4, pp.393-398, Aug. 1998.

[10] J.H. Luo, C.N. Wang, and T. Chiang, "A Novel All-Binary Motion Estimation (ABME) With Optimized Haredware Architectures," IEEE Trans. Circuits Syst. Video Technol., vol.12, no.8, pp.700-712, Aug. 2002.

[11] M. Bierling, "Displacement estimation by hierarchical block matching," *Proc. SPIE*, vol. 1001, pp. 942–951, 1988.

[12] B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vector," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, pp. 148–157, Apr. 1993.

[13] K. T. Choi, S. C. Chan, and T. S. Ng, "A new fast motion estimation algorithm using hexagonal subsampling pattern and multiple candidate search," in *Proc. ICIP*, 1996, pp. 497–500.

[14] C.N. Wang, S.W. Yang, C.M. Liu and T. Chiang, "A Hierarchical Decimation Lattice Based on *N*-Queen With an Application for Motion Estimation," IEEE Signal Processing Letters., vol.10, no.8, pp.228-231, Aug. 2003.

[15] C.N. Wang, S.W. Yang, C.M. Liu and T. Chiang, "A Hierarchical *N*-Queen Decimation Lattice and Hardware Architecture for Motion Estimation," IEEE

Trans. Circuits Syst. Video Technol., vol.14, no.4, pp.429-440, April 2004.

[16] Y.-L. Chan and W.-C. Siu, “New adaptive pixel decimation for block motion vector estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 113–118, Feb. 1996.

[17] Y. K.Wang, Y. Q.Wang, and H. Kuroda, “A globally adaptive pixel-decimation algorithm for block-motion estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 1006–1011, Sept. 2000.

[18] A. V. Oppenheim, R. W. Schafer and J. R. Buck, “Discrete-Time Signal Processing”, Prentice Hall

[19] V. Bhaskaran and K. Konstantinides, “Image and Video Compression Standards: Algorithm and Architectures”, Kluwer Academic.

[20] Iain E. G. Richardson, “H.264 and MPEG-4 Video Compresion: Video Coding for Next-generation Multimedia”, WILEY.

[21] Swee Yeow Yap; McCanny, J.V., “A VLSI architecture for variable block size video motion estimation” Circuits and Systems II: Express Briefs, IEEE Transactions on [see also Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on] Volume 51, Issue 7, July 2004 Page(s):384 - 389

[22] Heiko Schwarz, and Thomas Wiegand, “The Emerging JVT/H.26L Video Coding Standard,” *in Proc. of IBC 2002, Amsterdam, NL, September 2002*, invited paper.

[23] Kurceren, R.; Karczewicz, M., “Synchronization-predictive coding for video compression: the SP frames design for JVT/H.26L,” *Image Processing. 2002. Proceedings. 2002 International Conference* on , Volume: 2 , 2002

[24] P. Pirsch, “VLSI architectures for video compression—A survey,” *Proc. IEEE*, vol. 83, pp. 220–246, Feb. 1995.

[25] P. M.Kuhn, "Fast MPEG-4 motion estimation: Processor based and flexible VLSI implementations," *J. VLSI Signal Processing Syst. Signal, Image, Video Technol.*, vol. 23, pp. 67–92, Oct. 1999.

[26] L. de Vos and M. Schobinger, "VLSI architecture for a flexible block matching processor," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, pp. 417–428, Oct. 1995.

[27] J. F. Shen *et al.*, "A novel low-power full-search block-matching motion-estimation design for H.263+," *IEEE Trans. Circuits Syst. Video Technol.*, no. 7, pp. 890–897, July 2001.

[28] K. M. Yang and L. Wu, "A family of VLSI designs for the motion ompensation block-matching algorith," *IEEE Trans. Circuits Syst.*, vol. 36, pp. 1317–1325, Oct. 1989.

[29] D. Hilbert, "Üeber die stetige abbildung einer linie aufein flächenstück," *Mathematische Annalen*, vol. 38, pp. 459–460, 1891.

[30] S. Kamata, M. Niimi, and E. Kawaguchi, "A method of an interactive analysis for multi-dimensional images using a Hilbert curve," The Transactions of the Institute of Electronics, Information and Communication Engineers D-II, vol. J77-D-II, pp. 1255–1264, July 1994.

[31] R. J. Stevens, A. D. Lehar, and F. H. Preston, "Manipulation and presentation of multidimensional image data using the Peano scan," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-5, pp. 520–526, Sept. 1983.

[32] B. Moghaddam, K. J. Hintz, and C. V. Stewart, "Space-filling curves for image compression," *Proc. SPIE Conf. Automatic Object Recognition*, vol. 1471, pp. 414–421, 1991.

[33] A. C. Ansari and A. Fineberg, "Image data ordering and compression using Peano scan and LOT," *IEEE Trans. Consumer Electron.*, vol. 38, pp. 436–445,

Aug. 1992.

[34] S. Kamata, “A study on data compression for gray images using a Hilbert scan,” The Transactions of the Institute of Electronics, Information and Communication Engineers D-II, vol. J80-D-II, pp. 426–433, Feb. 1997.

[35] Y. Wang and H. Kuroda, “Hilbert scanning search algorithm for motion estimation,” *IEEE Trans., Circuits Syst. Video Technol.*, vol. 9, pp. 683–691, Aug. 1999.

[36] http://iphome.hhi.de/suehring/tml/download/old jm/JM9.2.zip

[37] http://www.m4if.org/resources.php#section26

[38] Zhibo Chen, Peng Zhou and Yun He, “Fast Motion Estimation for JVT,” Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6) 7th Meeting: Pattaya II, Thailand, 7-14 March, 2003

[39] Hsien-Wen Cheng and Lan-Rong Dung, "A Vario-Power ME Architecture Using the Content-Based Subsample Algorithm," IEEE Transactions on Consumer Electronics, Volume 50, Issue 1, pp.349 - 354, Feb 2004.

[40] Hsien-Wen Cheng and Lan-Rong Dung, "EFBLA: A Two-phase matching algorithm for FAST motion estimation," IEE Electronic Letter, Volume 40, Issue 11, pp.660 - 661, May 2004

[41] Hsien-Wen Cheng and Lan-Rong Dung, "A Power-Aware Motion Estimation Architecture Using Content-based Subsampling," Journal of Information Science and Engineering (accept)

[42] Hsien-Wen Cheng and Lan-Rong Dung, "A Content-Based Methodology for Power-Aware Motion Estimation Architecture," IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing (accept)

[43] Hsien-Wen Cheng and Lan-Rong Dung, "EFBLA: A Two-Phase Matching

Algorithm for Fast Motion Estimation," IEEE Pacific-Rim Conference on Multimedia, Dec. 2002.

[44] Hsien-Wen Cheng and Lan-Rong Dung, “A content-based motion estimation algorithm for power-aware architecture,” IEEE International Symposium on Image and Signal Processing and Analysis, Sep. 2003.

[45] Hsien-Wen Cheng and Lan-Rong Dung, "A Power-Aware Architecture for Motion Estimation," the 14th VLSI Design/CAD Symposium, 2003, Taiwan.

[46] Hsien-Wen Cheng and Lan-Rong Dung, "A Novel Vario-Power Architecture of Motion Estimation Using A Content-Based Subsample Algorithm," IEEE Workshop on Signal Processing Systems, Aug. 2003.

[47] Hsien-Wen Cheng and Lan-Rong Dung, "A Power-Aware ME Architecture Using Subsample Algorithm," IEEE International Symposium on Circuits and Systems, May 2004