Very Low Bit-Rate Video Coding Focusing On Moving Regions Using Three-tier Arbitrary-Shaped Pattern Selection Algorithm

Manoranjan Paul, Manzur Murshed, and Laurence Dooley

Gippsland School of Computing and IT, Monash University, Churchill Vic 3842, Australia

E-mail: {Manoranjan.Paul,Manzur.Murshed,Laurence.Dooley}@infotech.monash.edu.au

Abstract

Very low bit-rate video coding using patterns to represent moving regions in macroblocks exhibits good potential for improved coding efficiency. Recently an *Arbitrary Shaped Pattern Selection* (ASPS) algorithm and its *Extended* version (EASPS) were presented, that used a dynamically extracted set of patterns, of the two different sizes, based on actual video content. These algorithms, like other pattern matching algorithms failed to capture a large number of *active-region macroblocks* (RMB) especially when the object moving regions is relatively larger in a video sequence. As the size of the moving object may vary, superior coding performance is achievable by using dynamically extracted patterns of a larger size. This paper, proposes a three-tier *Arbitrary Shaped Pattern Selection* (ASPS-3) algorithm that uses three different pattern sizes for very low bit rate coding. Experimental results show that ASPS-3 exhibits better performance compared with other pattern matching algorithms, including the low-bit rate video coding standard H.263.

Keywords: *Video coding, pattern matching, moving regions, low bit rate, arbitrary shaped objects.*

1. Introduction

Reducing the transmission bit-rate while concomitantly retaining image quality continues to be a challenge for efficient very low bit-rate video compression standards, such as H.263 [4]. These standards are however unable to encode moving objects within a 16×16 pixel *macroblock* (MB) during *motion estimation* (ME), resulting in all 256 residual error values being transmitted for *motion compensation* (MC) regardless of whether there are moving objects. One solution is to sub-divide the MB and apply ME and MC to each sub-block. With a sufficient number of blocks, the shape of a moving object can be accurately represented, but this has a high processing expenditure [1].

The MPEG-4 [3] standard first introduced the concept of content-based coding, by dividing video frames into separate segments comprising a background and one or more moving objects. To address the limitations of [1], Wong *et al.* [13] exploited the idea of partitioning the MBs via a segmentation process that again avoided handling the exact shape of moving objects, so that popular MB-based motion estimation techniques could be applied. [13] classified each MB into three distinct categories: 1) *Static MB* (SMB): MBs that contain little or no motion; 2) *Active MB* (AMB): MBs which contain moving object(s) with little static background; and 3) *Active-Region MB* (RMB): MBs that contain both static background and part(s) of moving object(s). SMBs and AMBs are treated in exactly the same way as in H.263. For RMB coding, it was assumed that the moving parts of an object might be represented by one of the eight predefined patterns P_1–P_8 in Figure 1. An MB is classified as RMB if using some *similarity* measure, the part of a moving object of an MB is well covered by a particular pattern. The RMB can then be coded using the 64 pixels of that pattern with the remaining 192 pixels being skipped as *static background*.

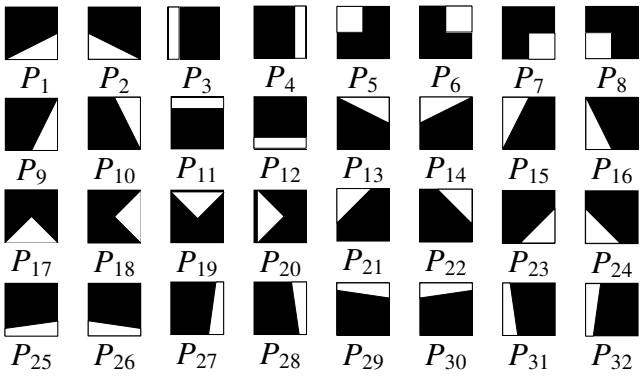

Figure 1: The pattern codebook of 32 regular shaped, 64-pixel patterns, defined in 16×16 blocks, where the white region represents 1 (motion) and black region represents 0 (no motion).

Other pattern matching algorithms have been reported [5]-[8], while Paul *et al.* in [9] proposed a new MB classification definition that outperformed [13]. Figure 1 shows the complete 32-pattern codebook, where each 64-pixel pattern is *regular*—bounded by straight lines, *clustered*—the pixels are connected, and *boundary-adjoined*. All these various pattern-matching algorithms approximate the actual shape of an object by assuming the moving regions of a RMB are similar to one or more of the patterns in the codebook (Figure 1). A key problem in using predefined patterns is finding the optimal number for coding.

An arbitrarily shaped region-based MC algorithm may offer a better solution. Yokoyama *et al.* **Error! Reference source not found.** applied image segmentation techniques to acquire arbitrarily shaped regions, though its performance was not guaranteed because it used a large number of thresholds. Paul *et al.* [10] proposed an *arbitrary shaped pattern selection* (ASPS) algorithm based on extracted arbitrary shaped same size pattern from the video contents, whose performance was better than the best pre-defined pattern-matching algorithm. Paul *et al.* [11] recently proposed *Extended* ASPS (EASPS) algorithm with two different size patterns, whose performance was better than the ASPS algorithm in very low bit rate video coding. Table I shows the MB types percentage generated by the EASPS algorithm, where SRMB means small size RMB and LRMB means large size RMB. While SMBs need no information, SRMBs need 25%, LRMB 50%, and AMBs 100% information to store or transmit, so any attempt to further reduce the number of AMBs leads to better compression.

Table I: Percentage of different MB types generated by the ASPS algorithm

Video sequences	SMB (%)	SRMB (%)	LRMB (%)	AMB (%)
Miss America	63	21	7	10
Suzie	29	23	18	29
Mother & Daughter	43	28	9	20
Car phone	22	28	23	27
Foreman	14	27	25	33
Claire	77	15	3	4

In this paper, an *Arbitrary Shaped Pattern Selection* (APS-3) algorithm with three pattern sizes {64,128,192}, i.e., three tiers, is proposed which divides the MB into 5 types; SMB, *Small* RMB (SRMB), *Medium* RMB (MRMB), *Large* RMB (LRMB) and a traditional AMB. The previously defined LRMB [10] is defined as a *medium RMB* (MRMB). Experimental results confirm that the overall performance of the new ASPS-3 algorithm with 5 types of MBs is not only superior to H.263 but also other contemporary pattern matching algorithms for very low bit rate video coding.

This paper is organized as follows. The video coding strategy using the three tier ASPS algorithm is described in Section 2, while some simulation results are analyzed in Section 3 and conclusions presented in Section 4.

2. Low Bit-Rate Video Coding Using ASPS-3

Prior to video coding, three *pattern codebooks* (PCs) with different pattern sizes have to be constructed. Using these PCs, actual coding will be performed. These PCs are constructed by using the different *candidate* RMB (CRMB) for each. The first PC consists of 8 patterns of 64 pixels each, the second PC consists of 4 patterns of 128 pixels each, and the third PC consists of 2 patterns of 192 pixels each. When the number of pixels in a pattern increases, it decreases the possible orientation of pixels that leads to a small number of patterns. As a result small patterns use a larger PC size and vice versa.

The whole pattern matching process can be divided into two phases, in first phase, the PCs are formulated on the basis of the actual video content, while in the second phase; the coding is undertaken using these content-dependent PCs.

2.1 Three Sets of PCs Generation

Let $C_k(x,y)$ and $R_k(x,y)$ denote the k^{th} MB of the current and reference frames, each of size W pixels$\times H$ lines, respectively of a video sequence, where $0 \le x, y \le 15$ and $0 \le k < W/16 \times H/16$. The moving region $M_k(x,y)$ of the k^{th} MB in the current frame is obtained as follows:

$$M_k(x,y) = T(|C_k(x,y) \bullet B - R_k(x,y) \bullet B|) \qquad (1)$$

where B is a 3×3 unit matrix for the morphological closing operation $\bullet$ [2], which is applied to reduce noise, and the thresholding function $T(v) = 1$ if $v > 2$ and 0 otherwise.

Algorithm ASPG(λ, χ, μ)

Parameters: $\lambda \in \{8,4,2\}$ *is the Number of patterns in a PC*; $\chi \in$ {CSRMB, CMRMB, CLRMB} *candidate RMBs*; $\mu \in$ {64,128,192} *Number of moving pixels in a pattern.*

Return: PC of λ *size.*

Step 1: Classify the χ into λ classes $C_1, C_2, \ldots, C_\lambda$ by any clustering method, e.g. FCM, using the gravitational centres of the χ.

Step 2: For $i = 1, 2, \ldots, \lambda$

Step 2.1: Calculate a temporary 2D array $T_i(x \times 16 + y) = \left(\sum_{j=1}^{|C_i|} C_{i,j}(x,y), x, y \right)$ and rank in descending order using the first field as the key, where $C_{i,j}$ is the j^{th} χ in class C_i and $0 \le x, y \le 15$.

Step 2.2: Set $P_i(x,y) = 0$ for $0 \le x, y \le 15$.

Step 2.3: For $l = 0, 1, \ldots, \mu$, $P_i(T_i(l,1), T_i(l,2)) = 1$.

Figure 2: The ASPG algorithm.

According to the number of '1' in an M_k, the k-th MB is classified as one of three CRMB types. *Candidate small* RMB (CSRMB): $8 \le \sum M_k < 128$. Candidate medium RMB (CMRMB): $128 \le \sum M_k < 192$ Candidate large RMB (CMRMB): $192 \le \sum M_k < 256$. CSRMB is used to generate the first PC of 8 patterns of 64-pixel each, CMRMB is used to generate the second PC of 4 patterns of 128-pixel each and CLRMB is used to generate the third PC of 2 patterns of 192-pixel using ASPG algorithm with $\lambda = \{8,4,2\}$ and $\mu = \{64,128,192\}$ correspondingly.

The *Arbitrary Shaped Pattern Generation* (ASPG) algorithm detailed in Figure 2 then generates the three sets of

PCs from CSRMBs, CMRMBs, and CLRMBs individually. . Let P_1, …, P_λ be the CRMBs for any of the above three classes, where λ is the pattern codebook size. Any clustering method, such as the fuzzy c-means (FCM) can be used in the ASPG algorithm. The clustering method classifies all CRMBs into λ classes using the *gravitational centre* (GC), $G(A)$, which for a 16×16 binary matrix A, is given by:-

$$G(A) = \begin{pmatrix} \frac{1}{\sum_{x=0}^{15}\sum_{y=0}^{15} A(x,y)} \sum_{x=0}^{15}\sum_{y=0}^{15} xA(x,y), \\ \frac{1}{\sum_{x=0}^{15}\sum_{y=0}^{15} A(x,y)} \sum_{x=0}^{15}\sum_{y=0}^{15} yA(x,y) \end{pmatrix}. \quad (2)$$

By using FCM, those CRMBs with less inter GC distance are placed in the same class. The ASPG algorithm then adds all the corresponding '1's of those CRMBs in the same class to provide the most populated moving region. To create the μ-most populated moving regions as a pattern, only the first μ-pixel positions are assigned '1' with the rest assigned '0'.

2.2 Coding

After obtaining the P_1, … ,P_λ patterns from a sequence using ASPG, the MBs are classified as SMB, SRMB, MRMB, LRMB, and AMB using the rules in Table II. For the best-matched pattern, the following similarity measure is used:-

$$D_{k,n} = \frac{1}{256}\sum_{x=0}^{15}\sum_{y=0}^{15} |M_k(x,y) - P_n(x,y)| \quad (3)$$

where $1 \le n \le \lambda$, M_k and P_n are the k^{th} macroblock and nth pattern respectively.

From the similarity measure, ASPS-3 selects only one from each PC for which the following condition is satisfied

$$D_{k,n} = \min_{\forall P_i \in \mathrm{PC}} (D_{k,i} \,|\, D_{k,i} < T_S). \quad (4)$$

where T_S is a similarity threshold.

For the SRMB, MRMB and LRMB three different pattern sets are used when calculate the $D_{k,n}$. To select the three best patterns for corresponding SRMB, MRMB, and LRMB, the corresponding T_S's values are also different. It is assumed in this paper that $T_S = \mu/256$, since if none of the μ-pixels of a particular pattern cover any part of a moving region, then the pattern similarity metric will be $\ge \mu/256$, e.g., T_s={0.25, 0.5, 0.75} when μ = {64,128,192}.

Note that, each CRMB is compared against all corresponding patterns of corresponding PC to maximize the image quality, as sometimes the extracted pattern from a cluster is unable to capture all the moving regions of the CRMBs in that particular cluster. There is always a possibility of misclassifying a CRMB as an AMB, if only the extracted pattern from a cluster is used to match against the CRMBs of the same cluster.

Since every SMB and the static region of a SRMB, MRMB, and LRMB are considered as having zero motion, they are omitted from the coding and transmission since they can be obtained from the reference frame. For each AMB, as well as the moving region of each SRMB, MRMB, and LRMB, motion vectors and residual errors are calculated using conventional block-based processing, with the obvious difference in having the shape of the blocks for the moving regions of SRMBs, MRMBs, and LRMBs as that of the best-match pattern.

To process the SRMB, MRMB, and LRMB, a motion vector is calculated from only the μ moving pixels of the best-match pattern. To avoid extra 8×8 blocks of DCT calculations for μ residual error values per SRMB, MRMB, and LRMB, these μ values are rearranged into an 8×8 block. This avoids unnecessary DCT block transmission, since for example, when μ = 64 and μ = 128 only one and two DCT blocks respectively need to be transmitted, while for μ = 192, this increases to three DCT blocks. A similar inverse procedure is performed during the decoding. Unlike fixed pre-defined pattern selection algorithms, in any arbitrary shaped pattern selection algorithm original patterns need to be transmitted.

From an optimum coding point of view, 5 MB types can be justified since an SMB is skipped in the H.263 standard. The remaining 4 MB types can thus be coded using two bits.

Table II: Classifying rules for MB types

MB Type	Conditions
SMB	$\sum M_k < 8$.
SRMB	$8 \le \sum M_k < 128$ and find a pattern P_n so that $D_{k,n} = \min_{\forall P_i \in \mathrm{PC1}} (D_{k,i} \,|\, D_{k,i} < 0.25)$.
MRMB	$128 \le \sum M_k < 192$ or CSRMB not classify due to threshold for SRMB and find a pattern P_n so that $D_{k,n} = \min_{\forall P_i \in \mathrm{PC2}} (D_{k,i} \,|\, D_{k,i} < 0.5)$.
LRMB	$192 \le \sum M_k < 256$ or CMRMB not classify due to threshold for MRMB and find a pattern P_n so that $D_{k,n} = \min_{\forall P_i \in \mathrm{PC3}} (D_{k,i} \,|\, D_{k,i} < 0.75)$.
AMB	Otherwise AMB

3. Simulation Results

The ASPS-3 algorithm along with a number of other low-bit rate coding algorithms has been tested on a large number of standard and non-standard video sequences of QCIF digital video formats [12]. In this paper, experimental results are presented using the first 100 frames of seven standard video test sequences. Full-search motion estimation and the H.263 recommended variable length coding were employed to obtain the encoding results using the new ASPS-3 approach, as well as the ASPS, EASPS, Fixed-8, and H.263. The ASPS-3 algorithm used λ = {8,4,2} T_s={0.25, 0.5, 0.75} for SRMB, MRMB, and LRMB respectively. Different λ combinations for SRMB, MRMB, and LRMB were also investigated, such as {8,4,4}, {8,2,2}, and {2,4,8}, though

the {8,4,2} combination consistently provided the best results. A small PC size for large patterns is sufficient to encode the moving regions because it increases the overlapping regions among the patterns. Conversely a large PC size is needed for small patterns, because there are less overlapping regions amongst the patterns.

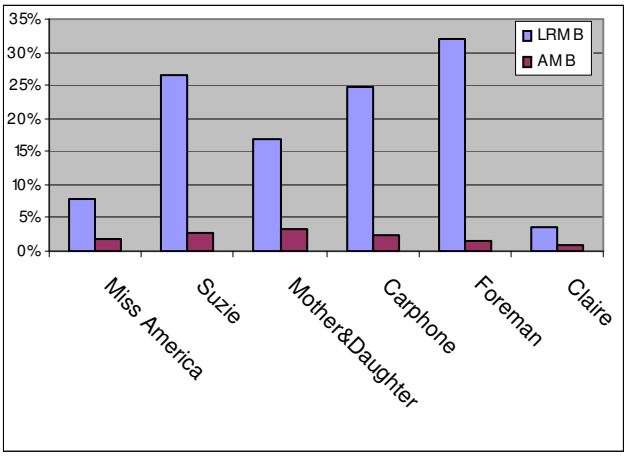

Figure 3: Percentage of LRMB and AMB generated by ASPS-3 algorithm for six standard video sequences.

Figure 3 shows that a significant number of AMBs are now classified as LRMBs and the total number of LRMBs varied from 4% to 32%, so justifying the motivation behind the ASPS-3 algorithm. The percentages of SMB, SRMB, and MRMB are similar to that of previous algorithms, so they are not included in Figure 3.

Table III confirms that ASPS-3 provides superior results compared to the H.263 standard, the Fixed-8 and ASPS algorithms for low bit rate video coding. For example, the ASPS-3 algorithm performs better than H.263 by reducing the bit-rate by 21.8% for the smooth motion *Mother and Daughter* sequence. ASPS-3 has proven to be a better algorithm than all pattern-matching algorithms, which used a predefined or arbitrary shape pattern set.

The computational complexity of any arbitrary shaped pattern selection algorithm ([10], [11]) is always greater than that of a pre-defined pattern selection algorithm [5] - [8] and [13], due to the clustering algorithm used in arbitrary pattern extraction. As a result, an arbitrary shaped pattern selection algorithm is unsuitable for real-time applications, though for archiving video and transmission delay tolerant systems, it can be used and its better compression and quality performance exploited.

It is interesting to observe that the shape of the 64-pixel small patterns for the *Miss America* sequence in Figure 4 is not far removed from the pre-defined pattern set in Figure 1. Clearly, the pattern set extracted directly from the video content provides better results than the pre-defined pattern set. Another interesting observation in Figure 4 is that the large pattern set (LP) is not clustered in one region and inter-pattern overlapping is relatively larger than in any of the other codebooks, thereby validating the choice of a small PC size for the LP

The coding performance comparison curve in Figure 5 shows that ASPS-3 algorithm is better than the ASPS, EASPS algorithms for very low bit rate video coding though a diminishing trend is observed as the bit rate increases. At higher bit rates, the performance of the ASPS-3 algorithm may actually become worse than other techniques for some video sequences, because of the additional overhead due to pattern identification bits.

Table III: Percentage of bit-rate reduction for same PSNR values using the ASPS-3, EASPS, ASPS, and Fixed-8 compare to H.263 for six standard sequences.

Video sequences	PSNR (dB)	Bit Rate (Kbps)			
		ASPS-3	EASPS	ASPS	Fixed-8
Miss America	35.5	20.8	18.8	15.2	6.3
Suzie	29.5	18.9	14.1	7.3	1.2
Mother&Daughter	29.0	21.8	18.2	14.4	5.6
Car phone	29.5	12.6	9.4	7.8	1.2
Foreman	26.0	16.5	13.3	6.6	2.2
Claire	33.5	14.3	13.4	10.4	3.0

4. Conclusions

In this paper, a novel three-tier *Arbitrary Shaped Pattern Selection* (ASPS-3) algorithm has been developed using three-extracted pattern sets based on actual video content to approximate the shape of different sized moving objects in a macroblock. By exploiting the arbitrariness of video objects, it has been proven to be a better pattern-matching algorithm compared to contemporary pre-defined and arbitrary shaped pattern-based algorithms. Experimental results proved that the ASPS-3 algorithm provided superior results for all video test sequences, compared with the ASPS, EASPS, Fixed-8 algorithms and digital video coding standard H.263. In particular, this performance was most significant at very low-rates, where a reduction of more than 20% was achieved for the same picture quality, compared with the H.263 standard.

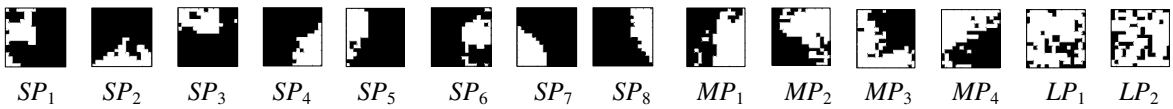

Figure 4: 64-pixel 8 small patterns (SP), 128-pixel 4 medium patterns (MP), and 192-pixel 2 large patterns (LP) are extracted using ASPS-3 algorithm from *Miss America*, where the white region represents 1 (motion) and black region represents 0 (no motion).

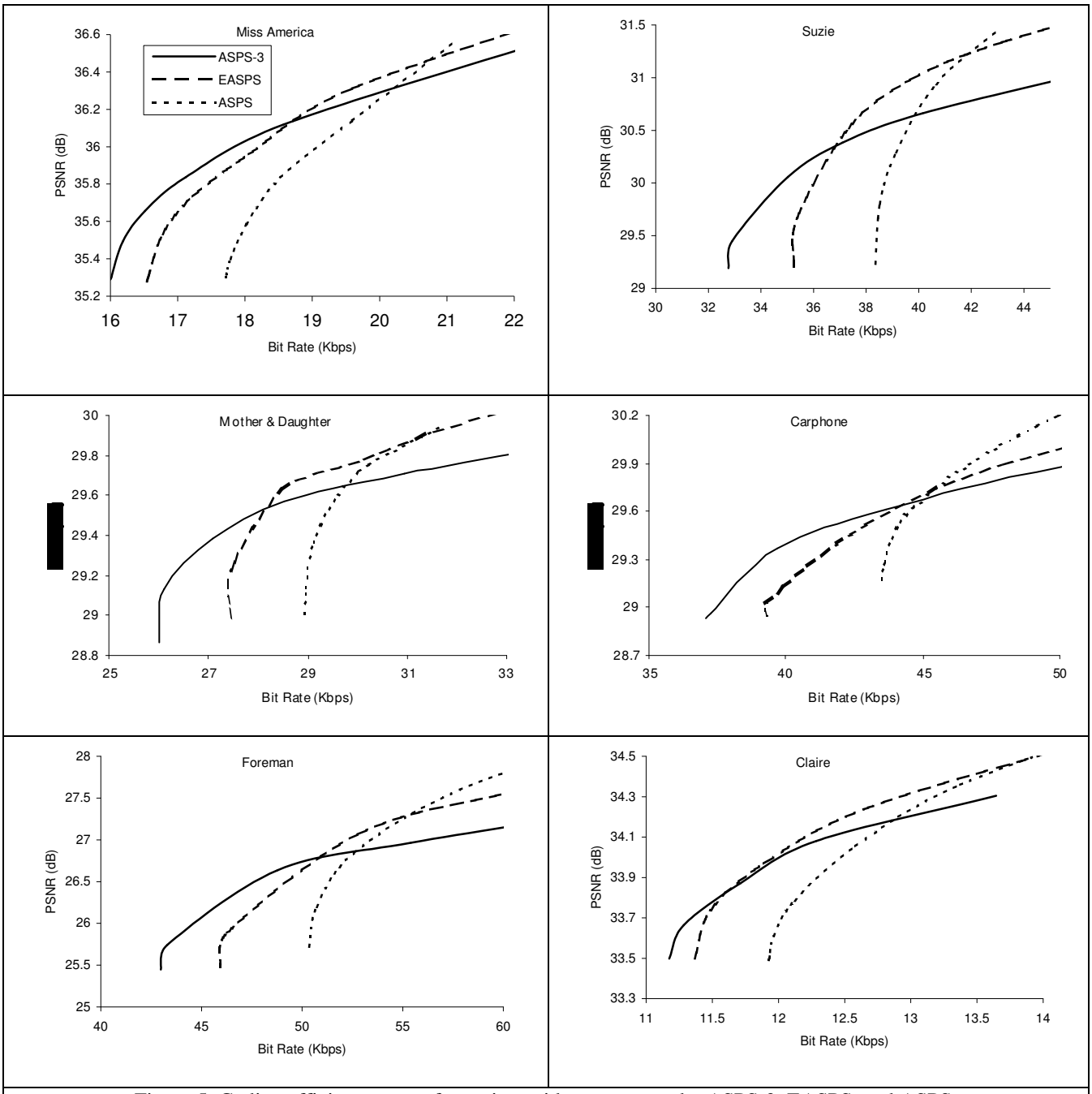

Figure 5: Coding efficiency curve for various video sequences by ASPS-3, EASPS, and ASPS.

5. References

[1] Fukuhara, T., K. Asai, and T. Murakami, "Very low bit-rate video coding with block partitioning and adaptive selection of two time-differential frame memories," *IEEE Trans. Circuits Syst. Video Technol.*, 7, 212–220, 1997.

[2] Gonzalez, R.C. and R. E. Woods, *Digital Image Processing*, Addison-Wesley, 1992.

[3] ISO/IEC N4030, MPEG-4 International Standard, 2001.

[4] ITU-T Recommendation H.263, "Video coding for low bit-rate communication," Version 2, 1998.

[5] Paul, M., M. Murshed, and L. Dooley, "A Low Bit-Rate Video-Coding Algorithm Based Upon Variable Pattern Selection," Proc. of 6th IEEE Int. Conf. on Signal Processing (ICSP-02), Beijing, Vol-2, 933–936, 2002.

[6] Paul, M., M. Murshed, and L. Dooley, "A new real-time pattern selection algorithm for very low bit-rate video coding focusing on moving regions," Proc. of IEEE Int. Conference of Acoustics, Speech, and Signal Processing (ICASSP-03), Hong Kong, Vol-3, III_397-III_400, 2003.

[7] Paul, M., M. Murshed, and L. Dooley, "A Real Time Generic Variable Pattern Selection Algorithm for Very Low Bit-Rate Video Coding," Proc. of IEEE Int. Conference on Image Processing (ICIP-03), Barcelona, Spain, 2003.

[8] Paul, M., M. Murshed, and L. Dooley, "A Variable Pattern Selection Algorithm with Improved Pattern Selection Technique for Low Bit-Rate Video-Coding Focusing on Moving Objects," Proc. of Int. Workshop on Knowledge Management Technique (IKOMAT-02), Crema, Italy, 1560–1564, 2002.

[9] Paul, M., M. Murshed, and L. Dooley, "Impact of Macroblock Classification on Low Bit Rate Video Coding Focusing on Moving Region," Proc. of Int. Conf. of Computer and Information Technology (ICCIT-02), Dhaka, Bangladesh, 465–470, 2002.

[10] Paul, M., M. Murshed, and L. Dooley, "An Arbitrary Shaped Pattern Selection Algorithm for Very Low Bit-Rate Video Coding Focusing on Moving Regions," Proc. of Fourth IEEE Pacific-Rim Int. Conference on Multimedia (PCM-03), 2003, Singapore.

[11] Paul, M., M. Murshed, and L. Dooley, "Very Low Bit Rate Video Coding Using an Extended Arbitrary Shaped-Pattern Selection Algorithm," submitted to Fifth Int. Conference on Advances in Pattern Recognition (ICAPR-03), 2003, India.

[12] Shi, Y.Q. and H. Sun, *Image and Video Compression for Multimedia Engineering Fundamentals, Algorithms, and Standards*, CRC Press, 1999.

[13] Wong, K.-W., K.-M. Lam, and W.-C. Siu, "An Efficient Low Bit-Rate Video-Coding Algorithm Focusing on Moving Regions," *IEEE trans. circuits and systems for video technology*, 11(10), 1128–1134, 2001.

[14] Yokoyama, Y., Y. Miyamoto, and M. Ohta, "Very Low Bit-Rate Video Coding Using Arbitrarily Shaped Region-Based Motion Compensation," *IEEE trans. circuits and systems for video technology*, 5(6), 500–507, 1995.