Real-Time H.264 Video Encoding in Software with Fast Mode Decision and Dynamic Complexity Control

YURI V. IVANOV AND C. J BLEAKLEY
SCHOOL OF COMPUTER SCIENCE AND INFORMATICS, UNIVERSITY COLLEGE DUBLIN, IRELAND

This paper presents a novel real-time algorithm for reducing and dynamically controlling the computational complexity of an H.264 video encoder implemented in software. A fast Mode Decision algorithm, based on a Pareto optimal MacroBlock classification scheme, is combined with a Dynamic Complexity Control algorithm that adjusts the MB Class decisions such that a constant frame rate is achieved. The average coding efficiency of the proposed algorithm was found to be similar to that of conventional encoding operating at half the frame rate. The proposed algorithm was found to provide lower average bit rate and distortion than Static Complexity Scaling.

Categories and Subject Descriptors: I. Computing Methodologies [**I.4 IMAGE PROCESSING AND COMPUTER VISION**]: I.4.2 Compression (Coding) (E.4)
General Terms: Algorithms
Additional Key Words and Phrases: H/264/AVC, rate-distortion, real-time, mode decision, fast mode decision, complexity, complexity control

1. INTRODUCTION

In recent decades, there has been explosive growth in the field of multimedia communications. A key enabler of this growth has been the development of efficient video coding standards and systems. The H.264 video coding standard, developed by the Joint Video Team (JVT [2003]), provides better coding efficiency at low bit rates [Wiegand at al. 2003] than previous standards, such as MPEG-2 and H.263. However, for some applications, its deployment has been impeded due to the significantly increased computational complexity of the encoder. As a result, most real-time H.264 video encoders are implemented in specially designed hardware, rather than on general purpose processors. Clearly, implementation in software in general purpose, programmable processors would provide a more flexible and, in many cases, cost effective solution.

This research was supported by University College Dublin.
Authors' addresses: Dr Yuri V. Ivanov, Movidia Ltd., 19 Mountjoy Square, Dublin 1, Ireland; Dr C.J. Bleakley, School of Computer Science and Informatics, University College Dublin, Belfield, Dublin 4, Ireland.

A large number of algorithms have been proposed for reducing the computational complexity of H.264 video encoding. Generally, these algorithms focus on decision prediction and early termination to reduce the average complexity of the most computationally demanding components of the video encoder. In almost all cases, these algorithms aim to reduce the total encoding time. The algorithms show significant variations in frame encoding times. Hence, these methods are not well suited to real-time applications in which the encoder must maintain a constant frame encoding time.

In contrast, almost all published research on real-time software H.264 encoders focuses on low-level Instruction Set Architecture (ISA) optimizations and/or global encoding parameter adjustments, such as reduced search size or frame rate. Although these approaches are very straightforward, they are far from optimal in terms of the coding efficiency achieved.

This paper presents an algorithm for reducing the computational complexity of H.264 video encoding and dynamically controlling complexity such that the encoder meets real-time constraints while maintaining optimal coding efficiency. A fast Mode Decision (MD) method is presented which reduces the computational complexity of H.264 encoding and is amenable to complexity control. The method improves on previously published fast MD schemes by means of Pareto-optimal MacroBlock (MB) class definition and Rate-Distortion (RD) cost based mode prediction metrics. A Dynamic Complexity Control (DCC) method is presented which, based on an encoding time prediction model, adjusts the fast MD encoding parameters from MB to MB such that real-time coding is achieved. Operating in tandem, the methods achieve the goal of real-time H.264 video encoding in software with high coding efficiency. To the authors' knowledge, their work is the first to propose use of a classification-based Dynamic Complexity Control algorithm. The main advantages of the approach are RD optimal complexity scaling, constant frame encoding complexity and low bookkeeping overhead.

The paper is organized as follows. Section 2 reviews related work in the field. Section 3 provides an analysis of the low complexity encoding problem, describes the proposed fast MD method and presents experimental results. Section 4 contains an analysis of the Dynamic Complexity Control problem, details the proposed DDC method and provides experimental results. Section 5 concludes the paper.

2. RELATED WORK

Since the H.264 standard was adopted and its complexity was studied, various methods have been proposed to decrease the complexity of the encoder. Generally speaking, the work done by researchers in the field can be divided into two categories - non-real-time or low complexity algorithms and real-time implementations. Published work in these categories is described in the following two sub-sections.

2.1 Low Complexity H.264

Most of the complexity of the encoder is due to Motion Estimation (ME) and Mode Decision (MD) [Saponara at al. 2002]. Thus optimization of these tools has the greatest impact on the overall computational complexity of the encoder.

A key innovation in H.264 is that it provides Various Block Sizes (VBS). The conventional MD algorithm goes through all possible block modes and selects the one that has the lowest RD cost [Wiegand et al. 2003]. Conventionally, the modes are examined sequentially in the order P16x16, P16x8, P8x16, P8x8, I16x16, I4x4, SKIP where P denotes Inter coding relative to a motion compensated MB from the previous frame and I denotes Intra coding relative to surrounding MBs in the current frame. The numbers denote the size of the sub-blocks within a given 16x16 MB. A SKIP occurs when the MB can be simply copied from the previous frame without residual coding.

The RD cost, J, is calculated using the Lagrangian formula [Everett 1963]:

$$J = D + \lambda R \qquad (1)$$

where R is the number of bits used for encoding, D is the distortion, calculated as Sum of Square Errors (SSE), and λ is the Lagrange multiplier. The Lagrange multiplier controls the relative importance of bit rate and distortion.

Fast MD algorithms can be divided into four categories - Early Termination (ET), Forward SKIP Prediction, MB classification and Other.

ET techniques assume that some block modes can be eliminated from the mode search by predicting (i.e. without performing ME) that they would not be selected after a full search. The idea of Forward SKIP Prediction is to perform the SKIP decision first, not last. This can result in significant computational saving for low motion sequences for which the SKIP rate is high (40–60%).

Combinations of these techniques with various prediction methods are proposed in [Yin et al. 2003; Han and Lee 2004; Ahmad et al. 2004; Li et al. 2005; Kim et al. 2006; Kannangara et al. 2006]. Of these, we found the most effective, when used with recent versions of the JM reference software, to be [Kim et al. 2006] and [Kannangara et al. 2006], which achieve a 53% and a 19%–67% complexity reduction respectively, with insignificant bit rate gain.

In the MB classification approach, MBs are analyzed and classified according to their features. Different H.264 encoding parameters are used for each class. For example, ‘difficult’ MacroBlocks, i.e. those that are hard to encode, are allocated to classes with more computationally complex H.264 settings.

This idea was investigated in [Deepak and Chen 2001; Zhao and Richardson 2003; Wang and Zhu 2005; Yu 2004; Kim and Kuo 2005; Hong et al. 2005; Feng et al. 2006]. Of these, the most effective appears to be [Wang and Zhu, 2005]. This technique is based on the identification of active and inactive regions of the frame based on a Frame Difference (FD) metric. The computed FD value is compared with an activity threshold and only active macroblocks are processed using full H.264 complexity settings. Results showing a 42% complexity reduction indicate that more sophisticated classification and, possibly, use of an additional metric are required to achieve further speed improvements.

It worth noting that [Hong et al. 2005] and [Feng et al. 2006] also a propose classification based approach but focus on encoders where Rate-Distortion Optimization (RDO) is on. Switching RDO on significantly increases computational complexity. Since this work focuses on low complexity algorithms, RDO is switched off.

Other methods proposed by the researchers are difficult to categorize since they are quite specific. For instance, fast multi-block selection [Chang et al. 2004] is based on the idea of detecting fast and slow moving areas of the frame and processing them differently. The method in [Kuo and Chan 2004] is based on the correlation of Motion Vectors across the various MB partitions. The results achieved for these methods are similar to those obtained for the ET and forward SKIP schemes.

All of these algorithms reduce the total encoding complexity but do not maintain a constant frame encoding time. In addition, none of these papers address the problem of making the encoding process amenable to complexity control.

2.2 Real-time H.264 Encoding

In contrast, real-time encoders must maintain a constant frame encoding time. Most papers on the topic describe instruction-level optimizations for particular processors. For example, [Kant et al. 2006], [We and Canhui 2006] and [Hsu et al. 2005] describe H.264 implementations on the Analog Devices' BF561, Texas Instruments TMS320C6416, and Pocket PC respectively.

Joint Rate-Distortion-Complexity (RDC) optimization frameworks for H.264 have been proposed for static complexity scaling in [Hu 2006] and [Strottup-Andersen at al. 2004]. These frameworks seek to reduce total encoding complexity with minimum RD loss by determining the best combination of encoding parameters at each complexity point. These methods are static in that the encoding parameters are fixed *a priori* and do not change during the encoding process.

A number of papers have proposed algorithms for Dynamic Complexity Control of the H.264 encoder suitable for real-time applications. In [Goto et al. 2006], a simple method is described for CPU workload smoothing based on Peak-Load-Suppressed Diamond Search ME. The authors' of [Berger et al. 2007] describe an H.264 encoding algorithm that operates on a MB time budget and adaptively eliminates sub16x16 MB sizes so as not to exceed the time limit. [Shen et al. 2007] proposes a method for dynamic complexity control for Intra coding. A method that limits the reference frame and mode search based on the predicted complexity and an assessment of the benefits of search depth is described in [Wu et al. 2007]. [Ates and Altunbasak 2008] describes a method that limits mode search based on a spatiotemporal activity metric and uses a Lagrangian parameter for trading off complexity and RD performance. Unlike the work herein, these last two methods do not incorporate forward SKIP prediction or a frame buffering technique. The lack of a forward SKIP decision significantly increases computational complexity. The lack of a frame buffer means that the frame encoding time budget must be set conservatively. A graph in [Ates and Altunbasak 2008] shows that the lack of a frame buffer results in 20% of clock cycles going unused. [Kaminsky et al. 2008] proposes a method that provides Constant Bit Rate (CBR) and constant complexity encoding. In this work, we seek to maximize RD performance subject only to the processor performance constraint. Use of CBR can lead to significant variation in visual quality during a single sequence.

The proposal in [Akyol et al. 2007] bears most similarity to the algorithm described herein. However, there are a number of key differences. Firstly, [Akyol et al. 2007] uses a Pareto analysis to identify the Early Termination thresholds. In contrast, this work uses a Pareto analysis to identify the optimal mode search parameters as part of a classification approach [Wang and Zhu 2005]. Secondly, [Akyol et al. 2007] uses a PID control loop to maintain constant complexity. This loop uses recent MB encoding time measurements to adaptively predict the encoding time of the current frame. During scene transitions, the average MB encoding time can change rapidly and significantly. This can lead to large errors in encoding time prediction. Figure 5 of [Akyol et al. 2007] shows some frames requiring almost double the target coding time. In this work, MB encoding times are calculated for each class. At a scene transition, the distribution of MB Classes changes but the MB encoding times for each class do not. Thus, encoding time prediction is much more accurate. Thirdly, in this work, we solve the bookkeeping problem referred to in [Akyol et al. 2007] by only requiring storage of the class decision for each MB (3 bits per MB). Fourthly, class allocation for the whole frame prior to encoding allows our proposed algorithm to achieve constant Rate-Distortion-Complexity encoding over the whole frame, avoiding the intra-frame distortion variation arising from the use of MB-by-MB methods.

3. FAST MODE DECISION

As discussed previously, MB classification combined with ET and Forward SKIP Prediction shows promise for low complexity encoding. Design of a MB classification scheme requires selection of the MB classes and definition of accurate class decision metrics. Unlike previous work in the area, we propose to select the MB classes based on a formal analysis of the coding efficiency and computational complexity of the H.264 encoding tools and parameters. The class decision metrics are then defined based on their computational complexity and accuracy in predicting MB classes.

3.1 Analysis

3.1.1 Complexity Scaling.

The H.264 standard utilises a number of encoding tools and provides a number parameters for control of these tools. The computational complexity of the encoder can be scaled simply by adjusting the encoding parameters.

The question that arises is: for a given computational complexity requirement, what combination of encoding tools and parameter settings provides the best coding efficiency? This question was partially addressed in [ISO4964 2002] and [Saponara et al. 2002] but not in sufficient detail for the purposes of coder design. The method employed herein is similar to that described in [Strottup-Andersen at al. 2004], i.e. reduction of the 3 dimensional Pareto problem to 2 dimensions. However, different encoding parameters are considered herein.

Since changing complexity effects both bit rate and distortion, the need arises to unify both quantities into a single metric. Based on the commonly used RD model, we introduce a coding efficiency metric that is dependent on visual quality loss and bit rate change relative to a reference full complexity encoder:

$$W = \Delta R + \mu \Delta D \qquad (2)$$

where ΔR is a bit rate change as a percentage, ΔD is PSNR loss in dB and μ is a constant, relating bit rate loss and distortion increase. Thus, for any given computational complexity C, the most efficient encoder can be identified as that providing minimum W.

The constant μ can be interpreted as the percentage increase in bit rate equivalent to a 1 dB loss in PSNR. Previous work [Bjontegaard 2001] determined that a 10% decrease in bit rate is roughly equivalent to a loss of 0.5 dB in PSNR. In this work, μ was determined experimentally. Bit rate and PSNR were measured for Carphone, Container, Paris and Akiyo [Xiph.org], QCIF and CIF, at QP settings of 26, 28, 30 and 32. The results were plotted and the gradient was determined by fitting a linear model. The gradient was found to vary between 3.62 and 29.6 with a mean of 12.9.

The encoding time and coding efficiency of the encoder were measured across a range of encoding tool settings: VBS 1-7; search range 1, 2, 4, 6, 8; and Hadamard on/off. For QCIF and CIF, a search range greater than 8 provides little PSNR or bit rate advantage, but does lead to a significant complexity increase [ISO4964 2002; Saponara et al. 2002]. Hence values greater than 8 were excluded from the analysis. RDO on was also excluded since it leads to a complexity of almost 3 times while improving W only slightly to –2.83. Use of QP as a complexity control parameter was also considered but the idea was rejected due to the adverse impact that increasing QP has on visual quality. In experiments, increasing QP from 28 to 30 was found to reduce complexity by only 3.5% but to decrease PSNR by 4.3 dB, on average (W=28.2). Hence, QP is set *a priori*, based on application demands, and is not used as a complexity control parameter.

Measurements of complexity, bit rate and visual quality were performed for various combinations of tool settings using the JM 9.5 reference encoder applied a range of 300 frame QCIF and CIF video sequences. The results are shown in Figure 1. Complexity C is normalized relative to encoding with the full encoding parameters define herein as: full search, full VBS, search range 8, CABAC, full Hadamard, sub-pixel accuracy on, de-blocking filter on, RDO off and QP=28. The RDC optimum encoding parameters correspond to points on the Convex Hull of Individual Minima (CHIM).

The maximum PSNR loss is 0.37 dB when complexity is scaled to 28% of full complexity, which corresponds to a 17% bit rate increase. Subjective evaluation of visual quality revealed no anomalies in the decoded sequences. The slight quality degradation measured between full and minimal complexity modes can only be noticed on still images.

Clearly, a complexity scaling encoder should use parameter settings that are RDC optimal, i.e. are on the CHIM. Allowing the encoder to scale to all points on the CHIM would add unnecessary complexity. Hence, we identified a subset of the CHIM points for use in a classification-based fast MD algorithm. For effective scaling, the subset should be evenly distributed and span the CHIM. Five MB classes were selected, as shown in Figure 2. Experiments were conducted to refine the choice of parameter settings for classes B and C.

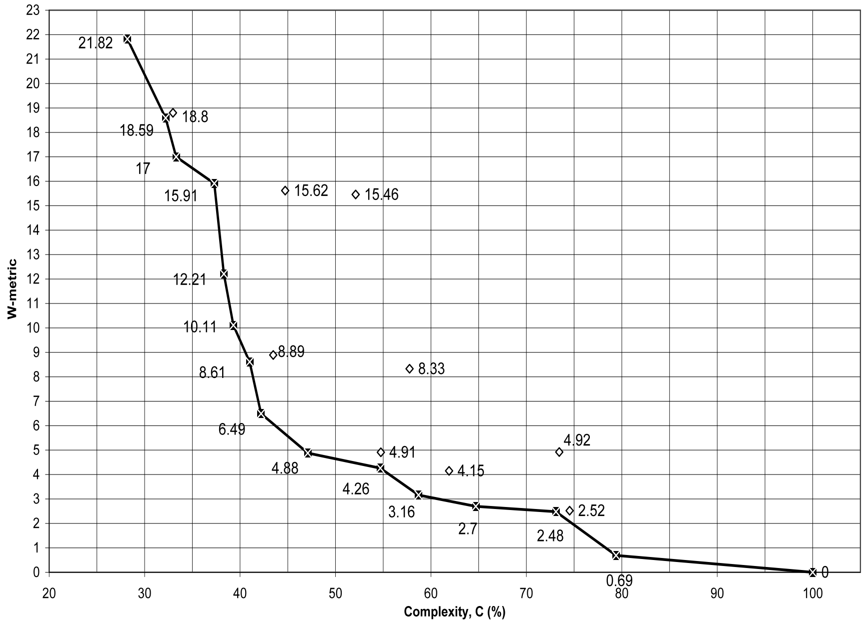


Fig. 1. Variation of *W* metric with encoding time.

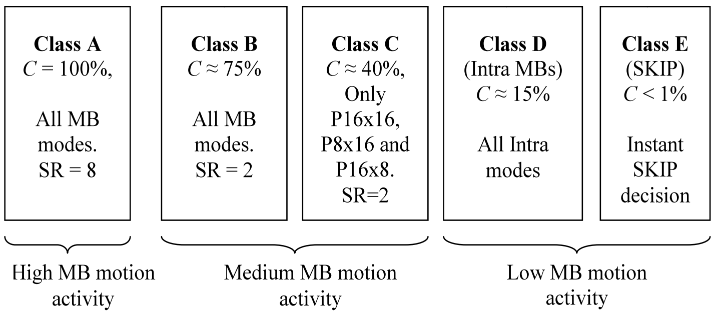


Fig. 2. MB classification.

3.1.2 Class Decision Metrics

Class decision metrics can be divided into three groups: those based on MB motion properties (e.g. average MV length, motion intensity), those based on statistical information (e.g. probabilities of MB modes) and those based on the properties of MBs (e.g. SAD, MAD, Frame difference, MB energy).

Metrics in the first group are low complexity in that they do not involve manipulation of pixels. Initial experiments and literature survey indicates that these metrics are weaker than visual metrics in that they do not provide good accuracy.

Statistical metrics are also low complexity. For various QCIF and CIF sequences, an analysis was performed of the probability of block mode transition from one frame to the next. The results are shown in the Table I. These findings were incorporated in the design of the fast MD algorithm by omitting searches for modes arising from low probability transitions.

Based on our analysis, some form of MB property metric is required. Metrics such as MAD, SSE and Frame Energy were excluded due to their high computational complexity. Use of the RD cost metric, *J*, as a class decision metric was studied in this work. RD cost has the significant advantage that it is calculated as part of the MD process and, in most cases, does not require an additional processing step. Thus, its incremental complexity cost is low.

Table I. Temporal Transition Probabilities for Different MB Modes, %

From\To	**SKIP**	**Inter (P-modes)**				**Intra (I-modes)**	
		P16x16	P16x8	P8x16	P8x8	I4x4	I16x16
SKIP	85.9	10.2	0.7	0.8	1.4	0.0	0.7
P16x16	40.5	40.8	3.3	3.5	10.6	0.2	0.9
P16x8	27.1	33.4	10.9	6.0	20.5	0.7	1.1
P8x16	27.4	31.0	5.2	12.2	22.3	0.8	0.8
P8x8	9.6	20.4	3.7	4.4	59.7	1.6	0.3
I4x4	25.1	12.9	2.6	3.6	25.4	24.4	5.6
I16x16	67.2	17.2	1.5	1.0	1.7	1.7	9.5

Experiments were conducted to determine if previously calculated values of the RD cost could be used as a prediction metric for ET. Pearson correlation coefficients, were calculated between *J* values obtained for MBs and their neighbours in the same and the previous frame after full search. In the experiments the optimal *J* for each MB was used. The Pearson coefficient was calculated as:

$$r_j = \frac{\sum_{i=1}^{N}(X_i - M_X)(Y_i - M_Y)}{(N-1)\cdot S_X \cdot S_Y} \tag{4}$$

where N is the number of 16x16 blocks in the video sequence, X_i is the RD cost of the current block and Y_i is the RD cost of the neighbouring block. For each value of j, the position of the neighbouring block is fixed relative to the current block. M_X, S_X, M_Y and S_Y

are the mean and standard deviation of X_i and Y_i respectively. The results obtained for a typical video sequence are provided in Figure 3.

The RD cost function J is highly correlated between neighbouring blocks. In comparison, correlation coefficients between Motion Vectors were reported in [Kossentini et al. 1997] to be around 0.4 for blocks in the same frame and around 0.15 for blocks in the previous frame. Based on this, it was decided to employ RD cost as a metric for ET and Forward SKIP Decisions.

		0.370		
	0.230	0.249	0.255	
0.500	0.632	0.843	0.614	0.488
	0.271	0.267	0.250	
		0.374		

		0.420	
	0.272	0.288	0.294
0.564	0.703	X	

Fig. 3. Correlation coefficients r_j for the current frame (right) and the previous frame (left) for Coastguard, CIF. X indicates the position of the current block.

In the case of SKIP decisions, which must be made for all MBs to ensure a low bit rate, calculation of the RD metric can be simplified by noting that RD cost is directly proportional to Distortion since Rate is approximately zero. Experiments using a range of video sequences were conducted to determine the correlation between Sum of Absolute Differences (SAD) and RD cost for SKIP MBs. The computed correlation coefficient was 0.985. This allows estimation of RD cost for SKIP MBs from SAD. Computational complexity was further reduced by use of partially computed SAD. The paper [Wang et al. 2004] proposes various SAD sub-sampling strategies. Correlation experiments for skipped macroblocks (SAD vs. J) were performed for various pixel patterns. It was found that the Quarter pixel pattern provides a mean correlation coefficient of 0.626. The Quarter pixel pattern is defined as:

$$SAD_{8x8} = \sum_{x=0,y=0}^{7,7} | s[2x,2y] - c[2x,2y] | \tag{5}$$

where s[x,y] is the received pixel and c[x,y] is the reconstructed pixel.

Use of the Frame Difference visual metric was also investigated [Wang and Zhu 2005]:

$$FD = \sum_{x=0}^{15} \sum_{y=0}^{15} \begin{cases} 1 & |c(x,y) - p(x,y)| \geq T_{\text{diff}} \\ 0 & |c(x,y) - p(x,y)| \leq T_{\text{diff}} \end{cases} \tag{6}$$

where $c(x,y)$ is the pixel value at position (x,y) for the MB in the current frame, $p(x,y)$ is the pixel of the previous frame and T_{diff} is an activity threshold, set equal to 10 [Wang and Zhu 2005]. Experiments over a range of video sequences indicate that the *FD* metric has a computational complexity of 46% that of SAD_{8x8} calculation. The metric was found to be effective for distinguishing between MBs with active motion and those with inactive motion.

3.2 Fast Mode Decision Algorithm

The proposed fast MD algorithm consists of two parts: Class Decision and Mode Search with ET. From the preceding analysis, a Class Decision algorithm was developed based on three metrics: *FD*, J_{prev} and SAD_{8x8}. The algorithm is depicted in Figure 4.

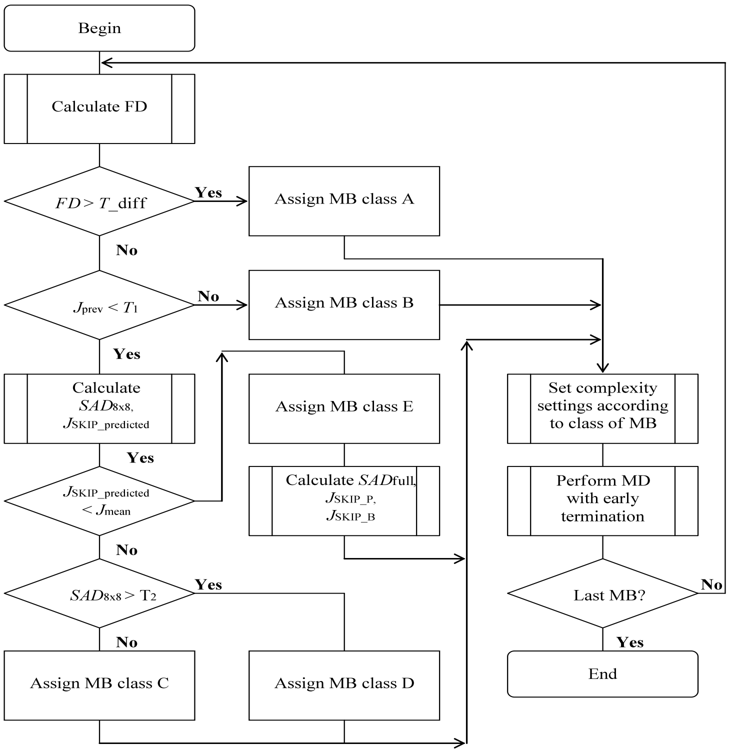


Fig. 4. MB class decision algorithm.

The FD metric is used to identify MBs with high motion activity. MBs for which FD exceeds a threshold are deemed to be active and are allocated to Class A. For inactive macroblocks, the RD cost of the same MB in the previous frame is compared to a threshold. MBs with high J_{prev} are deemed to have been coded with poor efficiency in the previous frame and are allocated to Class B. The RD cost in the case of a SKIP decision is predicted according to:

$$J_{SKIP\_predicted} = \alpha \cdot SAD_{8x8} + C_1 \quad (7)$$

where α and C_1 are constants. If the predicted RD cost of a SKIP decision is less than the mean RD cost in the previous frame, i.e, $J_{SKIP\_predicted} < J_{mean}$, then a SKIP decision is made and the MB is allocated to Class E. Utilization of mean J avoids localizing high and low quality areas (the 'convergence problem') and produces more consistent results.

If a SKIP decision is not made, then the partial SAD result is compared to a threshold. If it exceeds the threshold, the MB is deemed to be static and encoded with low quality in the previous frame and is allocated to Class D. Otherwise the MB is deemed to be static and encoded with high quality in the previous frame and is allocated to Class C.

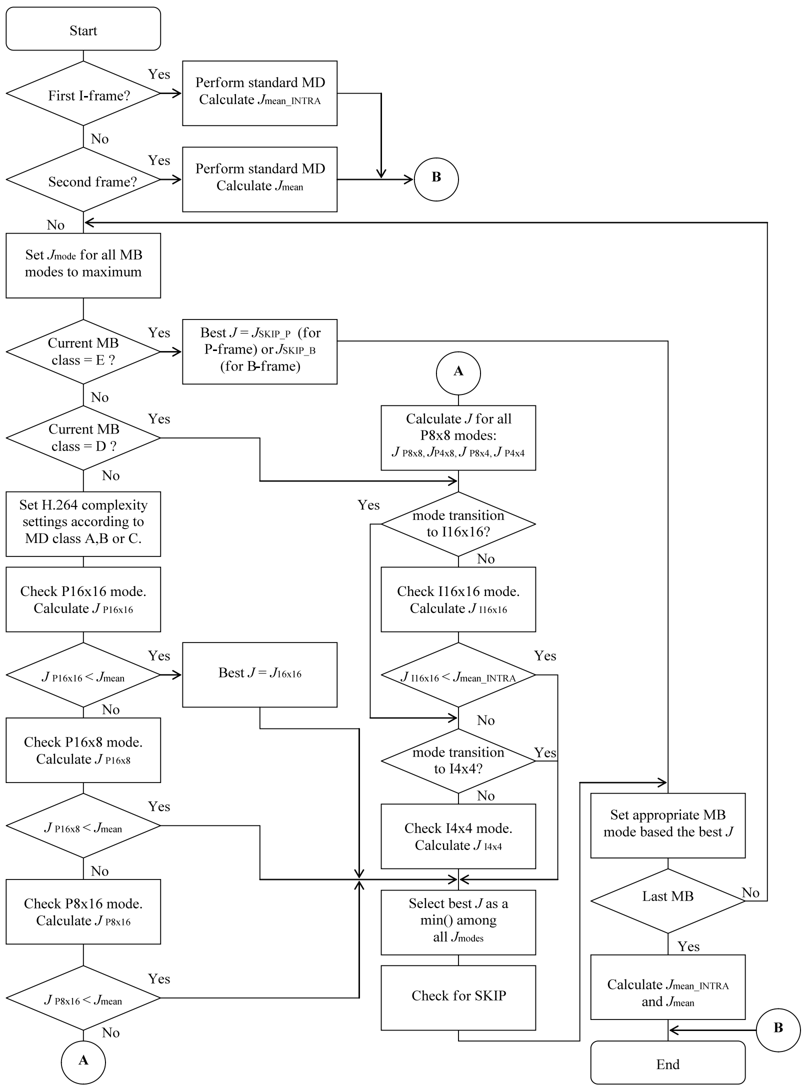


Fig. 5. Fast MD algorithm with ET.

In the case of a SKIP decision (Class E), the value of J must be re-estimated to ensure J_{prev} and J_{mean} are accurate for the purposes of Class Decisions in the next frame. For P- and B- frames separate equations are used:

$$J_{SKIP\_P} = \beta \cdot SAD + C_2 \tag{8}$$

$$J_{SKIP\_B} = \chi \cdot SAD \tag{9}$$

The values of the coefficients in Eqs. (7-9) were derived from correlation experiments using a linear approximation. The thresholds were derived from experiments with test video sequences. The final values for the coefficients and the thresholds are α=10.12; β=2 .099, C_1=–560.06, C_2=–252 .089, χ = 0.57, T_1=2300 and T_2=400.

The Mode Search algorithm uses ET based on RD cost. For P-blocks, ET occurs when $J \geq J_{mean}$ and for I-blocks when $J \geq J_{mean\_INTRA}$, i.e. the mean RD cost of I-blocks in the previous frame. The following mode transitions with low probability are excluded from the search: {Skip, Inter16x16, Inter16x8, Inter8x16, Intra16x16} to {Intra4x4}, and {Inter8x8} to {Intra16x16}. ET operates only after mean J has been calculated for I- and P- macroblocks. Thus, for the studied GOP structure, the first two frames are processed with the conventional MD algorithm.

The complete Fast MD algorithm is shown in Figure 5.

3.3 Results

The performance of the proposed fast MD algorithm was compared with that of the reference JM9.5 encoder [JM9.5]. QCIF and CIF video sequences (300 frames each) with different degrees of motion and spatial complexity were encoded at 30 fps using the test configurations given in Table II.

Table II. Test Configurations Used in Experiments

Encoding parameter	**Config. A**	**Config. B**
GOP structure	IPPP	IBBP
Number of ref. frames	1	5
RDO	Off	on
Entropy coder	CABAC	
QP	28, 32, 36 and 40	

Bit rate change ΔBR, PSNR drop $\Delta PSNR$ and total encoding time change Δt were measured and averaged across QP settings. The results are provided in Tables III and IV. The minus sign indicates an improvement for the new method.

$$\Delta Bits = \frac{Bits_{method} - Bits_{JM}}{Bits_{JM}} \cdot 100\% \qquad (10)$$

$$\Delta PSNR = PSNR_{JM} - PSNR_{method} \qquad (11)$$

$$\Delta t = \frac{t_{method} - t_{JM}}{t_{JM}} \cdot 100\% \qquad (12)$$

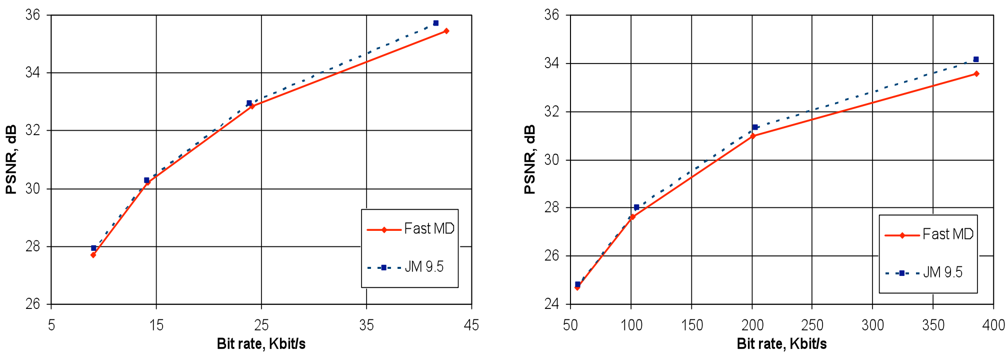
RD curves for various video sequences are shown in Figure 6. The solid line shows the RD curve of the proposed fast MD algorithm, while the dashed line shows the RD curve produced by the reference encoder.

Table III. Experimental Results for Configuration A

Video sequence	**Δ*BR*, %**	**Δ*PSNR*, dB**	**Δ*t*, %**	***W* metric**
Table tennis, QCIF	2.52	0.25	–60.1	5.77
News, QCIF	1.54	0.21	–65.7	4.27
Container, QCIF	2.21	0.22	–71.3	5.07
Akiyo, QCIF	–0.26	0.18	–72.6	2.08
Hall, CIF	–2.06	0.13	–60.2	-0.37
Paris, CIF	0.72	0.12	–58.4	2.28
Mean	*0.98*	*0.18*	*–64.71*	*3.18*

Table IV. Experimental Results for Configuration B

Video sequence	**Δ*BR*, %**	**Δ*PSNR*, dB**	**Δ*t*, %**	***W* metric**
News, QCIF	1.33	0.24	–62.3	4.45
Hall, QCIF	–0.99	0.10	–66.0	0.31
Akiyo, QCIF	–2.11	0.09	–70.0	-0.94
Mobile, CIF	0.07	0.25	–56.8	3.32
Paris, CIF	2.17	0.23	–66.3	5.16
Mean	*0.93*	*0.18*	*–64.28*	*2.46*



(a) *(b)*

Fig. 6. RD curves for video sequences encoded by the algorithm. (a) Table tennis, QCIF and (b) Mobile, CIF both test config. B.

The accuracy of the class decision metrics was assessed for configuration B. Class decisions (except Classes A and B since they include all VBS modes) were compared with the final mode selected for the same MB by the reference encoder. The results are shown in Table V.

Table V. Accuracy of the Class Decision Metrics, %

Video sequence	**Class C**	**Class D**	**Class E**
News, QCIF	96.4	82.7	96.9
Hall, QCIF	95.8	88.3	92.4
Akiyo, QCIF	94.8	97.4	99.1
Mobile, CIF	96.9	57.0	81.2
Paris, CIF	95.1	87.7	92.6
Mean	*95.8*	*85.62*	*92.44*

3.4 Discussion

It can be seen from the results that the proposed algorithm provides significant complexity reduction with minimal coding efficiency loss. The highest W metric is 5.77 for the Tennis video sequence, while W for the original JM encoder from the Pareto curve is about 10. The RD curves of the fast MD algorithm are almost the same as those produced by the reference coder.

For both encoding configurations, total encoding time is reduced by roughly 60–73% depending on the video sequence (except for Mobile at 56%). The highly textured sequence Mobile and the high motion sequence Tennis have greatest $\Delta PSNR$ and least Δt, as was expected. Visual examination of all decoded video sequences revealed no anomalies or blocking artefacts.

On low motion sequences (i.e. Akiyo, Hall) the bit rate is reduced since the algorithm produces a slightly higher SKIP rate than the original coder. The quality degradation of 0.1–0.25 dB is reasonable.

The evaluation of the accuracy of the Class Decisions indicates that the metrics are accurate. Lower accuracy is achieved for the Intra decision (class D), especially for the Mobile sequence. Improvement might be achieved by altering the threshold criterion, possibly by means of an adaptive threshold.

The proposed algorithm at the 35% complexity point provides a W metric similar to standard JM operating at 68% of its full encoding complexity (as defined in Section 3.1). Alternatively, at the complexity point of 38%, standard JM produces W=15.81, in contrast to W=3.18 achieved by the proposed method.

4. DYNAMIC COMPLEXITY CONTROL

As discussed previously, low complexity H.264 encoding algorithms are aimed at reducing total complexity rather than at meeting a fixed frame rate constraint. Frame encoding time can vary substantially, e.g. between a low motion, mainly Class E, frame and a high motion, mainly Class A, frame. In this section, we propose a Dynamic Complexity Control algorithm that adjusts class decisions so that encoding meets a fixed frame encoding time constraint. As will be seen, the classification-based MD algorithm described above lends itself particularly well to solution of the DCC problem.

4.1 Analysis

Figure 7 and Table VI show how MD time varies from frame to frame for a high motion sequence, Carphone, and a low motion sequence, Akiyo, when using the fast MD algorithm described in the previous section.

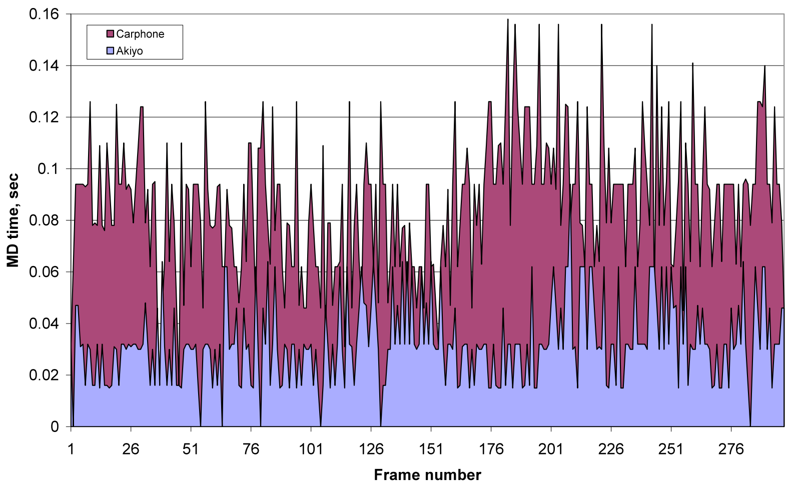


Fig. 7. MD time per frame using Fast MD for Carphone and Akiyo, QCIF.

Table VI: MD time per frame for fast MD with ET

Video sequence	MD time per frame, sec			
	min	Max	mean	$\frac{\text{max}}{\text{min}}$
Carphone, QCIF	0.016	0.158	0.058	9.87
Akiyo, QCIF	0.004	0.094	0.028	23.5

The problem of maintaining a constant frame encoding time for real-time applications can be formulated as: Set the parameters of the fast MD algorithm such that the frame encoding time deadline is met and coding efficiency loss is minimized, i.e.

$$\text{Set } \{P_i\} \text{ such that } T_{actual} < T_{\text{frame\_limit}} \text{ and } W \text{ is minimum} \tag{13}$$

where P_i are the encoding control parameters, T_{actual} is the actual frame encoding time and T_{frame_limit} is the frame encoding time limit which depends on the frame rate.

Three solutions to the complexity control problem were considered – Static Complexity Scaling, Truncated Time Scheduling and Scene Complexity Estimation.

In Static Complexity Scaling, the encoding parameters are set such that the encoding time for the most computationally complex frame is less than the frame limit. This approach guarantees that the real-time constraint is met. However, coding efficiency is sub-optimal since the same control parameters are used for all MBs in the sequence, regardless of scene complexity. When coupled with a fast MD algorithm, such as that described above, the approach leads to unused processing time in the case of easy-to-code frames, e.g. those with low motion activity. Table VII shows the unused processing time for various sequences using Static Complexity Scaling based on the Carphone sequence.

Table VII. Unused Processing Time, % for Several QCIF Video Sequences

Video sequence	**Unused processing time, %**
Carphone, QCIF	24.35
News, QCIF	29.02
Akiyo, QCIF	28.37

In Truncated Time Scheduling, encoding proceeds using a low complexity coding scheme and the control parameters are set so that, on average, the encoding process meets the frame deadline. If the frame deadline is exceeded, coding is truncated by simply SKIPing the remaining MBs. This leads to less unused time but produces regions (bottom-right) with high distortion when coding high motion frames.

The method adopted in this paper, uses complexity control based on Scene Complexity Estimation. In this approach, the frame is pre-processed to obtain an estimate of the frame encoding time based on scene complexity (Frame Complexity Prediction). The individual MB encoding control parameters are then selected such that the estimated frame encoding time meets the deadline and the time budget is distributed between the MBs such that coding efficiency is maximized. A one-frame buffer is used to allow for inaccuracies in encoding time estimation.

Frame Complexity Prediction estimates frame encoding time $T_{predicted}$, based on some measure of scene complexity, S_{frame}:

$$T_{predicted} = f\left(S_{frame}\right) \tag{14}$$

In the previous section, a number of visual complexity metrics were proposed and their performance assessed for Class Decision. In order to reduce computational complexity, we investigated the use of these Class Decisions as scene complexity metrics.

Experiments with various video sequences indicate that for the fast MD algorithm described above, encoding time for a MB can vary by a factor of up to 44 - 55 within a single sequence. The degree of variation in encoding time is reduced significantly when the measurements are classified according to the Class Decision, see Table VIII. Hence, it is proposed to predict the frame encoding time as a weighted sum of the number of MBs in each class:

$$T_{predicted} = \sum_{i=1}^{5} n_i t_i + t_0 \qquad (15)$$

where n_i is the number of MBs in class i, t_i is the mean encoding time for MBs assigned to class i and t_0 is a constant time overhead.

Table VIII. MD Time for Different MB Classes for Fast MD with ET

Video sequence	**MD time per MB, msec**	**MB class**				
		A	***B***	***C***	***D***	***E***
Carphone, QCIF	Min	0.45	0.18	0.27	0.07	0.07
	Max	3.74	3.45	3.87	0.41	0.41
	max/min	*8.31*	*19.16*	*14.33*	*5.61*	*5.61*
Akiyo, QCIF	Min	1.03	0.23	0.23	0.10	0.07
	Max	3.00	3.03	3.14	0.36	0.98
	max/min	*2.89*	*13.12*	*13.62*	*3.49*	*13.44*

Several strategies for adaptive calculation of t_i over the previous N frames were investigated:

1. Mean encoding time for all MBs in class i
2. Mean encoding time for all MBs in class i which were not involved in promotion or demotion (see later)
3. Mean encoding time for all MBs in class i which were not Early Terminated

The proposed models were assessed across a range of QCIF video sequences (Carphone, Hall and Akiyo) under configuration A. In the experiments, values of N=1, 5 and 10 were tested. The best results were achieved for N=5, as shown in Table IX where $T_{predicted}$ is the predicted frame encoding time and T_{actual} is the measured frame encoding time.

Table IX. Experimental Results of Adaptive Frame Prediction Model Estimation

Video sequence	Method	Prediction error, %			Pearson correlation r between $T_{predicted}$ and T_{actual}
		min	*max*	*average*	
Carphone	1	0.02	33.04	6.95	0.751
	2	0.13	33.75	7.73	0.771
	3	8.21	85.61	23.35	0.496
Hall	1	0.02	46.70	3.95	0.949
	2	0.21	43.90	5.46	0.967
	3	4.18	88.62	22.94	0.625
Akiyo	1	0.01	28.8	2.31	0.969
	2	0.02	29.13	3.42	0.983
	3	2.67	66.93	14.41	0.768

From the experimental results it can be clearly seen that strategy 3 is very inaccurate. The other two methods provide accurate prediction with an average difference between predicted and actual times of about 7%. Since strategy 2 has a higher Pearson correlation coefficient, it was chosen for implementation in the main DCC algorithm.

We propose to apply complexity control by adjusting the Class Decisions made by the Fast MD algorithm. Computational complexity is reduced by demoting MBs to lower class, e.g. from Class A to B. Alternatively, computation complexity is increased by promoting MBs to higher class, e.g. from Class E to D. This approach maintains a single framework for low complexity coding, encoding time prediction and for complexity control. In addition, the classes are known to be on the Pareto curve and so are optimal in terms of coding efficiency. Thus, on average, promotion improves RD performance while, on average, demotion reduces RD performance by the least amount possible given the required reduction in computational complexity.

4.2 Dynamic Complexity Control Algorithm

The overall DCC algorithm proceeds as follows:

1. The frame is processed, making Class Decisions for all MBs according to the previous Fast MD algorithm.
2. The frame encoding time is estimated based on the Class Decisions.
3. If the estimated frame encoding time exceeds the time quota for the frame, demotions are performed until the budget is met. Alternatively, if the estimated frame encoding time is less than the time quota for the frame, promotions are performed until the budget is met.
4. Fast MD based coding is performed according to the revised Class Decisions.

As can be seen in Figure 1, the gradient of the CHIM of the Pareto curve is less steep for more computationally complex classes. There is less coding efficiency loss in demoting higher complexity classes. Thus demotions proceed right-to-left and promotions proceed left-to-right as per the Pareto Curve. In some cases, it may not be necessary to demote all MBs within a given class. Since demotion probably means a bit rate and distortion increase, demotion should start with MBs that have lowest J_{prev} within the class. In this way, the most effectively coded MBs are demoted first, minimizing coding efficiency loss. Similarly, promotion with a class should start with the MBs that have highest J.

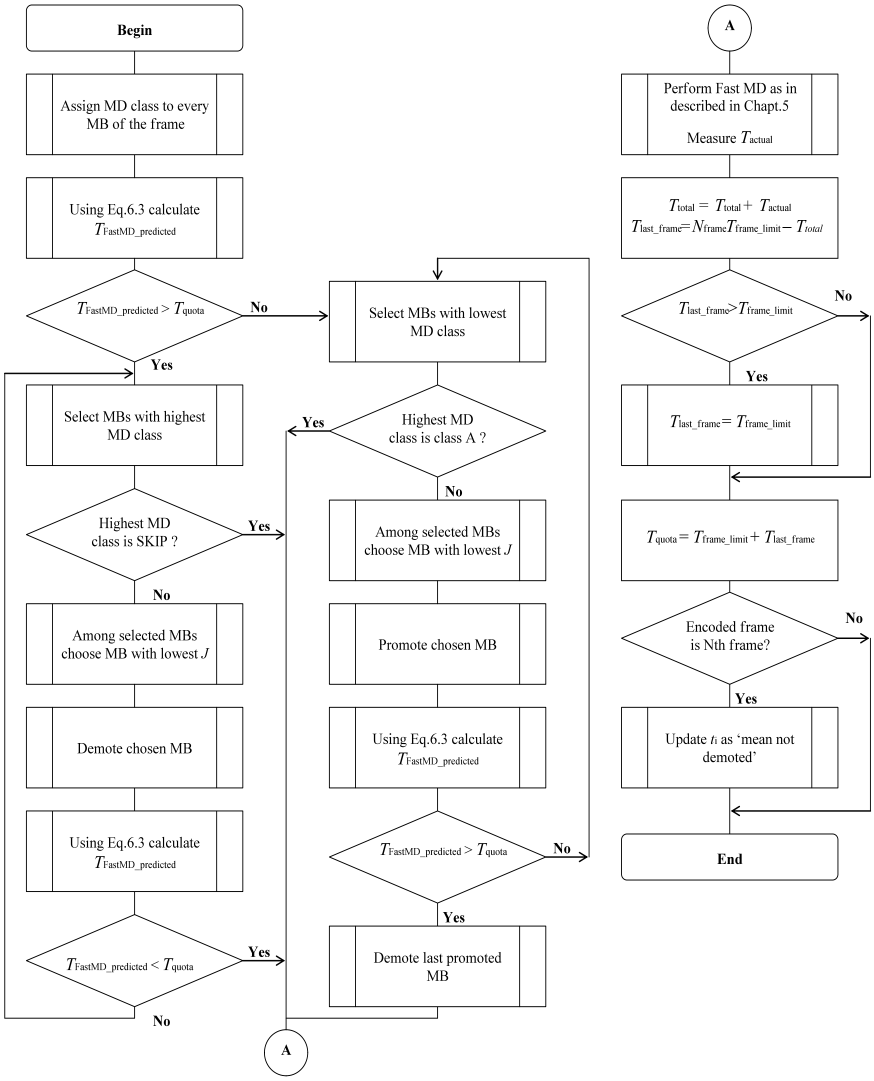


Fig. 8. The Dynamic Complexity Control algorithm.

A single frame buffer allows for inaccuracies in the frame encoding time estimation. The encoding time quota for each frame is calculated as:

$$T_{quota} = T_{\mathrm{frame\_limit}} + \min\left(n \cdot T_{\mathrm{frame\_limit}} - T_{total}, T_{\mathrm{frame\_lmit}}\right) \qquad (16)$$

where n is the number of frames processed so far and T_{total} is the total encoding time so far. In order to synchronize the buffer, the last frame encoding time cannot exceed the frame encoding time limit.

The final algorithm for Dynamic Complexity Control algorithm is shown in Figure 8.

4.3 Results

A real-time H.264 system combining fast MD and DCC and a system with Static Complexity Scaling (SCS) were implemented using JM and experimentally tested. Several QCIF video sequences of 300 frames each were encoded with an "IPPP" GOP structure. Reference encoding used all seven VBS, search range of 8, CABAC entropy coder and RDO off. In the experiments QP was set to 28. The algorithm was tested under conditions, where the value of T_{frame_limit} was set to allow real-time encoding at 15, 20 and 30fps on a reference 3GHz Pentium IV PC with 1GB RAM. No assembly level optimizations were applied. Encoding tools used for the Static Complexity Scaling implementation are given in Table X.

The bit rate increase, quality degradation and W metric obtained are compared to the results of non-real time full search JM (i.e. C=100% point on the Pareto curve) in Tables XI-XIII.

In order to further assess the performance of the algorithm, the mean prediction error T_{error} and mean adjusted time T_{adjust} were calculated as percentages and are included in Table XIV.

$$T_{error},\% = \frac{100}{T_{seq}} \sum_{i=1}^{i=N_{seq}} \left| T_{predicted\_posterior}(i) - T_{actual}(i) \right| \qquad (17)$$

$$T_{adjust},\% = \frac{100}{T_{seq}} \sum_{i=1}^{i=N_{seq}} T_{predicted\_prior}(i) - T_{actual}(i) \qquad (18)$$

where $T_{predicted\_posterior}(i)$, $T_{predicted\_prior}(i)$ and $T_{actual}(i)$ are the predicted encoding time after promotion/demotion, the predicted encoding time before promotion/demotion and the actual encoding time for frame i, respectively. N_{seq} is the number of frames in the sequence and T_{seq} is the total encoding time for the sequence. T_{error} is a measure of the mean accuracy of the Frame Complexity Prediction method, lower is better. T_{adjust} is a measure of the mean encoding time adjustment made by the promotion/demotion method.

Positive T_{adjust} indicates overall time savings, meaning more demotions than promotions. Negative values indicate available time, meaning more promotions than demotions.

Table X. H.264 Settings for SCS

H.264 settings	**SCS**		
	15 fps (C=64%)	20 fps (C=47%)	30 fps (C= 28%)
VBS modes	All VBS modes	P16x16, P8x8, all Intra modes	P16x16, all Intra modes
Search range	6	4	1
Hadamard transform	on	on	off

Table XI. ΔBit Rate, % for DCC and SCS Approaches

Video sequence	**DCC**			**SCS**		
	15fps	20fps	30fps	15fps	20fps	30fps
Carphone	0.05	1.04	15.25	–0.30	3.68	14.25
Table tennis	2.48	3.85	11.16	2.24	4.48	23.52
Coastguard	0.26	0.87	6.48	2.46	3.48	15.71
News	0.84	0.76	5.90	4.07	5.59	20.89
Salesman	–0.5	-1.50	5.60	3.84	5.67	22.55
Grandmother	–1.23	-3.08	-1.82	–1.09	0.95	12.96
Mother&Daughter	–0.7	-1.00	1.68	–2.07	–0.13	11.69
Hall	0.03	-1.73	-4.41	5.72	6.54	17.98
Akiyo	–1.15	-1.80	-1.85	0.76	2.96	18.9
Mean	*0.008*	*-0.287*	*4.22*	*1.73*	*3.69*	*17.6*

Table XII: ΔPSNR, dB for DCC and SCS Approaches

Video sequence	**DCC**			**SCS**		
	15fps	20fps	30fps	15fps	20fps	30fps
Carphone	0.11	0.28	0.49	0.20	0.22	0.53
Table tennis	0.24	0.24	0.46	0.10	0.12	0.37
Coastguard	0.06	0.08	0.13	0.06	0.08	0.20
News	0.09	0.12	0.38	0.18	0.21	0.46
Salesman	0.03	0.07	0.24	0.13	0.15	0.30
Grandmother	0.02	0.19	0.18	0.08	0.10	0.26
Mother&Daughter	0.06	0.17	0.32	0.17	0.20	0.50
Hall	0.01	0.03	0.20	0.12	0.20	0.30
Akiyo	0.01	0.02	0.16	0.15	0.16	0.35
mean	*0.07*	*0.13*	*0.28*	*0.13*	*0.16*	*0.36*

Table XIII: Coding Efficiency Loss W for DCC and SCS (also, percentage difference in coding efficiency loss of DCC relative to SCS at the same frame rate negative values indicate improvement using DCC)

Video sequence	**DCC**			**SCS**		
	15fps	20fps	30fps	15fps	20fps	30fps
Carphone	1.48	4.78	21.61	2.9	6.54	21.14
Table tennis	5.6	6.97	17.14	3.54	6.04	28.33
Coastguard	1.03	1.91	8.17	3.24	4.52	18.31
News	2.01	2.32	10.85	6.41	8.32	26.87
Salesman	–0.11	–0.59	8.75	5.53	7.62	26.45
Grandmother	–0.97	–0.61	0.52	-0.05	2.25	16.34
Mother&Daughter	0.08	1.20	5.84	0.14	2.47	18.19
Hall	0.15	-1.34	-1.81	7.28	9.14	21.88
Akiyo	–1.16	–1.54	0.23	2.71	5.04	23.45
mean	*1.72*	*2.64*	*7.92*	*3.52*	*5.77*	*22.32*
difference (%)	-51%	-54%	-65%	-	-	-

Table XIV: Mean Prediction Error and Mean Adjusted Time, 30 fps DCC

Video sequence	T_{error}, %	T_{adjust}, %
Carphone	3.39	74.95
Table tennis	5.55	50.59
Coastguard	3.88	98.27
News	5.88	21.11
Salesman	6.28	18.04
Grandmother	2.56	13.74
Mother&Daughter	3.13	17.00
Hall	6.40	5.75
Akiyo	3.64	-2.92

5.4 Discussion

Firstly, we should consider that full, conventional JM encoding (C=100%, as defined in Section 3.1.1) can only achieve 9 fps in real-time on the reference PC.

As described previously, the computational complexity of the conventional encoder can be decreased by statically reducing the search size using SCS. For a frame rate of 15 fps, the encoding parameters much be scaled such that C=64%, see Table X. To achieve 20 fps, C must be 47% and for 30 fps, C must be 25%. As can be seen from Tables XI and XII (SCS), 15 fps and 20 fps are achieved with relatively small increases in bit rate and distortion. However, the increases become large in the case of 30 fps. There is a mean increase in bit rate of 17.6% and a mean increase in distortion of 0.35 dB, both relative to full, conventional JM encoding. For Table Tennis, the bit rate increase is particularly bad, 23.5%. For News, distortion rises by a very noticeable 0.46 dB. Overall, Table Tennis has worst coding efficiency loss of W=28.3 (Table XIII).

The proposed DCC algorithm allows complexity to be scaled dynamically for each individual MB such that coding efficiency is maintained while meeting the real-time constraint. At 15 fps, DCC shows almost no loss in coding efficiency, on average, relative to the full, conventional encoder, W=1.72, see Tables XI-XIII (DCC). This is roughly half of the average coding efficiency loss in the SCS case for 15 fps. As the frame rate increases to 20 fps, the high motion sequences cannot be encoded in the available time without large numbers of MB demotions. This causes the coding efficiency to deteriorate for Carphone, Table Tennis and Coastguard, see Table XIII (DCC). Nevertheless the average coding loss across all sequences, W=2.54, is less than half that of the SCS method, W=5.77. At 30 fps, DCC demands a large number of MB demotions for all sequences, except Akiyo. This can be seen in Table XIV, where T_{adjust} is positive for all except Akiyo. Akiyo has such low motion that it still has more MB promotions than demotions (negative T_{adjust}). This increase in demotions, causes average bit rate and distortion to increase further, see Tables XI and XII (DCC). As would be expected, the increase is greater for the higher motion sequences. Since Akiyo and Hall have so few MB demotions, their bit rate and distortion are almost unchanged relative to full, conventional JM encoding, even though the system is operating at more than 3 times the frame rate, Tables XI and XII. Overall, at 30 fps, the mean coding efficiency loss of DCC, relative to conventional JM encoding and averaged across all sequence, is almost a third of that for SCS, i.e. W=7.9 as opposed to W=22.3, a 65% improvement.

Table XIV shows that the mean error in the MB encoding time prediction is less than 7% in all cases at 30 fps. There were no cases of buffer overrun in any of the tests.

Visual examination of all decoded video sequences revealed no anomalies or blocking artefacts. Frame rate can be increased to 40 fps using DCC. However, RD loss is large since almost all MBs are demoted to Class E, instant SKIP, due to the lack of encoding time. SCS cannot achieve this frame rate without modification of the encoder.

6. CONCLUSION

The problem of real-time H.264 video encoding in software with high coding efficiency was investigated. A fast Mode Decision encoding method based on Pareto optimal MB classification was proposed for reducing the total complexity of H.264 video encoding. A Class Decision algorithm was used together with a Rate-Distortion cost metric for ET and Forward SKIP Prediction. A novel Dynamic Complexity Control algorithm was proposed. Three methods for frame encoding time prediction were proposed and assessed. An MB Class promotion/demotion scheme was proposed for RDC optimal complexity adjustment.

The complete real-time encoding algorithm incorporating the fast MD and DCC methods was implemented and experimentally assessed. The performance of the algorithm was compared with that of full, conventional H.264 encoding and that of Static Complexity Scaling, an alternative method for reducing encoder complexity. At 20 fps, the average coding efficiency of the proposed algorithm was found to be similar to that of full, conventional encoding operating at 9 fps (-0.3% bit rate increase and 0.13 dB PSNR loss). The proposed algorithm was found to provide lower average bit rate and distortion than SCS at all frame rates studied (15, 20 and 30 fps). As frame rate was increased from 15 to 30 fps, the coding efficiency of the proposed algorithm deteriorated less quickly than that of SCS. At 30 fps, the coding efficiency loss of the proposed algorithm was 65% less than that of SCS.

Future work includes improving the *J* prediction model for skipped MBs, and improving the class demotion/promotion scheme by adding criteria that would prohibit the algorithm from generating too many SKIP and Intra decisions when the processing capabilities of the CPU are limited.

ACKNOWLEDGMENTS

This work was sponsored by University College Dublin.

REFERENCES

AHMAD, A., KHAN, N., MASUD, S., AND MAUD, A. 2004. Selection of Variable Block Sizes in H.264. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP'04*), 3, 173–176.

AKYOL, E., MUKHERJEE, D. AND LIU, Y. 2007. Complexity control for real-time video coding. In *Proceedings of IEEE International Conference on Image Processing* (*ICIP'07*), 1, 77-80.

ATES, H.F., AND ALTUNBASAK, Y. 2008. Rate-Distortion and complexity optimized motion estimation for H.264 video coding, *IEEE Transactions on Circuits and Systems for Video Technology*, 18, 2, 159-171.

BERGER, G., GOEDEKEN, R., RICHARDSON, J. 2007. Motivation and implementation of a software H.264 real-time CIF encoder for mobile TV broadcast applications, *IEEE Transactions on Broadcasting* 53, 2, 584-587.

BJONTEGAARD, G. 2001. Calculation of average PSNR differences between RD curves. *ITU-T VCEG Meeting*, document VCEG-M33, Austin, TX.

CHANG, A., WONG, P. H. W., YEUNG, Y. M. AND AU, O. C. 2004. Fast multi-block selection for H.264 video coding. In *Proceedings of IEEE International Symposium on Circuits and Systems* (*ISCAS'04*), 3, 817–820.

DEEPAK, S., AND CHEN, T. 2001. Classification based mode decisions for video over networks. *IEEE Transactions on Multimedia* 3, 41-52.

EVERETT, H. 1963. Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. *Operations Research* 11, 399-417.

FENG, B., ZHU, G., AND LIU, W. 2006. Fast adaptive Inter Mode Decision method for P slices in H.264. In *Proceedings of IEEE 3rd International Conference on Consumer Communications and Networking* (*CCNC'06*), 2, 745-748.
GOTO, K., HATABU, A., NISHIZUKA, H., MATSUNAGA, K., NAKAMURA, R., MOCHIZUKI, Y., AND MIYAZAKI, T. 2006. H.264 Video encoder implementation on a low-power DSP with low and stable computational complexity. In *Proceedings IEEE Workshop on Signal Processing Systems Design and Implementation* (*SIPS '06*), 101–106.
HAN, K., AND LEE, Y. 2004. Fast macroblock mode decision in H.264. In *Proceedings of IEEE Region 10 Conference* (*TENCON'04*), 347-350.
HONG, Z., CHENG-KE, W., YANG-LI, W., AND YONG, F. 2005. Fast Mode Decision for H.264/AVC based on Macroblock Correlation. In *Proceedings of IEEE 19th International Conference on Advanced Information Networking and Applications* (*AINA'05*), 775-780.
HSU, K. W., XIANG, L., AND CHOPRA, R. 2005. An IC design for real-time motion estimation for H.264 digital video. In *Proceedings 48th Symposium on Circuits and Systems,* 1489–1493.
HU, Y., LI, Q., MA, S. AND KUO, C.C.J. 2006. Joint Rate-Distortion-Complexity optimization for H.264 motion search, In *Proceedings of IEEE International Conference on Multimedia and Expo* (*ICME 2006*), 1949-1952.
ISO4964. 2002. Implementation studies group of ISO/IEC. Main Results of the AVC Complexity Analysis. *ISO/IEC JTC1/SC29/WG11*, Klagenfurt.
JM9.5. JVT reference software JM 9.5, available on-line at: http://iphome.hhi.de/suehring/tml.
JVT. 2003. Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG. *Draft ITU-T recommendation and final draft international standard of joint video specification* (*ITU-T Rec. H264|ISO/IEC 14496-10 AVC*), document JVT-G050d35, 7th Meeting, Pattaya, Thailand.
KAMINSKY, E. AND GROIS, D. AND HADAR, O. 2008. Dynamic computational complexity and bit allocation for optimizing H.264/AVC video compression, *Journal of Visual Communication and Image Representation*, 19, 1, 56-74.
KANNANGARA, C. S., RICHARDSON, I. E.G., BYSTROM, M., SOLERA, J. R., AND ZHAO, Y., MACLENNAN, A., AND COONEY, R. 2006. Low-complexity skip prediction for H.264 through Lagrangian cost estimation. *IEEE Transactions Circuits Systems Video Tech.,* 16, 2, 202-208.
KANT, S., MITHUN, U. AND GUPTA, P.S.S.B.K. 2006. Real-time H.264 video encoder implementation on a programmable DSP processor for videophone applications. In *The Digest of Technical Papers of the International Conference on Consumer Electronics* (*ICCE'06*), pp. 93-94.
KIM, C., AND JAY KUO, C.C. 2005. A feature-based approach to fast H.264 Intra/Inter Mode Decision. In *Proceedings IEEE International Symposium Circuits and Systems* (*ISCAS'05*), 1, 308-311.
KIM, Y., CHOE, Y., AND CHOI, Y. 2006. Fast Mode Decision algorithm for H.264 Using AZCB prediction. In *Proceedings International Conference on Consumer Electronics* (*ICCE'06*), 33-34.
KOSSENTINI, F., LEE, Y.W., SMITH, M.J.T, AND WARD, R.K. 1997. Predictive RD optimised Motion Estimation for very low bit-rate video coding. *IEEE Journal on Selected Areas in Communications* 15, 9, 1752–1763.
KUO, T.Y., AND CHAN, C.H. 2004. Fast Macroblock partition prediction for H.264/AVC. In *Proceedings IEEE International Conference on Multimedia and Expo* (*ICME'04*), 675–678.
LI, G. L., CHEN, M. J., LI, H. J., AND HSU, C. T. 2005. Efficient motion search and mode prediction algorithms for Motion Estimation in H.264/AVC. In *Proceedings IEEE International Symposium on Circuits and Systems* (*ISCAS'05*), 5481 – 5484.
SAPONARA, S., BLANCH, C., DENOLF, K., AND BORMANS, J. 2002. Data transfer and storage complexity analysis of the AVC Codec on a tool-by-tool basis. *ISO/IEC JCT1/SC29/WG11*, M8547, Klagenfurt.
SHEN, J.S., CHEN, C.H., TSAI, C.M., AND LIN, C.W. 2007. Computation-Aware Intra-mode Decision for H. 264 Coding and Transcoding, In *Proceedings of Ninth IEEE International Symposium on Multimedia Workshops* (*ISMW'07*). 427-433.
STOTTRUP-ANDERSEN, J., FORCHHAMMER, S. AND AGHITO, S.M. 2004. Rate-distortion-complexity optimization of fast motion estimation in H.264/MPEG-4 AVC, In *Proceedings of International Conference on Image Processing* (*ICIP '04*), 1, 111-114.
WANG, C.N., YANG, S.W., LIU, C.M., AND CHIANG, T. 2004. A hierarchical N-Queen decimation lattice and hardware architecture for Motion Estimation. *IEEE Transactions on Circuits and Systems for Video Technology* 14, 4, 429–440.
WANG, H., AND ZHU, Z. 2005. Fast Mode Decision and reduction of reference frames for H.264 Encoder. In *Proceedings of International Conference on Control and Automation* (*ICCA'05*), 1040-1043.
WE, Z., AND CANHUI, C. 2006. Realization and optimisation of DSP based H.264 encoder. In *Proceedings of IEEE International Symposium on Circuits and Systems* (*ISCAS'06*), 4.
WIEGAND, T., SULLIVAN, J. S., BJONTEGAARD, G., AND LUTHRA, A. 2003. Overview of the H.264/AVC video coding Standard, *IEEE Transactions on Circuits and Technology* 13, 7, 560-576.
WU, M., FORCHHAMMER, S., AND AGHITO, S.M. 2007. Complexity control of fast motion estimation in H.264/MPEG-4 AVC with rate-distortion-complexity optimization, *Proceedings of SPIE*, 6508, 650824.
XIPH,ORG. Test media, available on-line at: http://media.xiph.org/video/derf/.

YIN, P., TOURAPIS, H.Y.C., TOURAPIS, A.M., AND BOYCE, J. Fast Mode Decision and Motion Estimation for JVT/H.264. In *Proceedings of IEEE International Conference on Image Processing* (*ICIP'03*), 853–856.

YU, A.C. 2004. Efficient block-size selection algorithm for Inter-frame coding in H.264/MPEG-4 AVC. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP'04*), 169-172.

ZHAO, Y., AND RICHARDSON, I. 2003. Macroblock classification for complexity management of video encoders. *Signal Processing: Image Communication* 18, 801-811.