Real-Time Software Based MPEG-4 Video Encoder

Weiguo Zheng, Ishfaq Ahmad and Ming L Liou

Multimedia Technology Research Center[1]
HKUST, Clear Water Bay, Hong Kong

ABSTRACT

Rapid improvements in general-purpose processors are making software-based video encoding solutions increasingly feasible. Software encoders for H.263 and MPEG-2 video standards are well documented, recently reporting close to real-time performance. However, MPEG-4 video, due to its higher complexity, requires more computational power, making its real-time encoding speed rather infeasible. Design of a fully standard-compliant MPEG-4 encoder with real-time speed on a PC entails optimizations at all levels. This includes designing efficient encoding algorithms, software implementation with efficient data structures, and enhancing computation speed by all possible methods such as taking advantage of the machine architecture. In this paper we report a software-based real-time MPEG-4 video encoder on a single-processor PC, without any frame skipping, profile simplifying tricks, or quality loss compromise. The encoder is a quintessence of a number of novel algorithms. Specifically, we have designed an algorithm for fast motion estimation algorithm. We have also designed an algorithm for the detection of all-zero quantized blocks, which reduces the complexity of DCT and quatization. To enhance the computation speed, we harness Intel's MMX technology to implement these algorithms in an SIMD (Single Instruction Stream, Multiple Data Stream) fashion within the same processor. On the 800 MHz Pentium III, our encoder yields up to 70 frames per second for CIF resolution video, with the similar picture quality as the reference VM software.

1. Introduction

MPEG-4 specifies coding, multiplexing and representation of synthetic, natural or hybrid information in object-based manner. It provides solutions in the form of tools and algorithms enabling functionalities such as content-based interactivities, efficient compression, object scalability, spatial and temporal scalability, and error resilience, etc. Due to improved coding efficiency of MPEG-4, a number of new applications using digital video, such as video conferencing, Internet video games or digital TV, are emerging on common PC as well as hand-held devices. For such application, efficient software encoding and decoding is highly essential.

Compared to its predecessor standards, MPEG-4 has additional coding modules that exacerbate its coding complexity. MPEG-4 supports arbitrary shape object based encoding described by alpha map. Multilevel alpha maps are frequently used to form different layers of image sequences for the final film. Coding of texture for arbitrarily shaped regions is required for achieving an efficient texture representation for arbitrarily shaped objects. In addition, MPEG-4 also provides multifunctional coding tools and algorithms that support a number of content based as well as other functionalities. MPEG-4 video group has developed Video Verification Models (VMs), which evolve through time by means of core experiments. New algorithms/tools are added to the VM and old algorithms/tools are replaced in the VM by successful core experiments. VM has gradually evolved more than ten versions [1].

The speed of the latest VM is very slow and far from being applicable in practical applications. Based on our testing, the encoding speed for CIF format video (rectangular frame) is usually less than 2 frames per second on Pentium III 800 MHz processor and Windows 2000 (CIF size rectangular frame simple head & shoulder sequences).

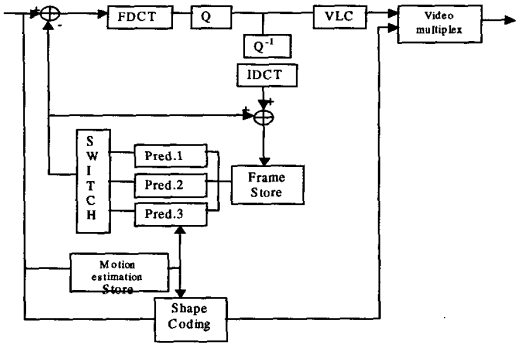

Fig.1: Block Diagram of MPEG-4 Video Encoder

[1] This work was supported by Research Grants Council of Hong Hong under contract # CRC98/01.EG05 and HKUST6228/99E.

The basic coding structure includes shape coding, which is used for arbitrarily shaped video objects (VOs), motion compensation as well as DCT-based texture coding. The block diagram is shown in Fig.1.

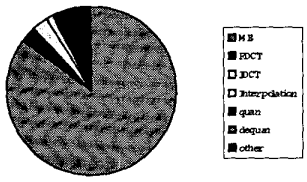

Fig. 2: Complexity comparison of components of MPEG-4 video encoder

Fig. 2 shows the complexity distribution among various modules of MPEG-4 video encoding using CIF size sequences. As can be observed, motion estimation occupies the largest portion of the complexity pie, and FDCT, IDCT hold the second position, followed by quantization and dequantization.

In this paper we report a software-based real-time MPEG-4 video encoder on a single-processor PC, without any frame skipping, profile simplifying tricks, or quality loss compromise. The main factors contributing to real-time video compression include:

1) Fast motion estimation;
2) Prediction of all-zero quantized block;
3) Fast transformation and quantization;
4) Single Instruction Multiple Data (SIMD) implementation.

2. Adaptive Motion Search With Elastic Diamond (AMSED)

Motion estimation (ME) is the most important part of the MPEG-4 encoder, since it significantly affects the output quality of the encoded sequence. This is also the most complex part with an overwhelming computational complexity compared with the other parts of the encoding process. In our testing, while using searching window size of ±16, motion estimation (full search) can swallow more than 90% of processing resource. A myriad of algorithms exist in the literature that aim to improve the speed and performance of motion estimation, such as three-step search, new three-step search, 2-D logarithmic search, conjugate directional search and hierarchical search [2], [3], [4], etc. The MPEG-4 Part-7 has adopted MVFAST (motion vector field adaptive fast search technique) as the core technology for fast motion estimation [5]. MVFAST can achieve substantial speedup comparing with full search; however, it doesn't mean the end of development of fast motion estimation algorithm. We propose a new algorithm (Adaptive Motion Search with Elastic Diamond, AMSED) to get more significant speedup. The major features of AMSED are:

- Adaptive threshold for stationary block.
- Definition of extended motion vector candidate list.
- Detection of Local Motion Activity.
- Motion search with elastic diamond.
- Adaptive threshold for half-stop.
- Keeping track of checking point history.
- Detection of MB belonging to same moving object.
- Adoption motion inertia in temporal domain.
- Use interpolated motion vector in motion candidate list for BVOP.

It is observed that the SAD of most of the stationary blocks at (0,0) is less than 512. Detection of such stationary blocks would eliminate the need to perform further motion estimation. However, due to the influence of noise, the SAD of stationary MB can be higher than 512. Instead of using static threshold, in AMSED we use set adaptive threshold TH0, enabling faster and more accurate detection of stationary block. The detail of TH0 setting is as following: if the SAD at (0,0) is less than 512, the MB will be considered as stationary MB. Otherwise, if all of its neighbour MBs (see Fig.2) are stationary MBs, the TH0 will be

$$TH0 = \max[\min(SADi), BLOCK\_SIZE*3]. \quad (1)$$

If the MBs are classed as active MBs, a motion vector candidate list (MVCL) is defined. It includes motion vectors from its neighbour MBs. Based on the observation of smoothness of motion field, local motion activity (LMA) is measured as:

$$l_i \quad |x_i - \bar{x}| \quad |y_i - \bar{y}|$$

$$\begin{cases} & L \le L1 \\ & \le \end{cases}$$

$(\bar{x}, \bar{y})$, which is the median vector in MVCL.

As we know, MVs are correlated in temporal domain too. We also use the so-called motion inertia property in AMSED

for a better MV prediction in temporal domain. This is done by testing the MVs inside the search range in the reference frame, and by selecting the closest predictor (MB position + MV) to the current MB position as the 4th MV candidate in MVCL.

There are two search patterns in AMSED: Large Diamond Search Pattern (LDSP) and Small Diamond Search Pattern (SDSP). LDSP is used to find raw motion vector quickly, and SDSP is used to refine motion vector prediction. In the elastic mode, the search switches between the LDSP and SDSP. If SDSP is executed for a certain times, the search pattern will change to LDSP. If LDSP finds that its center has the minimum SAD in the current round, the search pattern changes to SDSP with the same search center (in order to refine the search), and so on.

In order to gain further speedup, AMSED keeps track of the checking points that have been accessed, in order to avoid unnecessary operations. An adaptive half-stop is applied to restrict the motion search once a good enough motion vector is found.

If the LMA is low, we start the search using the SDSP; otherwise, we check the predictors in MVCL and select the one with minimal SAD as the new search center, and search with LDSP. The search will be stop when the search center is found to be minimal in SDSP or the SAD is less than half-stop threshold.

For a BVOP, the motion vectors can be interpolated from its reference VOPs. The interpolated motion vectors are used as prediction MV for the BVOP. We found that the interpolated motion vectors usually provide a good prediction of motion vector for BVOP, and the motion search can be terminated quickly.

The experiments are done using the VM software as reference encoder, with TM5 rate control, search windows size ±16, and at 384kbps bit rate. All testing sequences are rectangular video with CIF format. There are two BVOPs between IVOP/PVOPs, and 15 VOPs in a GOV.

As indicated in Table 1, AMSED outperforms MVFAST with faster speed while yielding the similar picture quality. The average speedup of AMSED to MVFAST is about 200%. Compared with full search, AMSED is faster from 500 times to 2800 times.

3. All-zero Quantized Block Prediction

In MC/DCT framework, forward DCT helps in removing the spatial redundancy by concentrating most relevant information to the lower most coefficients in DCT domain. Quantization is basically a process for reducing the precision of the DCT coefficients. The lower precision always implies a low bit rate in compressed data stream and higher distortion. The quantization process involves division of integer DCT coefficient value by integer quantization scales. The result is rounded to the closest integer with certain rules. The quantization scale is chosen so as to minimize the perceived distortion in the reconstructed picture, using the principles based on the human visual system. In practice, most quantized DCT coefficients become zero. In particular, at low bit rate, many of these blocks have all of their coefficients as zero after quantization of the motion compensated DCT blocks. If we can predict a block that will become all-zero after quantization (which is called all-zero block in this paper) we can skip the DCT, quantization, dequantization, and inverse DCT for this block. This can result in significant reduction in computational complexity.

Using a different approach than [6] and [7], we combine properties of DCT transformation and quantization of H.263 mode and MPEG mode, and use Sum of absolute value (SAV) for 8×8 block that is to be transformed and quantized.

$$\sum' \sum' |p(i \quad |$$

SAV in all zero block prediction can be implemented by SIMD efficiently.

There exist several fast DCT/IDCT algorithms that can be used for SIMD implementation [9]. The AAN algorithm is suitable for SIMD implementation of DCT/IDCT because of its ability to decompose data to regular structures [10]. For further optimization, the transpose and prescaling can be implemented in scan/de-scan and quantization/de-quantization routines.

5. Conclusions

The experiment conditions have been described in Section 2. The experiment results are shown in Table 2. As we can see from the table, the speed is improved significantly, and the quality is as same as the reference encoder.

In addition to the techniques mentioned above, we also improved 4 -MV motion estimation, half-pixel motion estimation, SIMD motion compensation, best-bounding box finding, and upsampling, etc. Based on the proposed techniques in this paper, our software-only MPEG-4 encoder encodes CIF resolution video at faster than real-time speed without loss of quality and compression efficiency. Our future research efforts focus on the improvement of rate control algorithm and arbitrary shape object encoding.

REFERENCES

[1] ISO/IEC JTC1/SC29/WG11 N3093, "MPEG-4 Video Verification Model Version 15.0."

[2] R.Li, B.Zeng, and M.L.Liou, "A New Three-Step Search Algorithm for Fast Motion Estimation," *IEEE Transactions on Circuits & Systems for Video Technology*, Vol.4, pp. 438-442, Aug. 1994.

[3] F.Dufaux and F.Moscheni, "Motion Estimation Techniques for Digital TV: A Review and a New Contribution," *Proceeding of IEEE*, vol. 83, pp. 858-876, June 1995.

[4] Jo Yew Tham, Surendra Ranganath, Maitreya Ranganath, and Ashraf Ali Kassim, "A Novel Unrestricted Center-Biased Diamond Search Algorithm for Block Motion Estimation," *IEEE Transations on Circuits & Systems for Video Technology*, Vol.8, No.4, pp. 369-377, August, 1998.

[5] ISO/IEC JTC1/SC29/WG11 N3324, "Optimization Model Version 1.0."

[6] I_Ming Pao, and Ming-Ting Sun, "Modeling DCT Coefficients for Fast Video Encoding," *IEEE Transactions on Circuits & Systems for Video Technology*, Vol.9, No.4, pp. 608-616, June 1999.

[7] Anurag Bist, Wei Wu and Albert Hsueh, "Intelligent Pre-Quantization in Motion compensated Video Coding", *ITU-T Q15-D-35*, Tampere, Finland, April, 1998.

[8] Intel, "Using Streaming SIMD Extensions in a Fast DCT Algorithm for MPEG Encoding," *AP-817*, 1999.

[9] Intel, "Intel Architecture Software Developer's Manual: Instruction Set Reference," *DOC. 243191*, 1999.

[10] Yukihiro ARAI, Takeshi AGUI and Masayusi NAKAJIMA, "A Fast DCT-SQ Scheme for Images," *The Transactions of the IEICE*, Vol. E71, No.11, pp. 1095-1097, Nov. 1988.

Table 1. Performance comparison of MVFAST and AMSED at 384 kbps

Sequence Name	Full Search		MVFAST		Proposed AMSED	
	Checking points	PSNR (dB)	Checking points	PSNR (dB)	Checking points	PSNR (dB)
Akiyo (cif)	676640320	41.45	475595	41.50	233564	41.46
Foreman (cif)	676514032	31.74	2403299	32.08	1342178	32.10
Silent (cif)	676578496	33.63	1470387	33.64	796245	33.58

Table 2. Comparison of benchmark for reference MPEG-4 VM video encoder and our optimized encoder with core profile @ level 2

Sequence Name	Sequence Type	# of Frames	PSNR (Y) (dB)		Speed (FPS)	
			VM	Optimized	VM	Optimized
Akiyo	Rect	300	41.45	41.43	1.93	71.52
Foreman	Rect	300	31.74	32.12	0.55	34.54
Silent	Rect	300	33.63	33.65	0.47	41.21