Video Codec for Mobile Devices Using SPIHT Encoding and Fuzzy Change Detector

Said E. El-Khamy, Fellow IEEE and Julie Islander,

Department of Electrical Engineering, Faculty of Engineering, Alexandria University, Alexandria 21544, Email: elkhamy@ieee.org, and julie_iskander@yahoo.co.uk

1. Introduction

Mobile video communication faces a lot of challenges [2]. First, the enormous amount of data generated by video makes the use of efficient coding techniques vital. Second, the resources of mobile terminals, the processing power and battery life are very limited and scarce resources. Third, the mobile channel is a hostile environment with high bit error rates caused by a number of loss mechanisms, like multipath fading, shadowing, and co-channel interference. Moreover, standard video compression systems such as H.263, H.264, MPEG2 and MPEG4 all rely on complex motion estimation schemes to achieve compression. The computational resources available to typical mobile devices are significantly less abundant than to a PC.

This paper presents a video codec for mobile videos. The codec is based on a video compression proposed in [1]. The codec, proposed here, is based on segmenting the video into scenes using a new fuzzy change detector. For each scene, one or more keyframes is chosen. The number and location of the keyframe is chosen according to the length and characteristics of the scene. Each frame is then differenced from the nearest keyframe. The difference frame is segmented into blocks. Each block is wavelet transformed and compressed using SPIHT. The keyframes are segmented and encoded without differencing.

The paper is organized as follows: Section 2 explains the fuzzy change detector. Section 3 describes the codec operation in more details. Section 4 shows the simulation results and conclusion.

2. Fuzzy Change Detector and Sequence Energy Tester

A scene [3-5] is a sequence of shots that belong together semantically. There are two kinds of shot boundaries in videos, abrupt shot changes called "cuts" and gradual transitions between two different shots. To test the performance of any scene cut detection technique the following terms must be defined first. Recall is the number of correctly detected cuts divided by the number of really existing cuts, and precision is the number of correctly detected cuts divided by the total number of detected cuts (including "false alarms").

The proposed scene cut detector is simple and fast. The objective of the detector is to divide the whole video into independent sequences to be encoded separately. This is done in 2 steps described next. The results are efficient since it keeps complexity low and the presence of this module in the codec improves efficiency. Table 1 shows the recall and precision of the FCD algorithm tested on different types of video sequences.

2.1 Fuzzy change detector:

The video as a whole is used to get the global cuts, thus dividing the video into sequences. The detector uses the normalized sum of absolute difference of histograms between each two successive frames on a pixel-by-pixel basis as the input to the fuzzy inference system, more details about fuzzy systems can be found in[7]. If the normalized difference is high then an abrupt cut is present.

Table 1: Recall and precision results

Sequence	no.of correctly detected	no of real cuts	total no. of detected cuts	Recall	Precision	Notes
onescene.avi (32 frames)	0	0	0	1	1	one scene sequence with a moving boy in a still back-ground
7cutmov.avi (244 frames)	7	7	7	1	1	part of demos from matlab library with 7 sharp cuts

shortmov.avi (22 frames)	3	3	3	1	1	short movie made from 3 different test sequences.
akiyo.avi (300 frames)	0	0	0	1	1	a news reader

2.2 Fuzzy sequence tester:

Using FCD, alone, was efficient enough to detect abrupt cuts and help divide the video file into sequences. But in the case of long scenes where motion occurs, the high level of changes between frames causes degradation in decoded video. This is because the codec depends on differencing frames not on complicated motion estimation and compensation. For each scene a keyframe is chosen and all the other frames are differenced from it. When long scenes with high level of changes between frames especially w.r.t. the keyframe, the encoded frame carries a lot of information that may be lost due to lossy encoding and decoding so another keyframe must be used. This will be further proved in the simulation results.

3. The Proposed Codec

The proposed encoder can be described as follows:

1. Initialize compression parameter, like no. of frames, block size, wavelet decomposition level, frame size, the number of iterations for encoding a keyframe block and the number of iterations for encoding a difference frame block.
2. For the specified block size, create the spatial orientation tree.
3. Using Fuzzy Change Detector (FCD)and Fuzzy Sequence Test (FST), get the frames representing end of scenes or high color distribution difference, and the number and location of keyframes to be encoded.
4. Non-Keyframes are encoded as difference frames from the keyframe. Encoding is done as follows:
    a. Divide the frame into blocks
    b. Each block is encoded using SPIHT[6].
5. Repeat for all scenes of the video.

The decoder can be done by inversing the operation of the encoder. The codec parameters are the number of frames, the block size, the level of wavelet decomposition, the frame size, the change frames numbers, the number of iterations for encoding a keyframe block and the number of iterations for encoding a difference frame block. The number of iterations of encoding a frame is related to SPIHT[7] i.e. the number of times the encoding process is repeated to control the rate and distortion of the frame. The number of frames and the frame size is determined from the video file being encoded. The change frames numbers are determined from the FCD and FST modules. The block size, the level of wavelet decomposition, the number of iterations for encoding a keyframe block and the number of iterations for encoding a difference frame block are variable and can be changed according to the desired target. They affect the average encoding/decoding time, the distortion and the compression rate.

4. Simulation Results

Experiments were done to choose the optimum parameters that minimize the average time and distortion while keeping a suitable compression ratio. They were obtained on a Pentium M processor 1.6 GHz. with 512 RAM on Windows XP SP2 using Matlab 7. Some samples of the results are shown in Figures 2 showing the effect of changing the block size on compression ratio (compressed file size to original raw file size), Mean Square Error (MSE) and average encoding/decoding time

The tests were done on the ‘akiyo’ sequence (first and second frames) of size 176x144. Figure 2 shows the results for block size 8x8. It was found from the experiments that for keyframes as the block size increase, Mean Square Error (MSE) increases a lot; encoding time increases but only slightly, whereas the decoding time decreases (especially at large number of iterations). Compression ratio stays the same. Compression ratio is affected by the number of iterations in the SPHIT algorithm. For difference frames, the MSE increases but only slightly. Encoding/decoding times changes but only slightly. Compression ratio increases with the increasing of the block size. So, the block size is chosen according to the limitations in the target system with tradeoffs between time, compression and distortion. As for the effect of the number of iterations, it is clear that as it increases the MSE decreases while the encoding/decoding time increases significantly and the compression ratio decreases significantly too. A good decision is to choose a large number of iterations to encode blocks in a keyframe while a lower number of iterations is used to encode the difference frames. This is due to the fact that keyframes carries more information then difference frames. Acceptable results are obtained using 8x8 blocks at 2 levels of decomposition.

Additional results are obtained from encoding and decoding a video sequence (the shortmov.avi file described in Table 1) using a 8x8 blocks and level 2 wavelet decomposition. Tables 2 shows the effect of using FCD to get keyframes. Table 3 shows the results when using the first frame of the video sequence as a keyframe. The results when using differencing each frame from the previous one as presented in [1] are shown in Table 4. Figure 1 shows sample frames illustrating these results.

(a) (b) (c)

Figure 1. Reconstructed 7th frame from sample sequence "shortmov.avi" using 8x8 block size, 2 level of avelet decomposition and 10 keyframe iterations and 2 difference frame iterations.
(a) using FCD as in Table 2 . (b) without FCD as in Table 3. (c) without FCD or keyframe as in Table 4.

Table 2. Results for using FCD

No. of iterations for keyframe	No. of iterations for difference frame	MSE	Average encoding time (seconds)	Average decoding time (seconds)	Compression Ratio
4	3	6	1960	1680	2
5	5	2.37	2235	1580	1.03
10	2	6.88	2580	1820	1.7

Table 3. Results for using only one keyframe in the video sequence (the first frame).

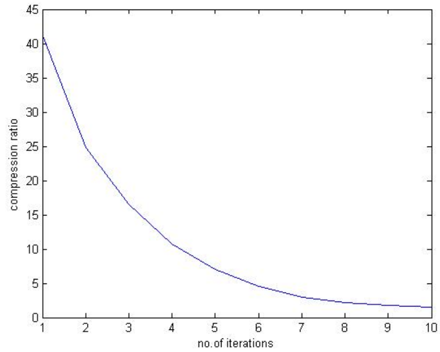
No. of iterations for keyframe	No. of iterations for difference frame	MSE	Average encoding time (seconds)	Average decoding time (seconds)	Compression Ratio
4	3	41	1270	1200	3.55
5	5	13.8	2190	1900	1.7
10	2	69.8	1138	975	3.9

Table 4. Results for differencing each frame from its previous one.

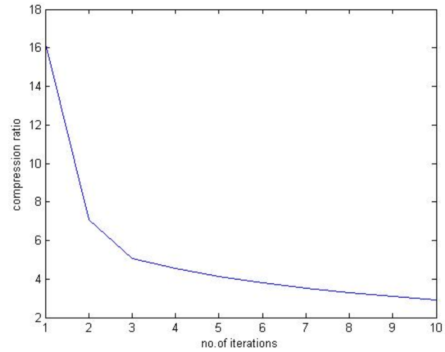
No. of iterations for keyframe	No. of iterations for difference frame	MSE	Average encoding time (seconds)	Average decoding time (seconds)	Compression Ratio
4	3	82.36	3420	1850	1.52
5	5	82.067	3786	1464	0.99
10	2	81.6	2670	1550	2.13

VI. Conclusion

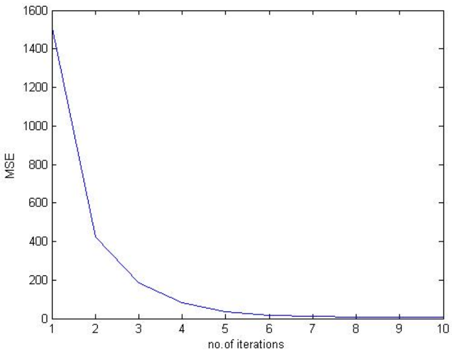
This paper presented a novel video codec that matches mobile device capabilities by reducing complexity while keeping the user quality of experience satisfactory. The codec uses wavelet decomposition and SPIHT to encode the video frames as difference from a chosen keyframe. Keyframes are chosen depending on sequence characteristics using two fuzzy inference systems. The results showed that using block size of 8x8 with 2 levels of decomposition we can get a balance between distortion, operation time and compression ratio.



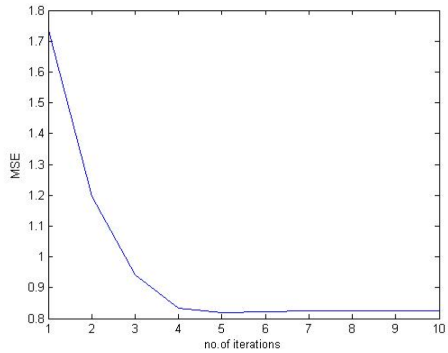
(a)



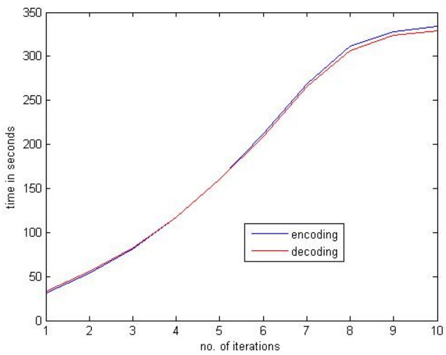
(d)



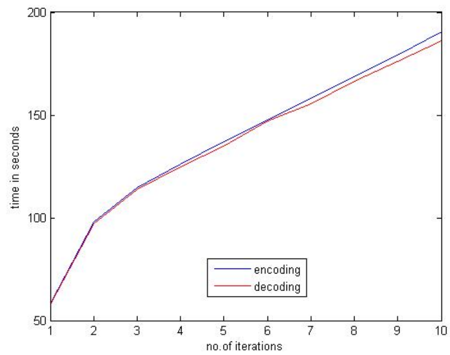
(b)



(e)



(c)



(f)

Figure 2. Effect of number of iterations of SPIHT on Compression ratio, MSE and Encoding/decoding time using 8x8 blocks and 2 levels of wavelet decomposition (a), (b) and (c): for keyframes, while (d), (e) and (f) are for difference frame

References

[1] E. S. Jackson and R. Peplow, "Video Compression System for Mobile Devices", 2003, sponsored by Thales Advanced Engineering, Armscor and The National Research Fund of South Africa.

[2] M. E. Almualla, C. N. Canugarajah and D. R. Bull,"Video Compression for Mobile Communications", Academic Press,2002.

[3] Ralph Ewerth and Bernd Freisleben ,"Video Cut Detection without Thresholds", *Proc. of 11th Int'l Workshop on Systems, Signals and Image Processing*, Poznan (Poland), pp. 227-230, 2004.

[4] Lienhart, R., "Reliable Transition Detection in Videos: A Survey and Practitioner's Guide", *International Journal of Image and Graphics*, Vol. 3, 2001, pp. 469-486.

[5] Yeo, B., and Liu, B., "Rapid Scene Analysis on Compressed Video", in *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 5, No. 6., 1995, pp. 533-544.

[6] Amir Said, William A. Pearlman, "A New, Fast, and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees", IEEE Trans. On circuits and systems for video technology, vol. 6, no. 3, pp 243-250, June 1996.

[7] E. Cox, "The Fuzzy Systems Handbook", Academic Press, NewYork, 1994.