Adaptive Key Frame Selection for Efficient Video Coding

Jaebum Jun, Sunyoung Lee, Zanming He,
Myungjung Lee, and Euee S. Jang

Digital Media Lab., Hanyang University
17 Haengdang-dong, Seongdong-gu, Seoul, 133-791, Korea
powerory@hotmail.com, profjang@gmail.com

Abstract. Recently, many researches on frame skipping are conducted to reduce temporal redundancy in video frames. As a simple method, fixed frame skipping (FFS) adjusts frame rate by skipping frame at regular intervals. To overcome the poor performance of FFS, variable frame skipping (VFS) has been introduced to exploit the temporal dependency between frames. In this paper, scene-adaptive key frame selection method with low complexity is proposed. The proposed method performed about 20 percent better in complexity with the better visual quality than the conventional video encoding. As a preprocessing method, the proposed technology can be used with any conventional video codec.

Keywords: variable frame skipping, frame interpolation, fame rate control.

1 Introduction

Many conventional video codecs tried to reduce temporal redundancy in video frames by means of motion estimation and motion compensation (MEMC). MEMC was performed as a part of video encoding process.

As an alternative approach, frame skipping (FS) method has been investigated. It is to further exploit the temporal redundancy of video frames by skipping some video frames in encoding. The skipped video frames may be generated at the decoder by repeating the previous video frame or by interpolating neighboring frames.

By employing FS in the coding process, one can allocate the more bits for each frame in encoding. By lowering the frame rate, the better picture quality can be obtained for the encoded frames. FS helps in downsizing the decoding time and the complexity by reducing the number of coded frames. These are the clear advantages of FS over MEMC.

The performance of FS is highly dependent upon the selection of coded (or uncoded) frames from the given video sequence. The most critical issue in FS is not to lose semantically important frames after FS. Therefore, a key element in FS is to distinguish repetitive or easily-interpolatable frames from the video sequence.

In FS, there are two representative approaches: fixed frame skipping (FFS) and variable frame skipping (VFS). By skipping frames at regular intervals, FFS is a useful technology for very low bit rate environment [1]. FFS is simple to implement, but suffers severe quality degradation due to the jerky effect in the sequence with high motion.

In the case of VFS, the interval of skipping frames can be changed depending on the similarity in video frames. Therefore, it can reduce the jerky effect when the bitstream is decoded and interpolated. Several methods are proposed in VFS [2]-[4].

Based on the similarity of the last two frames in the previous group of pictures (GOP), Song[2] defined the variable frame rate of the current GOP. However, the method showed a limited performance in the presence of high motion in a scene, since the frame rate of the current GOP is predicted only from the previous GOP information and the range of the variable frame rate was not flexible enough.

Pejhan[3] proposed a dynamic frame rate control mechanism. He tried to separate files containing motion vectors at the low frame rates. However this method has to analyze the entire encoding video sequence for adjusting frame rates. Due to the complexity that all the frames are to be analyzed, it would be difficult to use the method in real time applications.

Kuo[4] proposed a VFS method that introduced interpolation in the encoding process. The results reported in [4] produced the better PSNR quality than conventional coding with no FS. In [4], the method is closely coupled with the encoding process targeted for low resolution and low bit rate applications.

In this paper, we proposed a straightforward skipping algorithm by adaptively selecting key frames depending on the similarity of video sequence. In designing the proposed scheme, we focused on two requirements: low computational complexity and efficient selection of key frames.

The main objective of our method is to build a framework that provides good enough visual quality to the user. We tried to reach the level of good enough quality not necessarily by minimizing the difference (e.g., PSNR) between the original and interpolated video frames, but rather by semantically interpolating in between key frames. In this case, the level of visual quality can only be estimated correctly by subjective test with independent viewers.

The remainder of this paper is organized as follows. In Section 2, the proposed method is described including frame skipping and frame interpolation processes. Experimental results are provided in Section 3. We summarized the paper and listed the future work in Section 4.

2 Adaptive Key Frame Selection

Our proposed system can be described as shown in Fig. 1. The proposed frame skipping (FS) process can be implemented either inside or outside of the encoding process, since the FS is not dependent upon the encoding process. After FS, only the

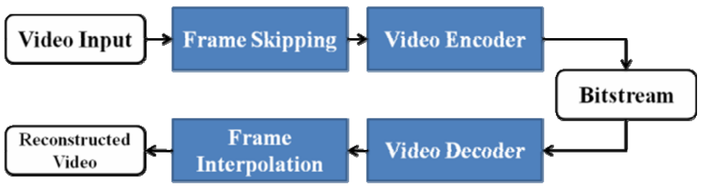


Fig. 1. Block diagram of the encoding and decoding process of the proposed system

selected frames will be encoded by the encoder. Once these encoded frames are decoded from the transmitted bitstream, frame interpolation is performed at the decoder side.

2.1 Frame Skipping

The proposed frame skipping algorithm can be described with the following three steps:

Step 1: Define the maximum cluster size.
Step 2: Analyze similarity between neighboring frames and form clusters.
Step 3: Select key frames.

The maximum cluster size (MCS) is defined in Step 1. A cluster is defined as a set of consecutive video frames with high similarity. The definition of MCS may be needed to meet the physical cluster size for application needs. For example, the MCS can be defined to be 30 to ensure that frame skipping does not last more than a second. The definition of the MCS does not affect the encoding/decoding process. It is only used to identify and form a cluster in Step 2.

The similarity among neighboring video frames is checked to identify clusters from the given video sequence in Step 2. In this paper, to measure distortion between two frames, PSNR is used as depicted in Fig. 2. For simplicity, tools such as sum of absolute difference (SAD) may be used instead. A cluster is identified when all the PSNR values of video frames in the cluster are higher than a certain threshold:

$$PSNR\ (i, i+1) \geq T \tag{1}$$

where $PSNR\,(i, i+1)$ is the PSNR value between the i-th and (i+1)-th frames and T is the threshold.

If $PSNR\,(i, i+1)$ is larger than the threshold, it is assumed that the i-th frame has little similarity to the (i+1)-th frame and a scene change is happened between the i-th and (i+1)-th frames as shown in Fig. 3. We have assigned an appropriate value to the threshold for each test sequence in empirical approach.

This clustering method would include a case that PSNR between the first frame and the last frame in a cluster is lower than the threshold. From the experiments, we found that such a case exists and clustering is still effective. When such a case happens, it means that only a small part of an entire frame is moving. This creates a gradual change in PSNR, which results in a great difference between the first and the last frames. However, it would be nicely interpolated when using motion vector information at the decoder side.

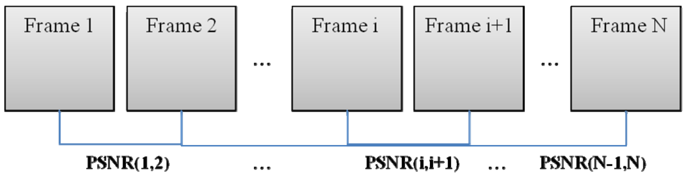


Fig. 2. PSNR computation between consecutive image frames

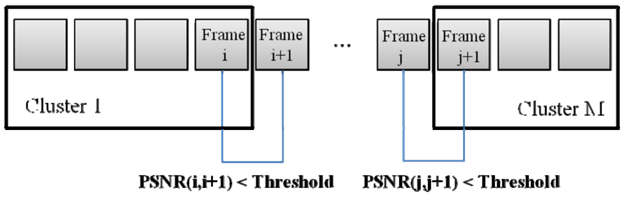


Fig. 3. Example of forming clusters

In Step 3, the first and the last frames in a cluster are candidates for the key frame as shown in Fig. 4 (a). In forming clusters, it is also possible for neighboring two clusters overlapped by one key frame as depicted in Fig. 4 (b). It is usually caused when the MCS is set too small and the actual cluster size is larger than the MCS. When two clusters are overlapped, the last frame of the first cluster will be the first frame of the second cluster. If three consecutive clusters are overlapped, the number of selected key frames becomes four.

The number of the skipped frames can be identified when reading the time stamp in each selected video frame. If there is a certain jump in time stamp between two consecutive frames, the decoder can detect how many frames are skipped by computing the difference of the two time stamps.

2.2 Frame Interpolation

When we use the frame skipping at the encoder side, skipped frames will be interpolated by using key frames at the decoder side. There are several methods to interpolate skipped frames such as repetition, bilinear interpolation, and motion compensated interpolation (MCI) [4], [5]. In this paper, we used an MCI-based method at the decoder side.

The proposed MCI-based interpolation method is performed to reproduce N skipped frames in between the first and the last key frames in a cluster. For the interpolation of the i-th skipped frame (i = {1, …, N}), the following procedure can be applied:

Step 1: [Bilinear interpolation] perform bilinear interpolation (cf. Eqn. 3) using the first and the last key frames to fill the i-th skipped frame.
Step 2: Select an MB in the last key frame with the following conditions: 1) a nonzero MV and 2) the smallest MV value among the unprocessed MBs.
Step 3: [First-frame background filling] Fill the collocated MB in the i-th skipped frame with the collocated MB in the first key frame according to Eqn. 4.
Step 4: [Last-frame background filling] Using the MB where the MV of the selected MB is pointing in the last key frame, fill the collocated area of the i-th skipped frame according to Eqn. 5.
Step 5: [Motion compensated bilinear interpolation] Find the area in the skipped frame with the scaled MV from the selected MB in the last key frame. Update the area by bilinear interpolation of the area which the MV of the selected MB in the last key frame is pointing and the selected MB in the last key frame according to Eqn. 6.
Step 6: Go to Step 2, until all MBs in the last key frame are processed.

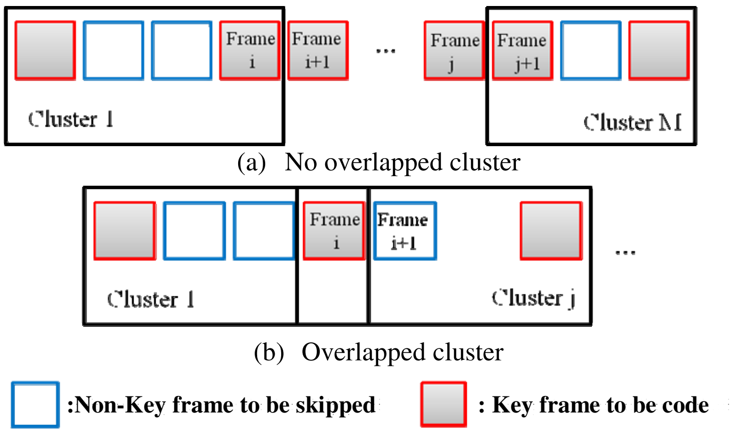


Fig. 4. Selecting key frames

The basic principle in interpolation is to use a two-pass interpolation. The first pass is the bilinear interpolation between the first and last key frames. In the second pass, we exploited the MV values in the last key frame to identify moving objects between the first and the last key frames.

When the *i*-th skipped frame is interpolated, interval ratio (R) should be computed based on the time stamps of the first, the last, and the *i*-th skipped frames:

$$R = \frac{T_i - T_{first}}{T_{last} - T_{first}} \tag{2}$$

where T_i, T_{first}, T_{last} are the time stamps of the *i*-th skipped, the first, and the last video frames, respectively.

For bilinear interpolation the following equation can be formulated:

$$P_i(x, y) = P_{last}(x, y) \times R + P_{first}(x, y) \times (1 - R) \tag{3}$$

where $P_i(x, y)$, $P_{last}(x, y)$, and $P_{first}(x, y)$ denote pixel values at the *x*-th row and the *y*-th column of the *i*-th skipped, the last, and the first video frame respectively.

The bilinear interpolation is a very simple method to reduce the computational complexity. However, the visual quality using bilinear interpolation is poor in the presence of fast moving objects in the first and the last key frames. In order to overcome this shortcoming, the second pass with the MVs in the last frame is performed.

In the second pass, we start with the MBs with small MV values to large MV values. The reason that we start with MBs with small MVs is due to the fact that moving objects with high motion is more influential to the visual quality of the interpolated frame. Therefore, the MB with the largest MV will be used last.

Once there is an MB selected with nonzero MV, three operations follow as shown in Fig. 5: first-frame background (FB) filling, last-frame background (LB) filling, and motion compensated bilinear interpolation (MCBI).

FB filling is to fill an MB in the skipped frame with the collocated MB in the first frame:

$$P_i(x, y) = P_{first}(x, y) \tag{4}$$

In LB filling, the area in the first key frame pointed by the selected MB in the last key frame is identified first. The collocated (to the pointed area in the first key frame) area in the skipped frame will be filled with the collocated area in the last key frame:

$$P_i(x + MV_x, y + MV_y) = P_{last}(x + MV_x, y + MV_y) \tag{5}$$

In MCBI, the filled area in the skipped frame can be found using the interval ratio (cf. Eqn. 2) and MV in the last key frame as shown in Fig. 5. The area will be filled by bilinear interpolation as follows:

$$\begin{aligned} &P_i((x + MV_x) \times R, (y + MV_y) \times R) \\ &= P_{first}(x + MV_x, y + MV_y) \times (1 - R) + P_{last}(x, y) \times R \end{aligned} \tag{6}$$

where MV_x, MV_y are motion vectors of macro block in the last key frame, which references the first key frame.

3 Experimental Results

For the evaluation of the proposed method, we used MPEG-4 simple profile (SP) as a test bed on a PC equipped with Intel Core2Duo 2.8GHz running Windows Vista. The test sequences used in the experiment are shown in Table 1. For MPEG-4 SP codec, we used the MPEG-4 reference software [6], where no optimization is performed. For simplicity of the experiment, we used only one MV per MB.

Table 2 shows the result of the proposed key frame selection. With three different MCS values for each test sequence, it is observed that the more frames will be skipped with the larger MCS value. This is more apparent in Akiyo and Container, whereas the variation in the number of selected key frames is marginal when the video sequence is highly active such as in Stefan.

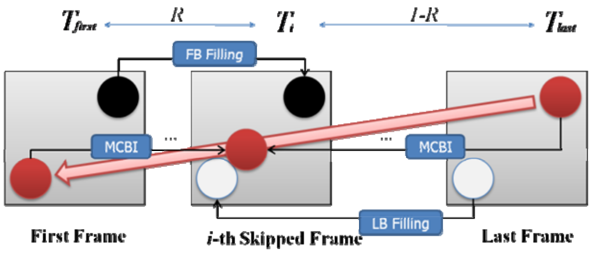


Fig. 5. MCI-based Interpolation

Table 1. Test Sequence

Sequence	*Resolution*	*Number of frames*	*FPS*
Akiyo	CIF (352x288)	300	30
Container	CIF (352x288)	300	30
Stefan	SIF (352x240)	300	30

Table 2. Key frame selection results

Test sequence	*MCS (frames)*	*T (dB)*	*Number of clusters*	*Selected key frames/ average fps*
Akiyo	3	38	85	215 (21.5)
	7	38	36	141 (14.1)
	12	38	24	124 (12.4)
Container	3	38.2	75	225 (22.5)
	7	38.2	40	150 (15.0)
	12	38.2	30	124 (12.4)
Stefan	3	23	11	289 (28.0)
	7	23	6	282 (28.2)
	12	23	5	280 (28.9)

We chose arbitrary thresholds (*T*) for clustering from sequence to sequence, keeping in mind that a high threshold is assigned when there is low in motion as shown in Table 1.

A large cluster is divided into multiple clusters when MCS decreases. For example, there are 24 clusters formed in Akiyo when MCS was set to 12, where as 61 more clusters formed when MCS was set to three with the same threshold. Overall, it is shown that the average frames per second (fps) can be controlled with MCS and T. In the table, the average fps varied from 12.4 to 28.9.

We have depicted a bar graph in Fig. 6, which indicates the selected key frames in Akiyo sequence. A bar in the figure indicates a selected key frame. If the distance between two neighboring bars is greater than one, there is a cluster. From the figure, it is clearly noticed that the number of the skipped frames is highly affected by MCS.

For the estimation of the computational complexity, we measured the encoding and decoding time in Table 3. Compared with the encoding time of MPEG-4 SP, the skipping time including encoding time of the proposed method is faster for all the test sequences. Our proposed skipping method requires only PSNR computation between two neighboring frames and reduces the coding time according to the number of the skipped frames. Furthermore, decoding time of the proposed method including the interpolation time is also comparable with the decoding time of MPEG-4 SP.

Table 4 shows the number of the coded frames for MPEG-4 SP and the proposed method at the similar bit rate. The reduced number of frames in the proposed method resulted in high visual quality, since more bits per frame can be assigned in encoding using the lower quantization parameter (Qp). Obviously, the better visual quality in key frames leads to the better visual quality in interpolation.

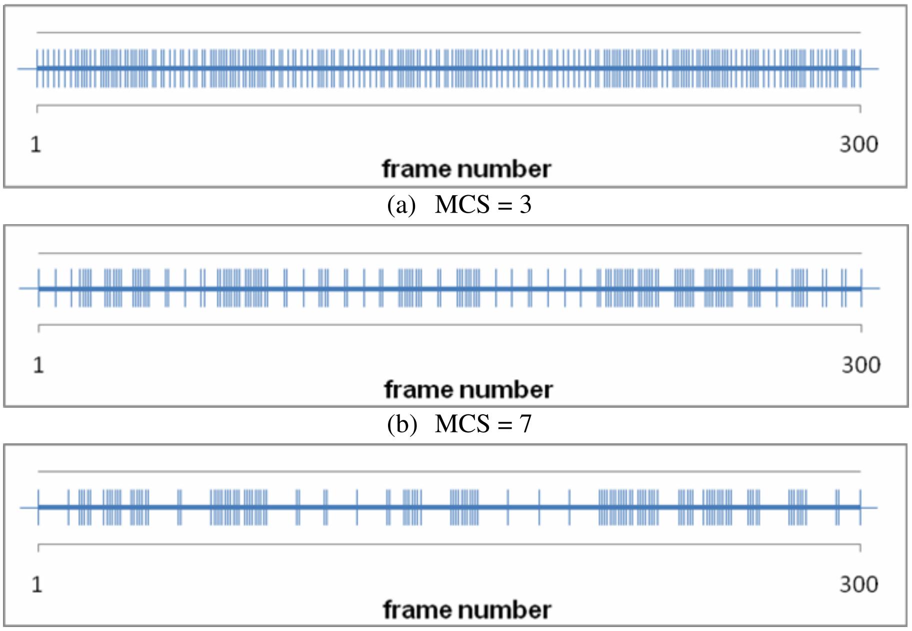


Fig. 6. Selected key frames in Akiyo

We evaluated the PSNR values between the original and reconstructed sequences using two methods (MPEG-4 SP and proposed interpolation method), where the MCS value is 7. For test sequences except Akiyo, the average PSNR values using two methods are quite close to each other. In the case of Akiyo sequence, the average PSNR value of the reconstructed sequence using the proposed method is 36.38dB, whereas that using MPEG-4 SP is 34.44dB as shown in Fig. 7. The results obtained in Akiyo sequence are rather surprising in that we did not expect the objective quality using the proposed method would excel that using MPEG-4 SP.

Fig. 8 shows the comparison of reconstructed key frames from MPEG-4 SP and the proposed method. If many frames are skipped as in the case of Akiyo and Container,

Table 3. Evaluation of encoding time and decoding time (in seconds)

	Test Sequence	*MPEG-4 SP*	*Proposed Method*		
			MCS=3	MCS=7	MCS=12
Encoding Time	Akiyo	14.64	11.90	8.42	7.78
	Container	19.56	18.87	14.78	13.04
	Stefan	33.08	31.81	31.63	32.22
Decoding Time	Akiyo	3.09	2.98	2.48	3.10
	Container	3.21	2.99	3.07	3.04
	Stefan	4.04	4.19	4.15	4.45

Table 4. Comparison of Qp between MPEG-4 SP and the proposed method

Test sequence	*Method*		*Qp*	*Coding frames*	*Bitstream size (byte)*
Akiyo	MPEG-4 SP		16	300	138,323
	Proposed Method	MCS=3	12	215	134,330
		MCS=7	9	141	127,858
		MCS=12	8	124	125,174
Container	MPEG-4 SP		16	300	228,041
	Proposed Method	MCS=3	14	225	210,438
		MCS=7	10	150	216,991
		MCS=12	9	124	218,381
Stefan	MPEG-4 SP		16	300	1,094,888
	Proposed Method	MCS=3	16	289	1,074,832
		MCS=7	16	282	1,060,963
		MCS=12	16	280	1,049,128

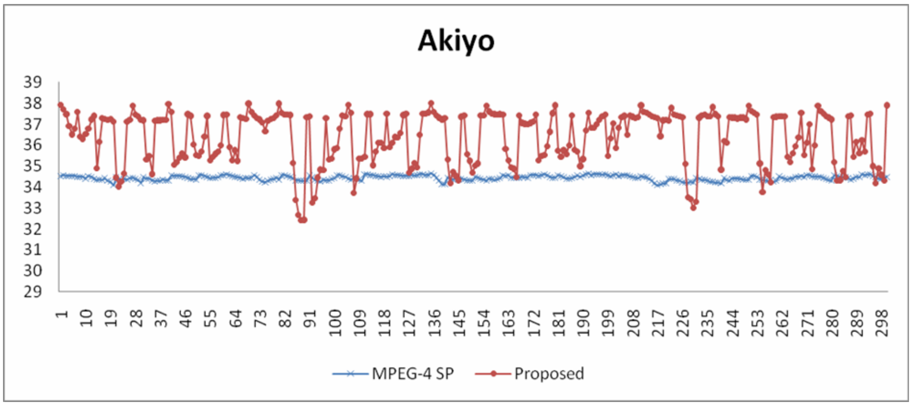
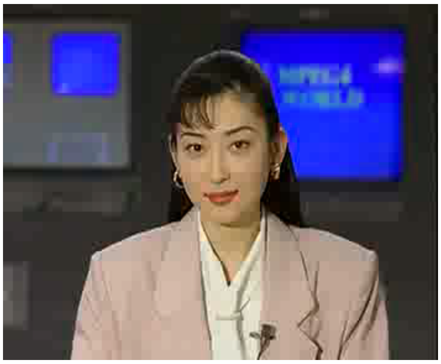


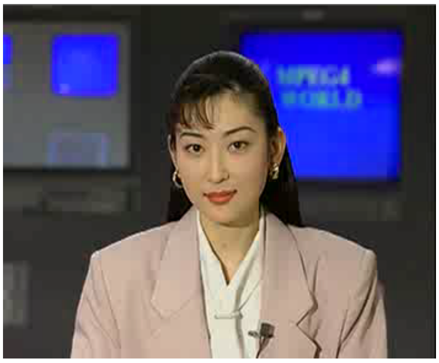
Fig. 7. Comparison of PSNR in Akiyo

the improved visual quality of the encoded key frames is apparent. In the case of Stefan, not many frames are skipped due to high motion. In such a case, there is no apparent gain in Qp.

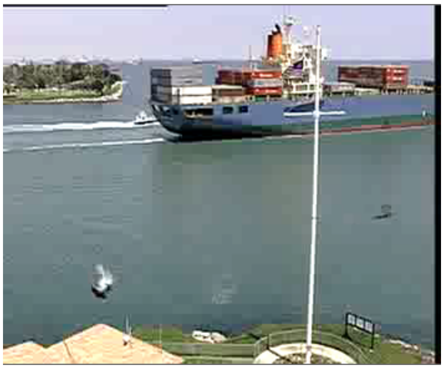
In Fig. 9, the screenshots of the first key frame, the interpolated frame, and the last key frame using the proposed method are compared with the corresponding frames using MPEG-4 SP. In Akiyo and Container sequences, the interpolated frames are better than the coded frames using MPEG-4 SP in that there are less visual artifacts due to the better key frames. It does not necessarily mean that the interpolated frame is close to the original skipped frame. As explained earlier, the main objective of this work is to reproduce the interpolated frame with semantically acceptable visual quality. In the case of Stefan, the interpolated frame is worse than the coded frame using MPEG-4 SP. It means that frame skipping and interpolation of video sequences with high motion is of little value.



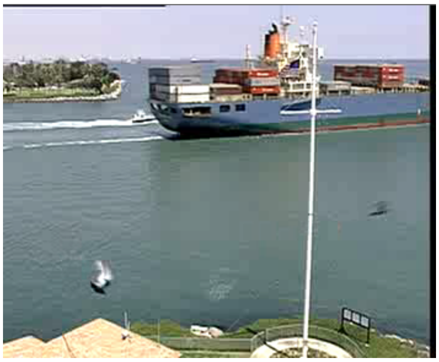
(a) Akiyo screenshot with MPEG-4 SP (Qp = 16)



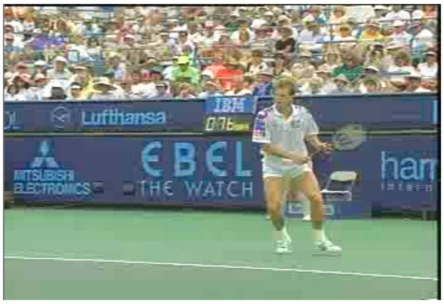
(b) Akiyo screenshot with the proposed method (MCS=7 & Qp =9)



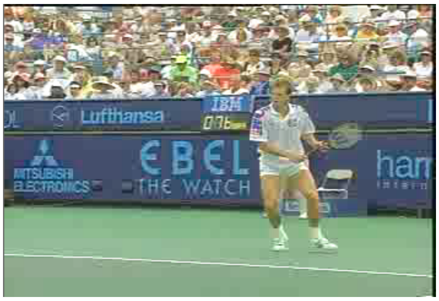
(c) Container screenshot with MPEG-4 SP (Qp = 16)



d) Container screenshot with the proposed method (MCS = 7 & Qp = 10)



(e) Stefan screenshot with MPEG-4 SP (Qp = 16)



(f) Stefan screenshot with the proposed method (MCS=7 & Qp =16)

Fig. 8. Screenshot of the selected key frames

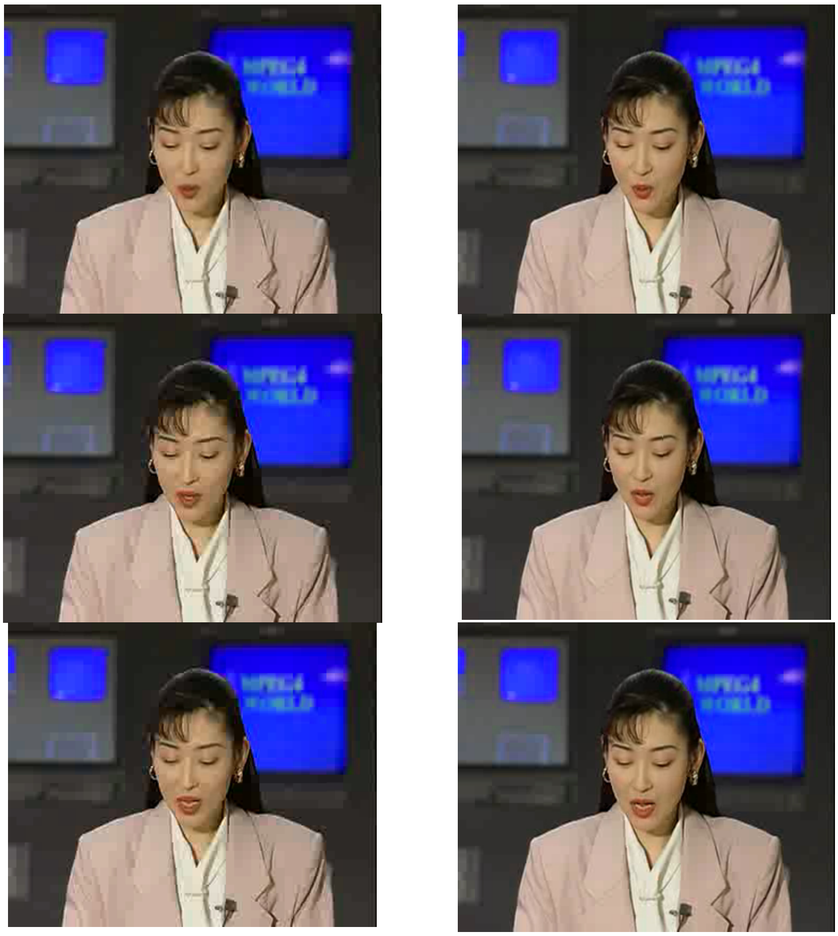


Fig. 9a. Akiyo screenshot

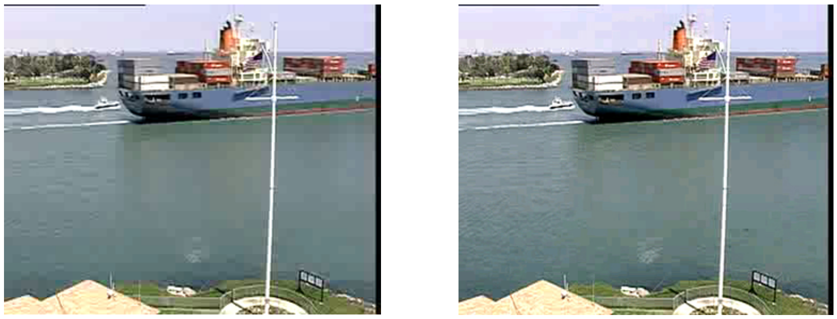


Fig. 9b. Container screenshot

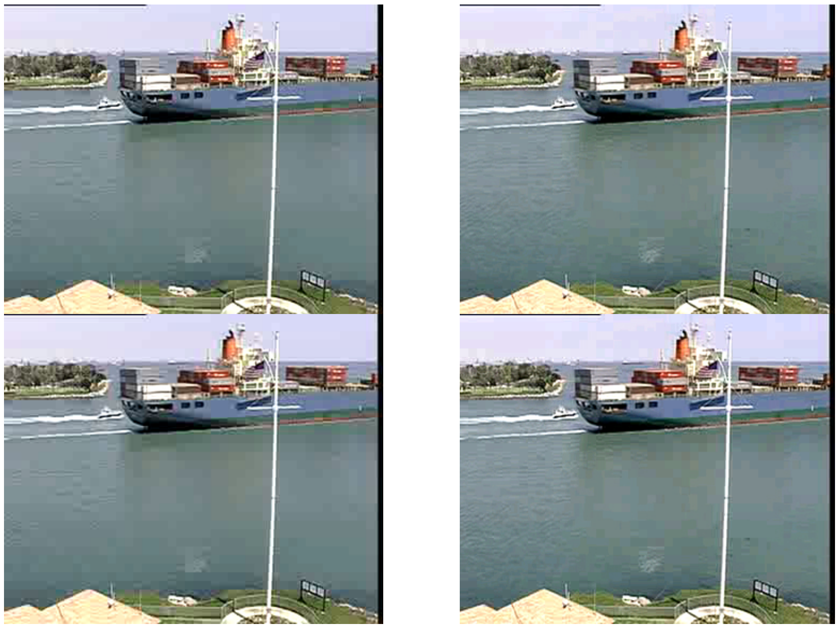


Fig. 9b. (*continued*)

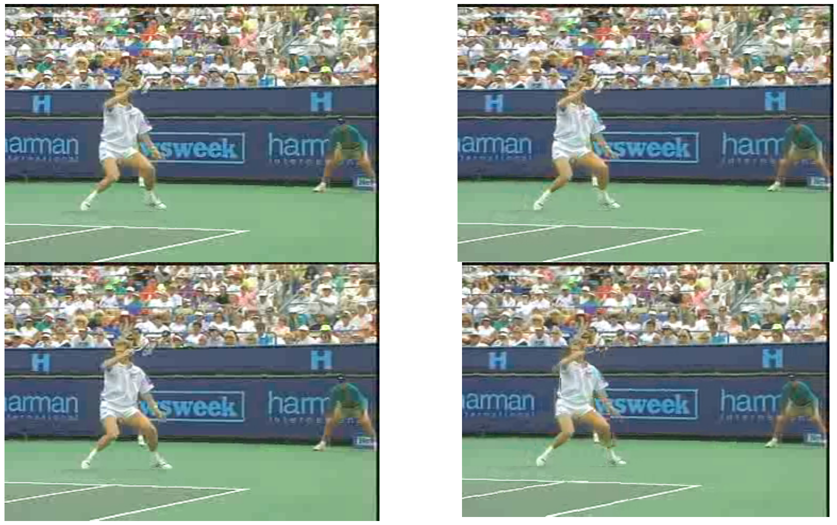


Fig. 9c Stefan screenshot

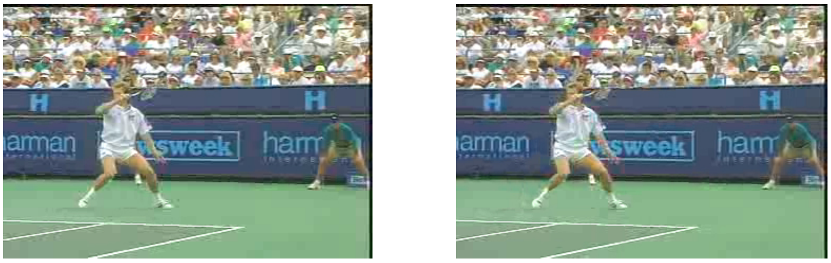


Fig. 9c. (*continued*)

Screenshots of the interpolated frames in the form of

MPEG-4	Proposed
Decoded frame	First key frame (decoded)
Decoded frame	Interpolated frame
Decoded frame	Last key frame (decoded)

We also conducted a subjective test (mean opinion score: MOS) with 40 viewers. The viewers are chosen from the first year undergraduate students who are taking C programming classes, where these viewers cannot be regarded as image processing experts. The viewers are given three video sequences: 1) the original test sequence and 2) two test video sequences using MPEG-4 SP and the proposed method (with an arbitrary order). After viewing, the viewer scored the test sequences between one (very corrupted) and five (undistinguishable with the original).

In this subjective test, we set the Qp value (16) in using MPEG-4 SP. For the test sequences of the proposed method, we used the cases, in which MCS is set to 3 for all test sequences with different Qp value as shown in Table 4.

Fig. 10 shows the results of the MOS of the blind test. In Akiyo and Container, the viewers gave more scores on the proposed method, while the best and the worst scores for the proposed method are the same or better than those for MPEG-4 SP. In the case of Stefan, the lower score was anticipated due to its difficulty in forming clusters. Considering that the interpolated frames using the proposed method are not

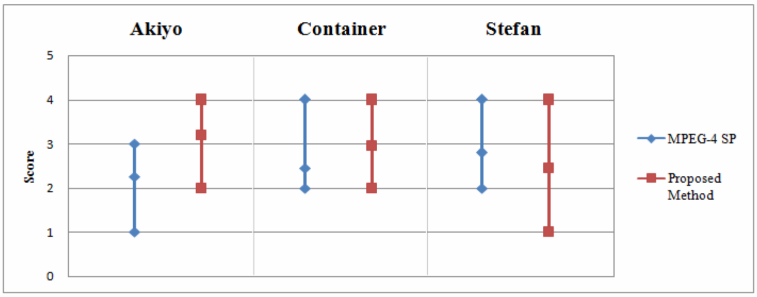


Fig. 10. MOS of the blind test

necessarily mathematically closer to the original frames than those using MPEG-4 SP, the subjective test results prove that the good enough quality reached using the proposed method.

4 Conclusion

In this paper, we proposed an adaptive key frame selection method to adaptively select key frames for the better interpolation at the decoder. From the experimental results, we showed that the proposed method is simple and efficient. Because of its low complexity at the encoder side as well as at the decoder side, the proposed method is well suited in real-time environment.

In this paper, the more focus has been given to the frame skipping part. Yet, a good frame interpolation is as important as a good frame skipping. Research on frame interpolation mechanism to yield good enough visual quality with low complexity should be continued in the future.

References

1. ITU -T Study Group 16.: Video codec test model, near-term, version8 (TMN8). ITU - Telecommunications Standardization Sector, Q15-A-59 (1997)
2. Song, H., Kuo, C.-C.J.: Rate control for low-bit-rate video via variable-encoding frame rates. IEEE Trans. Circuits Syst. Video Technol 11(4), 512–521 (2001)
3. Pejhan, S., Chiang, T.-H., Zhang, Y.-Q.: Dynamic frame rate control for video streams. ACM Multimedia 1, 141–144 (1999)
4. Kuo, T.-Y.: Variable Frame Skipping Scheme Based on Estimated Quality of Non-coded Frames at Decoder for Real-Time Video Coding. Ieice Trans. Inf. and Syst E88-D (12), 2849–2856 (2005)
5. Kuo, T.-Y., Kim, J., Kuo, C.-C.: Motion-compensated frame interpolation scheme for H.263 codec. In: Circuits and Systems, ISCAS 1999, vol. 4, pp. 491–494 (1999)
6. MPEG-4 Reference Software, http://mpeg.nist.gov/cvsweb/MPEG-4/MPEG4RefSoft/Video/natural/microsoft-2.5-040207-NTU/