

CODING METHOD FOR EMBEDDING AUDIO IN VIDEO STREAM

Harri Sorokin, Jari Koivusaari, Moncef Gabbouj, and Jarmo Takala

Tampere University of Technology
Korkeakoulunkatu 1, 33720 Tampere, Finland

ABSTRACT

In this paper, a new method for synchronous coding of digital audio and video signals is proposed. The audio signal is embedded within video signal to produce synchronous hybrid signal that is further encoded. The receiver is able to extract the audio signal from the hybrid signal, and thus, is able to reconstruct two separate signals, the audio and the video. The proposed method enables synchronous coding, transmission, storage, and playback of a video signal and the associated audio signal. The scheme can be used with the current and the developing coding standards to enable the accurate synchronization for the audio and the video data.

Index Terms— video compression, synchronization, embedding

1. INTRODUCTION

A presentation of video and audio data to produce a high-quality audiovisual sensation requires accurate timing control. Although video and audio data are processed with separate coding systems, they need to be played back almost simultaneously. Human perception can recognize that the video data is not displayed in sync with the audio data if the timing difference between audio and video is a few tens of milliseconds.

Traditionally audio and video signals are compressed with separate encoders, e.g., as in MPEG-2 [1], and the resulting audio and video streams are multiplexed with additional header information to construct audiovisual stream that is transmitted through a communication channel or stored for later use. At the receiver side, the audiovisual stream is demultiplexed and decoded for playback by separate decoders. Therefore, additional information to obtain synchronous playback of audio and video signals is required to be transmitted from the sender (encoder) to the receiver (decoder). In standard coding systems, synchronization of audio and video streams is accomplished by attaching time stamps and clock references to the compressed bit stream. Time stamps and clock references can be used to obtain correct playback time for the video signal and the associated audio signal at the decoder side. However, the accurate synchronization can be disturbed, e.g. by jitter in network transmission or by delays

in packet arrival times. Also, the synchronization information can be easily corrupted by new multimedia services such as transcoding and audio/video editing of the compressed data.

The synchronization problem can also be solved by embedding the audio information within the video data to obtain synchronous hybrid stream. In [2], audio bits are embedded in the spatial domain by increasing the mean value of the 4×4 blocks when the audio bit is equal to "1". If the audio bit is equal to "0" the mean value remains unchanged. However, it seems that the introduced audio embedding and extraction method does not work correctly and the audio data will be corrupted and reconstruction errors will be perceived in video sequences. For universal video data, it is not sufficient to measure only the similarity between the neighboring blocks to determine if the mean value of the current block was increased or not, i.e., audio bit was set as "1" or "0". In [3], a method is described, which embeds audio information into the coefficients that are located near the DC coefficient. However, the audio data is most likely to be corrupted by the quantization operation because of the lack of suitable audio embedding operations. The bit positions containing the audio information may be truncated by the quantization and the audio information can be lost.

In [4], 4-dimensional discrete cosine transform (DCT) is used as the basis of the video compression. This means that DCT is not only used in the spatial domain but it is also utilized in time domain and 3-component color domain. The encoding scheme embeds an audio signal into the related video signal and produces synchronous hybrid signal that is further encoded. The decoding scheme is able to extract the embedded audio signal as well as the related video signal from the synchronous hybrid signal. Therefore, the synchronous presentation time of the encoded audiovisual material can be maintained through the whole coding operation from the source encoding system to the destination decoding system. Consequently, additional information about the presentation time and multiplexing of video and audio signals are not necessary required for synchronization purposes.

In this paper, a novel embedding scheme is proposed where video compression based on 4-dimensional DCT is used as carrier for audio signal. The audio is embedded by modifying the DCT coefficients of video blocks accord-

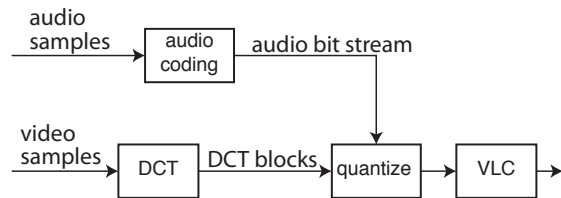


Fig. 1. Principal block diagram of embedding audio in DCT domain.

ing to audio data. Improvements are obtained by carefully selecting the modified coefficient so that the effect on the quality remains as small as possible. The proposed method relies on computing the total parity of all the coefficients in the quantized DCT block. The proposed coding scheme for digital audio and video allows synchronous coding, transmission, storage, and playback of a video signal and the associated audio. Our experiments show that the embedding scheme results in significant improvement in compression rate compared to previous embedding schemes. In the first experiment, we used 4D DCT with $4 \times 4 \times 3 \times 3$ blocks as in [4]. However, the method is not dependent on DCT geometry and, therefore, we show also an experiment with public domain MPEG-4 codec XviD [5] where we used 8×8 blocks.

2. EMBEDDING SCHEMES

Embedding audio signal into video serves two purposes; first, it allows synchronous transfer of the hybrid signal and, second, it allows simultaneous coding and decoding of audio and video. The principal idea of the embedding methods considered here is to carry out the embedding in the quantized DCT domain as illustrated in Fig. 1. In particular, no bits are added to the resulting video stream as the audio information is included by modifying the quantized DCT coefficients according to the audio data. The audio stream transmission is lossless because the embedding takes place after the quantization operation. In this sense, if audio compression is needed a separate audio codec can be applied before the embedding as shown in Fig. 1.

Modifications to the quantized DCT blocks can lead to severe distortions in the reconstructed image blocks. Therefore, moderate embedding will be used and we decided that one audio bit per block is sufficient. Examples of audio capacities obtained by this approach are listed in Table 1. The raw audio data is shown only as illustrative example. Since the embedded data can be perfectly recovered in the decoding process, compressed audio as well as any other type of data can be used as the payload.

In the following sections, we discuss audio embedding schemes and propose a new method.

Table 1. Embedding capacity of video and coding scenarios. The characteristics of raw audio data is given only as an example as the proposed method supports any kind of digital data, e.g., compressed audio.

Scenario	Capacity	Embedding example
4D DCT 352×288	63 kbps	mono 8 bits @ 6 kHz
4D DCT 720×486	219 kbps	mono 8 bits @ 24 kHz
XviD 352×288	71 kbps	mono 8 bits @ 8 kHz
XviD 720×486	251 kbps	mono 8 bits @ 24 kHz

2.1. Interleaving of Audio and Video

The traditional method for audio/video transmission is to interleave audio and video data. We will use this approach later as a reference in our experiments. However, in our experiments we simplify the implementation and we omit all the transport stream overhead and synchronization; the audio corresponding to the time dimension of the DCT blocks is written as such after the compressed video. This gives an optimistic bit rates for the method but it serves as an important reference with undistorted video and total bit rate.

2.2. Magnitude Method

The embedding scheme described in [4] embeds the data using coefficient magnitudes in two fixed positions in the DCT blocks. The embedding procedure modifies one of the two DCT coefficients such that the embedded bit can be obtained as the comparison result of the magnitudes. The problem is that depending on the coefficients, the magnitude of the modification can be really large resulting in huge loss in visual quality. Also, the location of the two coefficients is important when considering the effects on increase in bit rate.

2.3. Simple Parity Method

Our first simple attempt to improve quality is to select a single coefficient and modify its parity bit to match the audio bit to be embedded. While this method has the advantage of smaller quality degradation, the original problem remains; the selection of the position of the DCT coefficients to modify. Still, this idea is used as the basis for the proposed method that is described next.

2.4. Proposed Method based on Total Parity

The loss of video quality is inevitable when the DCT coefficients are modified. This problem can be minimized by carefully selecting the modified coefficient so that the effect on the quality remains as small as possible. On the other hand, the loss of quality can be compensated if the the DCT coefficient is selected and modified such that lower bit rate is achieved.

For example, a coefficient right in the middle of the quantization planes can be quantized to both one and zero parity without any effect to quantization noise. A coefficient with absolute value of one can be modified to zero, which in turn increases variable length coding efficiency and, thus, reduces bit rate.

The proposed method relies on computing the total parity of all the coefficients in the quantized DCT block. As we are considering the total parity, there is no need to select a particular coefficient to modify and, therefore, the embedding procedure is free to choose any coefficient that minimizes the quality degradation and/or decreases bit rate. The total parity is then modified through the parity of the chosen coefficient.

This method creates room for optimizing performance. However, as such the method can embed only one bit per block. However, this is still considered to be sufficient for many applications.

The visual quality is measured as PSNR defined by

$$\text{PSNR} = 10 \log_{10} \left(\frac{x_{max}^2}{\frac{1}{N} \sum (x(n) - \hat{x}(n))^2} \right) \quad (1)$$

where $x(n)$ are the pixels in the original frame, $\hat{x}(n)$ are corresponding reconstructed pixels and x_{max} is the maximum representable value (typically 255). The effect of a single block to the sum can be given as

$$J_0 = \|\mathbf{x} - \hat{\mathbf{x}}\|^2 = \|\mathbf{F}\mathbf{c} - \mathbf{F}\hat{\mathbf{c}}\|^2 = \|\mathbf{c} - \hat{\mathbf{c}}\|^2 \quad (2)$$

where \mathbf{x} is the original image block in vector form, the columns of \mathbf{F} are the DCT basis functions and \mathbf{c} is the cosine transform of \mathbf{x} . Similarly, vector $\hat{\mathbf{c}}$ contains the corresponding reconstructed quantized coefficients and $\hat{\mathbf{x}}$ is the reconstructed image block. When embedding modification $s \in \{-1, 1\}$ is applied to the k th DCT coefficient, c_k , when a quantization step of q_k is used, the error contribution becomes

$$J = \|\mathbf{c} - (\hat{\mathbf{c}} + s \cdot q_k \mathbf{e}_k)\|^2 \quad (3)$$

$$= \|\mathbf{c} - \hat{\mathbf{c}}\|^2 - s \cdot 2q_k(c_k - \hat{c}_k) + q_k^2 \quad (4)$$

$$= J_0 + q_k^2 - s \cdot 2q_k(c_k - \hat{c}_k) \quad (5)$$

where \mathbf{e}_k is k th column of an identity matrix and the embedding error is extracted as

$$\Delta J = q_k^2 - s \cdot 2q_k(c_k - \hat{c}_k). \quad (6)$$

Since loss of quality cannot be avoided, the first priority is to reduce bit rate such that the loss can be compensated. The savings are based on attempts to remove non-zero coefficients and to create longer zero-coefficient runs for the variable length coding.

If no coefficient can be forced to zero, the second priority is to minimize the quality loss. This is done by minimizing J among the non-zero AC coefficients. The zero coefficients are excluded such that bit rate is not increased by introducing new coefficients and breaking the zero runs. Embedding

to DC coefficient is to be avoided, if possible, since differential coefficient coding is typically used for DC. However, if anything else fails, it is used as the last choice.

The proposed embedding algorithm does one of the following steps in decreasing order of priority:

1. If the total parity of the block equals to data bit, do nothing!
2. Find an AC coefficient c_k with magnitude of one that minimizes ΔJ when $s = -c_k$. Set $c_k = 0$.
3. Find an AC coefficient c_k and modifying direction s such that ΔJ is minimized. Set $c_k = c_k + s$.
4. Select s such that ΔJ is minimized for the DC coefficient c_0 . Set $c_0 = c_0 + s$.

It should be noted that the method tries first to increase the number of zeros in the quantized DCT coefficients, which would be beneficial for VLC and will decrease the bit rate. It should also be noted, that the embedding method does not add any bits to the video stream thus the bit stream is compatible with the original video codec, i.e., the proposed method can be used with standardized video codecs. Audio bit extraction and detection of the proposed method is straightforward operation. In the decoder, the total parity of received block equals to received audio bit. If the total parity of received block is 1 then the audio bit is "1", otherwise the audio bit is "0".

3. EXPERIMENTS

In order to compare the performance of the proposed method, we implemented the methods with C++. First we carried out comparisons with video compression based on 4D DCT compression to compare the performance against the magnitude method from [4]. In this codec, $4 \times 4 \times 3 \times 3$ blocks were used, i.e., the used dimensions were two spatial dimensions (4×4), time, and color.

As 4D DCT codecs are seldom used, we carried out another experiment against more practical codecs. In order to compare the proposed method against standard methods, we used the public domain codec, XviD [5], and we added our embedding method to the codec. For simplicity, pure intra frame coding was used, i.e., predicted frames were not used.

The video data, which was used as stimulus data in the experiments, includes various scenarios from fairly static sequences to highly detailed and fast moving sequences. The low quality (352×288) video set consists of 13 sequences from [6] with average length of 10 seconds each. The medium quality (720×486) set is composed of 10 sequences from [7] with average length of 9 seconds each. All the sequences have the frame rate of 30 frames per second.

The synchronization of the audio and video was verified by video and matching audio sequences from [8] according to

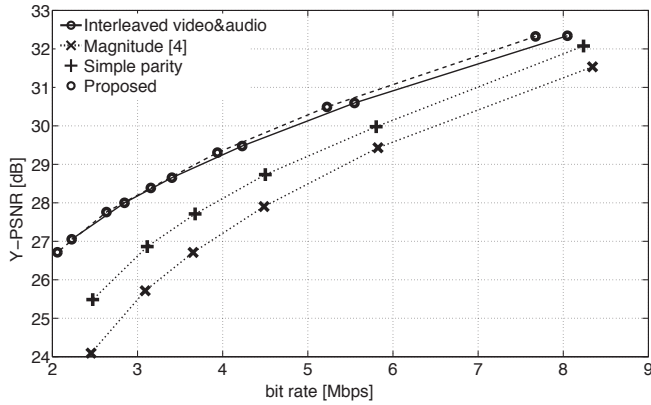


Fig. 2. 4D DCT coding comparison with a single 720x486 sequence.

the examples in Table 1. The performance simulations illustrated here were run at full capacity using non-related audio data. However, compressed audio data could be used as well, e.g., MP3 encoded audio can be embedded without modifications. The performance of embedded cases were evaluated from the single coded video file where the audio was included.

In order to compare the performance for traditional methods, we used audio-video interleaving as the reference, i.e., video signal and audio signal are both encoded with separate encoders and interleaved into a single data stream. However, in the experiment, we omitted the transport protocol, i.e., neither headers nor control bits were included in the comparison and only the payload was counted. In this sense, the reference is optimistic in terms of bit rate. In addition, we did not use audio encoding as the proposed method embeds data bits regardless of the contents. This does not change the results as we use the same audio data also in the reference interleaving.

The video quality was estimated with the aid of PSNR compared to the original video compression. The average bit

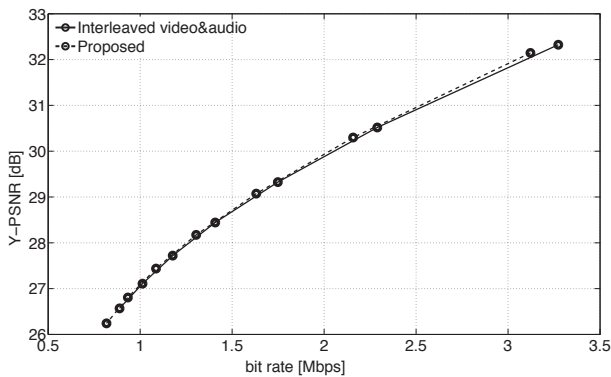


Fig. 3. Averaged results for 13 concatenated video sequences of size 352x288 with 4D DCT coding.

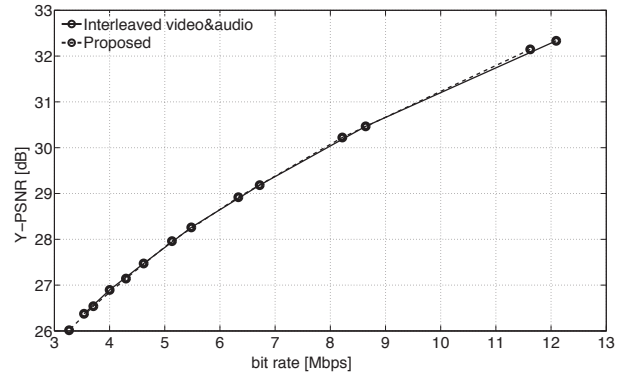


Fig. 4. Averaged results for 720x486 4D DCT coding.

rate was estimated by combining the sizes of the compressed video and the source audio.

In order to estimate the complexity of the proposed method, we compiled the XviD code to ARM Cortex-A8 microprocessor. The XviD code included our method, thus by exploiting the ARM software development tools and profiler, we obtained the clock cycles of the execution.

4. RESULTS

We carried out several simulations to obtain rate-distortion curves for each of the methods. The results are presented with PSNR as the function of the bit rate. The PSNR-rate curves were created by varying the quantization parameters and in the curves, markers show the simulations run with the different quantization parameters.

The first experiment was carried out with 4D DCT coding and the PSNR-rate curves can be seen in Fig. 2. It can be seen that the method based on coefficient magnitude [4] has the lowest PSNR-rate performance. Our first attempt, the simple parity method, provides indeed an improvement but falls far below the practical performance curve, which is represented by the interleaved video and audio. The proposed method shows even improvement compared to the reference method. This is due to the fact that in many of the DCT blocks, there are coefficients with a value of one, which are converted to zeros resulting in reduction in bit rate. Based on the results from this experiment, we did not compare the magnitude method and simple parity methods in other tests.

We compared the performance of the reference method and the proposed method with smaller video sizes. We concatenated 13 sequences with various characteristics of size 352x288 and used the resulting sequence as stimulus. The resulting curves can be seen in Fig. 3. Similar test was carried out with larger video size and the results are shown in Fig. 4. This time 10 different sequences of size 768x486 were concatenated as a single stimulus. The proposed method shows a small coding gain when using 4D DCT. However we have to

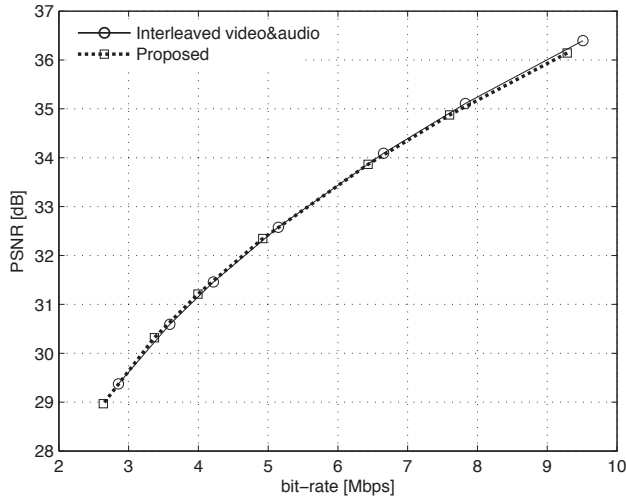


Fig. 5. Averaged results for 10 different videos of size 720×486 with XviD coding.

remember that, in the reference system, protocol overhead is omitted although it would be present in practical systems.

A more practical experiment was carried out by using the XviD codec where 8×8 blocks were used. The average performance is depicted in Fig. 5 where the curves were computed by using stimulus consisting of 10 sequences of size 768×486 . It should be noted that the performance of the proposed method was brought low by a single sequence in the set. However, majority of the sequences followed closely to the reference as shown in Fig. 6.

On the other hand, significant savings were obtained with a rather static 720×486 "News Reader" sequence. The results from this sequence are shown in Fig. 7, which shows clear rate-distortion improvements in lower bit rates.

We compiled the modified XviD C++ code to ARM Cortex-A8 and used profiler to measure the complexity of the added functionality due to the proposed method. The em-

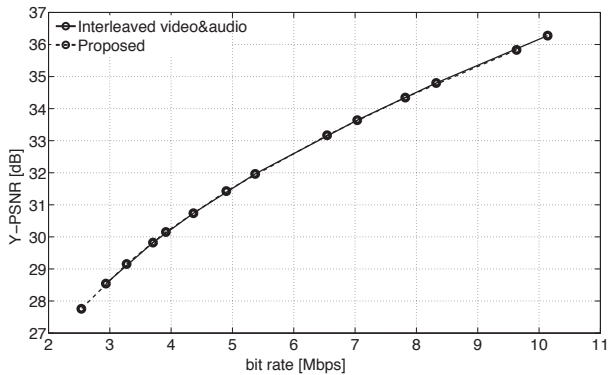


Fig. 6. A typical example of PSNR-rate performance of single 720×486 sequence in XviD codec.

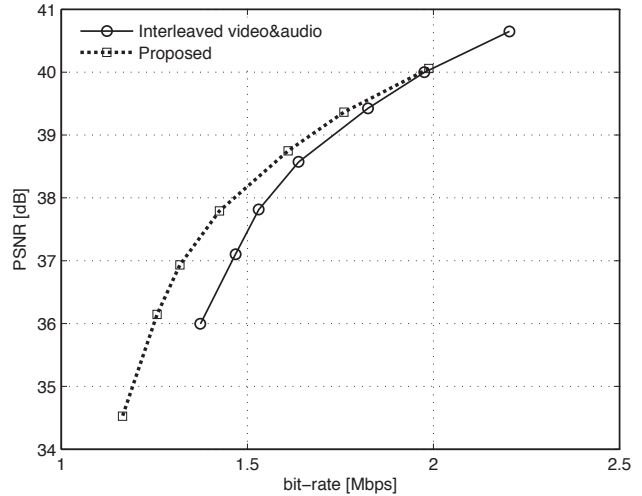


Fig. 7. An example of coding gain in XviD coded video. "News reader" sequence of size 720×486 .

bedding method requires 1190 clock cycles per an 8×8 DCT block, which is a bit less that it is needed for carrying out the quantization; the quantization of an 8×8 DCT block takes ca. 1500 cycles, i.e., 23 instructions per coefficient. Both the functions require the same amount of memory access, but quantization uses four times more writes than the proposed method. It should be noted that the developed code was not optimized and more efficient code could be developed fairly easily. All in all, it can be stated that the overall complexity of the proposed method compared to the entire XviD codec is negligible.

5. CONCLUSIONS

We have addressed the issue of embedding audio data into DCT domain video. A total parity method was proposed for embedding data optimally gaining synchronous playback and parallel audio-video coding. The method provides lossless data transport without any additional bits added to the video bit stream. The simulations show that the proposed embedding scheme can even provide coding gain in rate-distortion performance compared to traditional interleaved audio and video streams. The proposed method can be included in the existing video codecs using DCT transforms and the data stream after embedding is still compatible with the original codec. The proposed method can be used to embedded compressed audio but any kind of digital data can be used. In the future, we are going to extend the method by exploiting also the predicted frames from the video stream. In addition, the method can be extended to embed more audio bits than one per a block of video data.

6. REFERENCES

- [1] ITU-T, “Information technology - Generic coding of moving pictures and associated audio information: Systems,” ITU-T recommendation H.220.0, 2006.
- [2] H. Chen, Y. Zhao, and L. Qi, “Audio-embedded video frequency in audio-video mixed signal synchronous compression and method of extraction,” Chinese patent CN1655616, Aug. 17 2005.
- [3] H. Chen, Y. Zhao, and L. Qi, “Inserted audio-video mixed signal synchronous coding technique,” Chinese patent CN1599464, March 23 2005.
- [4] L. Qi, H. Chen, and Y. Zhao, “New synchronization scheme between audio and video,” in *ACIS Int. Conf. Software Eng., Artificial Intelligence, Networking, Par./Distr. Comput.*, Qingdao, China, July 30 – Aug. 1 2007, vol. 3, pp. 26–29.
- [5] XviD, “XviD web site,” <http://www.xvid.org/>.
- [6] XIPH, “Xiph web site,” <http://media.xiph.org/video/derf/>.
- [7] Video Quality Expert Group, “VQEG ftp test sequences,” <ftp://ftp.crc.ca/crc/vqeg/TestSequences/>.
- [8] Video Quality Expert Group, “VQEG ftp test sequences,” ftp://vqeg.its.bldrdoc.gov/SDTV/ANSI_T1_801_01/.