# A DATA EMBEDDING SCHEME FOR H.263 COMPATIBLE VIDEO CODING

*Jie Song and K. J. R. Liu*

Department of Electrical Engineering and Institute for System Research
University of Maryland at College Park, MD 20742
E-mail:{jsong, kjrliu }@isr.umd.edu

**ABSTRACT**

We propose a novel data embedding scheme for H.263 video coding. The proposed method, which can protect important information such as picture header and motion vector, is specifically suitable to noisy and bandwidth limited channels such as in wireless communications. The method can exactly recover the motion vectors of corrupted macroblocks, therefore has better performance than other error resilient methods.

## 1. INTRODUCTION

Video compression algorithm such as H.263 can compress video sequences efficiently by exploiting both spatial and temporal redundancy. However, the compressed information is very sensitive to transmission errors and losses caused by channel impairment. Furthermore, the error introduced by the channel may propagate to the following video frames because of INTER-frame compensation used in the current video coding standards. The retransmission of damaged frames using Automatic Repeat reQuery(ARQ) is not suitable in real-time video communication. Forward Error Control coding(FEC) may protect the data up to some degree but will waste the limited bandwidth.

Numerous methods have been proposed for error concealment in video communication when some data is lost or corrupted by exploiting the redundancy of video signals [1]. However, even sophisticated concealment techniques cannot totally avoid image degradation, and the accumulation of several small errors can also result in poor image quality. Normally, the image content of a corrupted Group of Macroblocks(GOB) is replaced by simply repeating the GOBs in the previous frame. This method works well for sequences with little motion. However, severe distortions are introduced for image regions containing highly active motion. In [2], one pseudo slice is added to one frame which is an erasure code of other slices in the same frame. Such a method works well to recover one slice of errors, but will increase the bit rate by more than 10 percent. An error tracking combined with feedback channel method is proposed in [3] to deal with bursty errors by sending INTRA mode coded macroblocks to stop error propagation, but the recovery speed depends on round-trip delay of the network, and in a heavy spatial error propagation case, this method cannot send enough INTRA-mode macroblocks in one frame to stop error propagation because of bitrate limitation. In [1], when errors occur at the decoder, the locations of the corrupted macroblock(MB)s are sent to the encoder through feedback channel, and the accumulated backward motion tracks of affected pixels in the following frames are stored. During the period between the errors occur and the arrival of the retransmitted data, any error concealment methods can be used. After the decoder receives the information of the corrupted MB's, the decoding process is redone for the frame with error and lossless recovery is achieved using the previous stored motion tracks. This method does not require INTRA-mode updating to stop the error propagation, but it still cannot provide very good performance when the motion is highly active. Other techniques for error resilient video coding can be found in [4][5].

The method proposed here aims at obtaining good recovery performance. In the proposed method, when errors occur, the video quality is better than any other error concealment methods because of exact motion vector recovery.

## 2. DATA EMBEDDING FOR H.263 VIDEO CODING

Our idea is motivated from information hiding for multimedia, or watermarking[6][7]. In information hiding, the objective is to embed some amount of data into the media such as audio, image or video without visibility and robust to the normal signal modification such as transformation, quantization and compression. Similarly, we want to embed some information into compressed H.263 video bitstream to protect the important video coding information such as motion vector and picture header. The difference between our method and information hiding is that we do not need to *hide* the information embedded.

### 2.1. DATA EMBEDDING AT HALF-PIXEL MOTION ESTIMATION

Now the question is how and where can we embed the data in the H.263 stream? There are several data hiding methods published for video coding such as in [8][9][10]. But these methods either can only embed several bits in one frame or are not compatible with the video coding standard. The method that we propose in this paper is very suitable for video transmission over bandwidth limited noisy channel in terms of better error recovery and channel utilization.

The idea is as follow: In the H.263 encoder, for every INTER mode coded macroblock(MB), an integer-pixel motion vector(MV) is found first by motion estimation as in Fig.1, where the block( with dashed line) at pixel A in the previous frame $K$ is the motion prediction of the current block( with solid line) in frame $K+1$. Then, half-pixel motion prediction is done as in Fig.2( see [5] for the detail of half-pixel motion estimaion). In normal H.263 encoding, the half-pixel prediction is found by looking for the minimum SAD(Sum of Absolute Difference) among half-pixels $1 \sim 8$ and A, as illustrated in Fig.2. In our method, we embed data by changing the half-pixel motion estimation as follow:

If the two bits we want to embed are: $b_n b_{n+1}$, we match $b_n$ to $MV_x$ and $b_{n+1}$ to $MV_y$:

0-7803-5471-0/99/$10.00©1999 IEEE

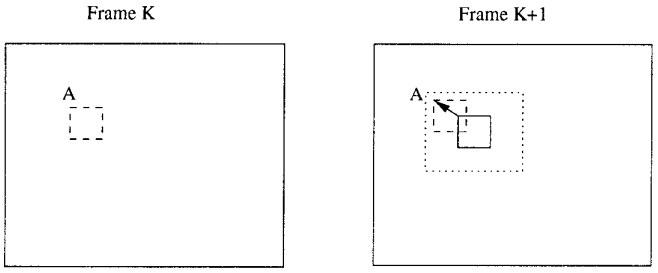



Figure 1: Integer-pixel motion prediction in H.263

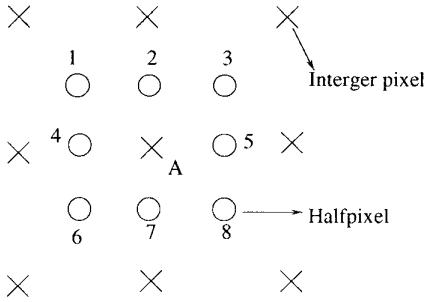



Figure 2: Halfpixel motion prediction in H.263

- if($b_n b_{n+1}$==00) then MV is at interger-pixel position A, no further half-pixel prediction.
- if($b_n b_{n+1}$==01) then MV is selected from half-pixels 2 and 7 with smaller SAD.
- if($b_n b_{n+1}$==10) then MV is selected from half-pixels 4 and 5 with smaller SAD.
- if($b_n b_{n+1}$==11) then MV is selected among half-pixels 1, 3, 6,and 8 with the smallest SAD.

In other words, we find the motion vector not from half-pixel with minimum SAD, but by matching the motion vector components $MV_x$ and $MV_y$ to integer-pixel positions if the bit to embed is '0' or half-pixel positions if the corresponding bit to embed is '1'. The residual block will be DCT transformed and quantized as usual.

For QCIF video of resolution 176×144, there are 99 macroblocks of size 16-by-16 in one frame, which means we can embed at most $99 \times 2 = 198$ bits per frame. In fact, because some MBs in a frame are not coded (where we just use MB in the previous frame at the same spatial location), or INTRA-mode coded, the number of INTER-mode coded macroblocks is less than 99. As a result, the number of bits we can embed in a frame is equal to twice the number of INTER-mode coded macroblocks. The number of INTER-mode coded MBs depends on the motion activity of video sequences, which we will show in simulation section.

Now that we can embed some number of bits in a frame, the question now is what kind of information to embed? Actually, how to efficiently use the information bits for embedding depends on the application environment. In this paper, we protect the important information such as motion-vector(MV) and picture header(PH) (the picture starting code is not protected). This is because if a picture header information is lost, the whole picture cannot be decoded correctly and the error will propagate to the following frames; the motion vector is also very important in video decoding.

| | | | |
|---|---|---|---|
| PH, $(X, Y)_{1,1}$ | $(X,Y)_{1,2}$ | ... | $(X, Y)_{1,11}$ |
| $(X,Y)_{2,1}$ | ... | ... | $(X, Y)_{2,11}$ |
| . | . | . | . |
| $(X, Y)_{9,1}$ | ... | ... | $(X, Y)_{9,11}$ |
| Parity code(modulo-2 sum) | | | |

Figure 3: Motion vector and picture header protection by parity code

The motion vectors are differentially Huffman coded as in the H.263 standard and arranged row by row as in Fig.3. There is one bit prefix for each macroblock to indicate whether it has motion vector(0) or not(1). If a macroblock does not have motion vector, it may be a macroblock that is not coded, which has motion vector $(0, 0)$ and no residual block information; or it is INTRA-mode coded macroblock which has motion vector $(0, 0)$ and the residual block is actually the macroblock itself. The rows of Huffman coded motion vectors are then modulo-2 sum coded as in Fig. 3. Because the lengths of rows are different, the shorter rows are padded with bit '0'. The generated parity code will be embedded in the motion vectors in the following frame. It should be noted that the number of bits that can be embedded in a frame is dynamic and is decided by the number of INTER-mode coded macroblocks in the frame. In case that the number of bits to embed is more than the number of bits one frame can embed, we just extend the data embedding to the next frame.

### 2.2. The ERROR RECOVERY AT DECODER

We assume that at most one GOB in a frame is corrupted when H.263 stream is transmitted over noisy channel. If there is no error, the decoder just works as a standard H.263 decoder.

When one GOB in frame $k$ is corrupted and therefore can not be decoded correctly, the decoder sends a NACK message with the location of this corrupted GOB to encoder. After frame $k+1$ arrives, the decoder can get the parity code of motion vectors in frame $k$ while decoding frame $k+1$ by matching the integer or half-pixel motion vector components $MV_x, MV_y$ to bit '0' or '1', respectively. Then the parity code combined with those Huffman coded MVs of those correctly received GOBs can be used to recover the motion vectors of the corrupted GOB (see Fig.3), finally the corrupted GOB is partially recovered by motion compensation from frame $k-1$, except that the residual blocks of the damaged GOB cannot be recovered at this time.

The problem of error propagation because of the loss of residual block information can be solved by either retransmission of the residual data of corrupted MBs or using the INTRA-mode updating method proposed in [3].

In general, assuming at most one GOB data is corrupted in one frame and the round-trip delay of the network is $D$ frame intervals (we assume no other errors occur during the round-trip period for simplicity), the error recovery scheme works as below:

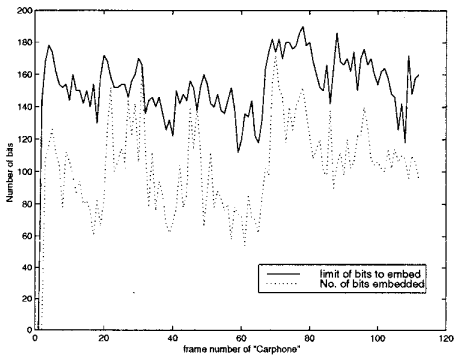
1. When errors occur at frame $k$, the decoder sends a NACK message to encoder with the location of the damaged GOB.
2. The motion vectors of the corrupted GOB can be recovered after frame $k+1$ is decoded using the data embedding scheme described above.
3. The motion compensation proceeds as usual except the residual data of the corrupted GOB is replaced by zero, the reconstructed frame is $f^{'}(n), n = k, k+1, \cdots, k+D-1$.
   - If INTRA-mode updating is used as in [3], then the error propagation will be fully recovered when the frame $k+D$ arrives.
   - If retransmission of residual data is used, the motion vectors and the residual block data in the affected areas from frame $k$ to frame $k+D-1$ are stored at the decoder.
4. After the residual data of the corrupted GOB in frame $k$ arrives with frame $k+D$, the retransmitted residual data is used to recover frame $f(k)$, and the accumulated motion compensations are reproceeded within the affected area to get the losslessly recovered frame $f(k+D-1)$.
5. The losslessly recovered frame $f(k+D-1)$ is used to decode frame $f(k+D)$, and the error propagation is fully stopped.

## 3. SIMULATIONS

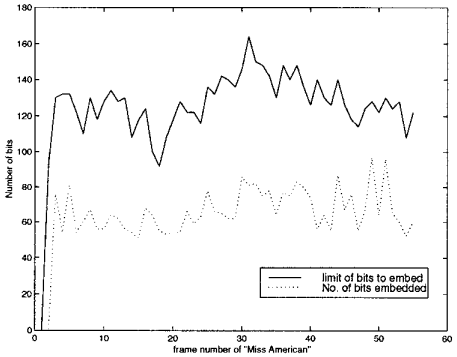
Simulations have been done using the base-mode H.263 on QCIF sequence *Carphone* coded at 10frames/s with bitrate 24kb/s and 48kb/s. In the simulation, no ARQ or INTRA-mode updating method is used for clarity.

Fig.4 shows the number of bits that can be embedded per frame, which is twice the number of INTER-mode coded macroblocks in one frame, and the actual number of parity bits embedded. We can see in most situations the parity bits can be totally embedded in the next frame. If the parity bits cannot totally be embedded in the next frame, we just extend the embedding to the following frame ; in this case, the decoder has to wait for one more frame to arrive to fully recover the motion vectors of the corrupted GOB.

In Fig. 5, the third GOB at frame 27 is corrupted. The image qualities are compared under three cases: (a) error free, (b) exact motion vector recovery using data embedding, and (c) The motion vectors of the corrupted GOB are simply replaced by the motion vectors of macroblocks above this damaged GOB. Even though more advanced motion-vector estimation techniques can be used in case(c), the exact MV recovery using data embedding in case(b) will always be better than MV estimation method in case(c), especially in highly active motion area. The same comparison has also been done at bitrate 48kb/s and the result is the same as that using 24kb/s, which means that the loss of residual block data has less significant effect on error propagation than the loss of motion vector does. Even without the residual data recovery, the exact motion vector recovery is good enough to have a pleasing video quality as shown in Fig.5(b). But the motion vector recovery using estimation from neighbouring macroblocks has poor quality because of the differential motion vector coding used in H.263, i.e., the motion vectors of those correctly received macroblocks following the damaged macroblocks can not be decoded correctly because of differential motion vector coding. Fig.6 is the PSNR comparison of the three cases described above.




(a) Carphone




(b)Miss American

Figure 4: The number of bits that can be embedded and actual number embedded per frame. (a) is from the sequence *carphone* which is highly active in motion, and (b) is from the sequence *Miss America* which has relatively smaller motion area

## 4. CONCLUSIONS

We consider using data embedding in H.263 to deal with error propagation of video transmission over bandwidth limited noisy channels. The proposed method focuses on providing better video quality when errors occur by exactly recovering the motion vectors of damaged GOB using data embedding scheme. The proposed method has advantages in terms of channel utilization, error recovery performance, and compatibility with H.263. One thing to note is that the proposed data embedding method has the trade-off of coding efficiency versus robustness to error propagation when compared with the original H.263 standard.

## 5. REFERENCES


[1] Y. Wang and Q.F. Zhu. *"Error control and concealment for video communication"*, Proc. IEEE, May. 1998, pp. 974-997.

[2] M. Khansari, V. Bhaskaran. *"A low-complexity error-resilient H.263 coder"*, IEEE ICASSP'97, pp. 2737-2740.

[3] E. Steinbach, N. Farber and B. Girod. *"Standard compatible extension of H.263 for robust video transmission in mobile environment"*, IEEE Trans. on CSVT,Vol.7, No.6, Dec. 1997, pp. 872-881.

(a) error free (b) data embedding (c) MV estimation

Figure 5: The reconstructed images of the sequence *Carphone* at decoder are compared when the third GOB is totally corrupted at frame 27. The images shown here correspond to three situations:(a) no error, (b) motion vector recovery using data embedding and (c) motion vector estimation from the macroblocks above the corrupted GOB. The prediction error of the damaged GOB in 27th frame is replaced by zero.


[4] N. Farber, B. Girod and J. Villasenor. *"Extensions of ITU-T Recommendation H.324 for Error-Resilent Video Transmission"*, IEEE Communications Mag., June 1998, pp. 120-128.

[5] ITU-T Rec. H263, *"Version 2, Video Coding for Low bitrate Communication"*,Jan., 1998.

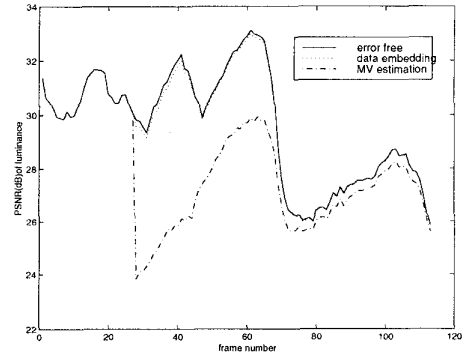
[6] W. Bender, D. Gruhl and N. Morimoto. *"Techniques for data hiding"*, Proceedings of the SPIE, San Jose, CA, February 1991, pp. 2420-2440.

[7] J. R. Smith and B. O. Comiskey. *"Modulation and Information Hiding in Images"*, Proceedings of the First Information Hiding Workshop, Cambridge, U.K., May 1996.

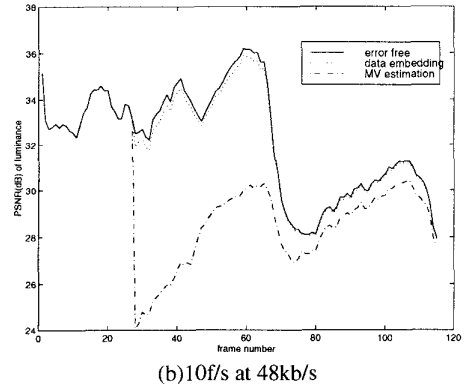
[8] F. Hartung and B. Girod. *"Digital Watermarking of MPEG-2 Coded Video in the Bitstream Domain"*, IEEE ICASSP'97, pp. 2621-2624.

[9] M. D. Swanson, B. Zhu and A. H. Tewfik. *"Data Hiding for Video in Video"*, IEEE ICIP'97, pp. 676-679.

[10] J. Lacy, S.R. Quackenbush, A.R.Reibman, D. Shur and J. H. Snyder. *"On combining watermarking with perceptual coding"*, IEEE ICASSP'98.


(a) 10f/s at 24kb/s




(b)10f/s at 48kb/s

Figure 6: The PSNR comparison at bit rate 24kb/s and 48kb/s, respectively when the 3rd GOB in 27th frame is damaged under three cases: (1) error free (2) exact motion vector recovery using data embedding, and (3) motion vector estimation from macroblocks above the damaged GOB. The prediction error of the damaged GOB is replaced by zero.