An Hybrid Error Concealment Method with Data Hiding for H.264 over Wireless Network

Renn-Hwa Hsieh
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
u881521@alumni.nthu.edu.tw

Long-Wen Chang
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
lchang@cs.nthu.edu.tw

ABSTRACT

With the development of internet, video communication becomes important in human's life. H.264 is a new standard for video coding. It is very suitable for video communication. However, in a network, especially in a wireless network, transmission error is inevitable so that packet loss and bit error occur very often. In this paper, we propose a new hybrid error concealment method with data hiding for H.264. It combines the advantages of temporal error concealment (TEC) and spatial error concealment (SEC) for the loss of macroblocks. Also, we use a data hiding technique to embed the directional information for each 8 x 8 block for directional interpolation. Our method is suitable for the error concealment in H.264 video communication over wireless network. Experimental results of two video sequences called foreman and news show that the proposed method outperforms the conventional TEC and SEC and their hybrid method.

Categories and Subject Descriptors

I.4.2 [**Image Processing and computer vision**]: **Compression (coding) (E.4)**

General Terms:

Security

Keywords: H.264, error concealment, data hiding, wireless network

1. INTRODUCTION

H.264[1,2] is a new standard for video coding. It is very suitable for video communication. However, in a network, especially in a wireless network, transmission error is inevitable so that packet loss and bit error occur very often. In H.264 standard, TEC (temporal error concealment) is only used in a P-frame and it takes use of the dependence of the current frame and its reference frame. In a P-slice, every macroblock (MB) has its motion vector (MV) so that it can reference its relative MB in the reference frame. When the current frame loses a MB (MV is lost), TEC selects a new MV for this lost MB in the reference frame. The best MB in the reference frame will be chosen to substitute the lost MB in the current frame. The algorithm to choose the best MV is called "Boundary Matching Algorithm" (BMA)[3,4].

Similar to TEC, SEC (spatial error concealment) is only used in an I-frame and the idea of SEC is to recover the lost MB by its surrounding MBs in the current frame with an interpolation algorithm. The interpolation algorithm in H.264 is bilinear interpolation (BI) that means to interpolate the lost pixel value in both vertical and horizontal direction. When a lost pixel is being interpolated, the four pixels of the surrounding MBs in vertical and horizontal direction are chosen.

BI is just a simple algorithm for SEC in H.264 standard and it can't recover edges because ecause it handles all kinds of MBs in the same way. No matter what the MB is smooth or complicated, the interpolation methods are all the same. The reconstructed MBs are usually "blurred" and the edges can't be recovered very well.

Many researchers try to improve the interpolation algorithm on SEC. One of the most popular refined algorithms is "Directional Interpolation" (DI) [5, 6]. Unlike BI, DI can interpolate a lost MB in 16 directions so that the reconstructed MB will be more suitable to its surrounding MBs. Because every lost MB can choose the most suitable direction to perform interpolation, better edges will be recovered than BI.

In this paper, we will propose a hybrid algorithm that uses TEC and SEC with data hiding to enhance the performance of error concealment[2-10].

2. Proposed Hybrid method of TEC and SEC with Data Hiding Method

As we mentioned above, SEC is used in an I-frame while TEC is used in a P-frame in the H.264 standard. However, in a P-frame, TEC is not the only choice because the lost MBs can also be reconstructed from their surrounding MBs by SEC algorithm. Besides this, some regions may be constructed by intra prediction (I-slice) in a P-frame and these regions' pixel value may be quite different to their correspondent regions' in the reference frame. We utilize Boundary Matching Algorithm (BMA) to select the error concealment mode for a lost MB to take their respective advantage. The BMA value is calculated from the sum of absolute difference of the outer boundary of the lost MB in the current frame and the inner boundary of the best MB (by "best MV") in the previous frame.

Mobimedia 2008 July 7-9, 2008, Oulu, Finland.
Copyright 2008 ICST ISBN 978-963-9799-25-7
DOI 10.4108/ICST.MOBIMEDIA2008.3917

Suppose a macroblock in the current frame i is lost. There are four candidate MBs in the reference frame that can be found by its 4 direct neighboring macroblocks with motion estimation. These four macroblocks are denoted as MB_{above}, MB_{below}, MB_{right}, and MB_{left}. Based on the 4 candidate macroblocks, we compute 4 values of sum of absolute difference by BMA and find the motion vector of the best matching macroblock as

$$MV_{best} = \min_{MV} \arg\{| F_{i-1}(MV)_{inner} - F_i(MB)_{outer} \}$$

where $MV \in \{MV_{above}, MV_{below}, MV_{right}, MV_{left}\}$.

$F_i(MB)_{outer}$ denotes the outer boundary of the lost MB and $F_{i-1}(MV)_{inner}$ denotes the inner boundary of the candidate MB that is found in the reference frame for the lost MB with motion estimation. The motion vector MV_{above} is the motion vector searched for MB_{above} with the motion estimation and so on. After we find the motion vector for the best matching among the 4 candidate MBs, we compute

$$BMA = | F_{i-1}(MV_{best})_{Inner} - F_i(MB)_{Outer} |,$$

Then, we choose the concealment mode by the BMA value. We set a threshold "t" and use "t" to examine whether "best MV" can make the "best error concealment".

The concealment mode will be "TEC" if the BMA value is smaller than the threshold "t" and there are at least 4 surrounding MBs of lost MB constructed by inter prediction. On the other hand, the concealment mode will be "SEC" if the BMA value is larger than the threshold "t" or there is less than 4 surrounding MBs of lost MB constructed by inter prediction. That is, if a surrounding MB is inter prediction mode Pre-Mode is 1; if it is intra prediction Pre-Mode_is set to 0. That is,

$$Pre\_Mode_i = \begin{cases} 0(intra\ prediction) \\ 1(inter\ prediction) \end{cases} \quad (i = 1...8)$$

The concealment mode can be determined as

$$Conceal\_Mode = \begin{cases} TEC\ if\ (BMA<t \text{ and } \sum_{i=1}^{8} Pre\_mode_i \geq 4) \\ SEC\ if\ (BMA \geq t \text{ or } \sum_{i=1}^{8} Pre\_mode_i < 4) \end{cases}$$

The following shows the step of mode selection:

1. Compute the BMA value from the sum of absolute difference of the outer boundary of the lost MB and the inner boundary of the concealed MB (by "best MV")
2. Compare the BMA value with the threshold value t
3. Choose TEC or SEC according to the result

We will propose three ideas to improve the performance of SEC. The first idea is to select more suitable reference pixels while directional interpolation is performed. The second idea is to divide a 16x16 MB into four 8x8 MBs to perform interpolation, and the third idea is to hide the directions of a MB for interpolation.

A. Select More Suitable Reference Pixels while Directional Interpolation Is Performed

When packet loss occurs during video communication, the whole slice may be lost together. To prevent MBs from being lost continuously, H.264 may use error resilience tools to rearrange the order of MBs, such as flexible microblock ordering (FMO). As Figure 1 shows, the black MBs may be in the same slice with the lost MB when FMO is used so that they may be lost at the same time. So, E1 and E2 may not be available for interpolation while D1 and D2 may be more possibly available.

Thus, we think it is more suitable to limit the reference pixels for directional interpolation to the outer boundary of the lost MB. As Figure 2 shows, the black region is the lost MB and the white region is the outer boundary of the lost MB for directional interpolation.

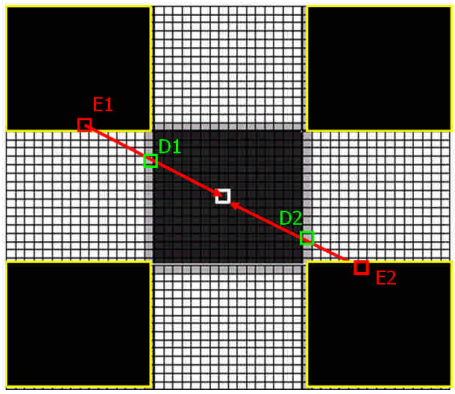


Figure 1: Problem on directional interpolation *(E1 and E2 may be lost with the lost MB)*

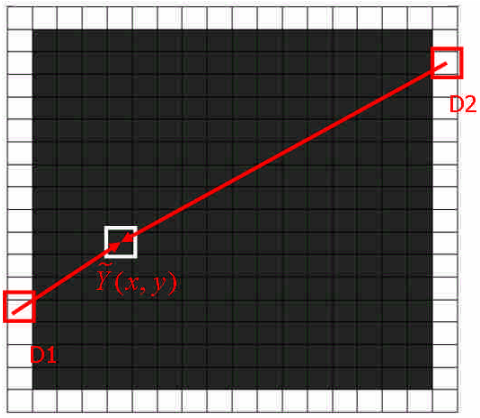


Figure 2: Limit the boundary for directional interpolation

B. Divide a 16x16 MB into four 8x8 MB to Perform Interpolation

The second idea we proposed is to divide a 16x16 MB into four 8x8 MBs to perform interpolation. Although DI can recover edges better than BI, sometimes a 16x16 MB may be "complicated" and may have not only one direction in itself. If we only use one direction to reconstruct a whole 16x16 MB, the performance may be bad in a complicated MB. So, we think it is better to divide a lost 16x16 MB into four 8x8 MBs to perform DI separately.

As Figure 3 shows, Block 0 consults the up-left corner's direction (Region 0), Block 1 consults the up-right corner's direction (Region 1), Block 2 consults the bottom-left corner's direction (Region 2), and Block 3 consults the bottom-right corner's direction (Region 3). Thus, each 8x8 MB has its interpolation direction.

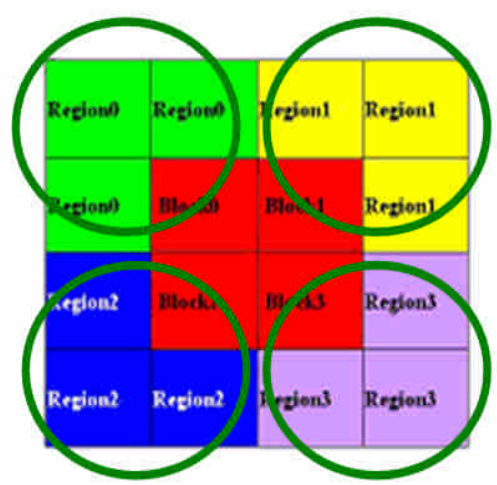


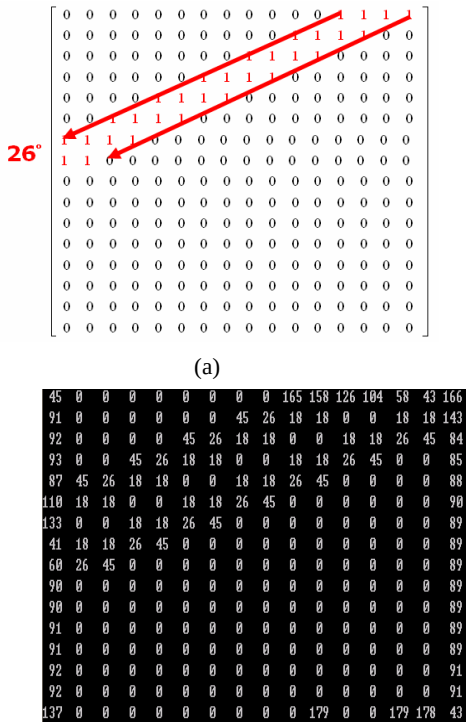
Figure 3: Divide a 16x16 MB into four 8x8 MBs to get their directions

C. Hide The Directions of A MB for Interpolation

The third idea we proposed is to employ data hiding technique and use the hidden data to get the "right" direction for DI. So, the idea we proposed for data hiding is to hide the direction of the MB itself. On the decoder side, when a MB is lost, we can't know the exact direction of the lost MB anymore, so the only way is to evaluate the lost MB's direction from its surrounding MBs. On the encoder side, every MB is still available so that we can calculate each MB's direction and "hide" a MB's "direction" into another MB. Thus, if a MB is lost on the decoder side, we still can find its direction in another MB and use this direction to perform DI. Although there are some overheads for data hiding, we think it is worthy to hide the direction of a MB. The direction evaluate from the lost MB's surrounding MBs may be "wrong", but the direction calculated from the MB itself is always "right." The following is our method on data hiding:

The direction θ calculated by Prewitt filter is not very precise. As Figure 4 shows, (a) is a 16x16 MB with pixel value "0" and "1". We can obviously see that the edge formed by "1" is 26° , but the result we get from Prewitt filter is not as good as we expect. The matrix in (b) is the direction θ of each pixel in the MB, and we can see that the θ may be 18° , 26° , and 45° . The result is not what we want because there is only one direction 26° in our human vision system. We think it is meaningless to define the direction too precisely and we will set 22.5° as a unit.

The data hiding algorithm we used is "even-odd" algorithm. Figure 5 shows a 4x4 DCT block and the data is hidden in the position (1, 1) of the block. If we want to hide "0", set the least significant bit to "even". If we want to hide "1", set the least significant bit to "odd". The meaning of the hidden data is shown on Table 1 where we define 8 directions and it takes 3 bits to hide a direction.



(a)

(b)

Figure 4: An example on Prewitt filter; (a) a 16x16 MB with the direction 26° (b) the direction θ of each pixel

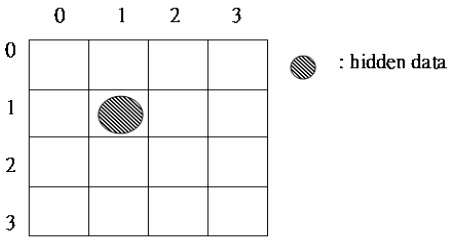


Figure 5: A 4x4 DCT block with the hidden data

Table 1: The information of the hidden data

Data	Direction
000	0°
001	23°
010	45°
011	68°
100	90°
101	113°
110	135°
111	158°

If we hide data in every MB, the overhead may be too large. So we only hide data in the MBs that have edges. We take a 16x16 MB as the unit to determine whether the data should be hidden in it. To examine whether a MB should hide data or not, we set a threshold "T". If the mean square error to the average value of a 16x16 MB is larger than T, we will hide the data of the MB. When a 16x16 MB has to hide data, it takes 12 bits (each 8x8 MB takes 3 bits, and totally takes 3x4=12 bits). In Figure 6, "0" represents the MBs without data hiding, "1" represents the MBs having data hiding.

To further reduce the overhead in data hiding, if the four 8x8 MBs in a 16x16 MB have the same direction, we just hide 3 bits (one direction) in a 16x16 MB. In Figure 7, "0" represents the MBs without data hiding, "1" represents the MBs have to hide data (12 bits), and "1" represents the MBs only have one direction (3 bits).

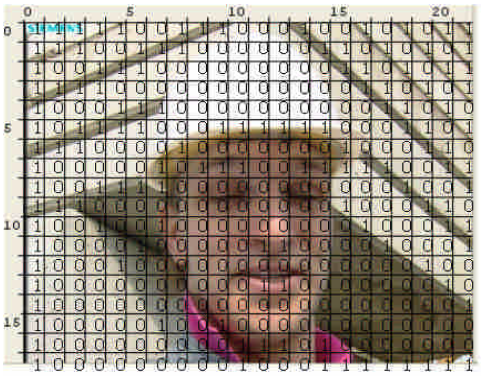


Figure 6: A bit map for each MB (0: repents the MBs without data hiding; black 1: with data hiding—12bits/MB)

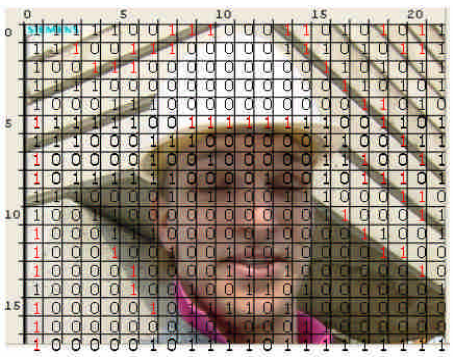


Figure 7: A new bit map for each MB (0: repents the MBs without data hiding; black 1: with data hiding—12bits/MB; red 1: with data hiding—3bits/MB)

As Figure 8 shows, there are sixteen 4x4 DCT block in a 16x16 MB so that a 16x16 MB can hide 16 bits. In Figure 9, the 0 to 15 array represents the 16 positions in a 16x16 that are chosen to hide data. D(0) is used to represent the bit map where "0" means no direction is hidden in the MB while "1" means the directions are hidden in the MB. D(1) is used to represent the direction type of the MB where "0" means single direction while "1" means multi direction. D(2) to D(13) are used to represent the directions in Table 1. If there is only one direction, only D(2) to D(4) will be used. Because there are at most 14 bits to be hidden in a MB, D(14) and D(15) will not be used. According to our experience, the probability of data hiding for a MB is about 37% and the probability of hiding only one direction for a MB is about 12%. A 16x16 MB takes about 4.71 bits overhead for data hiding in average (1 + 1*0.37 + 12*0.25 + 3*0.12= 4.71).

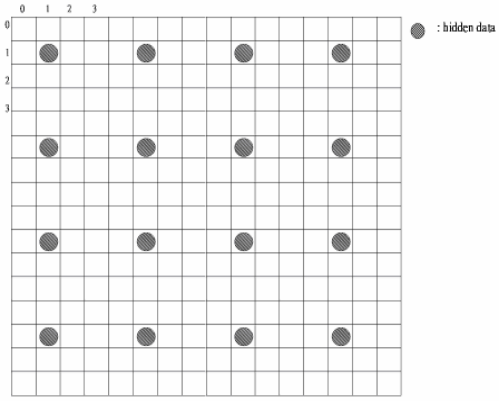


Figure 8: A 16x16 MB in DCT domain

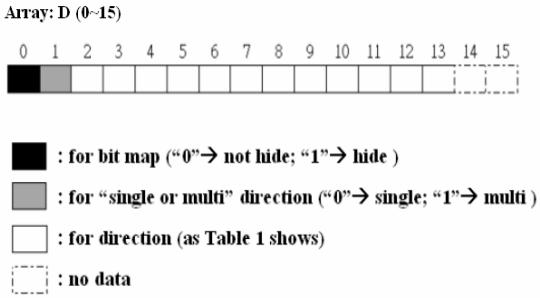


Figure 9: The usage of the hidden data

On the decoder side, if the embedded data is found, we will use the embedded direction to perform DI. Otherwise, we will calculate each 8x8 MB's direction from its neighborhood region. Figure 10 shows the whole process of error concealment under our proposed method and the following is the expression of the whole procedure of the error concealment on the decoder side:

When a 16x16 MB is lost in an I-frame:

(1) Divide the 16x16 MB into four 8x8 MBs.
(2) Get the data hidden in the relative MB of another slice.
(3) If the hidden data is available, go to step (4);
(4) Otherwise, use 8x8 DI to conceal the lost MB.
(5) Decode the hidden data.
(6) Perform the DI according to the direction decoded from step (4) .

When a 16x16 MB is lost in a P-frame:

(1) Calculate the BMA value of the 16x16 MB. .
(2) If the BMA value is larger than threshold "t", go to step (3) Otherwise, use TEC to conceal the lost MB.
(3) Divide the 16x16 MB into four 8x8 MBs.
(4) Get the data hidden in the relative MB of another slice.
(5) If the hidden data is available, go to step (6); Otherwise, use 8x8 DI to conceal the lost MB.
(6) Decode the hidden data.
(7) Perform the DI according to the direction decoded from step (6).

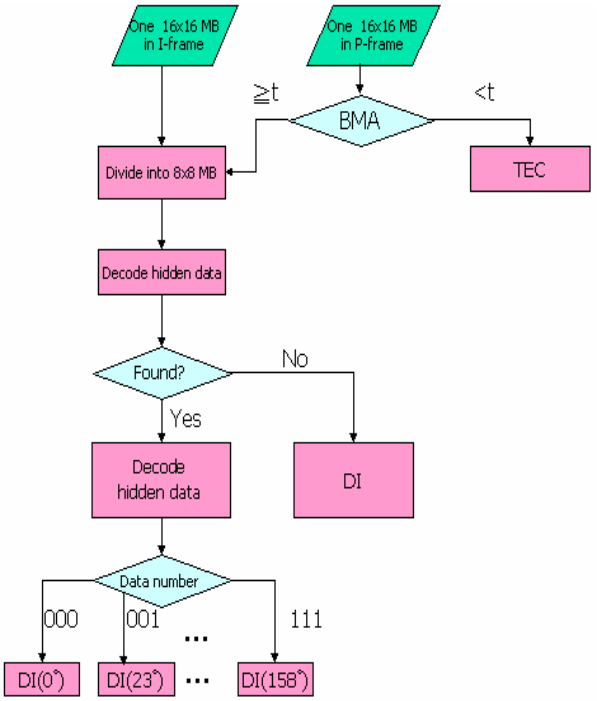
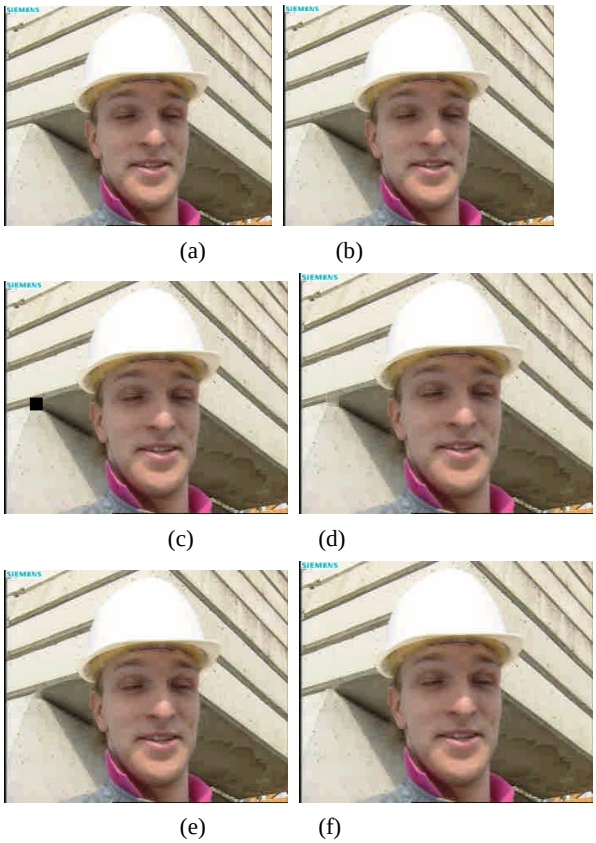


Figure 10: Whole procedure on the decoder

3. Experimental Results

Our simulation is implemented under JM12.2 tool and "foreman" and "news" in "CIF" format are the video sequences we used to test the performance of our proposed method. We used three kinds of QP value (20, 24, and 28) and three kinds of loss ratio (5%, 10%, and 15%) to compare our proposed method with bilinear interpolation and directional interpolation.

We first show two examples on "foreman" and "news" to show that the method we proposed can achieve a better performance in the complicated regions. In both Figure 11 and Figure 12, (a) is the original frame without error, (b) is the frame to be transmitted after data hiding, (c) is the frame received on the decoder side, (d) is the reconstructed frame by bilinear interpolation, (e) is the reconstructed frame by directional interpolation, and (f) is the is the reconstructed frame by our proposed algorithm. Figure 13 is the zoom in image of Figure 11. We can see from (a) that the lost MB has 3 edges in the beginning. Bilinear interpolation can't recover edge so that it looks blurred in the reconstructed MB. Directional interpolation uses only one direction so that the other directions can't be recovered. The performance of our proposed method is much better than other methods.



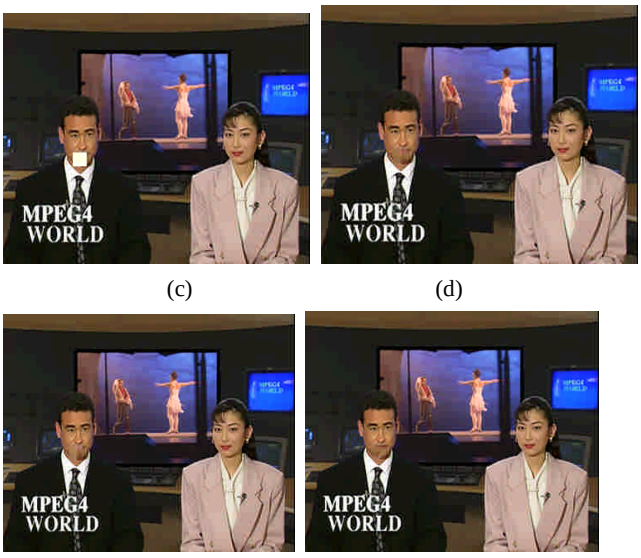
(a) (b)

(c) (d)

(e) (f)

Figure 11: The first frame of foreman (CIF size, QP=28) (a) original; (b) with data hiding; (c) lost; (d) BI; (e) DI; (f) proposed

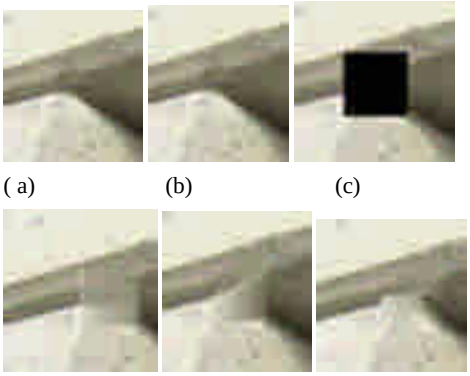
(a) (b)



(c) (d)

(e) (f)

Figure 12: The first frame of news (CIF size, QP=28) (a) original; (b) with data hiding; (c) lost; (d) BI; (e) DI; (f) proposed



(a) (b) (c)

(d) (e) (f)

Figure 13: Zoom in image of the first frame of foreman (CIF size, QP=28);(a) original; (b) with data hiding; (c) lost; (d) BI; (e) DI; (f) proposed

Figure 14 is the zoom in image of Figure 12. We can see from (a) the lost MB has 3 edges in the beginning. BI can't recover edge so that it looks blurred in the reconstructed MB. DI chose the direction 90° which is not the direction of the lost MB so that the performance is bad. Our proposed method is much better than other methods in recovering the shape of the human face.

(a) (b) (c)

(d) (e) (f)

Figure 14: Zoom in image of the first frame of news (CIF size, QP=28) (a) original; (b) with data hiding; (c) lost; (d) BI; (e) DI; (f) proposed

Then, we simulate the environment over wireless network with three loss ratios 5%, 10% and 15%. On Table 2, we can see DI (8x8) is better than DI (16x16) and our proposed method with data hiding is better than DI (8x8). The higher the loss ratio, the more improvement in the PSNR we can achieve. Figure 15 and Figure 16 shows the comparison of the performance for various error concealment algorithms for bit loss. It shows that the proposed method outperforms the conventional TEC and SEC and their hybrid method.

Table 2: PSNR (db) of two sequences under different QP and loss ratios

Test Sequences	QP	Original (dB)	PSNR (dB)	Loss Ratio		
				5%	10%	15%
Foreman	28	37.37	BI	33.80	31.44	30.07
			DI (16x16)	35.17	34.03	32.67
			DI (8x8)	35.80	34.36	33.36
			proposed	36.02	34.84	33.95
	24	39.82	BI	34.64	30.89	30.37
			DI (16x16)	36.94	35.15	33.54
			DI (8x8)	37.17	35.28	34.02
			Proposed	37.41	35.87	34.44
	20	42.59	BI	35.26	32.21	30.55
			DI (16x16)	37.14	35.32	33.75
			DI (8x8)	38.17	35.80	34.21
			Proposed	38.28	36.63	34.99
News	28	38.26	BI	30.30	28.68	27.33
			DI (16x16)	30.99	29.14	27.68
			DI (8x8)	31.35	29.31	27.80
			Proposed	31.63	29.93	28.59
	24	40.98	BI	30.90	29.09	27.62
			DI (16x16)	31.22	29.30	27.77
			DI (8x8)	31.70	29.60	27.94
			Proposed	31.75	30.01	28.54
	20	43.34	BI	31.36	29.55	28.09
			DI (16x16)	32.02	29.98	28.38
			DI (8x8)	32.40	30.21	28.51
			Proposed	32.77	30.89	29.44

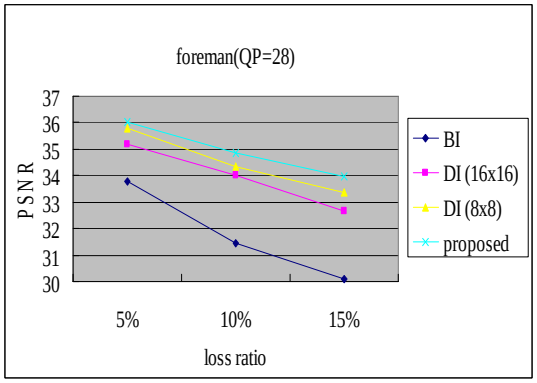
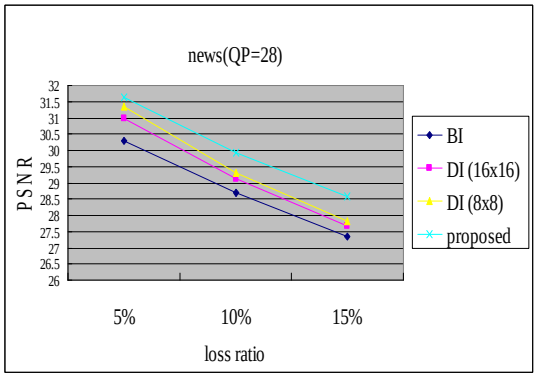


Figure 15: Performance of each error concealment algorithm in foreman *for QP=28;*



(a)

Figure 16: Performance of each error concealment algorithm in news *for QP=28.*

4. Conclusion

In the video communication over wireless network, packet loss is inevitable. Besides this, the devices are usually moving, such as mobile phone and other handheld devices, and the loss ratio will increase with the motion speed of the device. In this paper, our proposed method first shows that it is better to reduce the MB size to execute directional interpolation in the spatial error concealment and it is also can be used in a P-frame if SEC is chosen for the lost macroblock. Our proposed method also shows that the performance is enhanced when data hiding technique is employed for spatial error concealment, especially when the loss ratio is higher. Therefore, it is very suitable to implement our method in H.246 video with error concealment over wireless network.

5. REFERENCES

[1] S. Wenger, "H.264 over IP," IEEE Trans. on Circuits and Systems for Video Technology, ,vol. 13, no. 7, pp. 645- 656, 2003

[2] T. Wiegand, G. J. Sullivan, G. Bjntegaard, A. Luthra, "Overview of the H.264 video coding standard," IEEE Trans. on Circuits and Systems for Video Technology, vol. 13, no. 7, pp. 560- 576, 2003.

[3] D. Agrafiotis, D. R. Bull, C. N. Canagarajah, "Enhanced Error Concealment With Mode Selection," IEEE Trans. on Circuits and Systems for Video Technology , vol. 16, no. 8, pp. 960 – 973, 2006.

[4] Yan Chen,)scar Au, .Chiwang Ho, Jiantao Zhou;, "Spatio-temporal boundary matching algorithm for temporal error" concealment". Proceedings. 2006 IEEE International Symposium on Circuits and Systems, 2006. ISCAS 2006 pp. 21-24 .

[5] Y. L. Xu, Y. H. Zhou, "H.264 video communication based refined error concealment schemes" , IEEE Trans. on Consumer Electronics , vol. 50, no. 4, pp. 1135 – 1141, 2004

[6] Y. L. Xu, Y. H. Zhou, "Refined video error concealment over wireless IP network" , proceedings of the IEEE Circuits and Systems Symposium on Emerging Technologies: Frontiers of Mobile and Wireless Communication , vol. 1, pp. 257 – 260, 2004

[7] X. Xiu, L. Zhuo, L. Shen, "A hybrid error concealment method based on H.264 standard," Proceeding of The 8th International Conference on Signal Processing, Vol. 2, pp. 16-20, 2006

[8] M. J. Chen, W. W. Liao, M. C. Chi, "Robust temporal error concealment for H.264 video decoder," proceedings of Internation conference on Consumer Electronics, 2006. ICCE '06., pp. 383 – 384, 2006.

[9] K. Donghyung, Y. Siyoung, J. Jechang, "A new temporal error concealment method for H.264 using adaptive block sizes", procceding of international conference on image processing, vol. 3, no.3, pp. 928-931, 2005.

[10] Li-Wei Kang, JJin-Jang Leou, "An error resilient coding scheme for H.264 video transmission based on data embedding" , procceding of ICASSP, vol. 3, no. 3, pp. 257-260, 2004.