Scalable Motion Vector Coding for MC-EZBC

Zhiping Hu[1], Mihaela van der Schaar[1] and Beatrice Pesquet-Popescu[2]
zphu@ucdavis.edu, mvanderschaar@ece.ucdavis.edu, pesquet@tsi.enst.fr
[1]Department of Electrical and Computer Engineering, University of California Davis
One Shields Avenue, Davis CA 95616, USA
[2]Signal and Image Processing Department, ENST Paris,
46 rue Barrault, 75634 Paris Cedex 13, France

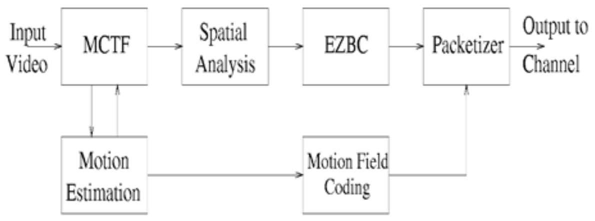
Abstract

For a scalable video coder to remain efficient over a wide range of bitrates, the motion information should be represented in a scalable manner. Indeed, especially at low bitrates tradeoffs should be made between motion information representation and texture information accuracy. Consequently, motion vector scalability has emerged as an important research topic, in conjunction with fully scalable video compression schemes. In this paper, a new scheme providing motion vector scalability is proposed. Starting from the MC-EZBC scalable wavelet video coder [1], multiple motion layers are constructed according to the importance of motion information. The overhead associated with this layered representation is proven to be negligible. Hence, the rate-distortion performance at high bit-rates is about the same as that of the scheme without motion information scalability, while at low bitrates the scheme benefits from this scalable coding strategy.

1. INTRODUCTION

Motion compensated wavelet video coding has emerged as an important research topic, due to its ability to provide easy bitstream adaptation to bandwidth variations and efficient transmission in different Quality-of-Service scenarios. A popular wavelet video coding scheme is MC-EZBC (Motion Compensated Embedded Zero Block Coding) that was proposed by Woods et al [1]. However, their implementation (see Fig.1) encodes the motion information in a non-scalable manner, which results in a reduced coding efficiency performance at low bit-rates as opposed to state of the art non-scalable coding techniques like that provided by the H.264 standard. Different tradeoffs between motion information and texture coding should be made for ensuring optimal rate-distortion (R-D) performance, depending on the bitrate and sequence characteristics.

Figure 1: Basic structure of MC-EZBC



Without some form of motion information scalability, only a very small bit budget can be allocated at low bitrates for texture coding. Hence, layered representations of the motion information are necessary to provide good R-D performance of scalable coders over a large range of bitrates.

In [9] and [2], scalable motion vector coding schemes are proposed, in which the motion vector fields are decomposed so that they can be progressively decoded. In [2] this is related to the 3-Step Search algorithm presented in [3]. In [4] another scalable motion vector coding scheme is proposed by Taubman at el. In this scheme motion parameters are scalably coded using techniques similar to those embodied in the JPEG2000 image compression standard [5], i.e. different quality layers are constructed and the decoder receives and uses only the most appropriate parameter quality layer. In [6], Hang at el [6] split the motion information in the MC-EZBC into a base layer and an enhancement layer, employing different motion vector block sizes.

In this paper, we propose an alternative algorithm for building and coding a layered representation of motion information in the MC-EZBC coder. We define a metric for determining the motion vector importance and then use this metric for constructing different importance motion vector layers.

We describe our scalable motion vector coding scheme in Section 2. In Section 3, experimental results are presented, and in the last section the conclusion and future work are given.

2. SCALABLE MOTION VECTOR CODING

2.1 New Motion Vector Importance Metric

In the motion estimation process of MC-EZBC, a full 5-level motion vector quad-tree is composed for every 64-by-64 block in the image. During the process the hierarchical variable size block matching (HVSBM) motion estimation algorithm is used to speed up the search process. After generating the full motion vector tree, a bottom-up merger [7] is utilized. The motion vector refining process is showed in Fig.2.

In our proposed scheme, the description of motion vector fields is transmitted in the form of multi-quality layers, using the spared bits to better encode the texture coefficients. Now the key problem is how to measure the importance of the motion vectors and thus to determine which motion vectors should be transmitted with a high priority, and how to ensure the encoding performance gain thanks to the spared bit to outperform the suffered performance due to the less accurate motion vector description.

The following measure is proposed for the importance of motion vectors:

$$DBEn = | BE\_F0 - BE\_F1 | \quad (1)$$

where $BE\_F0 = \frac{1}{N}\sum(\text{frame}_1(x,y) - \text{frame}_0(x,y))^2$

$$BE\_F1 = \frac{1}{N}\sum(\text{frame}_1(x,y) - \text{frame}_0(x-dx, y-dy))^2$$

N: number of pixels in a motion block
frame_1: current frame
frame_0: reference frame
dx, dy: motion vector components
BE_F0: quadratic error measure between the block in the current frame and the block *without* displacement in the reference frame
BE_F1: quadratic error measure between the block in the current frame and the matched block in the reference frame
DBEn: difference between BE_F0 and BE_F1 for the *n*-th block.

BE_F1 represents the incurred block error when the motion vector is adopted, while BE_F0 quantifies the block error when no motion vector is used. Thus, *DBEn* measures the incurred distortion due to the inaccurate motion representation. We use *DBEn* for classifying the motion vector *n* in different priority layers. The motion vector *n* is called significant if its *DBEn* value exceeds a predetermined threshold. Otherwise, it is called insignificant and it will not be transmitted.

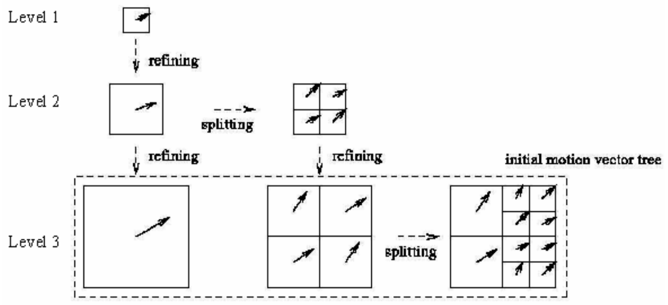


Figure 2: HVSBM algorithm detailed over 3 levels

To ensure encoding performance gains due to this layered motion vector representation, optimal motion information truncation points should be determined for different bit rates Rn at a specific spatial resolution: determine the truncation points n such that

$$\min D_{avn}(R=R_{MVn}+R_{textn}<=R_{avn}) \quad (2)$$

where R_{avn} and D_{avn} are the total available rate and distortion for a GOP, and the R_{MVn} and R_{textn} are the rate budgets for the motion vector field and texture coefficients. However, the coding distortion of a block cannot be directly obtained at the motion estimation stage, since the overall distortion will be available only after the motion information is available to perform the spatio-temporal subband analysis.

Thus, instead of solving the rate-distortion optimization problem of (2) in one step, the equation (3) is used to approximately solve the optimization problem indirectly:

$$\min (\Delta D_{mvn}(\Delta R_{mvn})+\Delta D_{textn}(\Delta R_{mvn})) \quad (3)$$

where the ΔD_{mvn} represents the suffered quality due to the motion information loss, ΔD_{textn} represents the distortion reduction in texture data, and ΔR_{mvn} designates the number of bits saved in the motion vector field. In the sequel, we eliminate for simplicity the subscript *n* in our notation.

To ensure an approximate solution for (3), two other parameters, Saved_MV and Suffered_PSNR are introduced in our scheme. Saved_MV is the upper bound of the lost motion vectors, which is measured by the lost motion vector percentage and Suffered_PSNR is the maximum permitted value of $\Delta D_{mv}(\Delta R_{mv})$ in (3) due to the motion information loss during the process of motion vector coding scan and its value should be less than $\Delta D_{text}(\Delta R_{mv})$ in (3) when ΔR_{mv} reaches its maximum value. In practice it is hard to get the exact value ΔR_{mv} during the process of motion vector coding. We approximately use the lost motion vector percentage to measure the value $\Delta R_{mv,}$ since the ΔR_{mv} is approximately proportional to the number of saved motion vectors.

2.2. Motion Vector Bitstream Configuration

Fig.3 depicts the employed motion information configuration. We use a GOP size of 16 and the number of temporal decomposition levels is 4, so there are 15 motion vector fields in one GOP.

Fig.4 represents the global motion vector bitstream layer configuration. The multi-layers are formed by motion vector coding scans with different parameter pairs of Saved_MV and Suffered_PSNR. The specific steps are described in the next section. Theoretically, any number of bitstream layers can be formed. In practice, we just form several layers and each layer is associated to a range of bitrates. This is motivated by the fact that the motion vector bitrate is negligible at high bitrates and thus the performance gain of the scalable strategy disappears.

The bitstream header consists of two parts (see Fig. 4): the MAP and the Layer Choose Marker. The MAP offers the motion vector quad tree information and the Layer Choose Marker offers the layers bit rate information to facilitate the encoder to select appropriate layers in different bitrate applications. For example, if the available bit rate is 128kbps, then the encoder will compare 128 with the Layer Choose Marker of the first layer: if the Layer Choose Marker is bigger than 128, this layer will be chosen to be transmitted; if not, the next Layer Choose Marker will be compared with 128 and the encoder will determine whether this layer should be transmitted till the Layer Choose Marker is bigger than 128. The value of the Layer Choose Marker is the upper limit of the bit range of this layer which Layer Choose Marker belongs to. Note that the Layer Choose Marker is the only additional overhead compared to the MC-EZBC coder without motion information scalability.

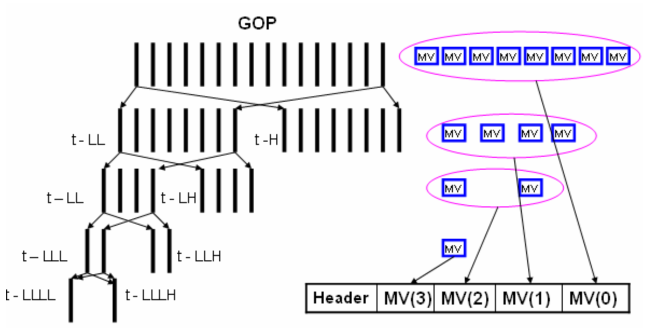


Figure 3. The basic motion information configuration in one GOP.

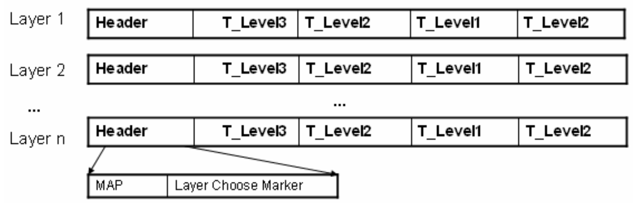


Figure 4. Motion vector global layer configuration.

2.3. Coding Steps

```
(1)   HVSBM motion estimation, quad-tree leaf
      pruning [7]; calculate DBEn values for each
      motion vector block using Eq. (1).
(2)   Initialization
      Threshold = 1;
      Saved_MV and Suffered_PSNR=the value for
      the highest temporal level of layer1.
(3)   Motion vector coding scan.
   for (i = 1; i <= the number of the layers; i++)
   {    for( j=1; j<=the number of temporal levels; j++ )
        {    while(saved_mv< Saved_MV &&
                  suffered_psnr < Suffered_PSNR)
             {
                  if(DBEn< Threshold )
                  {
   //this block is insignificant and does not need to be transmitted
                       mvx = 0;
                       mvy = 0;
                  }
   // Update the threshold and begin the next scan
                  Threshold++;
             }
             Update Saved_MV and Suffered_PSNR for the next
                  lower temporal level;
        }
        Update Saved_MV and Suffered_PSNR for the
             next layer;
   }
(4)   Entropy coding of each layer.
```

The difference between adjacent motion vectors is encoded using an arithmetic coding [8].

3. EXPERIMENTAL RESULTS

Four layers are considered in our simulations (layer1~layer4 for 150kbps ~ 500kbps). Each layer spans 50kbps bitrate except layer1, used under 150kbps.

The results in Fig.5 and Fig.7 illustrate the fact that the proposed scheme outperforms the original MC-EZBC substantially at low bitrates. At high bitrates (500kbps – see Fig.6 and 8), for the "Foreman" sequence there is no performance loss, while for the "Coastguard" there is only 0.01dB loss due to the overhead of the scheme. Tab.1 and 2 present more precisely the gains of the scalable scheme. In Fig. 9 and 10 we present the PSNR results for bitrates ranging from 150kbps to 500kbps, using two typical video sequences Foreman and Coastguard (CIF, 30 fps).

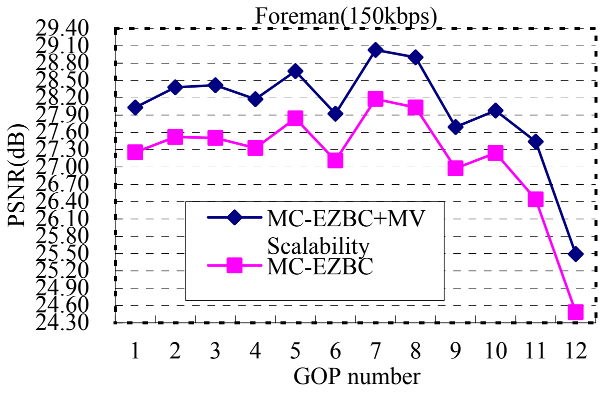


Figure 5: PSNR vs. GOP number of the CIF Foreman sequence at bit-rate 150kbps

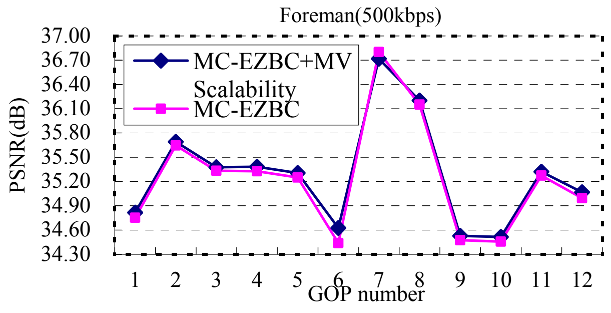


Figure 6: PSNR vs. GOP number of the CIF Foreman sequence at bit-rate 500kbps

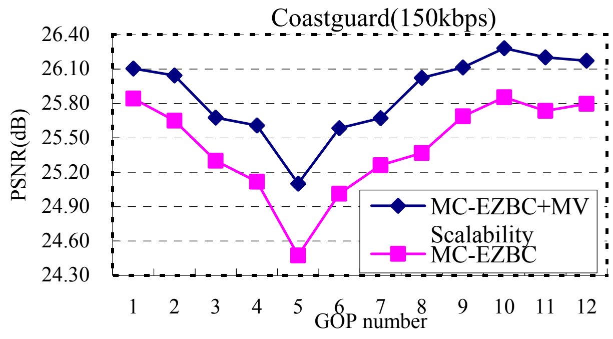


Figure 7: PSNR vs. GOP number of the CIF Coastguard sequence at bit-rate 150kbps

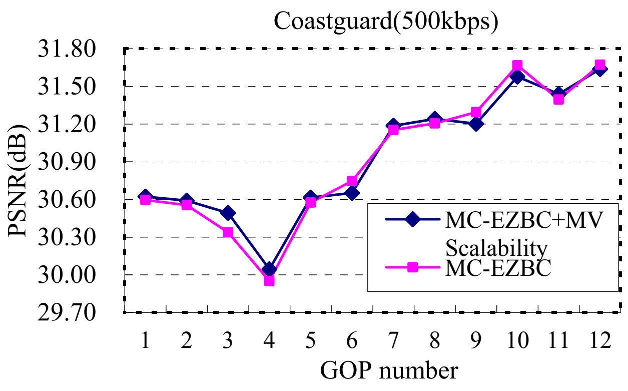


Figure 8: PSNR vs. GOP number of the CIF Coastguard sequence at bit-rate 500kbps

	Rate(kbps)	MV bits	PSNR
MC-EZBC+	150	704952	28.01
MC-EZBC	150	753032	27.16
MC-EZBC+	500	729862	35.26
MC-EZBC	500	753032	35.24

Table 1. Overall PSNR results for Foreman (12 GOPs, bitrates of 150kbps and 500kbps)

	Rate(kbps)	MV bits	PSNR
MC-EZBC+	150	638140	25.85
MC-EZBC	150	675984	25.42
MC-EZBC+	500	656308	30.92
MC-EZBC	500	675984	30.93

Table 2. Overall PSNR results for Coastguard (12 GOPs, bitrates of 150kbps and 500kbps)

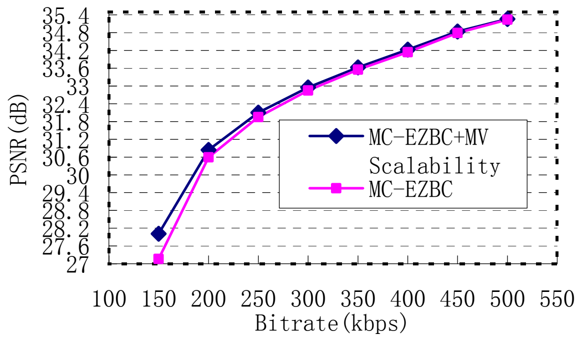


Figure 9: R-D curve for the CIF Foreman sequence

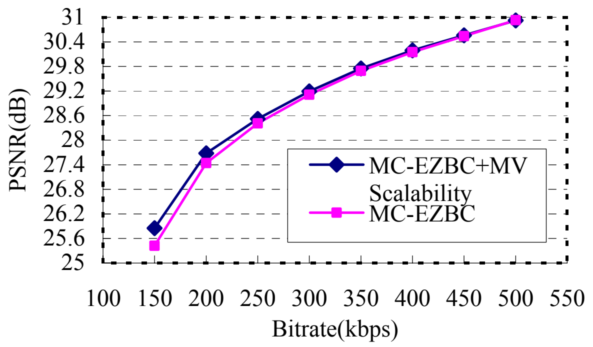


Figure 10: R-D curve for the CIF Coastguard sequence

4. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a new algorithm to measure the importance of motion vectors and based on this, we have built multiple motion information layers that introduce motion information scalability in the MC-EZBC algorithm. With our scheme, a better rate bit allocation tradeoff can be achieved between the motion vector field and the texture coding. Improved performance of this scheme can be especially observed at low bit-rates.

In the future, we plan to improve our algorithm by also considering the texture characteristics. Joint tradeoffs between the accuracy of motion information and texture coding representation could then be performed at a given bit-rate.

5. REFERENCES

[1]. J.W. Woods and P.S. Chen, "Improved MC-EZBC with Quarter-pixel Motion Vectors", ISO/IEC/JTC1 SC29/WG11 doc no. m8366, Fairfax, May 2002.

[2]. V. Valentin, M. Cagnazzo, M. Antonini, M. Barlaud, "Scalable Context-Based Motion Vector Coding for Video Compression", Proc. PCS, St. Malo, France, April 2003.

[3] T. Koga, K. Iinuma, A. hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," in Proceedings NTC, IEEE, 1981.

[4] D. Taubman and A. Secker, "Highly Scalable Video Compression with Scalable Motion Coding", Proc. ICIP, Barcelona, Spain, Sept. 2003.

[5]. D. Taubman and M. Marcellin, "JPEG2000: Image Compression Fundamentals, Standards and Practice." Boston: Kluwer Academic Publishers, 2002.

[6]. H.-M. Hang, S.S. Tsai, and T. Chiang, "Motion Information Scalability for MC-EZBC: Response to Call for Evidence on Scalable Video Coding", ISO/IEC JTC1/SC29/WG11 MPEG July 2003, Trondheim.

[7]. S.-J. Choi and J.W. Woods, "Motion-Compensated 3D Subband Coding of Video," *IEEE Trans. on Image Processing*, vol. 8, no. 2, pp. 155-167, Feb. 1999.

[8]. I. H. Witten, R. M. Neal, and J. G. Cleary, "Arithmetic Coding for Data Compression", Comm. of the ACM, June 1987, Vol. 30, No. 6.

[9]. V. Bottreau, M. Bénetière, B. Felts and B. Pesquet - Popescu, *A Fully Scalable 3D Subband Video Codec*, Proc. ICIP, Thessaloniki, Greece, Oct. 2001