

# Decoder side information generation techniques in Wyner-Ziv video coding: a review

Yuan Jia · Yangli Wang · Rui Song · Jiandong Li

Published online: 10 October 2013  
© Springer Science+Business Media New York 2013

**Abstract** In Wyner-Ziv (WZ) video coding, decoder side information (SI) takes a key role in a WZ video codec among other building blocks. In this paper, we review the decoder SI generation techniques according to the information utilized in the SI generation process. Specifically, motion compensated extrapolation based approaches, motion compensated interpolation based approaches, and hash-based decoder motion estimation approaches in which auxiliary information is sent by the encoder side. We also review the approaches for decoder SI refinement. The review will help us get insight in the current development of the topic and understand how new and better decoder SI generation approaches can be designed.

**Keywords** Wyner-Ziv video coding · Slepian-Wolf theorem · Side information · Motion compensated extrapolation · Motion compensated interpolation · Decoder motion estimation

## 1 Introduction

With the emergence of new applications, such as wireless video surveillance and sensor networks, where low complexity encoders are a must, low complexity video encoding has attracted much research efforts. As one of such promising techniques, Wyner-Ziv (WZ) video coding, or coding of video with decoder side information (SI), has been intensely studied in the past decade.

The theoretic foundation of WZ video coding is laid by the Slepian-Wolf (SW) theorem [49] for lossless distributed source coding, and the WZ rate distortion function [59] for source coding with decoder SI. The SW theorem states that for lossless compression of two correlated information sources  $X$  and  $Y$ , the joint

---

Y. Jia (✉) · Y. Wang · R. Song · J. Li  
State Key Lab. of ISN, Xidian University, Xi'an, Shaanxi 710071, China  
e-mail: jiyuan@xidian.edu.cn

entropy  $H(X, Y)$  can still be achieved even when the two sources are separately encoded, but are jointly decoded. And WZ rate distortion function states that, the rate distortion function of  $X$ ,  $R^*(D)$  when SI  $Y$  is only available at the decoder, and that of  $X$ ,  $R_{X|Y}(D)$  when SI  $Y$  is available at both the encoder and decoder satisfy  $R^*(D) \geq R_{X|Y}(D)$ , which indicates that there is generally a rate loss for scenarios where SI  $Y$  is not available at the encoder. But in certain cases, e.g., Gaussian sources with mean squared error distortion metric, the equality can be achieved. The two theorems suggest that low complexity encoding, with possibility a high complexity decoder is possible. Even though the theoretic foundation was established in as early as the middle seventies of last century and Wyner pointed out that SW coding is closely related to channel coding [58], due to the lack of powerful channel codes, practical SW coding and WZ coding was only possible after Pradham and Ramchandran published their pioneering work in 1999 [44]. After that, different WZ video coding approaches have emerged based on SW coding using more powerful channel codes such as turbo codes and low density parity check (LDPC) codes [3, 52]. For the background and development of WZ video coding, the readers are referred to [22, 45, 60], and also to [23] for more recent development in this field.

In WZ video coding, among other building blocks, such as SW coding, correlation channel modeling, and etc., decoder SI plays a key role. As it is involved not only in the decoding process but also in the decoder reconstruction process, as a result, it affects not only the bits required for correctly decoding, and hence the efficiency of a WZ video codec, but also the quality of the reconstructed video at decoder. The decoder SI  $Y$  here means any information available at the decoder when a compressed source frame  $X$  is to be decoded. More concretely, decoder SI is a frame (or its transform coefficients) which is generated at the decoder and can be taken as a corrupted version of source frame  $X$  over a virtual correlation channel. Obviously, decoder SI  $Y$  can be estimated with already decoded intra (also called key frames) or inter frames (also called WZ frames) exploiting the spatial or temporal correlation which exists between neighboring pixels within a frame or among temporally adjacent frames. As in popular WZ video coding frameworks, decoding is generally carried out on a bit plane to bit plane basis, so the bit plane(s) already decoded can also be used directly as SI or used in a so called decoder SI refinement process. To date, there are lots of approaches proposed in the literature for SI estimation. According information utilized, basically, they can be classified into the following categories: Motion compensated extrapolation (MCE), motion compensated interpolation (MCI) and hash based decoder motion estimation. Although there is some literature such as [22, 45] which reviews the techniques in WZ video coding, decoder SI estimation is never treated in detail. We will review in this paper the decoder SI generation approaches in the categories we have classified, We will also review approaches for decoder SI refinement (SIR). From which, we can get insight in the current development of the field, and in how new and better decoder SI generation approaches can be designed. Note, as we have indicated in the above, decoder SI can also be generated by exploiting the spatial correlation among neighboring pixels in a video frame, and indeed, there are approaches in this category, for example, [24, 50, 61], and so forth. We mainly focus on the above classified approaches in this paper. In Section 2, we describe a generic WZ video codec. In Section 3, MCE based approaches are reviewed. Following that, MCI based methods are presented in great detail in Section 4. In Section 5,

hash based decoder motion estimation methods are described. In Section 6, various decoder SI refinement strategies are presented. We conclude the paper in Section 7.

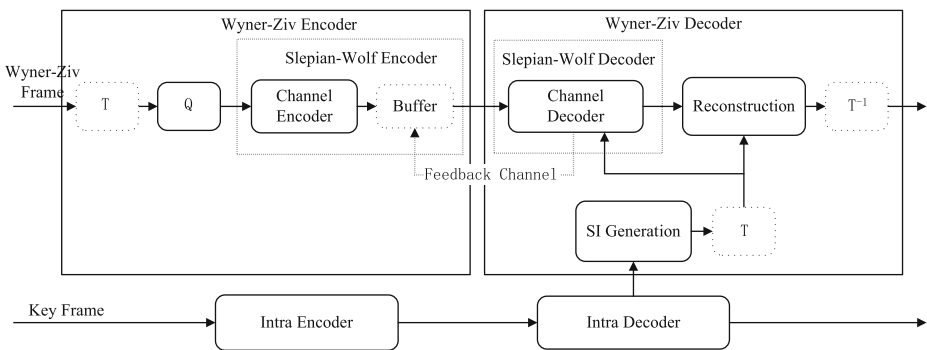
## 2 Generic WZ video codec

Figure 1 shows a general WZ video codec. The central is a channel code based Slepian-Wolf (SW) codec. The modules denoted in the figure as  $T$  and  $T^{-1}$  are respectively forward and inverse transform modules, as such the codec works in transform domain.  $Q$  represents the quantization module controlling the distortion and rate of the system. The SI generation module is responsible for generating SI at the decoder side, which is the focus of this review. The reconstruction module reconstructs the WZ frame using SI and channel decoded quantized source.

The SW decoder is composed of a channel decoder, while the SW encoder consists of a channel encoder and a buffer. Generally, systematic codes are used in SW coding.

When coding, a video sequence is divided into key frames and WZ frames, a key frame is intra encoded and decoded with a conventional video codec, for example H.264, as shown in the lower part of the figure. A WZ frame is coded after transform and quantization with a systematic channel code. The systematic bits of the channel encoder are discarded, while the output parity bits from the channel encoder go through a puncturing process to control the rate. The punctured parity bits are then temporally stored in a buffer and are sent to the decoder upon decoder request.

At the decoder, SI  $Y_t$  is first generated for a WZ frame  $X_t$  through techniques which are reviewed in the sections that follow, using already decoded adjacent frames. With SI and an assumed model for the correlation between the source and its SI, channel decoding is carried out to recover the quantized source. The channel decoding can work in an iterative way. Specifically, the channel decoder requests a chunk of parity bits a time and performs decoding. If the decoded source is “good” enough, it will terminate the process, otherwise, it will requests more parity bits from the encoder. The request-decode process is repeated till a predefined fidelity is reached. Reconstruction of the WZ frame is then carried out with the decoded quantized source and SI.



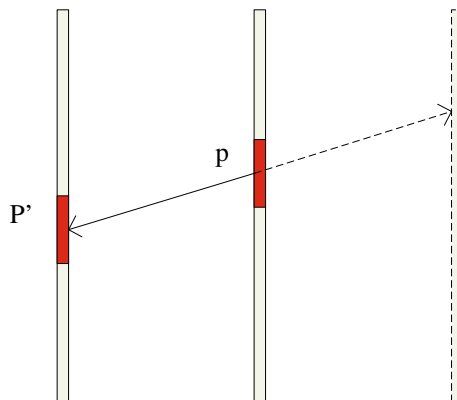
**Fig. 1** Illustration of a generic WZ video codec

Note, when the dash-line marked transform modules are dropped from Fig. 1, it turns into a pixel domain WZ video codec. When the feedback channel is removed, it becomes an un-directional WZ video codec, which is suitable for low complexity and low delay applications. In such a scenario, MCE based SI generation will be more often used. In addition, mode decision as well as other functions can also be added to the codec, which are not shown in the figure for clarity. Hash generation, SI refinement and etc., which are covered in this paper, are also not shown in the figure.

### 3 MCE based decoder SI estimation

In MCE based methods, SI for the current WZ frame is estimated using two past decoded frames  $\hat{X}_{t-1}$  and  $\hat{X}_{t-2}$ . As the current WZ frame is not available, motion field for the current WZ frame is deduced from those in the previous decoded frame,  $\hat{X}_{t-1}$ , may it be a key frame or a WZ frame, assuming a linear motion model, and SI for the current WZ frame is then obtained with the deduced motion vectors and  $\hat{X}_{t-1}$  and/or  $\hat{X}_{t-2}$ . In [41], such an SI estimation method was proposed. In the method,  $\hat{X}_{t-1}$  is first divided into  $m \times m$  (with  $m$  being a constant and usually taking 8 or 16) blocks. A motion vector is then searched in  $\hat{X}_{t-2}$  using superposition of the blocks. Next, motion field smoothing is performed by averaging all neighboring motion vectors around a block. This helps remove outlier motion vectors. As there is no information available for the current WZ frame, so a linear motion model has to be assumed, pixels in  $\hat{X}_{t-1}$  are projected to the current WZ frame using the motion field obtained and the block based linear motion model. Figure 2 illustrates the projection process, where a block in  $\hat{X}_{t-1}$  centered at  $\mathbf{p}$ , with a matched block in  $\hat{X}_{t-2}$ , centered at  $\mathbf{p}'$ , is forward projected in the current WZ frame. Finally, a post processing process is carried out to handle multi-covered and uncovered pixels in the current WZ frame. If a pixel is estimated by more than one pixel in the previous frame, an average of the values is taken as the estimation for the pixel. All positions in the current WZ frame which has no estimation are then filled with spatial extrapolation using pixel values from left, up and up-left neighbors, by scanning the current WZ frame from top to bottom and left to right. The extrapolation process is iterated till every pixel position has estimation. As linear motion only holds for video sequences with slow motion,

**Fig. 2** Motion estimation and projection



this method can only give reasonable results for those kind of video sequences, but it does have low delay. In [62], an autoregressive model is applied to improve the SI quality in MCE based SI estimation. The motion vector,  $\mathbf{u}_{t-1,t-2}(k, l)$  is first found for a block  $B_{t-1}(k, l)$  in  $\hat{X}_{t-1}$  with reference to  $\hat{X}_{t-2}$ . Denote the matching block by  $B_{t-2}(\tilde{k}, \tilde{l})$ . Assume each pixel in  $B_{t-1}(k, l)$  is approximated as a linear combination of a square spatial neighborhood in block  $B_{t-2}(\tilde{k}, \tilde{l})$ , centered on the corresponding pixel pointed by the motion vector  $\mathbf{u}_{t-1,t-2}(k, l)$ . In other words, each pixel in  $B_{t-1}(k, l)$  can be interpolated as

$$\hat{X}_{t-1}(m, n) = \sum_{-r \leq (i, j) \leq r} \alpha_{i, j} \cdot \hat{X}_{t-2}(\tilde{m} + i, \tilde{n} + j) + g(m, n) \quad (1)$$

Here  $\alpha_{i, j}$  is the forward auto-regressive coefficient, and is assumed constant for all pixels in block  $B_{t-1}(k, l)$ ,  $\hat{X}_{t-1}(m, n)$  is the estimation for pixel located at  $(m, n)$  in the previous frame,  $(\tilde{m}, \tilde{n})$  represents the corresponding position in frame  $\hat{X}_{t-2}$  pointed by motion vector of  $B_{t-1}(k, l)$ , and  $g(m, n)$  is the additive Gaussian noise.  $r$  is defined to be the radius of the AR filter. The forward AR coefficient  $\alpha_{i, j}$  can be estimated using the two previous frames and the motion vector for  $B_{t-1}(k, l)$  using least mean square algorithm.

By assuming that  $B_t(k, l)$  and its co-located block  $B_{t-1}(k, l)$  has the same AR coefficient, and same motion vector,  $B_t(k, l)$  can then be estimated with the AR coefficient, the motion vector, and the previous frame.

In the method, another AR coefficient called backward AR coefficient is also estimated from  $B_{t-2}(\tilde{k}, \tilde{l})$  with respect to  $B_{t-1}(k, l)$ . Another forward AR coefficient is deduced from the backward AR coefficient, and the average of output of the two AR estimation is taken as the SI for the current WZ frame.

As there are more than one pixels involved in the extrapolation of a pixel in the current WZ frame, the method greatly improves the quality of SI. An improvement of up to 3.5 dB in the quality of SI is reported compared with the previous method.

As already seen from preceding description, MCE based SI estimation methods only use past temporal adjacent frames of a WZ frame, this makes such methods low delay. On the other hand, as the motion vectors estimated with only past temporal frames are less reliable, so MCE based SI estimation methods generally achieve only poor performance.

#### 4 MCI based decoder SI estimation

In MCI based methods, SI for the current WZ frame is estimated using one past and one future frame which is already decoded. Different from MCE based methods, the two frames in MCI based methods are generally key frames. To keep the correlation as high as possible, a key-WZ-key frame structure is typically used. This structure is very like the I (P)-B-I (P) structure in conventional video coding. The key to MCI based SI estimation methods is how to determine the motion field between the current WZ frame and the past and future reference frame. The methods in this category can be differentiated by (1) the motion model used, and (2) the motion estimation strategy. As in conventional video coding, block based translational motion model is mostly applied in MCI based SI estimation. Also found are

pel-recursive based motion model, mesh based affine motion model, and higher order motion model. 3D mesh based model is also considered in a paper. The strategies applied for motion estimation include direct motion vector, bi-directional direct motion vector; symmetric motion estimation, reliable motion estimation and true motion estimation and so forth. The situation of MCI based SI estimation is also very like that of frame rate up conversion in conventional video coding, which has already been extensively explored, for example, [17, 29, 31, 54]. In the subsections that follow, various MCI based SI estimation methods are reviewed according to their applied motion model.

### 4.1 Block based translational motion

#### 4.1.1 Direct motion

As in conventional video coding, a very simple SI estimation method [33, 34] is using direct motion vectors, so they can be called direct motion vector based methods. In the first configuration as illustrated in Fig. 3, for each block  $B_t(k, l)$  in the current WZ frame, a motion vector  $\mathbf{u}_{t+1,t-1}(k, l)$  is searched for its co-located block  $B_{t+1}(k, l)$  in the next decoded key frame  $\hat{X}_{t+1}$  with reference to  $\hat{X}_{t-1}$ , the previous decoded key frame. A forward motion vector  $\mathbf{u}_{t,t-1}(k, l)$ , and a backward motion vector  $\mathbf{v}_{t,t+1}(k, l)$  is calculated assuming constant motion.

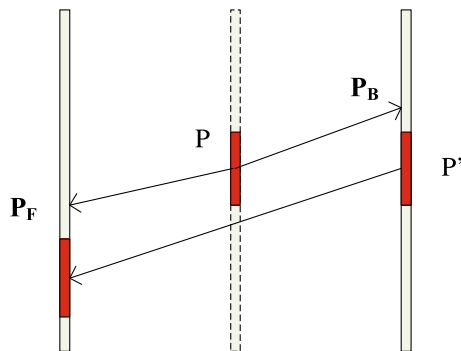
$$\mathbf{u}_{t,t-1}(k, l) = \frac{\mathbf{u}_{t+1,t-1}(k, l)}{2} \tag{2}$$

$$\mathbf{v}_{t,t+1}(k, l) = -\frac{\mathbf{u}_{t+1,t-1}(k, l)}{2} \tag{3}$$

SI for a pixel in  $B_t(k, l)$  is calculated by averaging the two blocks directed by the two motion vectors in the previous and the next decoded key frame, respectively.

The above process can also be carried out from frame  $\hat{X}_{t-1}$  to  $\hat{X}_{t+1}$ , and results in a so called bidirectional direct motion vector based method [33, 34]. The two interpolated blocks resulted from each direction are averaged to get the SI for a block in the current WZ frame.

**Fig. 3** Illustration of direct motion vector based SI estimation



#### 4.1.2 MCTI

A very popular MCI based SI estimation method is proposed in [10]. It is called Motion compensated temporal interpolation (MCTI), which includes the following steps:

1. A motion vector is estimated for every block in  $\hat{X}_{t+1}$  with reference to  $\hat{X}_{t-1}$  (forward motion estimation in the paper).
2. Using the trajectories of the motion vectors, a symmetric motion vector is selected for each block in  $Y_t$ , the SI for the current WZ frame  $X_t$ , by selecting the one with intersection in  $Y_t$  closest to the center of the block.
3. The symmetric motion vector is split into a forward motion vector, and a backward one, assuming constant motion.
4. The two motion vectors are further refined in a small area, keeping symmetric during this refinement process (called bi-directional motion estimation step in the paper).
5. Vector median filtering is performed over the two motion vector fields, which is called spatial smoothing of the motion vectors in the paper, to remove outlier motion vectors.
6. Average the two motion compensated blocks in  $\hat{X}_{t-1}$  and  $\hat{X}_{t+1}$  to produce  $Y_t$  (the bi-directional motion compensation step in the paper).

MCTI is combined with a global motion estimation method in [4]. In the method, global motion is estimated at the encoder using SIFT (scale invariant feature transform) feature extraction, and a perspective motion model. The SI generated with global motion compensation and MCTI is fused at the decoder. The SI is also refined during the successive decoding process. However, feature extraction and global motion estimation is computationally very complex, so this method loses the low complexity encoder feature of a WZ video codec.

#### 4.1.3 True motion estimation with 3DRS

Not like the situation in conventional video coding, where any block in the reference frame which minimizes the sum of absolute difference (SAD) criterion can be taken as the matched block for a block in the current frame of interest. In MCI, the estimation of so called true motion is the key to obtaining the best interpolation quality. Taking this into consideration, in [57], a 3DRS based true motion estimation approach is applied for SI estimation. Like in block based motion estimation algorithms (BMA), in 3DRS, the possible motion vector for a block in the current frame is from a candidate set  $CS^{\max}$ , in the reference frame, defined by

$$CS^{\max} = \{ \mathbf{c} \mid -N \leq c_x \leq N, -M \leq c_y \leq M \} \quad (4)$$

where  $N$  and  $M$  are constants limiting search area. Contrary to conventional full search BMA, which evaluates all possible candidate vectors in  $CS^{\max}$ , the 3DRS motion estimation method takes spatial and temporal prediction vectors (hence 3D),

from a 3D neighborhood, and update prediction vectors, which is defined as follows,

$$CS_{3DRS}(\mathbf{x}, t) = \begin{cases} \mathbf{c} \in CS^{\max} \mid \mathbf{c}_0 = \bar{\mathbf{0}}, \\ \mathbf{c}_1 = \mathbf{d}(\mathbf{x} - (W, H)^T, t), \\ \mathbf{c}_2 = \mathbf{d}(\mathbf{x} - (-W, H)^T, t), \\ \mathbf{c}_3 = \mathbf{d}(\mathbf{x} + 2(-W, H)^T, t - 1), \\ \mathbf{c}_4 = \mathbf{d}(\mathbf{x} + 2(W, H)^T, t - 1) \\ \mathbf{c}_5 = \mathbf{c}_1 + \mathbf{u}_a(\mathbf{x}, t) \\ \mathbf{c}_6 = \mathbf{c}_2 + \mathbf{u}_b(\mathbf{x}, t) \end{cases} \quad (5)$$

where  $W$  and  $H$  are the block width and height, respectively.  $\mathbf{u}_a(\mathbf{x}, t)$  and  $\mathbf{u}_b(\mathbf{x}, t)$  are update vectors, and are taken from a limited update set  $US(\mathbf{x}, t)$ .

$$US(\mathbf{x}, t) = \left( \begin{pmatrix} \pm 1 \\ 0 \end{pmatrix} \right) \cup \left( \begin{pmatrix} 0 \\ \pm 1 \end{pmatrix} \right) \quad (6)$$

here  $\mathbf{d}$  is the displacement of the block. For each set of candidates, 5 SAD of block are calculated between the source frame and its corresponding block and the 4 neighbors of the corresponding block in the reference frame in one iteration. The corresponding block will move to the location with minimum SAD in the reference frame before the next iteration. The recursion converges when the block does not move. After all branches converge, the displacement with the minimum SAD within the branches is the motion vector of the block.

Besides the two update branches in the original 3DRS algorithm, the authors of [57] also proposed to use other two inter-frame update recursions.

As 3DRS based motion estimation tends to find a motion field which is very close to true motion, so the SI method in this paper, can achieve better rate-distortion performance compared to other existing methods.

#### 4.1.4 Variable size block based reliable motion estimation

A variable size block based motion estimation strategy is used in [7]. Starting from  $8 \times 8$  blocks, four neighboring blocks will be merged into one  $16 \times 16$  block if the motion vectors in these blocks are in the same direction. The direction of a motion vector is calculated as

$$\theta = \arctan \frac{u_y}{u_x} \text{ if } u_x \neq 0 \quad (7)$$

Otherwise,  $\theta$  is  $\pm 90^\circ$  according to the sign of  $u_y$ . Further, the motion vectors are classified into 8 classes according to  $\theta$ , there is an additional class for zero motion vectors. Two motion vectors are said to be having the same direction if they are in the same class. Four neighboring  $16 \times 16$  blocks can further be merged into one  $32 \times 32$  block if the motion vectors in these blocks are also in the same direction. After the merging, a new motion vector is estimated for the merged block. Besides the merging process, another splitting process is also carried out if no merging has been done for a  $8 \times 8$  block and its eight neighboring blocks, and the directions of the motion vectors in the neighboring blocks are classified into more than four classes, the  $8 \times 8$  block will be split to four  $4 \times 4$  blocks. Again a new motion vector is estimated for each  $4 \times 4$  block. After the motion estimation process, bi-directional MCI is carried



out using the motion vectors estimated, and the previous and the next decoded key frame. With this method, the authors reported about 0.4 dB improvement in overall WZ video codec performance over the one with fixed size  $8 \times 8$  MCI.

Seeing that motion for area with little texture can not be reliably estimated, a variable size block based motion estimation strategy, with no-texture and stationary area detection is applied in [55]. In the method, Identification of no-texture area is carried out based on the analysis of spatial intensity gradient, which indicates spatial intensity variation. Before such areas are identified, the magnitude of the spatial intensity gradient (also referred to as gradient for brevity) for every pixel in  $\hat{X}_{t+1}$  is computed with

$$G(\mathbf{x}) = |G_x(\mathbf{x})| + |G_y(\mathbf{x})| \quad (8)$$

where  $G_x(\mathbf{x})$  and  $G_y(\mathbf{x})$  are respectively the horizontal and vertical component of gradient at pixel position  $\mathbf{x}$ , computed by finite difference:

$$\begin{cases} G_x(\mathbf{x}) = \hat{X}_{t+1}(\mathbf{x}') - \hat{X}_{t+1}(\mathbf{x}) \\ G_y(\mathbf{x}) = \hat{X}_{t+1}(\mathbf{x}'') - \hat{X}_{t+1}(\mathbf{x}) \end{cases} \quad (9)$$

here  $\mathbf{x}' = (x+1, y)$  and  $\mathbf{x}'' = (x, y+1)$  are two neighbors of pixel position  $x$ . The resulting pixel-by-pixel magnitude is stored in a buffer for the later identification of no-texture blocks.

To determine whether a block  $B$  with size  $K \times K$  ( $K \times K$  can be  $32 \times 32$ ,  $16 \times 16$ , or  $8 \times 8$ ) is no-texture or not, the mean,  $m_0$  and variance,  $\sigma_0^2$  of the gradients within it are computed. If  $\sigma_0^2 < T_{nt}$  with  $T_{nt}$  being a pre-specified threshold,  $B$  is classified as no-texture. Otherwise, the pixels in  $B$  are divided into two groups, group 1 contains pixels satisfying  $|G(\mathbf{x}) - m_0| < 2T_{nt}/3$ , group 2 contains the remaining pixels in the block. The mean,  $m_1$  and variance,  $\sigma_1^2$  for group 2 are computed once again. If  $\sigma_1^2 < T_{nt}$  holds,  $B$  is still classified as no-texture, otherwise, not.

Stationary area for the current WZ frame  $X_t$  is estimated by first computing the frame difference between its two adjacent decoded key frames  $\hat{X}_{t+1}$  and  $\hat{X}_{t-1}$ , dividing the difference frame into non-overlapping  $4 \times 4$  blocks, and then determining whether a  $4 \times 4$  block is stationary or not based on SAD within it. The pixel-by-pixel frame difference is computed by

$$FD(\mathbf{x}) = \hat{X}_{t+1}(\mathbf{x}) - \hat{X}_{t-1}(\mathbf{x}) \quad (10)$$

If  $\sum_{\mathbf{x} \in B} |FD(\mathbf{x})| < T_{st}$  holds for a  $4 \times 4$  block  $B$ , it is classified as stationary, otherwise not. Here  $T_{st}$  is another pre-specified threshold. For a block with size  $K \times K$  ( $K \times K$  can be  $32 \times 32$ ,  $16 \times 16$ , or  $8 \times 8$ ), which is multiples of  $4 \times 4$ , if all  $4 \times 4$  sub-blocks within it are classified as stationary, this block is taken as stationary, otherwise, not.

The estimation of variable size block based motion from  $\hat{X}_{t+1}$ , with reference to  $\hat{X}_{t-1}$  is then carried out as follows. First frame  $\hat{X}_{t+1}$  is divided into non-overlapping  $32 \times 32$  blocks and a motion vector is estimated in the reference frame  $\hat{X}_{t-1}$  for each block. Then according to the estimated motion and whether it is a no-texture block, a block may be further split into four  $16 \times 16$  child blocks, and motion estimation is conducted again on each  $16 \times 16$  block. This process is repeated until further splitting of a block is not necessary or the minimum block size  $4 \times 4$  has been reached.

The estimation of motion vector for a particular block  $B$  is conducted as follows. First zero motion vector is examined. If the block is stationary classified using the criterion described as above, a zero motion vector is assigned to block  $B$ , and no further motion estimation is conducted to the block. Otherwise, the no-texture identification process also described as above is next conducted for block  $B$ . If  $B$  is not classified as no-texture, a motion vector for block  $B$  is estimated in the reference frame  $\hat{X}_{t-1}$  by searching for the minimum SAD within a large search window  $[-M, M]$ . Otherwise, two cases are differently treated to estimate its motion. In the first case in which  $B$  is of size  $32 \times 32$ , motion estimation is conducted as other blocks which are not classified as no-texture. In the other case where  $B$  is of size less than  $32 \times 32$ , then its motion vector is searched within a smaller search window  $[-L, L]$ .

In the block split process, if a block has a zero motion vector or it is classified as no-texture, it is not split. For any other block, the average of the minimum SAD per pixel in it, denoted by  $m_{MinSAD}$ , is computed and compared with pre-specified threshold  $T_d$ , if  $m_{MinSAD} < T_d$  holds, then the motion vector estimated till now for the block will be kept, and no further split is applied. Otherwise, it is divided into smaller blocks and gone through further processing.

The approach in [55] also applies a different post processing of motion field, which leads to even better SI quality.

After the above steps, every  $4 \times 4$  block in the current WZ frame is assigned two motion vectors assuming linear motion, and SI is obtained by averaging the two  $4 \times 4$  blocks in  $\hat{X}_{t-1}$  and  $\hat{X}_{t+1}$  directed to by the motion vectors.

#### 4.2 Pel-recursive motion

In [16], a pel-recursive motion based SI estimation method was proposed. Block based motion estimation as those conducted in [10] is first carried out. With a pel-recursive motion estimation algorithm derived from the Cafforio-Rocca (CR) one [15], a dense motion vector field is then obtained. This is carried out in three steps. First, using the two already estimated block based motion vector fields for the current WZ frame,  $\mathbf{V}$  and  $\mathbf{U}$ , two pel-recursive motion vectors  $\mathbf{v}(\mathbf{p})$ ,  $\mathbf{u}(\mathbf{p})$  are estimated for every pixel  $\mathbf{p}$  in a block. When  $\mathbf{p}$  is at the top left corner of the block, the motion vectors estimated for the block,  $\mathbf{v}_B$ ,  $\mathbf{u}_B$  (the motion vector of a block  $B$  in motion vector field  $\mathbf{V}$  and  $\mathbf{U}$ , respectively) is used as an initialization for the pixel. Otherwise, a weighted average of the left, up, and up-right neighboring vectors, with different weights according to whether the neighbors are in the same block or not. The initialization vectors are denoted by  $\mathbf{v}^{(0)}(\mathbf{p})$ ,  $\mathbf{u}^{(0)}(\mathbf{p})$  respectively. Second, a motion vectors validation step is carried out. The motion compensated prediction errors associated to  $\mathbf{v}^{(0)}$ ,  $\mathbf{u}^{(0)}$  to zero vector, and to  $\mathbf{v}_B$ ,  $\mathbf{u}_B$  are computed, and the one with the least error is chosen as the validated vector. As in the original CR algorithm, a threshold is used to penalize the reset of an estimated vector. Finally, refinement of the validated motion vectors  $\mathbf{v}^{(1)}$ ,  $\mathbf{u}^{(1)}$  is carried out, by adding a correction  $\delta_v$ ,  $\delta_u$ , and minimizing the following cost function

$$J(\delta_v, \delta_u) = \left[ \hat{X}_{t+1}(\mathbf{p} + \mathbf{v}^{(1)} + \delta_v) - \hat{X}_{t-1}(\mathbf{p} + \mathbf{u}^{(1)} + \delta_u) \right]^2 + \lambda_1 \delta_v^2 + \lambda_2 \delta_u^2 \quad (11)$$

As in the original algorithm,  $\delta_v$  and  $\delta_u$  can be determined using a first order expansion of  $\hat{X}_{t+1}$  and  $\hat{X}_{t-1}$ , respectively. The steps are repeated till no pixels are updated or a pre-specified amount of loops have been run.

Using the above algorithm, the authors reported an improvement of up to 0.6 dB in the quality of SI, but minor improvement in the overall WZ video codec is observed.

In [26], pel-recursive motion is estimated with a different approach, and the interpolated frame using the pel-recursive motion and another one using OBMC [25] are both used in a multi-hypothesis WZ decoder.

### 4.3 Affine motion

Based on an affine motion model, a triangular mesh based SI estimation method was proposed in [28]. A regular triangular mesh is first constructed in the decoded key frame  $\hat{X}_{t+1}$ . The motion of each triangle vertex  $(dx_j, dy_j)$ ,  $j = 0, 1, \dots, N-1$ ,  $N$  is the number of vertices, is then estimated by minimizing the sum of squared error (MSE),

$$MSE = \sum_{\mathbf{s} \in \Omega_1, \mathbf{s} + \mathbf{u}(\mathbf{s}) \in \Omega_2} \left[ \hat{X}_{t-1}(\mathbf{s}) - \hat{X}_{t+1}(\mathbf{s} + \mathbf{u}(\mathbf{s})) \right]^2 \tag{12}$$

where  $\mathbf{s}$  is a point in a frame,  $\mathbf{u}(\mathbf{s}) = (dx(\mathbf{s}), dy(\mathbf{s}))$  is the displacement of point  $\mathbf{s}$ ,  $\Omega$  defines the support for estimation. For triangle mesh, the displacement of every point  $\mathbf{s}$  within a triangle  $T$  is calculated through the displacements of its vertices, by

$$\begin{cases} dx(\mathbf{s}) = \sum_{j \in T} w_j(\mathbf{s}) dx_j \\ dy(\mathbf{s}) = \sum_{j \in T} w_j(\mathbf{s}) dy_j \end{cases} \tag{13}$$

where  $w_j(\mathbf{s})$  are the weights. The SI is finally obtained using linear interpolation of  $\hat{X}_{t-1}$  and  $\hat{X}_{t+1}$  along motion trajectories at each pixel assuming that motion is uniform in the time period.

Observing that mesh based motion model is only effective to area where motion fields varies smoothly, and is not effective to complicated motion, and object boundaries, so a hybrid interpolation is adopted in the paper. Specifically, a block based motion is also estimated from  $\hat{X}_{t+1}$  to  $\hat{X}_{t-1}$ , and interpolation for every pixel in the current WZ frame is selected by computing the MSE in a small window around the pixel as in the motion estimation process, and if mesh based interpolation gives smaller MSE, then the pixel is interpolated by mesh based interpolation, otherwise, by block based interpolation, or vice versa. With this strategy, an up to 1dB performance gain with respect to block-based SI estimation methods is reported.

### 4.4 Higher order motion

Taking into consideration the acceleration of motion between adjacent frames in a video sequence, in [42] a high order motion interpolation (HOMI) based SI estimation method is proposed. Instead of using two adjacent key frames, this method uses four already decoded key frames  $\hat{X}_{t-3}$ ,  $\hat{X}_{t-1}$ ,  $\hat{X}_{t+1}$ , and  $\hat{X}_{t+3}$  for SI estimation of the current WZ frame  $X_t$ . Assume  $B_t(\mathbf{p})$  be a block in the current WZ frame centered on  $\mathbf{p}$ , the two motion vectors for  $B_t(\mathbf{p})$  directed to  $\hat{X}_{t-1}$  and

$\hat{X}_{t+1}$  be  $\mathbf{u}(\mathbf{p})$ , and  $\mathbf{v}(\mathbf{p})$ , respectively. Denote  $\mathbf{p}+\mathbf{u}(\mathbf{p})$  by  $\mathbf{p}'$ ,  $\mathbf{p}+\mathbf{v}(\mathbf{p})$  by  $\mathbf{p}''$ , respectively. Let  $B_{t-1}(\mathbf{p}')$  and  $B_{t+1}(\mathbf{p}'')$  be the two blocks directed by the two motion vectors respectively. A motion vector is searched for  $B_{t-1}(\mathbf{p}')$  in  $\hat{X}_{t-3}$ , and is denoted by  $\tilde{\mathbf{u}}(\mathbf{p})$ , minimizing the following matching criterion:

$$\sum_{\mathbf{q}} |B_{t-1}(\mathbf{p}') - B_{t-3}(\mathbf{p}' + \mathbf{q})| + \lambda \|\tilde{\mathbf{u}}(\mathbf{p}) - 3\mathbf{u}(\mathbf{p})\| \quad (14)$$

In the equation,  $\lambda$  is a constant, use of the second term is to penalize too large deviations from the linear model. Similarly, a motion vector  $\tilde{\mathbf{v}}(\mathbf{p})$  is searched for  $B_{t+1}(\mathbf{p}'')$  in  $\hat{X}_{t+3}$ . Next, with the four locations  $\mathbf{p}+\mathbf{u}(\mathbf{p})$ ,  $\mathbf{p}+\tilde{\mathbf{u}}(\mathbf{p})$ ,  $\mathbf{p}+\mathbf{v}(\mathbf{p})$  and  $\mathbf{p}+\tilde{\mathbf{v}}(\mathbf{p})$ , piecewise cubic Hermite interpolation is employed to determine a new location, name it  $\hat{\mathbf{p}}$  in the current WZ frame  $X_t$ . The last step of this method consists in choosing the interpolated trajectory passing closest to the block center and in assigning the associated vector to the position  $\mathbf{p}$ , and the new motion vectors now directed to  $\hat{X}_{t-1}$  and  $\hat{X}_{t+1}$  are  $\hat{\mathbf{u}}(\mathbf{p}) = \mathbf{u}(\mathbf{p})+\mathbf{p}-\hat{\mathbf{p}}$  and  $\hat{\mathbf{v}}(\mathbf{p}) = \mathbf{v}(\mathbf{p})+\mathbf{p}-\hat{\mathbf{p}}$ , respectively.

With this method, an improvement of up to 0.5 dB in PSNR in SI quality is achieved.

#### 4.5 3D model-based motion

Targeting video conferencing applications, in [9], 3D face model is used in the decoder SI estimation for head and shoulder video sequences. Here, by warping the 3D face model to the previous and next decoded key frame, the parameters of the 3D face model in the two frames are obtained, these parameters are then interpolated to get the model parameters of the 3D face in the current WZ frame. A further texturing mapping process which takes textures from the previous and the next decoded key frame and maps them onto the 3D face model for the current WZ frame is carried out, and the SI for the foreground object of the current WZ frame is generated. For the background area in the current WZ frame, block based MCI is used to generate the SI. Motion refinement with the successive decoding process proposed in [8] is also adopted in the scheme. A 0.5 dB to 0.75 dB improvement in PSNR compared to other state of the art WZ video coding schemes is reported.

### 5 Hash based decoder SI generation

In hash based decoder SI generation, SI is estimated in a way very like the motion compensated prediction in conventional video coding. Different from conventional video coding, in WZ video coding, current frame is not available when this process takes place. To make motion estimation still possible, some information about the current frame should be available at the decoder. To achieve this, parity bits, whether they are produced during the normal WZ encoding process, or they are specially generated solely for estimation of motion at the decoder side, are the first resource which can be utilized at the decoder side. This usually leads to joint SI estimation and WZ decoding, as will be seen clearly in the methods that follow. Hash codes generated at the encoder, which can be interpreted as a signature of pixels in the current WZ frame, are the second resource for this purpose. To ease the review, methods in this section are classified into explicit and implicit hash based ones.

## 5.1 Explicit hash

### 5.1.1 Parity bits as hash

PRISM [46, 47], a very famous WZ video coding framework in this field, is the first to our knowledge which uses parity bits as hash. In PRISM, frames are grouped in GOPs, as in conventional video coding. The first frame in a GOP is intra encoded with a conventional video coder. The other frames in the GOP are WZ encoded. Each of these frames is divided into  $8 \times 8$  blocks. A mode is then determined for the block according to SSE (sum of squared error) calculated with the block and its co-located block in the previous frame. A block can take one of the 3 classes, *SKIP*, *INTRA*, and *INTER*. If the block is classified as *INTRA* mode, it is encoded the same way as the first frame in the GOP using a conventional video coder. A *SKIP* block is not coded. An *INTER* block is further classified into 16 sub-classes, using a uniform quantizer of 16 levels applied to the SSE already calculated. The mode of a block and the sub-classes for an *INTER* block are all encoded and sent to the decoder. Each *INTER* block is then transformed with a  $8 \times 8$  DCT, and quantized with a uniform quantizer. The quantized DC and the first 14 quantized AC coefficients in the Zigzag scanning order are syndrome encoded, the remaining AC coefficients are encoded with run length coding and Huffman coding. A 16bit CRC is also calculated for the 15 syndrome encoded quantized coefficients, which is used in the motion estimation process at the decoder side for SI generation.

At the decoder, an intra frame and *INTRA* blocks in an *INTER* frame are decoded the same way as in conventional video coding. A *SKIP* block is reconstructed from its co-located block in a previous reconstructed frame. For an *INTER* block, a half pixel motion estimation is conducted to search for a matching block. The reference is the previous decoded frame, and the search window is centered at the zero motion vector in the reference window. At each point in the search window, the candidate matching block is transformed by DCT, and the first 15 transform coefficients, along with the syndrome for the block to be decoded are fed to a channel decoder. The decoder output and the 15 DCT coefficients of the candidate block are used again to recover the 15 quantized coefficients. 16-bit CRC is then generated for the recovered quantized coefficients, and is compared with the CRC transmitted by the encoder. If the CRCs match, the motion search terminates, otherwise, the motion search continues using the next candidate block in the search window, and the process repeats. If the motion search fails to find a match for an *INTER* block, error concealment of the block is conducted. For a successfully decoded block, reconstruction is then carried out.

It can be clearly seen from the above process, the auxiliary CRC used in PRISM makes motion estimation at the decoder side feasible, however, a 16 bits per  $8 \times 8$  block is a very heavy overhead in the method. In addition, though the parity bits in PRISM is solely for motion estimation at the decoder side to generate SI, they are never called hash codes in PRISM.

### 5.1.2 Other hash codes

A method that explicitly uses hash codes is first proposed in [1] to realize a low latency WZ video codec. In the method, frames are also grouped in GOPs. The first frame in a GOP is intra encoded with a conventional video coder. The other

frames in the GOP are WZ encoded. Each frame is first block DCT transformed and the DCT coefficients are then quantized with a uniform quantizer. The lower frequency coefficients in all blocks of a frame are then grouped into coefficient bands. The coefficient bands are encoded using a rate compatible punctured turbo code based SW encoder, by first extracting bit planes of each coefficient band, and then fed the bit plane to the SW encoder one after another. The higher frequency coefficients in a block are taken as hash code word for the block, and are entropy encoded. To save bits, the encoder also stores for the previous frame the quantized higher frequency coefficients, and the distance between the coefficients of a block in the current frame and those of its co-located block in the store is calculated. Hash code word is only sent for those blocks with distance greater than a pre-specified threshold.

At the decoder, if no hash code word has been sent for a block, its co-located block in the previous frame is taken as SI. For all other blocks, reconstruction of the blocks using the higher frequency coefficients is conducted, and motion search is carried out using the reconstructed block as source and the previous decoded frame as reference. The matched block is then taken as SI for the block, and channel decoding of the low frequency coefficients is then carried out. Due to lack of information of the current frame, this method is only effective for sequences with low motion activity and achieves poor performance for sequences with high motion activity.

Use of the lower frequency DCT coefficients as hash code word is proposed by the same authors in another paper [2].

To mitigate the drawbacks of the previous method, an adaptive hash based approach is proposed in [12]. Here, DC and some low frequency AC coefficients in zigzag scanning order are selected as hash code word for a block. The coefficients are also quantized before encoding and being sent to the decoder. An adaptive method is also designed, which selects different number of low frequency coefficients as hash code word for different blocks of a frame to achieve optimal rate-distortion performance, according to the quantization step size, and the energy distribution of the DCT coefficients.

In [19] and a following work [21], the  $b$  most significant bit planes of a WZ frame, which are entropy coded, are sent to the decoder and are used as hash information for decoder SI generation.

To get reliable motion with only the partially reconstructed WZ frame from only the  $b$  most significant bit planes, overlapped block motion estimation (OBME) strategy is employed. Specifically, a WZ frame is divided into overlapping spatial blocks  $B$  of size  $K \times K$ , with overlap  $\varepsilon$  where  $1 \leq \varepsilon \leq B$ . The best match of each such block is searched in the two already decoded adjacent key frames using a matching criterion that minimizes the pixel error ratio (PER) on the  $b$  most significant bit planes between a current block  $B$  and a candidate match  $R$  in the reference frame.

After OBME, each pixel  $\mathbf{x}$  in the WZ frame corresponds to a number of overlapping blocks  $B_i$ , each of which is linked with their best matching block  $R_i$  in one of the key frames. The co-located pixels in  $B_i$  form candidate predictor set  $T = \{\psi_0, \psi_1, \dots, \psi_{K-1}\}$ . Pixels in  $T$  are weight averaged to get the SI for  $\mathbf{x}$ . The authors proposed three different methods to determine the weights in calculating  $y$  for  $\mathbf{x}$ .

Targeting low complexity medical applications, In [20], a down-sampled WZ frame, which is intra coded with a conventional codec, is taken as hash.

At the decoder side, OBME as in [19, 21] is carried out for blocks in the decoded and up-sampled hash frame, with reference to a past and a future key frame. SI is then generated by multi-hypothesis motion compensated prediction. By averaging the matched blocks found in OBME. Considerations are made to not include blocks in this process which have a large SAD. For such blocks, their co-located blocks in up-sampled hash frame is used.

There are also other efforts on hash based decoder SI generation. For example, in [30], hash for a block is generated through different methods, specifically spatial sub-sampling, low frequency DCT coefficients, DCT thresholding, and edge detection, and their performance is evaluated. In [35], two hash codes for decoder SI generation was proposed, one uses N-queen spatial sub-sampling, the other uses DCT coefficients. Also in [18], three approaches, namely, down sampling, DCT-based and feature-based hash codes on the motion estimation accuracy at the decoder side was evaluated. Anbu and Zymnis [6] also proposed using low frequency DCT coefficients as hash code word.

In [51], theoretical modeling of hash based SI generation at the decoder side is proposed. It assumes that only translational motion exists in consecutive frames of a video sequence, and the high frequency DCT coefficients hash code word used in [1] can be approximated by filtering the original video frame with an ideal band pass filter in frequency domain. Using a power spectral density (PSD) tool and based on the results obtained in [48], the rate-distortion performance of hash based SI generation at the decoder side is analyzed. And the results show that, at low rates, due to there is not enough information of the current frame, reliable motion estimation could not be achieved. At medium to high rates, hash based methods can be as accurate as those used in conventional encoder SI estimation.

## 5.2 Implicit hash

As can be seen from the preceding paragraphs, MCE and MCI based decoder SI generation methods can not work well, due to the lack of information of the current frame to be decoded, and if motion between two frames is not linear as supposed in the two kinds of methods. For explicit hash codes based methods, they are effective only when enough information on the current frame is available, which results in much overhead and a degradation in rate-distortion performance. The CRC based joint decoder SI generation and decoding, as in the PRISM framework, though it causes much overhead, it provides some insight for this problem: To achieve good performance, decoder SI generation and decoding should be carried out in an iterative way. Working on a block by block basis, there is no connection between blocks in PRISM, so a block can be decoded without any information of its adjacent blocks. This makes joint decoder SI generation and decoding feasible with only moderate computational complexity. For other popular WZ video frameworks based on turbo codes [3], or LDPC codes [52], as they typically work on bit planes of a whole frame, a joint decoder SI generation and decoding approach is not feasible any more, as even when block based translational motion model is employed, a run for decoder SI generation and decoding takes a huge amount of computation. In the [63] the authors proposed a joint decoder SI generation and decoding methods, by ignoring the performance loss of turbo codes and working on a block by block



basis. Specifically, they used turbo codes to channel encode transform coefficients of a block, and joint SI generation and decoding is carried out at the decoder side. To tackle the joint SI generation and decoding problem when channel codes work on a whole frame, Varodayan et al. [53] proposed an unsupervised decoder motion learning method, where no explicit motion is estimated, instead a block is predicted by all possible blocks in the search window of the reference frame with a probability as weight. An Expectation Maximization (EM) based strategy is applied in the joint decoder SI generation and decoding process. Basically it works as follows.

Suppose  $\mathbf{U}$  be the motion field between  $X_t$  and  $X_{t-1}$ , and the residual of  $X_t$  with respect to motion compensated  $X_{t-1}$  is treated as independent Laplacian noise  $Z$ .

The a posteriori probability distribution of source  $X_t$  at the decoder based on parameter  $\theta$  is

$$P_{app} \{X_t\} \equiv P \{X_t; \theta\} = \prod_{i,j} \theta(i, j, X_t(i, j)) \tag{15}$$

where  $\theta(i, j, w) = P_{app} \{X_t(i, j) = w\}$  defines a soft estimate of  $X_t(i, j)$  over possible luminance values  $w \in \{0, \dots, 2^b - 1\}$ , with  $b$  being bits per pixel of the video,  $(i, j)$ , the pixel position.

The aim of the decoder is to estimate the a posteriori probability distribution of the motion  $\mathbf{U}$ ,

$$P_{app} \{\mathbf{U}\} \equiv P \left\{ \mathbf{U} \mid \hat{X}_{t-1}, S; \theta \right\} \propto P \{\mathbf{U}\} P \left\{ \hat{X}_{t-1} S \mid M; \theta \right\} \tag{16}$$

The second step of the above equation follows by Bayes' Law. Note, we have changed from  $X_{t-1}$  to  $\hat{X}_{t-1}$ , as at the decoder, only  $\hat{X}_{t-1}$ , the decoder reconstruction of the source  $X_{t-1}$  is available.  $P\{\mathbf{U} \mid \hat{X}_{t-1}, S; \theta\}$  in the above equation is the probability of observing motion  $\mathbf{U}$  given that it relates  $X_t$  (as parameterized by  $\theta$ ) to  $\hat{X}_{t-1}$ , and also given  $S$  the syndrome.

In the E-step of the EM algorithm applied in the method, the motion field distribution  $\mathbf{U}$  is updated with reference to the source model parameters by (before re-normalization):

$$P_{app}^{(n)} \{\mathbf{U}\} := P_{app}^{(n-1)} \{\mathbf{U}\} P \left\{ \hat{X}_{t-1}, S \mid \mathbf{U}; \theta^{(n-1)} \right\} \tag{17}$$

To simplify this operation, in [53], the syndrome  $S$  is ignored in this step, and the motion field is approximated by a block-by-block motion vectors  $\mathbf{u}(k, l)$ . For a block  $\theta^{(n-1)}(k, l)$  with top left pixel located at  $(k, l)$ , the distribution on  $\mathbf{u}(k, l)$  is updated by (before re-normalization):

$$P_{app}^{(n)} \{\mathbf{u}(k, l)\} := P_{app}^{(n-1)} \{\mathbf{u}(k, l)\} P \left\{ \hat{X}_{t-1}((k, l) + \mathbf{u}(k, l)) \mid \mathbf{u}(k, l); \theta^{(n-1)}(k, l) \right\} \tag{18}$$

where  $\hat{X}_{t-1}((k, l) + \mathbf{u}(k, l))$  is the  $m$ -by- $m$  block of  $\hat{X}_{t-1}$  with top left pixel at  $((k, l) + \mathbf{u}(k, l))$ .  $P\{\hat{X}_{t-1}((k, l) + \mathbf{u}(k, l)) \mid \mathbf{u}(k, l); \theta^{(n-1)}(k, l)\}$  is the probability of observing  $\hat{X}_{t-1}((k, l) + \mathbf{u}(k, l))$  given that it was generated through vector  $\mathbf{u}(k, l)$  from  $X_t(k, l)$  as parameterized by  $\theta^{(n-1)}(k, l)$ .



In the M-step of the method, the soft estimate  $\theta$  is updated by maximizing the likelihood of  $X_{t-1}$  and syndrome  $S$ .

$$\theta^{(n)} := \arg \max_{\Theta} P\{X_{t-1}, S; \Theta\} = \arg \max_{\Theta} \sum_{\mathbf{u}} P_{app}^{(n)}\{\mathbf{U} = \mathbf{u}\} P\{X_{t-1}, S|\mathbf{U} = \mathbf{u}; \Theta\} \tag{19}$$

where  $\Theta$  denotes, the summation is over all configurations  $\mathbf{u}$  of the motion field. As true maximization is intractable, so in [53], it is approximated by generating soft SI  $\psi^{(n)}$ , followed by an iteration of channel decoding to yield  $\theta^{(n)}$ .

The probability that the soft SI has a value  $w$  at pixel  $(i, j)$   $\psi^{(n)}(i, j, w)$  is determined by

$$\begin{aligned} \psi^{(n)}(i, j, w) &= \sum_{\mathbf{u}} P_{app}^{(n)}\{\mathbf{U} = \mathbf{u}\} P\{X_t(i, j) = w|\mathbf{U} = \mathbf{u}, \hat{X}_{t-1}\} \\ &= \sum_{\mathbf{u}} P_{app}^{(n)} p_z(w - \tilde{X}_{\mathbf{u}}(i, j)) \end{aligned} \tag{20}$$

where  $p_z(Z)$  is the probability mass function of the independent additive noise  $Z$ , and  $\tilde{X}_{\mathbf{u}}$  is the previous reconstructed frame compensated through motion configuration  $\mathbf{u}$ .

The calculated  $\psi^{(n)}$  is then used in the channel decoding and soft reconstruction of  $X_t, \theta^{(n)}$ , as already described above.

The EM process is initialized by  $P_{app}^{(0)}\{\mathbf{u}(k, l)\}$ , determined by experiments and takes the following form in the paper,

$$P_{app}^{(0)}\{\mathbf{u}(k, l)\} := \begin{cases} \left(\frac{3}{4}\right)^2 & \text{if } \mathbf{u}(k, l) = (0, 0) \\ \frac{3}{4} \cdot \frac{1}{80} & \text{if } \mathbf{u}(k, l) = (0, *), (*, 0) \\ \left(\frac{1}{80}\right)^2 & \text{otherwise} \end{cases} \tag{21}$$

The key idea of this approach is that, an explicit motion field is not estimated, instead only the probability distribution that a block with top left pixel located at  $(k, l)$  which has a motion vector  $\mathbf{u}(k, l)$  is estimated.

The idea of soft SI proposed in the previous approach is combined with MCI based SI estimation in [40] to improve the performance of a WZ video codec.

### 6 Decoder SI refinement

As in most popular WZ video frameworks, quantized pixels or transform coefficients of a frame are encoded bit plane after bit plane. Decoding is carried out hence in the same order. After one or more bit planes are decoded, they can be used to reconstruct the current frame, even though it is a coarse one. The coarse reconstruction contains very important information of the frame being decoded, and can be used to refine the SI estimated by any methods we have reviewed in previous sections. The

refined SI is used for the decoding of the bit plane that follows. This decode-refine-decode process can be repeated till all the bit planes are decoded. In this section, we will review typical decoder SI refinement methods in WZ video coding.

The first decoder SI refinement was proposed in [8], which is called motion compensated restoration (MCR). Firstly, SI estimated by standard MCI is used for channel decoding and reconstructing a WZ frame, which is called partially decoded picture in the paper. Secondly, a second MCI, called MCR is executed, which takes the partially decoded picture as source, and the previous and next adjacent decoded key frame as well as SI estimated in MCI as reference, and then a best match is found for every aligned block in the partially decoded block in previous decoded key frame, next decoded key frame, and the motion compensated average of the past and the future frame, and the result from MCI. Finally, decoder reconstruction is built again using the refined SI. This process is iterated till no further obvious improvement is observed. Although only a minor improvement of about 0.15 dB for a variety of sequences is achieved by the method, to the best of our knowledge, this is the first of this kind of methods in the literature.

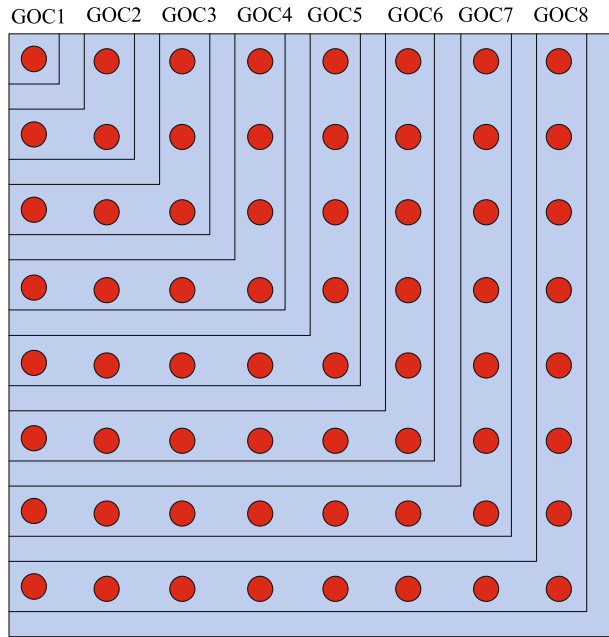
A similar SI refinement method was proposed in [11], for pixel domain DVC. It differs from [8] in that, only blocks with error bigger than a threshold, which is computed by the reconstructed block with respect to its co-located block in SI, is refined, and a different matching criterion and strategy for searching matching blocks is employed.

A similar SI refinement method as the one in [8, 11] was independently proposed in [32].

An SI refinement scheme for transform domain WZ video coding was proposed in [27]. In the approach, a WZ frame is divided into regular blocks, and each block is encoded and decoded independently. The transform coefficients in a block is further divided into groups (called groups of coefficients-GOC, see Fig. 4), and are transmitted and decoded in stages, one GOC a stage. At the first stage, the first GOC is transmitted, as at this stage no coefficients for the current frame have been received. SI is simply the co-located block in the previous frame. After decoding of the first GOC, SI is estimated with the decoded GOC as source and the previously decoded frame as reference. At a later stage, the GOC is decoded with the SI estimated in a previous stage. Again, after decoding, motion estimation is carried out to get better SI for the decoding and reconstruction of other GOCs. The authors reported that, for better SI refinement, and hence better overall rate-distortion performance, a high-to-low-frequency GOC ordering is better than a low-to-high-frequency GOC ordering, with about 1dB improvement in PSNR.

A similar SI refinement approach as the one described in the preceding paragraph was proposed for transform domain WZ video coding [14]. In which, after DC coefficient band is decoded, DME is carried out, with DC in each block of the current WZ frame as source, and the previous and next adjacent decoded key frame as reference. Full search block matching is applied. To achieve this, a possible matching block is first transformed with DCT, and then SAD is calculated with the DCs in both blocks. After motion estimation, forward, backward, and bi-directional motion compensation is carried out. With the three possible SI, and the original SI obtained by MCI, the best SI is selected block by block by choosing the block which results in least SAD calculated with the DCs in a block. The refined SI is used for decoding of all the AC coefficient bands.

**Fig. 4** Arrangement of GOCs



In another paper [13], the same authors extended the scheme to also AC coefficients bands, and proposed another iterative SI refinement scheme. In this scheme, when DC band is decoded, the same SI refinement process as that in [3] is carried out. With newly refined SI other AC coefficients bands are then decoded. Next, with the newly decoded AC coefficients bands, the already decoded DC band and SI, a partial WZ frame is reconstructed. Now motion estimation is performed again with the partial WZ frame as source, and the two adjacent already decoded key frames as reference. After motion estimation, best prediction is chosen again from backward, forward, bi-directional, and previous SI, using SAD as a measure. With the newly refined SI, another AC coefficients band is then decoded. This process is repeated till all coefficients bands are decoded.

Yet another decoder SI refinement approach for the transform domain DVC was proposed in [39]. It comprises three modules: block selection for refinement, candidate blocks searching, and new SI creation. In the approach, DC coefficient band is decoded with an initial SI  $Y$  estimated by MCI using the previous and the next adjacent decoded key frames. For any AC coefficient band, it is decoded after the SI refinement is performed, and hence, its SI may come from the initial estimation  $Y$ , or from the output of the SI refinement process  $Y'$ .

Before an AC coefficient band  $b$  is to be decoded, reconstruction of the current WZ frame is carried out, with the so far decoded  $b - 1$  bands, and results in  $R_n^{b-1}$  for a  $4 \times 4$  block  $n$ . In the reconstruction, the not yet decoded coefficient bands are filled from the initial SI. Next, for each  $4 \times 4$  block  $n$ , and band  $b$ , the sum of squared errors (SSE)  $\varepsilon_n^b(0)$  is computed between  $R_n^{b-1}$  and the co-located block  $Y_n$  in the initial SI. Only if  $\varepsilon_n^b(0)$  is bigger than a predefined threshold which is 100 in the paper, should the block be selected for further processing.

The next step of the approach is to find a best match for a block  $n$  selected in the previous step. This is carried out with a  $9 \times 9$  search window, centered on the central position of the block under processing, in the initial SI. Again SSE is used as a matching criterion, and this process will results in 80 SSEs, besides  $\varepsilon_n^b(0)$ , denoted by  $\varepsilon_n^b(k)$  with  $k = 1, 2, \dots, 80$ . Only if a candidate block satisfies  $\varepsilon_n^b(k) < \varepsilon_n^b(0) \cdot P$ , should the block be considered.  $P$  is a constant in the equation and takes a value 0.2 in the paper. For a block which does not even have one candidate satisfying the equation, it is dropped without further processing.

The creation of new SI module is now executed, which simply takes all candidate blocks found in the previous step, and compute a weighted average over them to generate the refined SI  $Y_n^r$ ,

$$Y_n^r = \frac{\int_k \left( Y_n^k \cdot \frac{1}{\varepsilon_n^b(k)} \right)}{\int_k \frac{1}{\varepsilon_n^b(k)}} \tag{22}$$

where  $Y_n^k$  is a candidate block found in the previous step.

In a mixed resolution WZ video coding framework [38], where B or some P frames between I and/or P frames in the conventional video coding technology like H.264 are treated as WZ frames. A WZ frame is first decimated by a factor of  $2^n \times 2^n$  ( $n$  is an integer) to get a low resolution (LR) frame. The LR current frame is then encoded with reference to other encoder reconstructed I or P frames which are also decimated, using a conventional encoding method. The residual signal that is computed between the original current frame and the decoded LR current frame after interpolation to get the same spatial resolution as other frames, is transformed, quantized and then WZ coded with a coset code based SW coder. At the decoder, with the interpolated LR reconstructed current frame denoted by  $F_0$ , and other already decoded frames in the frame store denoted by  $FS$ , SI estimation for the residual of the current frame and the decoding of the residual process is iterated as follows,

$$F_i^{HR} = SS(F_i, FS) \tag{23}$$

$$F_{i+1} = D_{WZ} (F_i^{HR} - F_0, b_{WZ}) + F_0 \tag{24}$$

where  $i = 0, 1, \dots, N - 1$  are the iterations,  $SS(F, FS)$  denotes the operation to yield a high resolution version  $F^{HR}$  of  $F$  based on  $FS$ , and  $D_{WZ}(RF, b_{WZ})$  is the WZ decoding operation yielding a corrected version of the residual frame based on a noise version  $RF$  using the WZ bit stream  $b_{WZ}$ . With this iterative residual SI estimation-decoding process better decoder reconstruction is obtained.

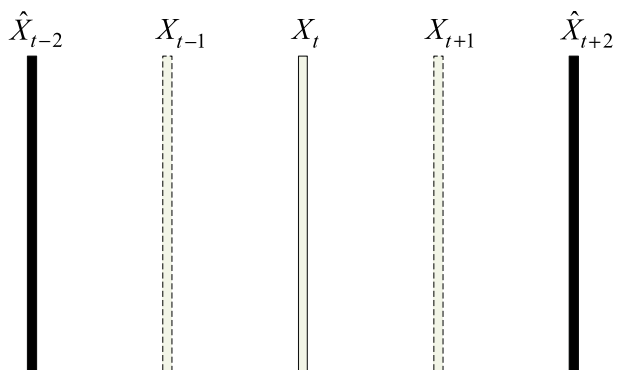
Using a similar codec configuration as the one presented in [41], and working in pixel domain, Adikari et al. [5] proposed a SI refinement method for WZ video coding. In the method, the SI for the highest bit plane is estimated with MCI. Once a bit plane is decoded, motion estimation is carried out with the already decoded bit plane(s) as source, and corresponding bit planes of the pixels in the previous decoded frame as reference. Both luminance and chrominance component are used in the motion estimation process. The new motion compensated SI obtained in such a manner will be used for the decoding of other lower bit planes. This process is repeated till all bit planes are decoded and the decoder reconstruction is built. The authors reported up to 3dB improvement in PSNR compared with the one in [41].

In [56], another SI refinement method was proposed. In the method, an initial SI is produced with MCE as in [41]. The initial SI is then interleaved and sent directly to an error estimation process, which treats 10 pixels as a block. If the error estimation results for a block is above a predefined threshold, all pixels in the block are flagged. Next the frame is de-interleaved, this makes the flagged pixel bursts to scatter. A temporal or spatial prediction algorithm is then utilized to re-fill the flagged pixels in the frame. The flag-filled output frame is then interleaved again and enters the next iteration of error estimation followed by flagging and prediction based flag-filling. This process is repeated several times. The use of spatial prediction and temporal prediction is alternated during the process with spatial prediction as initial step. The resulted SI is then fed to channel code based SW decoder for the decoding of the current frame.

For a WZ video coding system with a GOP size larger than 2, the quality of the SI estimated by MCI or the kind will be degraded as the temporal distance of a WZ frame to a key frame is now not 1 any more. For example, with Fig. 5, GOP size is 4, WZ frame  $X_{t-1}$  has a distance of 3 to key frame  $\hat{X}_{t+2}$ ,  $X_t$  has a distance of 2 to both key frames  $\hat{X}_{t-2}$  and  $\hat{X}_{t+2}$ , respectively, while  $X_{t+1}$  has a distance of 3 to key frame  $\hat{X}_{t-2}$ . In such a situation, MCI will not be as effective as when GOP size is 2, and a pyramid decoding structure is desirable. That is,  $X_t$  is first decoded, with  $\hat{X}_{t-2}$  and  $\hat{X}_{t+2}$ , for SI estimation, and then  $X_{t-1}$  is decoded with  $\hat{X}_{t-2}$  and  $\hat{X}_t$ , the reconstruction of  $X_t$ , for SI estimation.  $X_{t+1}$  is decoded in a similar manner. In such a case, the quality of the SI for  $X_t$  should be worse than that for  $X_{t-1}$  and  $X_{t+1}$  due to larger temporal distance from the already decoded key (WZ) frames. To cope with this, an SI refinement approach was proposed in [43] to the central frame in a GOP, i.e. the frame  $X_t$  when GOP size is 4 as illustrated in Fig. 5. It is carried out after the whole GOP is decoded, and then HOMI [42] is applied to estimate a better SI for  $X_t$ , another round of SW decoding with parity bits in the original decoding process, and the new SI and reconstruction is followed. With this approach, an improvement of up to about 2.7 dB in the SI quality is obtained, and up to 0.7 dB in decoded frames as well as a bit rate reduction of up to 15 % is obtained.

In [36, 37], a decoder side multi-resolution motion refinement (MRMR) scheme was proposed, where the decoder uses the already decoded lower resolution data to refine the motion estimation, which in turn greatly improves the SI quality as well as the coding efficiency for the higher resolution data. Theoretical analysis is

**Fig. 5** Pyramid coding structure



given in the paper to show that the scheme is feasible. A practical WZ video codec implementation with MRMR is also presented in the paper.

## 7 Conclusion and possible research directions

In this paper, we have reviewed various decoder SI generation methods in WZ video coding. Specifically, MCE based, MCI based and hash based methods. We have also reviewed SI refinement in WZ video coding. As can be seen from the description, for MCE based methods, with only previous temporal information of the current frame available, only limited performance can be achieved, even though this results in a similar coding structure as in conventional video coding. For MCI based methods, as both previous and future temporal information are available when SI is generated, so this kind of methods can get better SI quality than MCE based ones. But this is based on the condition that the temporal distance between two key frames is not very far. This also limits the performance of such WZ video codec. In addition, a coding structure of MCI based methods is very similar with the B frame coding in conventional video coding, so MCI based methods will cause further delay. Explicit hash based approaches are very similar with MCE based ones, with low complexity and low delay, and also similar performance. Note that there is an overhead for such approaches, as the main purpose of hash codes is for decoder SI generation. For these methods to achieve better SI quality, more hash code words have to be sent which will result in heavy overhead. Implicit hash based methods avoid the overhead of its counterpart, but they are computationally very complex.

From the review, we can see that, in MCI based SI generation, better quality SI requires more accurate motion model for the underlying video content, and basically translational motion model is only effective for slow motion. When some information about the current WZ frame is available at the decoder side, as in the hash based decoder SI generation scenario, how to reliably estimate motion for a WZ frame with reference to previously decoded key or WZ frame is critical for good SI. In joint decoding and SI generation approaches, the difficulty lies in estimating block based or dense motion using parity bits for a whole frame. As a result, Researches on more accurate motion model, motion estimation with only partial information, and how to estimate local motion from parity bits for a whole frame are all possible topics which can be pursued in the future.

**Acknowledgements** This work was supported in part by the National Science and Technology Major Project of China (2011ZX03001-007-01), the National Basic Research Program of China (2009CB320404), the Fundamental Research Funds for the Central Universities (72115046), and the 111 Project (B08038).

## References

1. Aaron A, Girod B (2004) Wyner-ziv video coding with low encoder complexity. In: Picture Coding Symposium 2004, PCS 2004. PCS—Tektronix, Inc., pp 429–433
2. Aaron A, Rane S, Girod B (2004) Wyner-ziv video coding with hash-based motion compensation at the receiver. In: IEEE International Conference on Image Processing, ICIP 2004. vol 2. IEEE, pp 3097–3100

3. Aaron A, Zhang R, Girod B (2002) Wyner-ziv coding of motion video. In: The thirty-sixth asilomar conference on signals systems and computers. Conference record of the asilomar conference on signals, systems and computers, vol 1. IEEE Computer Society, pp 240–244
4. Abou-Elailah A, Dufaux F, Farah J, Cagnazzo M, Pesquest-Popescu B (2013) Fusion of global and local motion estimation for distributed video coding. *IEEE Trans Circuits Syst Video Technol* 23(1):158–172
5. Adikari ABB, Fernando WAC, Arachchi HK, Weerakkody WARJ (2006) Sequential motion estimation using luminance and chrominance information for distributed video coding of wyner-ziv frames. *Electron Lett* 42(7):398–399
6. Anbu A, Zymnis A (2005) Hash-aided motion estimation. Tech. rep., Stanford University
7. Argyropoulos S, Thomos N, Boulgouris NV, Strintzis MG (2007) Adaptive frame interpolation for wyner-ziv video coding. In: IEEE 9th workshop on Multimedia Signal Processing, 2007, MMSP 2007, 1–3 Oct 2007. IEEE, pp 159–162
8. Artigas X, Torres L (2005) Iterative generation of motion-compensated side information for distributed video coding. In: IEEE International Conference on Image Processing, ICIP 2005, vol 1. IEEE, pp 833–836
9. Artigas X, Torres L (2006) An approach to distributed video coding using 3d face models. In: IEEE international conference on image processing, ICIP 2006. IEEE, p 284
10. Ascenso J, Brites C, Pereira F (2005) Improving frame interpolation with spatial motion smoothing for pixel domain distributed video coding. In: 5th EURASIP conference on speech and image processing, multimedia communications and services. Smolenice, Slovak Republic, pp 1–6
11. Ascenso J, Brites C, Pereira F (2005) Motion compensated refinement for low complexity pixel based distributed video coding. In: IEEE conference on Advanced Video and Signal Based Surveillance AVSS 2005, 15–16 Sept 2005. IEEE Computer Society, pp 593–598
12. Ascenso J, Pereira F (2006) Adaptive hash-based side information exploitation for efficient wyner-ziv video coding. In: IEEE International Conference on Image Processing, ICIP 2007, vol 3. IEEE, pp III29–III32
13. Badem MB, Fernando WAC, Martinez JL, Cuenca P (2009) An iterative side information refinement technique for transform domain distributed video coding. In: IEEE International Conference on Multimedia and Expo, ICME 2009. IEEE Computer Society, pp 177–180
14. Badem MB, Mrak M, Fernando WAC (2008) Side information refinement using motion estimation in dc domain for transform-based distributed video coding. *IEE Electron Lett* 44(16):965–966
15. Cafforio C, Rocca F (1983) The differential method for image motion estimation. *Image Sequence Processing and Dynamic Scene Analysis*, NATO ASI Series, Vol. 2. 1983. pp. 104–124
16. Cagnazzo M, Maugey T, Pesquet-Popescu B (2009) A differential motion estimation method for image interpolation in distributed video coding. In: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2009, 19–24 Apr 2009. IEEE, pp 1861–1864
17. Choi BT, Lee SH, Ko SJ (2000) New frame rate up-conversion using bi-directional motion estimation. *IEEE Trans Consum Electron* 46(3):603–609
18. Dehy R, Pratz G, Wu H (2005) Hash codes for motion estimation of video. Tech. rep., Stanford University
19. Deligiannis N, Munteanu A, Clerckx T, Cornelis J, Schelkens P (2009) Overlapped block motion estimation and probabilistic compensation with application in distributed video coding. *IEEE Signal Process Lett* 16(9):743–746
20. Deligiannis N, Verbist F, Clossifides A et al (2012) Wyner-ziv video coding for wireless lightweight multimedia applications. *EURASIP J Wirel Commun Netw* 2012:106
21. Frederik V, Deligiannis N, Jacobs M, Barbarien J et al (2013) Probabilistic motion compensated prediction in distributed video coding. *Multimed Tools Appl* 66(3):405–430
22. Girod B, Margot A, Rane S, Rebollo-Monedero D (2005) Distributed video coding. *Proc IEEE* 93(1):71–83
23. Guillemot C, Pereira F, Torres L, Ebrahimi T, Leonardi R, Ostermann J (2007) Distributed monoview and multiview video coding. *IEEE Signal Proc Mag* 24(5):67–76
24. Guo M, Lu Y, Wu F, Li S, Gao W (2007) Distributed video coding with spatial correlation exploited only at the decoder. In: IEEE International Symposium on Circuits and Systems, ISCAS 2007, 27–30 May 2007. IEEE, pp 41–44
25. Huang X, Forchhammer S (2008) Improved side information generation for distributed video coding. In: IEEE international workshop on Multimedia Signal Processing, MMSP2008, 8–10 Oct 2008. IEEE, pp 223–228



26. Huang X et al (2011) Multi-hypothesis transform domain Wyner-Ziv video coding including optical flow. In: IEEE international workshop on Multimedia Signal Processing, MMSP 2011, Oct. 2011. IEEE, pp 1–6
27. Kaul S, Padilla M, Trejo O (2004) Wyner-ziv video coding with increasingly refined motion estimation. Tech. rep., Stanford University
28. Kubasov D, Guillemot C (2006) Mesh-based motion-compensated interpolation for side information extraction in distributed video coding. In: IEEE international conference on image processing, ICIP 2006, 8–11 Oct 2006. IEEE, pp 261–264
29. Lagendijk RL, Sezan MI (1992) Motion compensated frame rate conversion of motion pictures. In: IEEE International Conference on Acoustics, Speech and Signal Processing, 1992 ICASSP-92 (Cat. No.92CH3103-9), 23–26 Mar 1992. IEEE, pp 453–456
30. Lee J, Moon T, Song K (2005) Hash aided motion estimation: various approaches. Tech. rep., Stanford University
31. Lee SH, Yang S (2002) Adaptive motion-compensated frame rate up-conversion. *Electron Lett* 38(10):451–452
32. Li Z (2005) New methods for motion estimation with applications to low complexity video compression. Doctoral dissertation
33. Li Z, Delp EJ (2005) Wyner-ziv video side estimator: conventional motion search methods revisited. In: IEEE International Conference on Image Processing 2005, ICIP 2005, 11–14 Sept 2005, vol 1. IEEE, pp 825–828
34. Li Z, Liu L, Delp EJ (2007) Rate distortion analysis of motion side estimation in wyner-ziv video coding. *IEEE Trans Image Process* 16(1):98–113
35. Lin Y, Song Y, Yeh S (2005) Scalable feature extraction for remote motion estimation. Tech. rep., Stanford University
36. Liu W, Dong L, Zeng W (2008) Wyner-ziv video coding with multi-resolution motion refinement: theoretical analysis and practical significance. In: Visual Communications and Image Processing 2008, VCIP 2008, vol 6822. SPIE, pp 1–11
37. Liu W, Dong L, Zeng W (2010) Motion refinement based progressive side-information estimation for wyner-ziv video coding. *IEEE Trans Circuits Syst Video Technol* 20(12):1863–1875
38. Macchiavello B, De Queiroz RL, Mukherjee D (2006) Motion-based side-information generation for a scalable wyner-ziv video coder. In: IEEE International Conference on Image Processing, ICIP 2007, vol 6. IEEE, pp VI413–VI416
39. Martins R, Brites C, Ascenso J, Pereira F (2009) Refining side information for improved transform domain wyner-ziv video coding. *IEEE Trans Circuits Syst Video Technol* 19(9):1327–1341
40. Martins R, Brites C, Ascenso J, Pereira F (2010) Statistical motion learning for improved transform domain wyner-ziv video coding. *IET Image Process* 4(1):28–41
41. Natario L, Brites C, Ascenso J, Pereira F (2006) Extrapolating side information for low-delay pixel-domain distributed video coding. In: 9th international workshop on visual content processing and representation, VLBV 2005, 15–16 Sept 2005. Lecture Notes in Computer Science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics). LNCS, vol 3893. Springer, pp 16–21
42. Petrazzuoli G, Cagnazzo M, Pesquet-Popescu B (2010) High order motion interpolation for side information improvement in dvc. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2010, IEEE, pp 2342–2345
43. Petrazzuoli G, Maugey T, Cagnazzo M, Pesquet-Popescu B (2010) Side information refinement for long duration gops in dvc. In: IEEE international workshop on Multimedia Signal Processing, MMSP 2010. IEEE Computer Society, pp 309–314
44. Pradhan SS, Ramchandran K (2003) Distributed source coding using syndromes (discus): design and construction. *IEEE Trans Inf Theory* 49(3):626–643
45. Puri R, Majumdar A, Ishwar P, Ramchandran K (2006) Distributed video coding in wireless sensor networks. *IEEE Signal Proc Mag* 23(4):94–106
46. Puri R, Majumdar A, Ramchandran K.: Prism: A video coding paradigm with motion estimation at the decoder. *IEEE Trans Image Process* 16(10):2436–2448 (2007)
47. Puri R, Ramchandran K (2002) Prism: a new robust video coding architecture based on distributed compression principles. In: Proceedings of the annual allerton conference on communication control and computing, vol 40. Citeseer, pp 586–595
48. Quazi AH (1981) An overview on the time delay estimate in active and passive systems for target localization. *IEEE Trans Acoust Speech Signal Process* 29(3):527–33
49. Slepian D, Wolf JK (1973) Noiseless coding of correlated information sources. *IEEE Trans Inf Theory* 19(4):471–480



50. Tagliasacchi M, Trapanese A, Tubaro S, Ascenso J, Brites C, Pereira F (2006) Exploiting spatial redundancy in pixel domain Wyner-Ziv video coding. In: IEEE International Conference on Image Processing, ICIP 2006, 8–11 Oct 2006. IEEE, pp 253–256
51. Tagliasacchi M, Tubaro S (2007) Hash-based motion modeling in wyner-ziv video coding. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2007 (IEEE Cat. No. 07CH37846). IEEE, pp 509–512
52. Varodayan D, Aaron A, Girod B (2005) Rate-adaptive distributed source coding using low-density parity-check codes. In: 39th asilomar conference on signals, systems and computer (IEEE Cat. No. 05CH37761), 30 Oct–2 Nov 2005. IEEE, pp 1203–1207
53. Varodayan D, Chen D, Flierl M, Girod B (2008) Wyner-Ziv coding of video with unsupervised motion vector learning. *Signal Process Image Commun* 23(5):369–378
54. Wang D, Vincent A, Blanchfield P, Klepko R (2010) Motion-compensated frame rate up-conversion-part ii: new algorithms for frame interpolation. *IEEE Trans Broadcast* 56(2):142–149
55. Wang Y, Jeong J, Wu C, Xiao S (2008) An approach to side information estimation for wyner-ziv video coding. In: 2008 international Congress on Image and Signal Processing (CISP), 27–30 May 2008. IEEE, pp 405–410
56. Weerakkody WARJ, Fernando WAC, Martinez JL, Cuenca P, Quiles F (2007) An iterative refinement technique for side information generation in dvc. In: IEEE International Conference on Multimedia and Expo, ICME 2007. IEEE pp 164–167
57. Wei-Jung C, Karam LJ, Abousleman GP (2006) Distributed video coding with 3d recursive search block matching. In: IEEE international symposium on circuits and systems, ISCAS 2006 (IEEE Cat. No. 06CH37717C). IEEE, pp 5415–5418
58. Wyner AD (1974) Recent results in the shannon theory. *IEEE Trans Inf Theory* IT-20(1):2–10
59. Wyner AD, Ziv J (1976) The rate-distortion function for source coding with side information at the decoder. *IEEE Trans Inf Theory* IT-22(1):1–10
60. Xiong Z, Liveris AD, Cheng S (2004) Distributed source coding for sensor networks. *IEEE Signal Proc Mag* 21(5):80–94
61. Yangli W, Jechang J, Chengke W, Song X (2007) Wyner-ziv video coding with spatio-temporal side information. In: IEEE International Conference on Multimedia and Expo, ICME 2007. IEEE, pp 132–135
62. Zhang Y, Zhao D, Ma S, Wang R, Gao W (2009) An auto-regressive model for improved low delay distributed video coding. In: Visual Communications and Image Processing 2009, VCIP 2009, vol 7257. SPIE, pp 18–22
63. Zhang Y, Zhu C (2007) Full search of side information in distributed video coding. In: Image and graphics, 2007. IEEE, pp 246–249



**Yuan Jia** received the B.S. degree from Xi'an Jiaotong University in 2002, and M.S. degree from Xidian University in 2006. She is currently working toward the Ph.D. degree under the direction of Prof. Jiandong Li in Xidian University, China. She is currently a lecturer at the State Key Laboratory of Integrated Service Networks of Xidian University. Her research interests include multimedia communication, wireless communication, multiple description video coding and distributed video coding.



**Yangli Wang** received the B.S. degree from Peking University in 1984, M.S. degree from Xidian University in 1991, and Ph.D. degree also from Xidian University in 2000. He is currently a Professor at the State Key Laboratory of Integrated Service Networks of Xidian University. His research interests are in the fields of image and video compression and communication, distributed video coding, and image processing.



**Rui Song** received the B.S., M.S. and Ph.D. degrees from Xidian University in 2003, 2006, and 2009 respectively. He is currently an associate professor at the State Key Laboratory of Integrated Service Networks of Xidian University. His research interests include image and video compression, super resolution image reconstruction, sparse representation of image, and VLSI architecture of multimedia chips.



**Jiandong Li** (SM'05) received the B.S., M.S., and Ph.D. degrees in communications and electronic system from Xidian University, Xi'an, China, in 1982, 1985, and 1991, respectively. He has been with Xidian University since 1985, where he was an Associate Professor from 1990 to 1994 and has been a Professor since 1994, a Ph.D. Student Supervisor since 1995, and the Dean of the School of Telecommunication Engineering since 1997. His current research interests and projects consist of mobile communications, broadband wireless internet protocol systems, ad hoc networks, cognitive and software radio, packet radio networks, self-organizing networks, and game theory for wireless networks. These research projects are funded by the 863 High Tech Project, the National Natural Science Foundation of China, the National Science Fund for Distinguished Young Scholars, the Teaching and Research Award Program for Outstanding Young Teachers in Higher Education Institutions of the Ministry of Education, China, and the Ministry of Information Industry. He is a Senior Member of the China Institute of Electronics and a Fellow of the China Institute of Communication. He was a member of the PCN Specialist Group for the China 863 Communication High Technology Program between January 1993 and October 1994 and from 1999 to 2000. He is also a member of the Communication Specialist Group for The Ministry of Information Industry.