EPITOME INPAINTING WITH IN-LOOP RESIDUE CODING FOR IMAGE COMPRESSION

Safa Chérigui ^{1,2}, *Martin Alain* ^{1,2}, *Christine Guillemot* ^{1}, *Dominique Thoreau* ^{2} *and Philippe Guillotel* ^{2}

^{1} *INRIA*
Campus de Beaulieu, 35042 Rennes Cedex France
firstname.lastname@irisa.fr

^{2} *Technicolor Research and Innovation*
Av. des Champs Blancs, 35576 Cesson-Sévigné France
firstname.lastname@technicolor.com

ABSTRACT

This paper describes a novel image coding scheme based on epitome inpainting. An epitome containing a factorized texture representation of the image is first coded and transmitted. The decoded epitome is then inpainted by propagating its structure and texture with extended H.264 Intra directional prediction modes and advanced neighbor embedding methods. The blocks to be inpainted are processed according to an adaptive scanning order which adapts to the image content and gives a higher priority to the blocks having strong structures. The inpainting step has been incorporated in the coding loop of an H.264 encoder, leading to a bitrate saving of up to 48.12% with respect to H.264 intra.

Index Terms: Intra coding, epitome, inpainting, H.264.

1. INTRODUCTION

The concept of epitome was first introduced in [1] by Jojic et al. as a condensed representation of the image texture. Different types of epitomes have then been proposed in the literature. While the epitome in [1] is obtained using a patch-based probability model learned from the input image, the authors in [2] construct instead the epitome with self-similar or repeatable texture patterns found using the KLT algorithm [3][4]. The epitome is in this case the union of epitome charts which are pieces of repeatable textures found in the image. The epitome is associated with a transform map that links patches in the epitome to original patches in the input image. The epitome construction has been improved in [5] by using a bi-directional similarity measure based on the notion of “completeness” and “coherence”.

In this paper we propose a novel image coding technique based on epitome inpainting. Note that the use of epitomes has already been considered in the literature for image coding. The authors in [2] encode the epitome and its associated transform map, which are then used by the receiver to reconstruct the image. The use of the epitome of [1] is also considered in [6][7] together with template matching for Intra prediction. In [8], the epitome constructed with a simple block matching, together with the associated map, is used to generate a reference image in the same vein as in H.264 temporal prediction.

The coding scheme described here significantly differs from previous approaches, since it relies on the inpainting of the epitome **with in-loop residue coding**. Note that in this regard, the method also differs from classical inpainting coding schemes [9][10], which only “fill holes” at the decoder side. In these schemes, the inpainted regions quality is traditionally assessed using a subjective measure, while our method ensures that an objective measure can be used. The approach implies a significant change with respect to classical coding architectures, since the image blocks are not processed any more in a raster scan order but instead according to an adaptive order which depends on the local image structure. The local image structure is taken into account in a priority term which allows deciding which block of size 8×8 should be inpainted first. A new scanning order of all the other blocks in the macro-block has then been defined depending on the position of the first inpainted block based on the priority term. The adaptive scanning order is needed to take into account the ”arbitrary” shape of the local neighborhood of the block to be inpainted. It also aims at giving a higher priority to blocks for which the local neighborhood has stronger structures.

In a nutshell, an epitome is first constructed using the method described in [8]. This epitome is coded with an intra coding scheme of type [8] and transmitted. The epitome texture and structure is block-per-block propagated using a set of possible and rate-distortion competing modes which include extended H.264 Intra directional prediction modes as well extra modes based on Locally Linear Embedding (LLE) [11] and the optimized Map-Aided Locally Linear Embedding technique [12]. Once a block has been predicted, its residue is computed and added to the predictor to reconstruct the current block before continuing the inpainting process. Note that here the transform map is not used. Experimental results show a significant PSNR-rate gain (up to 48.12% using the Bjontegaard measure).

The rest of this paper is organized as follows. Section 2 reviews the related work on intra image coding and epitome construction. Section 3 describes the proposed coding scheme based on epitomes with in-loop residue coding. The experimental results are discussed in section 4.

2. BACKGROUND

2.1. Intra image coding

The H.264/AVC standard defines three types of intra prediction modes depending on the block size: the Intra-4x4, Intra-8x8 and Intra-16x16 [13]. For Intra prediction of 16x16 blocks, four intra prediction modes can be used while for Intra prediction of 4x4 and 8x8 blocks, nine modes have been defined. In addition to the so-called “DC” mode which consists in predicting the entire block from the mean of neighboring pixels, eight directional prediction modes have been specified. The prediction is done by simply propagating (or interpolating) the neighboring pixel values along the specified direction. The best mode is then selected using a rate distortion optimization (RDO).

The improvement of these Intra prediction modes has received a lot of attention in the past years. For example, the authors in [14] consider bi-directional intra prediction (BIP) which combines two existing prediction modes. To be able to do so an alternative coding order was proposed **within a macroblock**, but the macroblocks are still scanned in a raster order. An adaptive scanning order for blocks and MB was introduced in [15] in order to maximizes the number of reference pixels available for intra prediction. The intra coding efficiency has also been improved by transposing temporal prediction methods, using block-matching (BM) [16] or template matching (TM) [17], to Intra prediction. An alternative spatial prediction algorithm based on TM has been described in [18]. A template is formed by previously encoded pixels in the close neighborhood of the block to be predicted. The best match between the template of the block to be predicted and candidate texture patches with the same shape, within a causal search window, gives the predictor for the current block to be coded. This approach has later been improved in [19] by averaging multiple TM predictors. Further PSNR-rate gains have been achieved by considering linear combinations of multiple patches (predictors) using for example the locally linear embedding (LLE) and optimized Map aided locally linear embedding (oMA-LLE) methods [11][12].

2.2. Epitome construction

The epitome construction method which we use here is the one described in [8] and which we briefly recall here. Note that one could also consider the method of [2] using the KLT algorithm instead of a simple BM algorithm to find the repeatable pieces of texture. But this second method has to retain in the transform map ϕ all the information about translations, rotation angles, scaling factors and reflections, which can make it really heavy and CPU time consuming.

In these approaches, the image is described by its epitome E and a transform (or correspondence) map ϕ. The epitome E is composed of disjoint pieces of texture called epitome charts. Each epitome chart corresponds to a specific region in the image. The set of all the epitome charts constitutes the epitome E.

The algorithm in [8] proceeds by first searching for self-similarities within the image. For each non-overlapping block in the input image, the method searches for the set of patches in the image with similar content. That is, for each block $B_i \in I$, a list of best matching patches (which may overlap) is constructed. Once all the lists have been created for all the image blocks, a dual list of the best matching blocks to a given patch $M(i, j)$ is also constructed.

The first epitome chart is then initialized by the texture patch which allows reconstructing the image with the highest fidelity. All the image blocks having this patch as one of the best matching patches in the list are replaced by this matching patch. The mean square reconstruction error is computed by setting the pixels which are not reconstructed to zero. In others words, the initialization of an epitome chart consists in searching in all the lists, the texture patch so far not used at the previous iterations that allows the minimization of the reconstruction error (the MSE).

Once initialized, the epitome chart has to be progressively grown in a way which optimizes the trade-off between the MSE of the reconstructed image and the number of pixels (as a rough measure of rate cost) in the epitome. The set of matched patches which overlap the current epitome chart is identified. For each possible candidate, the image blocks which can be reconstructed by the enlarged area of the epitome chart are determined from the corresponding list. The candidate minimizing a rate-distortion cost (that is decreasing the most the MSE under the constrained of a limited number of additional pixels in the epitome chart) is selected. The current chart keeps on growing until there are no more matched patches overlapping the current chart which can represent other blocks. When the current chart cannot be extended anymore and when the whole image is not yet full represented by the current epitome, another epitome chart is initialized at a new location in the image. The process ends when the whole image is reconstructed by the epitome.

Note that the epitome charts are padded in order to be aligned with the block structure (here with the 8×8 block grid) of the codec used.

2.3. Image coding based on epitome

As mentioned in the introduction, epitomes have already been considered for image coding, the epitome being simply used as extra reference texture for classical TM-based Intra prediction [6], or for generating a reference image which is then used for H.264 interframe prediction [8]. In [2], the epitome is used only (i.e. without transmitting a prediction residue) with the correspondence map to reconstruct the image on the receiver side. The PSNR-rate performance of this scheme, with no residue coding, is however limited, as shown in [8]. The goal was here to push further the idea of epitome-based image coding, but considering a completely different approach (using inpainting with in-loop residue coding) which does not require the transmission of the correspondence map. This new approach implies a radical change in the coding architecture.

3. EPITOME INPAINTING WITH IN-LOOP RESIDUE CODING

The main steps of the coding scheme are depicted in Fig. 1. The epitome is generated using the method described in section 2.2 and encoded **in a raster scan order** using the image coding algorithm described in [12] which is based on an H.264 Intra coder plus extra Intra prediction modes have been added, using block matching within the same frame, as well as locally linear embedding (LLE) based methods. The LLE based methods can be regarded as extensions of TM using multiple patches which are linearly combined in order to best approximate the template of the block to be predicted.

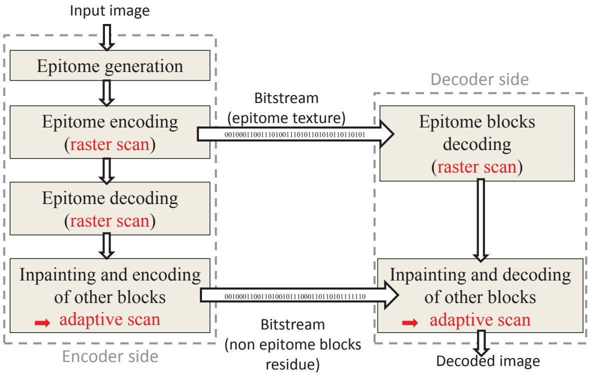

Fig. 1. Main steps of the coding scheme.

3.1. Adaptive processing order

The epitome, corresponding to pieces of the image, is decoded and used to predict by inpainting the missing blocks. Inpainting implies changing the scanning order to the blocks in the encoder. An **adaptive scanning order** is used, based on the priority order proposed in [20] for object removal, adapted here to fit with the block structure of the encoder. Blocks surrounded by strong linear structures and a larger number of reliable pixels will have the highest priority and be processed first.

The algorithm first inpaints the block Ψ_B having the higher priority $P(\Psi_B)$ defined as follows. Let Ψ_B be a block located on the contour (also called front line) $\delta\Omega$ of the target region Ω to be filled, its priority is defined as

$$P(\Psi_B) = \alpha \times \log(C(\Psi_B)) + (1-\alpha) \times \log(D(\Psi_B)) \qquad (1)$$

The first term $C(\Psi_B)$ is a confidence term defined as the ratio of known pixels in the neighborhood (template) Ψ_T of the block Ψ_B with respect to the total number of pixels within the block plus its template Ψ_{B+T}. The template can have different shapes and sizes, hence $\Psi_T = \Psi_{T_i}, i = 0 \ldots N-1$ with Ψ_{T_i} denoting one of the shapes and sizes shown in Fig. 2. The confidence term $C(\Psi_B)$ is computed as:

$$C(\Psi_B) = \frac{card(\Psi_T)}{card(\Psi_{B+T})} \qquad (2)$$

This term thus favors neighborhoods with a high number of known pixels.

The second term $D(\Psi_B)$ gives the maximum strength of the isophotes hitting all pixels on the front line separating the template and the block to be inpainted. This term is computed as

$$D(\Psi_B) = \max(D(p), p \in \delta\Omega \cap \Psi_{B+T}) \qquad (3)$$

where $D(p)$ computes on each pixel on the front line the inner product between the perpendicular to the image gradient and the normal to the front line as

$$D(p) = \frac{|\nabla I_p^{\perp} \cdot n_p|}{\sigma} \qquad (4)$$

This term is higher when the isophote (the orthogonal to the gradient) is perpendicular to the front line. The parameter σ is a normalization factor (*e.g.*, σ = 255 for a typical grey-level image), and n_p is a unit vector orthogonal to the front line $\delta\Omega$ in the point p.

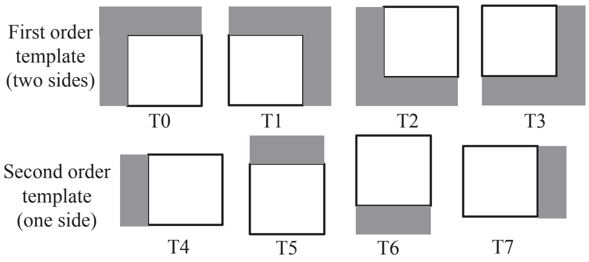
First order template (two sides)
T0
T1
T2
T3
Second order template (one side)
T4
T5
T6
T7

Fig. 2. Set of templates (in grey) used to compute the priority of a block (in white).

The parameter α controls the weight given to each of the terms $C(\Psi_B)$ and $D(\Psi_B)$. The priority function $P(\Psi_B)$ is computed only on the previously encoded pixels, which ensures that the same processing order can be derived at the decoder side without the need to transmit any side information.

Figure 3 displays an example of adaptive block processing order based on the proposed priority cost function.

3.2. Inpainting or prediction modes

The inpainting of the epitome is performed by propagating texture and structure using methods similar to intra prediction methods, adapted here to account for the fact that the blocks are not processed any more in a raster scan order. The methods used here are based on extended multi-directional H.264 Intra prediction modes in competition with extra modes based on block matching (BM) and LLE/oMALLE.

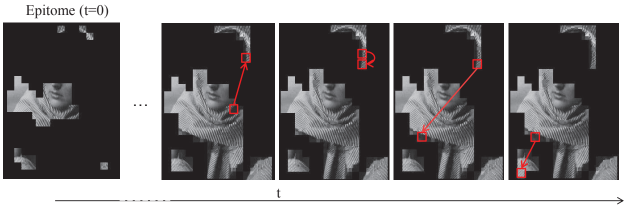

Fig. 3. Example of adaptive block filling order on input image Barbara with the proposed epitome inpainting method.

The H.264 directional modes have been adapted so that they follow the orientation of the template of the current block (see Fig. 2). Here we consider the four directions of the first order templates defined as T0, T1, T2 and T3 in Fig. 2. The direction of T1 is the one already used to define the directional modes in H.264. The modes associated to the three others directions are obtained by applying respectively three rotations of 90°, 180° and 270° on the classical intra modes. Fig. 4 gives the directional modes obtained for the T1 template (upper right neighborhood). The BM algorithm does not rely on the template form so it is not affected. The only change is the shape of the search window SW that is defined in the causal part of the image. Similarly, the LLE-based method searches the K-NN blocks in the search window by computing distances between templates [11], hence here with shapes which vary from one block to the other.

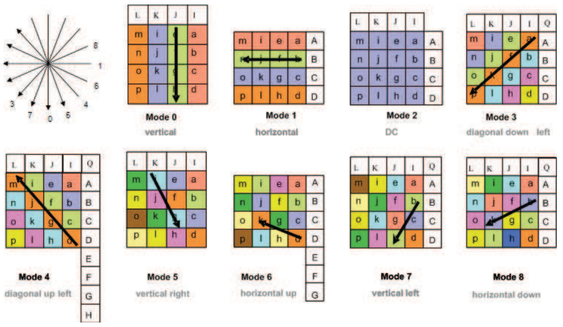

Fig. 4. Adapted directional modes for intra prediction with an upper right neighborhood.

Once the best mode has been selected and the prediction obtained, the residue of the block is computed, transformed, quantized and entropy coded with the H.264 encoding tools. At the decoder side the residue is obtained by inverse transform and dequantization. The block is immediately reconstructed by adding the decoded residue to the prediction. The reconstructed block is then available in the causal part of the image and the new front line is taken into account when computing the new filling order for the inpainting of the next block.

3.3. Integration in an H.264 codec

To validate the approach, the inpainting step has been integrated in an H.264 codec with the block size set to 8x8. The epitome is padded to fit with the 8x8 block grid and the prediction methods described above are applied. The best inpainting mode between H.264 extended intra prediction modes, BM, LLE and oMALLE, is selected based on an RDO criterion.

However, the H.264 prediction step operates at a macroblock (MB) level. Hence, the algorithm inpaints the first 8×8 block with the highest priority term within the macro-block. The processing order of the remaining blocks of the macro-block is then defined according to the position of the first processed block and the orientation of the causal neighborhood. The scanning order within the MB is obtained by applying three rotations of 90°, 180° and 270° on the order defined in H.264 (see Fig. 5). Note that if several templates are available to encode a block within a MB, the same priority criterion as for the epitome inpainting is used to determine the one for encoding.

Finally the modification of the raster scan order implies major changes for the entropy coding (CABAC) for which the context has also been adapted.

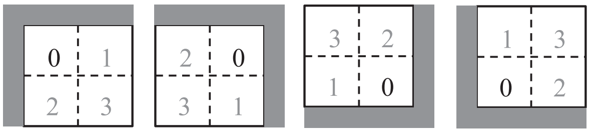

Fig. 5. Scanning order within a MB. In grey is represented the causal part of the image neighboring the MB.

4. SIMULATIONS AND RESULTS

The algorithm was implemented in the Key Technical Area (KTA) software [21]. This software was also used as reference for the simulations. Objective assessments of the method were obtained using the Bjontegaard metric [22] at low (Quantization parameter (QP) = 26, 31, 36, 41) and high bitrates (QP = 16, 21, 26, 31). Note that in the following results, the measured bitstream does take into account both the epitome texture and the non-epitome block residue bitstreams. The method was tested on a set of four images: Barbara, Building, Spincalendar and Wool.

First, the scheme based on epitome inpainting with in-loop residue encoding is compared to the classical H.264 intra coding scheme. Table 1 shows the Bjontegaard measures obtained with the priority order with $\alpha = 0.5$ and $\alpha = 0.9$ (Eq. 1). Results show a drastic reduction of the bitrate with an average of 32.34 percent of rate reduction at low bitrate and 21.99 at high bitrate when $\alpha = 0.5$. The results are even better when $\alpha = 0.9$.

Second, the method is compared to a modified H.264 codec integrating the additional BM and LLE/oMALLE modes used with a raster scan order [12]. The performances of this reference modified codec are given in the column "Raster scan" of Table 2 in comparison with a standard H.264 codec using Bjontegaard measures. The use of the BM and LLE/oMALE modes already gives interesting results with an average of 33.54% and 22.28% of bitrate reduction at low and high bitrates respectively. However, one can observe an extra bitrate reduction when using the inpainting with the adaptive scanning order.

Figure 6 displays partially inpainted prediction images of Spincalendar obtained at the same iteration but using two different QPs (i.e. different bit rates) to encode the residue. One can observe that at high bit rates, the image structures are obviously better reconstructed, hence lead to a different scanning order. However in both cases, overall, the scanning order follows the angles of the calendar strong edges.

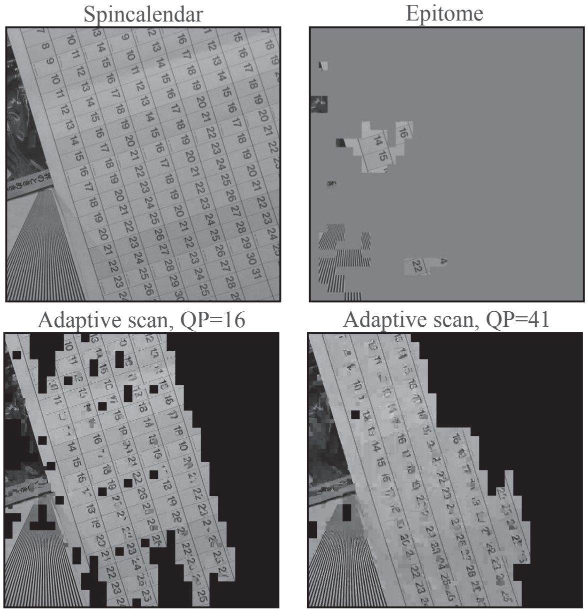

Fig. 6. On top row: input image and its epitome. On bottom row: Prediction images during encoding, at the same iteration but different QPs.

Table 1. Comparison between the inpainting scheme integrated with neighbor embedding and H.264 intra modes

Adaptive scanning	Low bitrates		High bitrates	
	$\alpha = 0.5$	$\alpha = 0.9$	$\alpha = 0.5$	$\alpha = 0.9$
	dB % rate	dB % rate	dB % rate	dB % rate
Barbara	0.98 -14.43	1.13 -16.16	0.81 -9.37	0.92 -10.58
Building	4.16 -46.11	4.40 -48.12	4.33 -32.79	4.48 -33.89
Spincalendar	2.87 -37.28	3.15 -39.99	2.74 -27.61	2.97 -29.92
Wool	2.19 -31.55	2.33 -33.39	1.69 -18.21	1.81 -19.39
Average	**2.55 -32.34**	**2.75 -34.42**	**2.39 -21.99**	**2.54 -23.44**

Table 2. Comparison between the inpainting scheme integrated with neighbor embedding and H.264 intra modes with additional neighbor embedding

	Low bitrates		High bitrates	
	Raster scan	Adapt. scan	Raster scan	Adapt. scan
	dB % rate	dB % rate	dB % rate	dB % rate
Barbara	1.19 -17.07	1.13 -16.16	0.92 -10.53	0.92 -10.58
Building	4.23 -46.78	4.40 -48.12	4.32 -32.80	4.48 -33.89
Spincalendar	2.87 -37.59	3.15 -39.99	2.64 -26.83	2.97 -29.92
Wool	2.28 -32.73	2.33 -33.39	1.76 -18.93	1.81 -19.39
Average	**2.64 -33.54**	**2.75 -34.42**	**2.41 -22.27**	**2.54 -23.44**

5. CONCLUSION

In this paper we introduced a novel compression scheme that relies on the use of an epitome, different prediction methods such as BM, LLE and more importantly a new adaptive block processing order. This new block processing order allows a better adaptation to the content of the image that benefits the coding performances. The multi-patches methods such as LLE also greatly benefit from the use of the epitome. When introduced in an H.264 codec the method was shown to be very efficient especially at low bitrates with up to 48.12 % of bitrate saving.

6. REFERENCES

[1] N. Jojic, B. J. Frey, and A. Kannan, "Epitomic analysis of appearance and shape", in *Proc. IEEE Conf. Comput. Vis. (ICCV)*, 2003, pp. 34–41.

[2] H. Wang, Y. Wexler, E. Ofek, and H. Hoppe, "Factoring repeated content within and among images.", *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 27, no. 3, Aug. 2008.

[3] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision.", in *Proceedings of Imaging Understanding Workshop*, 1981.

[4] J. Shi and C. Tomasi, "Good features to track.", in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 1994.

[5] D. Simakov, Y. Caspi, E. Shechtman, and M. Irani, "Summarizing visual data using bidirectional similarity", in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2008, pp. 1–8.

[6] Q. Wang, R. Hu, and Z. Wang, "Improving intra coding in H.264/AVC by image epitome", in *Advances in Multimedia Information Processing (PCM)*, 2009, pp. 190–200.

[7] Q. Wang, Z. Wang, and B. Hang, "Intra coding and refresh based on video epitomic analysis", in *IEEE Int. Conf. on Multimedia and Expo (ICME)*, 2010, pp. 452–455.

[8] S. Cherigui, C. Guillemot, D. Thoreau, and P. Guillotel, "Epitome-based image compression using translational sub-pel mapping", in *IEEE Int. Workshop on Multimedia Signal Processing (MMSP)*, Oct. 2011, pp. 1–6.

[9] D. Liu, F. Wu, S. Li, and Y. Q. Zhang, "Image compression with edge-based inpainting", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17-10, pp. 1273–1287, Oct. 2007.

[10] Z. Xiong, X. Sun, and F. Wu, "Block-based image compression with parameter-assistant inpainting", *IEEE Trans. Image Proc.*, vol. 19-6, pp. 1651–1657, Jun. 2010.

[11] M. Turkan and C. Guillemot, "Image prediction based on neighbor embedding methods", *IEEE Trans. on IP*, vol. 21, pp. 1885–1898, 2011.

[12] S. Cherigui, C. Guillemot, D. Thoreau, P. Guillotel, and P. Perez, "Map-aided locally linear embedding for image prediction", in *IEEE Int. Conf. Image Process. (ICIP)*, 2012, pp. 2909–2912.

[13] T. Wiegand, G. J. Sullivan, B. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13-7, pp. 560–576, Jul. 2003.

[14] T. Shiodera, A. Tanizawa, and T. Chujoh, "Bidirectional intra prediction", in *ITU-T SG16/Q.6 VCEG, VCEG-AE14, Marrakech, Morocco*, January 2007.

[15] T. Palfner and T. Wedi, "Advanced scanning for intra frame coding", in *IEEE Int. Conf. Image Process. (ICIP)*, 2006, pp. 3121–3124.

[16] S.L. Yu and C. Chrysafis, "New intra prediction using intra macroblock motion compensation", in *JVT-C151, 3rd meeting of Joint Video Team (JVT)*, May 2002.

[17] S. Cherigui, C. Guillemot, D. Thoreau, P. Guillotel, and P. Perez, "Hybrid template and block matching algorithm for image intra prediction", in *IEEE Conf. Acoustics, Speech, and Signal Process. (ICASSP)*, 2012, pp. 781–784.

[18] T. K. Tan, C. S. Boon, and Y. Suzuki, "Intra prediction by template matching", in *IEEE Int. Conf. Image Process. (ICIP)*, 2006, pp. 1693–1696.

[19] T. K. Tan, C. S. Boon, and Y. Suzuki, "Intra prediction by averaged template matching predictors", in *IEEE Conf. Consumer Comm. Network. Conf. (CCNC)*, 2007, pp. 405–409.

[20] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by examplar-based image inpainting", *IEEE Trans. Image Proc.*, vol. 13-9, pp. 1200–1212, Sept. 2004.

[21] "KTA Software, Ver. JM 14.2 KTA 1.0. Available: http://iphome.hhi.de/suehring/tml/download/KTA/".

[22] G. Bjontegaard, "Calculation of average psnr differences between rd curves", in *document VCEG-M33, ITU-T VCEG Meeting*, April 2001.