Epitome-based image compression using translational sub-pel mapping

Safa Cherigui [1,2], Christine Guillemot [1], Dominique Thoreau [2], Philippe Guillotel [2] and Patrick Perez [2]

[1] *INRIA/IRISA, Université de Rennes 1*
Campus de Beaulieu, 35042 Rennes Cedex France
firstname.lastname@irisa.fr

[2] *Technicolor Research and Innovation*
Av. Belle Fontaine, 35576 Cesson-Sévigné France
firstname.lastname@technicolor.com

***Abstract*—This paper addresses the problem of epitome construction for image compression. An optimized epitome construction method is first described, where the epitome and the associated image reconstruction, are both successively performed at full pel and sub-pel accuracy. The resulting complete still image compression scheme is then discussed with details on some innovative tools. The PSNR-rate performance achieved with this epitome-based compression method is significantly higher than the one obtained with H.264 Intra and with state of the art epitome construction method. A bit-rate saving up to 16% comparatively to H.264 Intra is achieved.**

Index Terms: Epitome, texture reconstruction, prediction, intra coding, H.264.

I. Introduction

This paper presents an image compression algorithm based on the concept of "epitome" first introduced by Jojic et al. [1]. An epitome is a condensed representation of an image (or a video) signal containing the essence of the textural properties of this image. Different forms of epitomes have been proposed, such as an image summary of high "completeness" [2] or a patch-based probability model learned either from still image patches [1] or from space-time texture cubes taken from the input video [3]. These probability models together with appropriate inference algorithms, are useful for content analysis inpainting or super-resolution. Another family of approaches makes use of computer vision techniques, like the KLT tracking algorithm, in order to recover self similarities within and across images [4]. In parallel, another type of approach has been introduced in [5] which aims at extracting epitome-like signatures from images using sparse coding and dictionary learning.

This paper considers approaches which aim at tracking self-similarities within an image, inspired by [4] and introduces a novel epitome construction method based on a block matching (BM) algorithm. The epitome is constructed from disjoint pieces of texture (coined "epitome charts" in [4]) taken from the original image and a transform map which contains translational parameters. Those parameters keep track of the correspondences between each block of the input image and a block of the epitome. An intra image compression scheme based on the epitome is then described. Most intra compression schemes are built on the local correlation between local neighborhoods in order to reduce the redundancy and minimize the encoded information of an image. However, they do not exploit the correlation of patches across nonlocal neighborhoods, except the alternative intra prediction based on Template Matching (TM) [6]. The TM method tries to find the best match to the causal neighborhood of the block to be predicted. Alternative methods based on sparse signal approximation have been considered in [7][8]. Fractal based compression methods have also been introduced in [9]. This approach consists in detecting the recurrence of texture patterns and tends to remove the redundancy within the image. Fractal compression is described as an iterated function system that approximate an image as the attractor of a set of recursive transformations on both geometry and color. Note that this method is only based on a partitioning of the original image and affine transformations. Our epitome approach can be also an alternate solution, which aims at reducing the transmission of redundant information by exploiting similar areas of an image. In the experiments reported here, we search to represent accurately the signal instead of determining the general appearance of the pattern like it is performed in texture synthesis algorithms [10].

Although earlier work [2][3] shows that epitomes are powerful tools for segmentation, denoising, recognition, indexing, and texture synthesis, further considerations, such as their mean description length, or robustness to quantization noise, need to be taken into account in a compression context. Indeed, the epitome needs to be stored and/or transmitted, together with the transformation map, when used in a video coding context. Both the epitome and the transformation map are then used by the decoder to predict or reconstruct the input image. First Intra prediction methods based on image epitome have been introduced in [11] where a prediction for each block is generated from the image epitome by Template Matching. An intra coding method based on video epitomic analysis has also been proposed in [12] where the transform map (vectors) is coded with fixed length code which are determined by the length and width of image epitome. The epitome image used by these two approaches [11][12] is based on EM (Expectation Maximization) algorithm [1] with a pyramidal approach. This kind of epitome image preserves the

978-1-4577-1434-4/11/$26.00 ©2011 IEEE

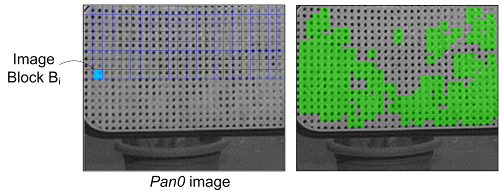


Fig. 1. A given image block B_i shown in blue and its matching patches highlighted in green with error tolerance ϵ.

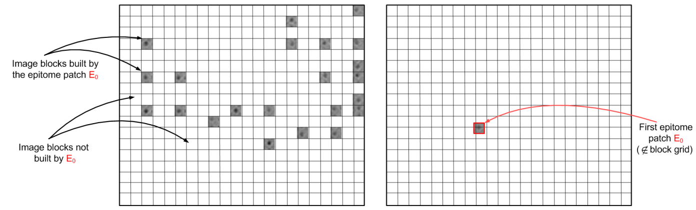


Fig. 2. Initialization chart step: the image subset reconstructed by the current chart (left), the current epitome initially composed of a single patch.

global texture and shape characteristics of original image but introduces undesired visual artefacts (e.g. additional patches which were not in the input image). The epitome construction approach described in this paper, similar to [4], does not introduce such artefacts since the epitome is only composed of patches taken from the original image. Besides, the epitome construction method described here has first been developed with full pixel accuracy and has then been extended to half-pel and quarter-pel accuracy. The rate-distortion performance of the resulting image compression algorithm is compared to the H.264 standard [13], which uses closed-loop spatial prediction with several directional modes. The prediction in H.264 is done by simply "propagating" the pixel values along the specified direction. This approach is suitable in presence of contours, the directional mode chosen corresponds to the orientation of the contour. However, it fails in more complex textured areas. The epitome-based approach is better suited to cope with complex textures since 2D patches are considered instead of oriented mono-directional interpolations.

The remainder of the article is organized as follows. The method to construct the epitome is first presented in section II. In section III, the proposed compression algorithm based on epitome approach is introduced. Different variants of our approach are compared to H.264 Intra standard and to the state of the art epitome analysis/synthesis solution from [4] in section IV.

II. Epitome construction and image reconstruction

An image is described by its epitome E and a transform map (or assignation map) ϕ. The epitome contains a set of charts that originates from the input image. The transform map indicates for each block of the image which patch in the texture epitome is used for its reconstruction.

Let Y denote the original image of size $N \times M$ pixels. The image Y is divided into a regular grid of blocks, and each block B_i is approximated from an epitome patch via an assignation map φ_i. The construction procedure is basically composed of three steps: 1)- finding self-similarities, 2)- creating epitome charts and 3)- improving the quality of reconstruction by further searching for best matching and by updating accordingly the transform map.

A. *Finding self-similarities*

The epitome construction method consists first in finding self-similarities within the image. Indeed, for each block in the input image, one searches the set of patches in the image with similar content. That is, for each block $B_i \in Y$ ($B_i \in$ block grid of image), we will determine from the input image a list $\mathrm{L}_{match}(B_i) = \{M_{i,0}, M_{i,1}, ...\}$ of matches (or matched patches) $M_{i,l}$ that approximate B_i with a given error tolerance ϵ. In the current implementation, the matching is performed with a block matching algorithm using an average Euclidian distance. An exhaustive search is performed on the entire image. Once all the Match lists have been created for the set of image blocks, a new list $\mathrm{L}^{'}_{match}(M_{j,l})$, indicating the set of image blocks that could be represented by a matched patch $M_{j,l}$ is built. Note that all the matching blocks $M_{j,l}$ found during the full search step are not necessarily aligned with the block grid of the image and thus belong to the "pixel grid". Fig. 1 shows the set of matched patches found for a given image block B_i in the " *Pan0* " image.

B. *Creating epitome charts*

The second step of the approach consists in selecting texture patches from the original image in order to construct the epitome charts, the set of all the epitome charts forms the epitome E. Each epitome chart represents specific areas of the image.

Initialization: Let n be the index of the current epitome chart EC_n. Initially, n is set to zero. Let $Y \in R^{N\times M}$ and $Y^{'} \in R^{N\times M}$ denote the input image and the image reconstructed by a given texture patch respectively. the epitome chart EC_n is initialized by the most representative texture patch of remaining no reconstructed image blocks. To initialize an epitome chart, we introduce the following selection criterion based on the minimization of the Mean Square Errors (MSE) criterion:

$$\min\left(\frac{\sum_{i=0}^{N}\sum_{j=0}^{M}\left(Y_{i,j}-Y^{'}_{i,j}\right)}{N\times M}\right) \tag{1}$$

The selected criterion (1) considers the prediction errors on the whole image. That is, this criterion is applied not only to image blocks that are approximated by a given texture patch but also to the image blocks which are not approximated by this patch. In our current implementation, a zero value is assigned to image pixels that have not reconstructed by this patch when computing the image reconstruction error. Thus, this criterion enables the current epitome chart to be

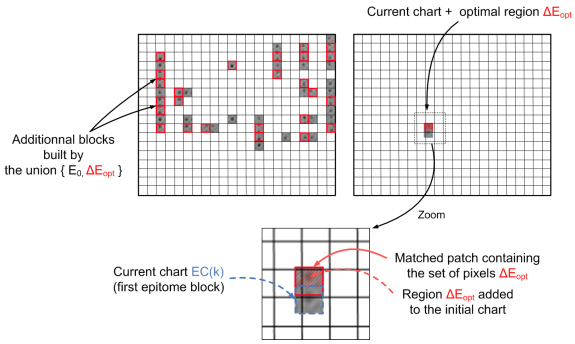


Fig. 3. Extension chart step: the image subset reconstructed by the current chart epitome and the optimal region ΔE_{opt} (left), the current epitome chart extended by the optimal region ΔE_{opt} (right).

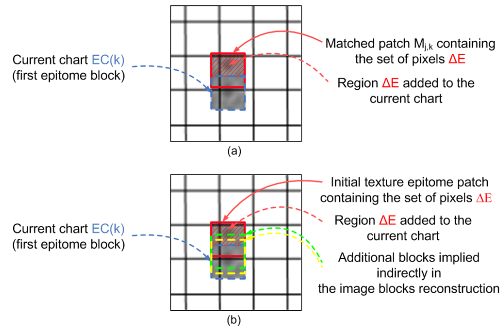


Fig. 4. Epitome chart extension : the initial method (a), the proposed algorithm (b)

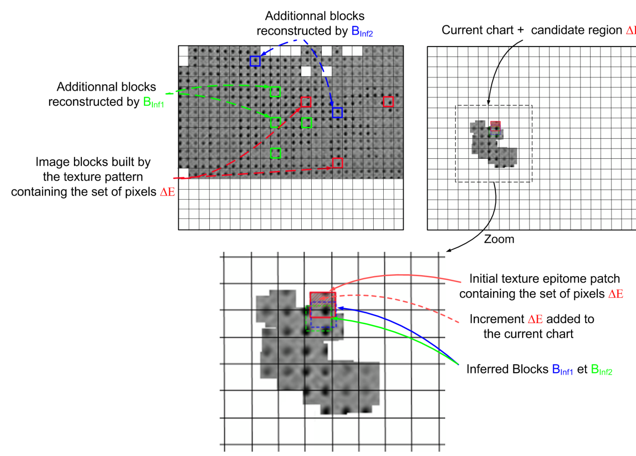


Fig. 5. Example that illustrates the consideration of inferred blocks while extending a chart: the subset image built by the current chart (top left), the current chart growing by an increment (top right).

extended by a texture pattern that allows the reconstruction of the largest number of blocks as well as the minimization of the reconstruction error. Fig. 2 shows the image blocks reconstructed once the first epitome patch is selected.

Extension: The current epitome chart EC_n is then progressively extended by a optimal region ΔE_{opt} from the input image, and each time the current epitome chart is enlarged, one keeps track of the number of additional blocks which can be predicted in the image, as depicted in Fig. 3.

Let k be the number of times where the current epitome chart is extended. The initial epitome chart $EC_n\left(k=0\right)$ corresponds to the texture patch retained at the initialization step. The epitome growth step proceeds first by determining the set of matched patches $M_{j,l}$ that overlap the current chart $EC_n\left(k\right)$ and represent other image blocks. Therefore, there are several growth candidates ΔE that can be used as an extension of the current epitome chart. Let m be the number of candidates found after k steps of the epitome chart extension. For each growth candidate ΔE, the additional image blocks that could be reconstructed is determined from the list $\mathrm{L}'_{match}(M_{j,l})$ related only to the matched patch $M_{j,l}$ containing the set of pixels ΔE. Then, we select the optimal candidate ΔE^k_{opt} among the set of candidates found, leading to best match according to a rate distortion criterion. Let $Y \in R^{N\times M}$ and $Y' \in R^{N\times M}$ denote the input image and the image reconstructed by the current epitome E_{curr} and a growth candidate ΔE respectively. Note that the current epitome is composed of previous epitome charts and the current epitome chart. This selection is indeed conducted according to a minimization of lagrangian criterion:

$$\min\left(D_{E_{curr}+\Delta E} + \lambda \times R_{E_{curr}+\Delta E}\right) \tag{2}$$

$$\text{with } E_{curr} = \sum_{i=0}^{n} EC_i$$

$$\Delta E^k_{opt} = \underset{m}{\operatorname{argmin}}\left(\frac{\sum_{i=0}^{N}\sum_{j=0}^{M}\left(Y_{i,j}-Y'_{i,j}\right)}{N\times M} + \lambda\left(\frac{E_{curr}+\Delta E_m}{N\times M}\right)\right)$$

The first term of the criterion (2) refers to the average prediction error per pixel when the input image is reconstructed by texture information contained in the current epitome $E_{curr} = \sum_i^n EC_i$ and the increment ΔE. As we have done in the initialization step, when the image pixels are impacted neither by the current epitome nor by the increment ΔE, a zero value is assigned. The second term of the criterion (2) corresponds to a rate per pixel when constructing the epitome, which is roughly estimated as the number of pixels in the current epitome E_{curr} and its increment ΔE divided by the total number of pixels within the image.

When the locally optimal increment ΔE^k_{opt} is selected, the current epitome chart becomes:

$$EC_n\left(k+1\right) = EC_n\left(k\right) + \Delta E^k_{opt} \tag{3}$$

Then, we keep on growing the current chart until there are no more matched patches $M_{j,l}$ that overlap the current chart and represent other blocks. Thus, when the current chart EC_n cannot be extended anymore and when the whole image is not yet represented by the current epitome, the index n is incremented by 1 and another epitome chart is initialized at a new location in the image. The process ends when the whole image is reconstructed by the epitome. The two first steps of the epitome construction method is summarized in Table I.

Epitome construction algorithm

Inputs: The source image Y, the error tolerance ϵ

1. FIND SELF-SIMILARITIES WITHIN THE IMAGE:

    For each image $B_i \in Y$ ($B_i \in$ block grid)

        Determine the set of matched patches in the image: $L_{match}(B_i) = \left\{M_{i,0}, M_{i,1}, \cdots\right\}$

    For each matched patch $M_{j,l}$ found ($M_{j,l} \in$ pixel grid)

        Determine the set of image blocks: $L'_{match}(M_{j,l})$ that could be approximated by $M_{j,l}$

2. CREATE EPITOME CHARTS:

    Init $n = 0$

    While all image blocks are not represented by the current epitome **Do**:

        Init an epitome chart EC_n by the most representative matched patch $M_{j,l}$

        Extend EC_n:

            Init $k = 0$

            While there are some matched patches $M_{j,l}$ that overlap $EC_n(k)$ **Do**:

                Determine the optimal increment ΔE^k_{opt}

                Extend the current epitome chart $EC_n(k+1) = EC_n(k) + \Delta E^k_{opt}$

                $k = k + 1$

            End Do

        $n = n + 1$

    End Do

3. REFINE THE MAP:

    For each image $B_i \in Y(B_i \in$ block grid)

        Search the best matching in the final epitome

    End

TABLE I
PSEUDOCODE OF EPITOME CONSTRUCTION ALGORITHM

***Inferred blocks**:* In this paper, we have improved the way of extending an epitome chart using so-called inferred blocks. Indeed, so far, while extending the current epitome chart by a large region, we considered only the blocks reconstructed by the initial texture epitome patch containing the set of pixels ΔE (Fig. 4.a). However, the current chart and a given candidate increment ΔE can provide other epitome patches that could represent additional image blocks. Note that the so-called inferred blocks do not require adding texture content other than ΔE in the current epitome. Indeed, these inferred blocks contain only a part of the increment ΔE (Fig. 4.b). Besides, we can notice that the number of inferred blocks also depends on the local spatial repartition of the pixels available in the current epitome. Fig. 5 illustrates more precisely the principle of inferred blocks after several extension chart steps.

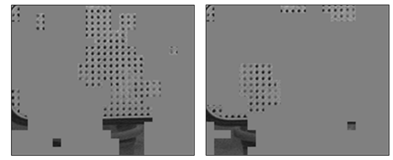
C. Improving the quality of reconstruction (map refinement)

Once the epitome construction is completed, a refinement is performed to obtain a better reconstruction quality. Indeed, a texture patch currently added to the epitome could provide better reconstruction of image blocks which have, in the previous steps, been matched with another epitome patch. That is the reason why, after the epitome is generated, a best matching search is conducted between each image block and the epitome (step 3 in Table I).

D. Matching with sub-pel accuracy

In order to improve the quality of the reconstructed image, we have explored the merit of sub-pel accuracy in the epitome construction. The quarter-pel is performed using separable FIR filters [13]. The algorithm proceeds in the following way:

1) First, a quarter-pixel interpolation is applied on the input image I_0 leading to $I_{0(\uparrow 4)}$.
2) A quarter-pixel search is then performed to find self similar content.
3) An epitome $E_{0(\uparrow 4)}$ is then extracted from $I_{0(\uparrow 4)}$ by proceeding as explained above.
4) Once the construction of the epitome $E_{0(\uparrow 4)}$ is completed, $E_{0(\uparrow 4)}$ is extended with pixels taken from the interpolated image $I_{0(\uparrow 4)}$. This padding is needed due to FIR filters constraints to extract E_0 from the pixels of $E_{0(\uparrow 4)}$ on a sub-sampling grid, E_0 corresponding to the original image resolution.
5) A refinement phase is carried out via block matching algorithm in order to possibly exploit the pixels added when extending $E_{0(\uparrow 4)}$.



(a) (b)

Fig. 6. Epitome construction example: full pel precison (a), quartel pel precision (b).

We can notice that the quarter-pel accuracy approach does a better job at factorizing out the texture redundancy of the input image, as shown in Fig. 6. Indeed, the epitome image size is reduced almost by half compared to the full pel precision.

E. Image reconstruction

Once the epitome image is generated, we can reconstruct an approximation of the original image from the epitome texture and the transform map. Due to the error tolerance ϵ, there are a lot of remaining differences between the input image and the reconstructed one. For image (or video) coding applications, it is necessary to further encode those remaining differences, as described in the remainder.

III. COMPRESSION ALGORITHM

The epitome based compression scheme is depicted in Fig. 7.a and Fig. 7.b. The purpose of this method is to use the reconstructed image from the decoded epitome as a reference frame to encode the current image. This coding scheme requires thus to encode not only the residue but also the epitome $\{$texture and transform map$\}$.

Epitome Texture encoding: The epitome texture is encoded using an encoder of type H.264 Intra (KTA reference software [14]). The epitome being built on the base of a pixel grid, a real padding has been performed for the epitome to be aligned with the image block structures and thus be better adapted to H.264 encoding. Thereby, there are no undesirable boundaries in the blocks. Since the initial epitome is padded with texture content taken from the original image, a refinement phase can

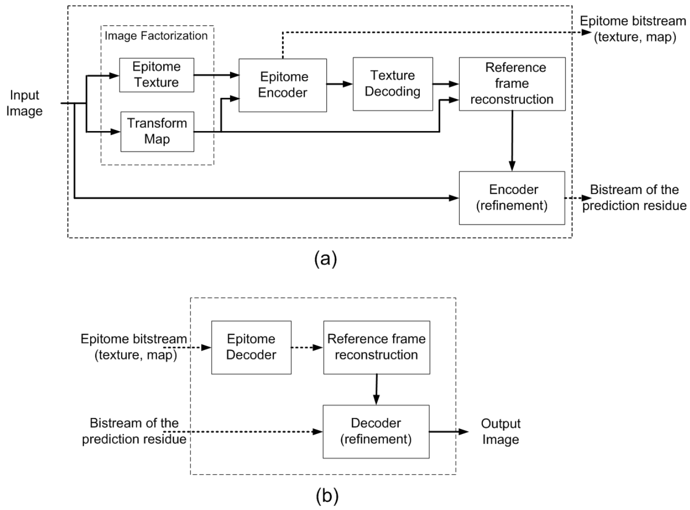


Fig. 7. Encoder epitome based synopsis (a), Decoder epitome based synopsis (b)

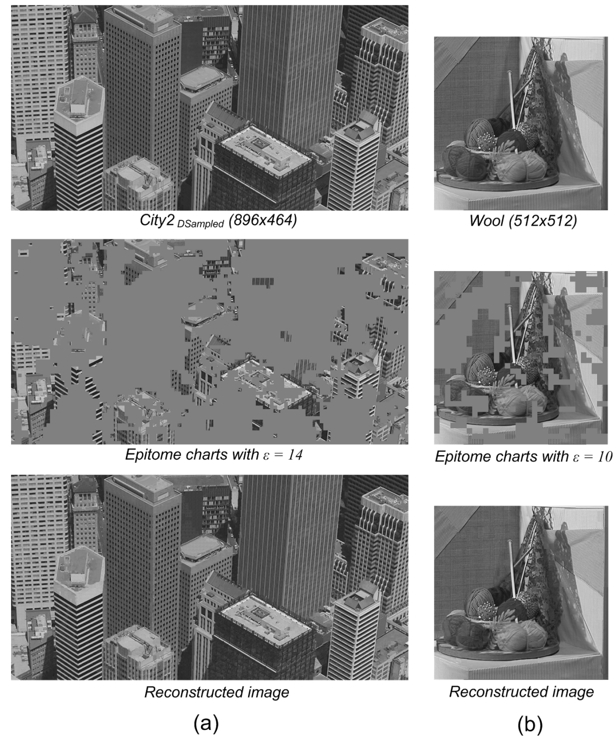


Fig. 8. Factoring results of two images: $City2_{DS}$ (down sampled version of City2) (a) and Wool (b)

be carried out via best matching algorithm in order to possibly exploit the pixels added to the initial epitome.

Transformation map encoding: The transform map indicates for each block of the image which patch in the texture epitome is used for its representation. To encode the transform map, a predictor based on a median of neighboring transform vectors is used. This enables us to encode the difference between the current block vector components and the prediction vector. The map coding has been assessed by coding the prediction error using a combination of a variable length coding (VLC) and a fixed length coding (FLC).

Residue encoding (refinement): At encoder side, once the original image is factored into an epitome texture and a map, an image prediction is built from the reconstructed (decoded) epitome texture and the map. Then, this image prediction can be used as a **reference frame** while encoding the current image (Fig. 7.a). This step corresponds to the residue encoding between the current image and the reference frame. Note that the Intra modes are also enabled to encode the original image when the epitome based prediction fails. Besides, the epitome bitstream (texture and vector map) needs to be sent to the decoder additionally to the bitstream of the prediction residue. At the decoder side, we decode first the epitome bitstream (texture, map) and reconstruct then a **full size picture**, which further serves as a reference frame for decoding dedicated to the current frame (Fig. 7.b).

IV. Simulation results

First, the proposed epitome construction method with a simple translational model is compared to the one described in [4]. To do this, the quality of the reconstructed image is measured without transmitting the prediction error between the current image and the image reconstructed from the decoded epitome texture and the map. With the image ”City2”, a reduction of 12.8% of the bitrate is achieved when encoding the epitome texture and the transform map. Compared to the state of the art approach [4], some improvements have been realized which consist in considering the inferred blocks at the chart extension step and padding the epitome with the structure blocks of H.264 standard.

The proposed epitome based compression algorithm (Fig. 7) has been then assessed comparatively to H.264 Intra. Here, the reconstructed image from the decoded epitome (texture, map) is used as a prediction image. The epitome based prediction was integrated into the KTA reference [14]. The detailed results obtained with the proposed algorithm at integer-pel accuracy for several images are presented in Table II. Using the Bjontegaard metric [15] (QP = 16, 21, 26, 31, 36, 41),

Image	Size	ϵ	Patch	Low-Bitrate		High-Bitrate	
				PSNR	% Rate	PSNR	% Rate
INTEGER-PEL ACCURACY							
City2	1792x944	14	16x16	+0.69	-12.34	+0.40	-3.80
$City2_{DS}$	896x464	14	8x8	+0.97	-11.56	+0.99	-8.25
Wool	720x576	10	16x16	+0.30	-4.61	+0.25	-2.52
Pan0	176x144	11	8x8	+0.64	-8.33	+0.71	-5.80
QUARTER-PEL ACCURACY							
Wool	720x576	10	16x16	+0.50	-7.87	+0.38	-3.91
Pan0	176x144	11	8x8	+1.29	-16.16	+1.21	-10.05

TABLE II
Comparison of H.264 Intra and the proposed compression algorithm at integer-pel and quarter-pel accuracy.

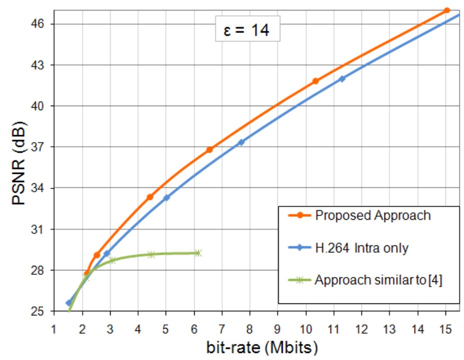


Fig. 9. Coding performances curves for $City2_{DS}$ image at integer-pel accuracy

average BD-rate gain (estimated on 4 quantizer steps: QP) with the proposed compression scheme at integer-pel accuracy are of 9.21% at low bitrates and of 5.09% at high bitrates. We can notice also that the coding performances are particularly improved when the quarter-pel accuracy approach is used (Table II). Indeed, a bit-rate saving up to 16% comparatively to H.264 Intra is achieved.

Fig. 9 shows the coding performance curves of the proposed coding algorithm in comparison with H.264 Intra only. This figure describes also an intermediate step representing the quality of the image prediction for a fixed error tolerance ϵ, according to the coding cost of the epitome (texture, map). This intermediate step can be assimilated to the approach described in [4]. We can observe that it is useless to quantify more precisely the epitome texture since the quality of the image prediction is not necessarily better. This is due to the fact that the error tolerance is fixed when constructing the epitome. Some examples of image factoring of our epitome construction method are presented in Fig. 8. These figures show respectively the original, the epitome and the reconstructed images.

V. Conclusion

In this paper, an epitome based prediction method has been introduced. This new approach of prediction offers interesting results compared to directional prediction modes of H.264. For complex textures, the epitome based compression algorithm turns out to be an alternative solution for Intra prediction. Future work will consider more complex transformations in order to improve the quality of the reconstructed image, as well as more efficient methods to code the transformation map. Moreover, the decoded image epitome only could be used as a reference frame in the encoder loop in order to optimize the current image encoding. Besides, we will focus on the speeding up of the epitome construction since which constitutes the main computational complexity for this new algorithm.

References

[1] N. Jojic et al, "Epitomic analysis of appearance and shape", *in Proc.IEEE Conf. Comput. Vis. (ICCV'03), pp.34-41, 2003.*

[2] D. Simakov, Y. Caspi, E. Shechtman, M. Irani, "Summarizing visual data using bidirectional similarity", *Computer Vision and Pattern Recognition, CVPR 2008.*

[3] V. Cheung, B. J. Frey, and N. Jojic, "Video Epitomes", *International Journal of Computer Vision, vol.76, No.2, Feb. 2008.*

[4] H. Wang, Y. Wexler, E. Ofek, H. Hoppe, "Factoring repeated content within and among images", *ACM Transactions on Graphics, SIGGRAPH 2008.*

[5] M. Aharon and M. Elad, "Sparse and Redundant Modeling of Image Content Using an Image-Signature-Dictionary", *SIAM J. Imaging Sciences, Vol. 1, No. 3, pp. 228-247, Jul. 2008.*

[6] T. K. Tan, C. S. Boon and Y. Suzuki, "Intra prediction by template matching", *ICIP 2006.*

[7] A. Martin, J-J. Fuchs, C. Guillemot and D. Thoreau, "Sparse representation for image prediction", *in Proc. of the European Signal Processing Conf. (EUSIPCO), 2007.*

[8] M. Turkan, C. Guillemot, "Image prediction: template matching vs. sparse approximation", *ICME 2010.*

[9] Y. Fisher, "Fractal Image Compression, Theory and Application", *Springer-Verlag, New York, 1995.*

[10] A. Efros, T. Leung, "Texture synthesis by non-parametric sampling", *in International Conference on Computer Vision, pages 1033-1038, 1999.*

[11] Q. Wang, R. Hu, Z. Wang, "Intra coding in H.264/AVC by image epitome", *PCM 2009.*

[12] Q. Wang, R. Hu, Z. Wang, B. Hang, "Intra coding and refresh based on video epitomic analysis", *ICME 2010.*

[13] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC", *Circuits and Systems for Video Technology, IEEE Transactions, Vol 13,7, 560-576, July 2003.*

[14] "JM 11.0" http://iphome.hhi.de/suehring/tml/download/KTA/

[15] G. Bjontegaard, "Calculation of average PSNR differences between RD curves", *document VCEG-M33, ITU-T VCEG Meeting, April 2001.*