LOCALLY LINEAR EMBEDDING METHODS FOR INTER IMAGE CODING

Martin Alain [1,2], *Safa Chérigui* [1], *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

Image prediction methods based on data dimensionality reduction techniques have been recently introduced for still images. These techniques have been proven efficient, especially when used in a H.264 framework. This paper introduces a natural extension of this work from spatial prediction to temporal prediction. Locally Linear Embedding and Optimized Map-Aided Locally Linear Embedding methods are adapted within H.264 in order to improve inter image prediction. The resulting prediction methods are compared with the H.264 motion estimation/compensation method, template matching methods and Adaptive Interpolation Filters, and are shown to bring significant Rate-Distortion performance improvements (up to 20.21 % rate saving with respect to H.264).

Index Terms: Motion estimation/compensation, adaptive filters, template matching, locally linear embedding, H.264.

1. INTRODUCTION

Video coding efficiency mainly relies on reducing temporal redundancy between a target frame and a reference frame. The main inter prediction technique relies on a block based motion compensation called block matching (BM), used for example in H.264 and HEVC. In this method the image to be encoded is split into blocks. The most correlated block is then searched in one or several reference frames. Once found this block is used as a prediction for the current block, and is pointed by a motion vector that is transmitted to the decoder. The motion vector can reach sub-pel accuracy thanks to fractional positions interpolation in reference frames. H.264 uses a 6-tap Wiener interpolation filter to yield half-pel positions and bilinear filter to compute quarter pel (QPel) positions [1].

Another technique close to BM has been designed, which exploits the correlation between the current block and its neighboring pixels. This set of pixels is called a template. Rather than looking for the most correlated block in reference frames, one looks for the most correlated template. The block adjacent to this template is used as prediction for the current block. This so-called template matching (TM) method [2, 3] can be reproduced at the decoder side, so no motion information needs to be transmitted.

In this paper we introduce a novel approach for inter prediction, based on Locally Linear Embeddings (LLE) [4]. This method derives from multi-patches techniques that already have multiple applications in still image prediction [5], error concealment [6, 7], image denoising [8] or inpainting [9]. The method introduced here has already been proven efficient for intra prediction in [5, 10]. It can be seen as an extension of the TM method. Although here the first step is not to look for the most correlated template but for the K nearest neighbor (K-NN) of the current template. The current template is then approximated as a linear combination of its K-NN, using the LLE. The linear combination coefficients are then applied on the blocks adjacent to the K-NN to yield the current block prediction. As for the TM method, no side information needs to be send to the decoder.

This method has been improved by taking into account the texture information of the whole current patch. A patch is defined as the union of a block and its adjacent template. The first step consists now in searching for the nearest neighbor (NN) of the current patch. Then the $K-1$-NN of the NN are found. The union of these K patches is used as described above to yield a prediction of the current block. This technique helps improving the quality of the prediction, though one should note that a so-called LLE-vector pointing to the first NN has to be transmitted to the decoder. This method was finally extended by testing several LLE-vectors, i.e by looking for the L-NN of the current patch. For every patch/LLE-vector, the $K-1$-NN are found and a prediction is produced. The best patch/LLE-vector is selected thanks to a Rate-Distortion Optimization (RDO) criterion. The optimized LLE-vector is then sent to the decoder.

LLE based methods can be compared with existing techniques that improve the fixed upsampling filters used in H.264. The so-called Adaptive Interpolation Filters (AIFs) [11] aim at minimizing error prediction by adapting Wiener filter coefficients using motion information. Contrary to LLE methods which are locally adaptive, AIFs rely on a global adaptation of the interpolation filter coefficients for each image.

The rest of the paper is organized as follows. Section 2 explains existing methods to improve inter prediction. Section 3 then describes in details LLE based methods. Section 4 describes the complete compression algorithm. Section 5 gives compression performances of our methods compared with standard H.264 scheme and improved H.264 version including AIFs.

2. BACKGROUND

2.1. Inter Prediction in H.264 P-images

Inter image prediction in H.264 [1] is based on the partition of the frames into macroblocks (MB) consisting of 16x16 luminance pixels. In P slices each MB can be subdivided in 16x8, 8x16 or 8x8 blocks sizes. When using 8x8 partition, blocks can be divided again into 8x4, 4x8 or 4x4 partition. This flexible mode partition is used for motion estimation. A BM algorithm is performed within reference frames in list L0 which yields a block prediction and a motion vector (MV). As reference frames are upsampled 4 times before being stored, motion compensation can reach QPel accuracy. For each partition, the motion vector and the reference frame index are coded and sent to the decoder. Moreover, the quantized transformed prediction error (residual) is coded for each block. Motion vectors are predictively coded, under the assumption that the motion field is continuous (at least locally). Thus a motion vector predictor (MVP) is computed as the median of available neighboring MVs. Only the difference between the MVP and the current MV is then coded.

2.2. Prediction based on Template Matching (TM)

An alternative to BM algorithm has been studied for both intra [2] and inter [3] prediction. The TM method is very close to the BM method, although this time it is not the pixels in the current block to be predicted which are used but the template pixels, on the top and to the left of the blocks (see Fig. 1). The union of the template X_k with its adjacent block X_u form the patch X. The underlying basic idea of the algorithm is to take advantage of a supposed correlation between the pixels in the block and those in its template. For inter prediction, the first step of the algorithm is to look for the NN of the template in a search window defined in one or more reference frames. Here and for the rest of this paper the metric used to find the NN is the sum of absolute difference (SAD). Once the NN X_k^{TM} of X_k is found, the adjacent block X_u^{TM} of this template is used as a prediction for the current block X_u. The main benefit of this method is that this process can be reproduced at the decoder side so that no side information (such as MV) needs to be sent to the decoder anymore. Although efficient in terms of bits reduction, this method suffers from limitations when the block and its template are not correlated enough. This can lead to low quality prediction.

This issue has been addressed in [12] with a so-called template matching averaging (TMA) method. In this method one looks for the K-NN of the current template and not only the first one. The prediction of the current block X_u is then obtained by averaging the K blocks adjacent to the K-NN of the template.

Although TMA provides better results than TM, a simple averaging does not lead to the best approximation of the template pixels. One can instead consider constrained least square approximation. Section 3 will present such prediction method, which can be considered as an extension of TM and TMA.

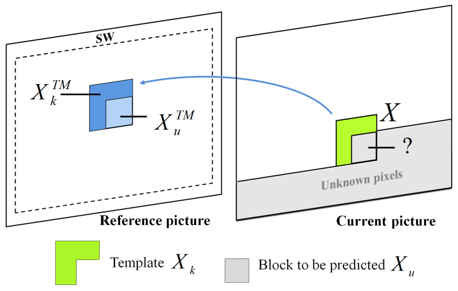

Fig. 1. Template Matching for inter prediction.

2.3. Adaptive Interpolation Filter

Another approach to improve temporal prediction and motion estimation is to enhance the interpolation filter used in H.264 to upsample reference frames. Thus, AIFs have been designed to better cope with aliasing and other noises (e.g. induced by quantization or camera flaws). The method is described in details in [11] and summarized below. Results show significant improvements when compared with H.264.

The AIF algorithm first estimates a motion vector with the standard filter. In a second time, for each sub-pel position SP, an independent filter h^{SP} is computed using the motion vector previously estimated. The filters coefficients are designed to minimize the prediction error energy. Finally new motion vectors are estimated using interpolated values yielded by previous filters. The last two steps can thus be iterated until satisfying quality level is reached. Once the filters coefficients are set, they need to be transmitted to the decoder.

3. PREDICTION BASED ON LLE

This section explains the algorithms for inter prediction based on LLE. These algorithms have been described in [10] for intra prediction. We first show how the LLE can be used for inter image prediction. The method is then improved by using the whole patch texture information.

3.1. LLE method

Prediction based on LLE can be seen as an extension of the TM algorithm described above. In fact the first step of the method is to look for the K-NN of the current template within a search window SW defined in the reference frames. The LLE then solves the corresponding constrained least square approximation [4] for the template pixel values. Thus the current template approximation is a linear combination of its K-NN. Finally in order to get the current block prediction the coefficients of this linear combination are applied to the blocks adjacent to the K-NN.

Let $\mathbf{A} = [\frac{\mathbf{A}_k}{\mathbf{A}_u}]$ denote a so-called dictionary represented by a matrix of dimension NxK. The columns of the dictionary $\mathbf{A}$ are constructed by stacking the K texture patches consisting in the K-NN of the current template and the K blocks adjacent to these K-NN. The sub-matrices $\mathbf{A}_k$ and $\mathbf{A}_u$ thus contain the pixel values of the templates and of the blocks respectively. Let $X = [\frac{X_k}{X_u}]$ be the vector composed of the known pixels of the template X_k and the unknown pixels of the current block X_u.

When using the LLE method, the prediction problem can be re-written as

$$\min_V \|X_k - \mathbf{A}_k V\|_2^2 \text{ s.t. } \sum_m V_m = 1. \tag{1}$$

where V denotes the optimal weighting coefficients vector which is computed as

$$V = \frac{\mathbf{D}^{-1}\mathbf{1}}{\mathbf{1}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{1}} \tag{2}$$

where $\mathbf{D}$ denotes the local covariance matrix (i.e., in reference to X_k) of the selected K-NN templates stacked in $\mathbf{A}_k$, and $\mathbf{1}$ is the column vector of ones. In practice, instead of an explicit inversion of the matrix $\mathbf{D}$, the linear system of equations $\mathbf{D}V = \mathbf{1}$ is solved, then the weights are rescaled so that they sum to one.

This method yield optimal weights regarding our problem but requires a high computational complexity. Interesting approaches were introduced in [6, 13] that reduce complexity but also the estimation quality. These methods will not be considered in this paper.

As for the TM method, the main benefit of this method is that no side information needs to be sent to the decoder, as it can perform the same operations as the encoder. So on one hand the bit cost is limited, but on the other hand the complexity is increased at both encoder and decoder side. It also suffers from the same limitations than the TM method. In fact when the template and the blocks are not correlated, the K-NN which are found on the base of the template are usually not suitable for the LLE, and the coefficients computed are not adapted to well predict the current block.

3.2. (Optimized) Map-aided LLE method: (o)MALLE

In order to improve the LLE based method, the proposed method takes into account the texture information in the current block X_u at the encoder side. Thus the patches stacked in $\mathbf{A} = [\mathbf{a}_0, ..., \mathbf{a}_{K-1}]$ are better selected since there is a better correlation between the templates $\mathbf{A}_k$ and the blocks $\mathbf{A}_u$. The coefficients learned by the LLE on the templates will be better suited to estimate the unknown pixels of the current block X_u. However since the texture information in X_u is only known at the encoder side a so-called LLE-vector $\vec{v}_{\text{NN}}$ needs to be sent to the decoder. This method is referred as Map-Aided Locally Linear Embedding (MALLE).

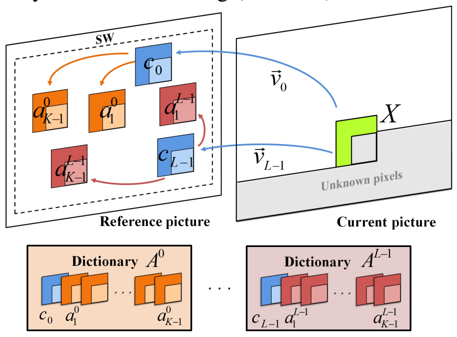

Fig. 2. Example of A^0 and A^{L-1} dictionaries construction from neighbors c_0 and c_{L-1} of the patch X (oMALLE method).

The MALLE method was quickly extended into the optimized Map-Aided Locally Linear Embedding (oMALLE) where not only one but several LLE-vectors are selected. More precisely, L best candidates patches $\mathbf{c}_0, ..., \mathbf{c}_{L-1}$ and their corresponding LLE-vectors $\vec{v}_0, ..., \vec{v}_{L-1}$ are found using a BM algorithm. For each candidate patch (or corresponding LLE-vector) a dictionary is determined and a prediction of the current block X_u is obtained using the LLE method (see Fig. 2). Among all the dictionaries $\mathbf{A}^0, ..., \mathbf{A}^{L-1}$, the one leading to the best prediction $\mathbf{A}^{l_{opt}}$ is selected. The corresponding optimized LLE-vector $\vec{v}_{l_{opt}}$ is then coded and sent to the decoder. The different steps of the oMALLE algorithm are detailed below:

- First the L-NN $\mathbf{c}_0, ..., \mathbf{c}_{L-1}$ of the entire patch X to be predicted are found. Each patch $\mathbf{c}_l, l = 0, ..., L-1$ is pointed by a LLE-vector $\vec{v}_l, l = 0, ..., L-1$.
- Then for each patch $\mathbf{c}_l$ a dictionary $\mathbf{A}^l$ is built by finding the patches $\mathbf{a}^l_1, ..., \mathbf{a}^l_{K-1}$ which are the closest to $\mathbf{c}_l$, i.e $\mathbf{A}^l = \left[\mathbf{c}_l, \mathbf{a}^l_1, ..., \mathbf{a}^l_{K-1}\right], l = 0, ..., L-1$. These first two steps are illustrated in Fig. 2.
- In a third step, the algorithm searches for L approximations of the template X_k as a linear combination of the K-NN templates in each sub-matrix $\mathbf{A}^l_k, l = 0, ..., L-1$. Thus the following approximation is solved L times using LLE:

$$\min_{V^l} \left\| X_k - \mathbf{A}^l_k V^l \right\|^2_2 \text{ s.t. } \sum_m V^l_m = 1. \tag{3}$$

  where V^l represents the weighting coefficients of the templates in the sub-matrix $\mathbf{A}^l_k$.
- Once the coefficients $V^l, l = 0, ..., L-1$ are known, they are applied to the blocks in the L sub-matrix $\mathbf{A}^l_u$ in order to produce L predictions $\hat{X}^l_u$ of the current block X_u. The best vector $V^{l_{opt}}$ is selected using a Lagrangian criterion, also called a rate-distortion optimization (RDO) criterion:

$$\min (D + \lambda \times R) \tag{4}$$

  where D represents the distortion between the original block and the reconstructed block by using the Sum of Square Errors (SSE) distance metric, λ represents a trade-off coefficient between the distortion and the bits R needed for coding the block. The parameter λ is defined in H.264 as $\lambda = 0.65 \times 2^{QP/3}$, with QP corresponding to the quantization parameter.

  The best LLE-vector is then coded and transmitted to the decoder, so that the exact same optimized prediction $\hat{X}_u = \mathbf{A}^{l_{opt}}_u V^{l_{opt}}$ can be produced at the decoder side.

3.3. Set of patches

Improvements provided by the oMALLE algorithm bring higher computational complexity, especially if the search window SW is large. In order to reduce this complexity a technique using a so-called set of patches (SP) has been designed, which uses only a subset of SW. Although the purpose was different, a similar technique was introduced in [14] to improve TM performances. Instead of considering all the patches only a set of S patches are selected by a TM algorithm in SW and stacked in the set of patches $\mathbf{S}$. Only these patches are then used to perform the algorithm. The use of the SP also offers new possibilities that are described in section 4.

The oMALLE method with a set of patches (oMALLE+SP) is synthesized as pseudo-code in Algorithm 1.

Algorithm 1 oMALLE method using a set of patches

```
Input: $X, S, K, L$
Output: current block prediction $\hat{X}_u$
Determine the $S$ nearest neighbors of the current template $X_k$, i.e the templates $\mathbf{s}_{k_0}, ..., \mathbf{s}_{k_{S-1}}$ in $SW$ such as $d_0 \leq ... \leq d_{L-1}$ with $d_i = ||X_k - \mathbf{s}_{k_i}||$
Stack in $\mathbf{S}$ the $S$ patches associated with the $S$ previous templates: $\mathbf{S} = [\mathbf{s_0}, ..., \mathbf{s_{S-1}}]$ with $\mathbf{s}_i = [\frac{\mathbf{s}_{k_i}}{\mathbf{s}_{u_i}}]$
Determine $L$ nearest neighbors of the entire patch $X = [\frac{X_k}{X_u}]$, i.e., the patches $[\mathbf{c}_0, ..., \mathbf{c}_{L-1}]$ in $\mathbf{S}$ such as $d_0 \leq ... \leq d_{L-1}$ with $d_i = ||X - c_i||$
for $l = 0 \rightarrow l = L-1$ do
    Determine $K-1$ neighbors close to $\mathbf{c}_l$, i.e., the patches $\left[\mathbf{a}^l_1, ..., \mathbf{a}^l_{K-1}\right]$
    $\mathbf{A}^l = \left[\mathbf{c}_l, \mathbf{a}^l_1, ..., \mathbf{a}^l_{K-1}\right]$
    Retrieve $\mathbf{A}^l_k = \left[\mathbf{a}^l_{k_0}, ..., \mathbf{a}^l_{k_{K-1}}\right]$ and $\mathbf{A}^l_u = \left[\mathbf{a}^l_{u_0}, ..., \mathbf{a}^l_{u_{K-1}}\right]$ from $\mathbf{A}^l$
    Solve the constrained least squares problem:
    $min_{V^l} \left\| X_k - \mathbf{A}^l_k V^l \right\|^2_2$ s.t. $\sum_m \mathbf{V}^l_m = 1$
end for
Select the optimum $l_{opt}$ minimizing the RDO criterion
Set $\hat{X}_u = \mathbf{A}^{l_{opt}}_u V^{l_{opt}}$
```

4. COMPRESSION ALGORITHM

The new prediction methods described in section 3 have been tested in the H.264 framework. Only the LLE and oMALLE+SP methods have been tested. The MALLE algorithm can in fact be obtained by setting the number L of tested vectors to 1 in oMALLE. The LLE method to be tested is introduced in the coding scheme in competition with the existing motion estimation and the intra modes. In fact the LLE based techniques suffer from some limitations and in some situations a simple BM algorithm might yield a better prediction, e.g. on smooth texture areas. On the opposite the LLE methods reconstruct well high frequency pseudo-periodic textures, which are difficult to predict with a BM algorithm.

Both prediction algorithms (BM and LLE) are performed only on 8x8 blocks. Then the best method is naturally selected thanks to an RDO criterion. Unlike intra prediction, H.264 inter prediction is not designed to support several modes. Thus the syntax needs to be slightly modified to transmit to the decoder a flag indicating the prediction method that has been chosen by the encoder. In practical terms, the rate is increased of 1 bit for each 8x8 block. Original motion estimation is always performed at QPel accuracy. The LLE methods were first tested with a full pel (FPel) accuracy. In this case, we compare the LLE coefficients in the weight vector V with those used by H.264 upsampling filters. In a second time the LLE methods were used with a QPel accuracy. In this case there is no competition between the LLE and H.264 interpolation filters but on the contrary the LLE technique uses the additional information provided by the upsampling.

When using the oMALLE+SP algorithm, a LLE-vector has to be coded. For this purpose the actual vector $\vec{v}_{l_{opt}}$ is not transmitted but rather the index of the patch $\mathbf{c}_{l_{opt}}$ in the set of patches $\mathbf{S}$. In fact the vector $\vec{v}_{l_{opt}}$ usually does not correspond to a "physical" motion, contrary to the BM vector. This means that the predictive encoding would not be efficient for the LLE-vector, hence the index coding instead. If the set of patches $\mathbf{S}$ consists of S patches, the additional rate cost for each 8x8 block using the oMALLE+SP method is:

$$R_{\text{oMALLE+SP}} = \log_2(S) \tag{5}$$

The parameter S thus becomes critical. In fact, as S increases, there are better chances to find interesting patches for the prediction, but it also increases the bit cost. In order to optimize the coding performance, S is always set to a power of 2.

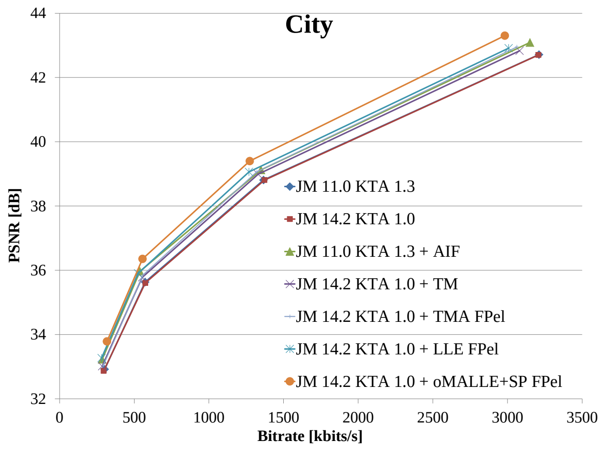

Fig. 3. Rate-Distortion curve for a CIF crop of the City sequence

5. SIMULATIONS AND RESULTS

The proposed schemes were implemented in the latest release of the KTA software [15], in order to be compared with H.264. LLE and oMALLE+SP algorithms were also compared with state of the art methods presented in section 2. AIF were not implemented in the latest KTA release [15] so a former version [16] was used for the simulations. Results of the two reference softwares are very similar so it makes sense to compare our methods with AIF even though they were not implemented in the same KTA version. This is shown for example on Fig. 3 where RD curves of the reference softwares are merged.

Simulations were run on a set of 8 videos with the encoder parameters defined in Table 1. All videos have a CIF format, except Pan0 that has QCIF format. The City sequence was cropped to fit CIF format. For each sequence 4 QP values were tested to draw Rate-Distortion (RD) curves and compute Bjontegaard measures [17]. For both LLE based methods and TMA, K was set to 64. For oMALLE+SP algorithm L and S were set to 256. Next results were obtained using 1 reference frame.

RD curves of the different algorithms for a crop of the City sequence are displayed on Fig. 3. Overall our methods outperform

Pamareter	**Setting**
Number of frame	10
Frame rate (frm/s)	30
Sequence	IPPP...
QP I	16, 21, 26, 31
QP P	20, 25, 30, 35
Mode intra allowed	4x4, 8x8, 16x16
Mode inter allowed	8x8 only, Skip
Search range	64

Table 1. Encoder parameters used for simulations

state of the art methods, which outperform H.264. The oMALLE+SP algorithm achieves the best performance at high bitrate by far, but at low bitrate its performance are really close to those of AIFs or TM(A) methods. In fact, at low bitrate high frequency pseudo-periodic textures well reconstructed by LLE are destroyed by quantization, thus limiting the PSNR gain for the LLE based methods. Furthermore at low bitrate the additional rate cost $R_{\text{oMALLE+SP}}$ damages oMALLE+SP performance. LLE based methods were here used at FPel accuracy, and both methods are better than AIF, so our LLE weight vector V is more efficient than AIFs or H.264 filters for this sequence.

In Table 2 are presented % rate reduction of the different algorithms with respect to H.264, computed with the Bjontegaard measure. On the average, oMALLE+SP method outperforms other algorithms, followed by LLE. For LLE methods used at FPel accuracy, at least one method provides better results than AIF. We can conclude that the LLE coefficients usually better fit than AIFs or H.264 filters coefficients, thanks to local adaptation. On the contrary TMA at FPel accuracy uses too simple weights to outperform AIFs. One can notice that LLE methods and TMA used at QPel accuracy usually outperform the use at FPel accuracy, which means that the information yield by interpolation filters can benefit the LLE and even a simple mean computation. However LLE based methods are not always efficient, especially for the Zebra sequence. In fact for this video it is difficult to find enough similar patches to compute a good prediction. On the opposite Pan0 and City sequences provide remarkable results and bitrate is reduced up to 15.78% and 20.21% for Pan0 and City respectively.

% rate			TMA		LLE		oMALLE+SP	
reduction	AIF	TM	FPel	QPel	FPel	QPel	FPel	QPel
Pan0	-9.36	-12.00	-5.11	-6.04	-6.33	-9.29	-10.72	-15.78
Foreman	-2.19	-1.52	-0.27	-3.46	-1.93	-3.83	-3.99	-2.60
City	-10.56	-7.37	-8.82	-14.53	-13.41	-15.18	-18.99	-20.21
Mobile	-2.61	-4.19	0.50	-1.85	-1.24	-5.61	-2.73	-7.53
Macleans	-2.68	-4.90	-1.99	-7.33	-5.60	-11.29	-10.18	-14.21
Rushes 1	-0.45	-2.95	0.25	-3.21	-3.94	-7.24	-7.37	-8.90
Rushes 2	-1.17	-1.56	-0.42	-3.36	-2.62	-5.71	-5.87	-7.96
Zebra	-2.32	-0.23	-0.48	-1.22	0.08	-1.16	-1.49	-1.71
Average	-3.92	-4.34	-2.04	-5.12	-4.37	-7.41	-7.67	-9.86

Table 2. Rate gains in % (Bjontergaard measure) with respect to H.264 for FPel and QPel accuracy

6. CONCLUSION

In this paper were introduced the LLE and oMALLE+SP methods for inter image prediction. These methods have been implemented in a H.264 framework and applied to P images. Simulations results show that significant improvement can be achieved with respect to H.264 (up to 20.31 % rate reduction), but also with respect to state of the art methods such as TM(A) and AIFs. Tests using LLE based methods at QPel accuracy show that it can increase the performances. Future work will extend these methods to B images and test multiple reference frames.

7. REFERENCES

[1] 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.

[2] 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.

[3] K. Sugimoto, M. Kobayashi, Y. Suzuki, S. Kato, and C. S. Boon, "Inter frame coding with template matching spatio-temporal prediction", in *IEEE Int. Conf. Image Process. (ICIP)*, 2004, vol. 1, pp. 465–468.

[4] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding", *Science*, vol. 290, pp. 2323–2326, Dec. 2000.

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

[6] Song Kwanwoong, Chung Taeyoung, Kim Chang-Su, Park Young-O, Kim Yongdeok, Joo Younghun, and Oh Yunje, "Efficient multi-hypothesis error concealment technique for h.264", in *IEEE Int. Symp. Circuits and Systems (ISCAS)*, 2007, pp. 973–976.

[7] J. Koloda, J. Ostergaard, S.H. Jensen, A.M. Peinado, and V. Sanchez, "Sequential error concealment for video/images by weighted template matching", in *Data Compression Conference (DCC)*, 2012, pp. 159–168.

[8] A Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising", in *IEEE Comp. Soc. Conf. Comp. Vision and Pattern Recog.*, 2005, pp. 60–65.

[9] 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.

[10] 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.

[11] Y. Vatis, B. Edler, D. T. Nguyen, and J. Ostermann, "Motion and aliasing-compensated prediction using a two-dimensional non-separable adaptive wiener interpolation filter", in *IEEE Int. Conf. Image Process. (ICIP)*, 2005, vol. 2, pp. 894–897.

[12] Y. Suzuki, C. S. Boon, and T. K. Tan, "Inter frame coding with template matching averaging", in *IEEE Int. Conf. Image Process. (ICIP)*, 2007, vol. 3, pp. 409–412.

[13] P. Amon, A. Hutter, E. Wige, and A. Kaup, "Intra prediction for lossless coding (proposal)", in *Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, Jan. 2013.

[14] M. Moinard, I. Amonou, P. Duhamel, and P. Brault, "A set of template matching predictors for intra video coding", in *IEEE Conf. Acoustics, Speech, and Signal Process. (ICASSP)*, 2010, pp. 1422–1425.

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

[16] "KTA Software, Ver. JM 11.0 KTA 1.3. Available: http://iphome.hhi.de/suehring/tml/download/KTA/".

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