A SET OF TEMPLATE MATCHING PREDICTORS FOR INTRA VIDEO CODING

Matthieu Moinard, Isabelle Amonou
Orange Labs
35512 Rennes

Pierre Duhamel, Patrice Brault
Laboratoire des Signaux et Systemes
91192 Gif-sur-Yvette

ABSTRACT
Prediction methods by template matching are often mentioned to improve video coding efficiency. They are based on a Markovian model to find the most similar patterns of texture in previously encoded information. These kinds of methods are more efficient than H.264/AVC intra prediction modes in many cases, such as complex texture coding. However, the template matching method is not always optimal as to the best predictor choice. This paper introduces the use of a sorted set of template matching predictors to propose different texture patterns. At the encoder side, the best predictor is selected by minimizing a rate-distortion metric and is transmitted to the decoder. Despite this needed additional side information, simulation results show that the proposed scheme further improves the coding efficiency of H.264/AVC up to 11%.

Index Terms—Video coding, template matching, intra prediction methods

1. INTRODUCTION
The spatial domain intra prediction specified in the H.264/AVC standard [1] relies on a DC prediction mode and 8 directional prediction modes for 4x4 and 8x8 luminance blocks. The reconstructed pixels surrounding the current block to be coded are extrapolated according to the mode direction (or averaged for DC mode) to form the sample predictor. This method works well for most sequences, but its weakness is that the extrapolation process makes the predictor roughly smooth, therefore increasing the residual error for complex texture.

In order to compensate for this drawback, recent studies on image and video coding – inspired by texture synthesis and inpainting methods – have been developed. All those methods tend to generate an image which is visually based on a source image. In this paper, we focus on a specific family of non-parametric techniques called Markovian models, based on the assumption that Markov random fields correctly model the properties of natural images. The algorithms based on Markov models are very simple because they are directly applied in the spatial domain. The output images are built iteratively by copying small patterns from a source image with the help of a metric based on neighboring pixel correlation. Initial texture synthesis models worked on a one-by-one pixel reconstruction such as [2][3], and were extended, among other things, to block pattern [4], multiresolution synthesis [5], and graphcut optimization [6]. Among the inpainting solutions, isophote-driven and ordering image-sampling process provide really impressive results such as [7].

Intra prediction by Template Matching (TM) was introduced in [8] and was improved in many contributions such as priority-based reconstruction in [9] (inspired by [7]), adaptive illumination compensation in [10] and averaged template matching in [11], and was compared to displacement intra prediction [12]. In typical implementations, template matching is considered as an additional intra prediction mode to be proposed to the same rate-distortion optimization process as other modes.

This paper is organized as follows. We briefly introduce template matching from the state of the art in Section 1. Section 2 describes our solution to improve template matching methods in video coding context, and how it is incorporated inside H.264/AVC codec. The simulation results and comparisons are discussed in Section 3.

2. INTRA PREDICTION BY TEMPLATE MATCHING
As described in [8], a template is a group of reconstructed pixels adjacent to the target block to be predicted (see examples on Fig. 3). In a causal search region $\Phi$, the most similar source template $\Psi_s$ is selected by minimizing the Sum of Squared Differences (SSD) with target template $\Psi_t$. Formally,

$$\Psi_s = \arg \min_{\Psi_s \in \Phi} SSD(\Psi_s, \Psi_t)$$ (1)

where function $SSD$ compares two templates pixel by pixel. Then, the source block associated with template $\Psi_s$ is defined as the predictor for template matching intra prediction mode. As shown in Fig. 1, an intra displacement vector derived implicitly, thus no side information is required to transmit it.

An enhancement of template matching, called Template Matching Averaging (TMA), where the $N$ most similar source templates are selected rather than only one, is described in [11]. The predictor is computed by averaging the $N$ corresponding source blocks. This process enables to smooth the predictor which is advantageous most of the time.
However, it produces a bad prediction when the \( N \) source blocks are quite different, hence causing a costly residual.

### 3. Set of Template Matching Predictors

#### 3.1. The Principle

In most template matching methods issued from the inpainting domain, the prediction process is the same at the encoder and decoder sides thus no side information is required. On the contrary, our template matching prediction method, using some of the original image data that are not available at the time of computation on the decoder side, needs to insert a side information into the bitstream.

Our new prediction method is proposed as an alternative intra mode to the video encoder. The first step of our method is common to encoder and decoder. It consists in building a set of predefined size \( N \) sorted from the most similar template to the least similar using Eq. 1 (see Fig. 2 for illustration) as in [11]. Then, at the encoder side, predictors \( c_n \) from \( L_N \) such that \( 0 \leq n \leq N \) are proposed to the rate-distortion optimization algorithm (e.g. H.264/AVC RD-opt [1]) taking into account the coding cost of index \( n \) and the residual. If the predictor \( c_n \) is better than the other modes (e.g. H.264/AVC intra modes) and than the other template matching predictors, then \( c_n \) is selected and \( n \) is binarized and transmitted using an arithmetic coder for each bit. At the decoder side, with list \( L_N \) and decoded index \( n \), the predictor is immediately available.

The size \( N \) of the list is an important parameter. As \( N \) increases, the chance to have better predictors available from the source image grows, but the coding cost of the index increases. Ideally, \( N \) should be large enough to get the same predictor as with intra displacement method (which is the best one), but short enough so that \( n \) is less expensive to code than an intra displacement vector.

This model tends to turn aside from usual template matching methods in a video coding context. Considering the set of predictors as a dictionary, analogies can be made with vector quantization. From this point of view, the dictionary is built for each target block with vectors produced from pixels of candidate predictors.

#### 3.2. Implementation in H.264/AVC Context

##### 3.2.1. New prediction mode

As in many other contributions, the intra template matching mode is considered as an additional intra mode in the H.264/AVC framework. Because H.264/AVC uses modes 0 to 8, the new mode is assigned value 9. The index \( n \) of the chosen predictor has to be coded when template matching mode is used. For ease of implementation, we make sure that \( N \) is a power of two, so that the index \( n \) is binarized into a fixed length code. Then, each bit of the index \( n \) is coded using an arithmetic coder.

A half-pixel search is included into our method in order to increase the number of potential candidates for the predictor set. Half-pixel images are computed with a bilinear interpolation, then half-pixel predictors have a slight smoothing effect.

##### 3.2.2. Encoding and signaling of the additional mode

In H.264/AVC, the intra block mode is two-step coded. First the \( c \) mode of current block \((0 \leq c \leq 8)\) is predicted from a most probable mode function \( P_1(a, b) \) depending on the mode \( a \) and \( b \) of the blocks above and left respectively from the current block as follows,

\[
P_1(a, b) = \begin{cases} 
DC & \text{if } a \text{ or } b \text{ unavailable} \\
\min(a, b) & \text{else}
\end{cases}
\]

(2)

A binary flag is set in the bitstream depending on whether \( c = P_1(a, b) \) is true. If false, a function \( F(a, b, c) \) is generated,

\[
F(a, b, c) = \begin{cases} 
c & \text{if } c < P_1(a, b) \\
c - 1 & \text{else}
\end{cases}
\]

(3)
Then, the result from \( F(a, b, c) \), ranging from 0 to 7, is simply binarized on 3 bits and then encoded using an arithmetic coder. A problem occurs if a new template matching intra mode 9 is added because \( F(a, b, c) \) will be represented on 4 bits, which is sub-optimal to code 9 symbols. Our solution consists in creating a second most probable mode function \( P_2(a, b) \) as follows,

\[
P_2(a, b) = \begin{cases} 
9 & a \text{ and } b \text{ unavailable} \\
\max(a, b) & \max(a, b) \neq P_1(a, b) \\
\alpha & \text{else, with } \alpha \neq P_1(a, b)
\end{cases}
\]

(4)

where \( \alpha \) can be set to any value in the range of 0 to 9 inclusive, and ensures that \( P_2(a, b) \) is different than \( P_1(a, b) \). Then, if \( P_1(a, b) \) is wrong, the second most probable mode function \( P_2(a, b) \) induces to signal an extra binary flag to indicate if \( c = P_2(a, b) \). If false, the previous \( F(a, b, c) \) is adapted as follows,

\[
F(a, b, c) = \begin{cases} 
c & c < P_1(a, b) \text{ and } c < P_2(a, b) \\
c - 2 & c \geq P_1(a, b) \text{ and } c \geq P_2(a, b) \\
c - 1 & \text{else}
\end{cases}
\]

(5)

to guarantee a value between 0 to 7 binarized on 3 bits.

3.3. The Computational Complexity

In terms of computational complexity, template search inside the source image is similar to the usual template matching methods, thus the decoding process has the same complexity. The introduction of a half-pixel search requires computation for pixel interpolation and increases search region size. However, the implementation of fast search methods such as in H.264/AVC motion estimation should decrease the computational time.

At the encoder, the computational complexity is increased because the rate-distortion optimization process has to be applied to all template matching predictors rather than only one for additional template matching intra prediction mode. The increase varies depending on the predefined size \( N \) of the predictor list \( \mathcal{L}_N \).

4. SIMULATION RESULTS AND COMPARISONS

In order to validate our solution, the proposed Set of Template Matching Predictor method (STMP) was compared with template matching in [8] and averaged template matching in [11], that averages four candidates. All of those algorithms were implemented in the JSVM 9.7 reference software, with Context-based Adaptive Binary Arithmetic Coding (CABAC) entropy coding and \( 8 \times 8 \) transform turned on. Template matching intra modes are used for \( 4 \times 4 \) and \( 8 \times 8 \) luma samples, templates are illustrated on Fig. 3. The search region sizes for \( 4 \times 4 \) and \( 8 \times 8 \) blocks are set to \( 32 \times 32 \) and \( 64 \times 64 \) pixels respectively, centered on top-left pixel of target block but reduced to causal neighborhood. The Bjontegaard metric [13] is used to measure the gain, with QP = 22,27,32,37.

![Fig. 3. Template shape for 4 x 4 and 8 x 8 blocks.](image-url)

The results are presented in Table 1. Our solution is always better than previous template matching methods, and the best results are obtained with 32 candidate predictors, when indexes \( n \) are coded using five binary symbols per block. In average, STMP with \( N = 32 \) is better than standard template matching by 2% and outperforms TMA by 1.28%. Half-pixel search is implemented for TM, TMA and STMP methods. Whereas the half-pixel search does not improve the results for TMA, we noticed that it improves TM and STMP by up to 1%. The benefits of the method are quite different according to the sequence, template matching methods being more efficient with the sequences containing rigid textures.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>TM</th>
<th>TMA</th>
<th>STMP with ( N = )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>8</td>
<td>16</td>
<td>32</td>
</tr>
<tr>
<td>CIF</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Foreman</td>
<td>7.11</td>
<td>8.88</td>
<td>10.18, 10.92, 11.53</td>
</tr>
<tr>
<td>Mobile</td>
<td>1.10</td>
<td>1.21</td>
<td>2.19, 2.53, 3.04</td>
</tr>
<tr>
<td>Paris</td>
<td>2.03</td>
<td>2.51</td>
<td>3.32, 3.65, 4.01</td>
</tr>
<tr>
<td>Templete</td>
<td>0.77</td>
<td>1.12</td>
<td>1.38, 1.68, 1.93</td>
</tr>
<tr>
<td>AVERAGE CIF</td>
<td>2.75</td>
<td>3.43</td>
<td>4.27, 4.69, 5.13</td>
</tr>
<tr>
<td>720p</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BigShips</td>
<td>2.52</td>
<td>3.15</td>
<td>3.49, 3.67, 3.83</td>
</tr>
<tr>
<td>City</td>
<td>3.98</td>
<td>5.64</td>
<td>6.58, 7.15, 7.61</td>
</tr>
<tr>
<td>Crew</td>
<td>1.83</td>
<td>2.46</td>
<td>2.48, 2.81, 3.08</td>
</tr>
<tr>
<td>Jets</td>
<td>2.59</td>
<td>3.11</td>
<td>3.04, 3.24, 3.31</td>
</tr>
<tr>
<td>Night</td>
<td>2.08</td>
<td>2.73</td>
<td>3.19, 3.53, 3.79</td>
</tr>
<tr>
<td>OldTownCross</td>
<td>2.41</td>
<td>2.90</td>
<td>4.31, 4.82, 5.20</td>
</tr>
<tr>
<td>ShuttleStart</td>
<td>1.40</td>
<td>1.82</td>
<td>2.04, 2.21, 2.38</td>
</tr>
<tr>
<td>AVERAGE 720p</td>
<td>2.40</td>
<td>3.12</td>
<td>3.59, 3.92, 4.17</td>
</tr>
</tbody>
</table>

Table 1. Results for different template matching methods expressed in rate decrease (%) using the Bjontegaard metric. TM: template matching, TMA: template matching averaging with four averaged predictors, STMP: set of template matching predictors with varying predictor number \( N \) (8,16,32). Half-pixel search is used for TM, TMA and STMP. Only the first 49 frames I are coded. Sequences have CIF (352 x 288 pixels) or 720p (1280 x 720 pixels) resolution.
Fig. 4 shows that the STMP method is used much more than other intra prediction modes, in particular for 8 × 8 blocks. These results reveal that our solution is less expensive despite the overhead due to the additional template matching index cost, especially for high quality video coding. However, the use of our SMTP method decreases with low bitrate. In this case, the reconstructed image is much more damaged than the original, hence the template matching predictors are not good.

The graphics on Fig. 5 indicates that from index 0 to index 12, the density function of selected template matching indexes decreases, and then their probabilities become equal. The assumption that the closest template gives the best predictor is most of the time wrong because all indexes are used, which validates our template matching model with a set of template matching predictors.

5. CONCLUSION

We have presented a new template matching algorithm that is based on the selection of the best predictor in a sorted set of predictors. Our solution was integrated in a H.264/AVC software as an extended intra prediction mode that improves the rate-distortion performance in comparison with the state of the art template matching methods. The experimental results show an average gain of 4.52% over H.264/AVC (at the same PSNR) with a maximum of 11.53%.

Extending the concept to inter prediction would provide an alternative to motion vector coding, which will be further investigated.

6. REFERENCES