Images are typically non-stationary signals. If prediction is applied in a linear fashion, it must be combined with a technique which takes this characteristic into account. In general, images can be either regarded as piecewise two-dimensional autoregressive processes or they are handled in a block-wise manner.

This paper presents a novel prediction technique, which treats the image data as an interleaved sequence generated by multiple sources. The challenge is to de-interleave the sequence and to compute prediction weights for each sub-source separately. The proposed approach adaptively determines the sub-sources based on the textures within the images.

The prediction method is incorporated in a framework for lossless image compression. It is based on least-mean-square filtering and achieves prediction-error entropies, which are comparable to those of least-squares approaches. In combination with a dedicated coding algorithm, the proposed approach shows a competitive compression performance for a wide range of different natural images.

Index Terms— linear prediction, lossless image compression, LMS filtering

1. INTRODUCTION

The prediction of image data can look back on a long history and a broad range of different approaches. Table 1 tries to categorise major approaches using a two-dimensional matrix and assigns corresponding publications. Column-wise, it differentiates between (i) simple (and fixed) predictors, (ii) predictors in which coefficients are optimised in a separate pass (offline) and have to be transmitted along to the decoder, and (iii) predictors with parameters, which are adapted on the fly (online). Row-wise, there are four possibilities: (i) a single predictor is used, (ii) one out of several predictors is newly chosen for each pixel, (iii) a common predictor setting is selected for a block of pixels, and (iv) the estimates of different predictors are combined (blended) in a certain way resulting in a joint estimation value.

The most important methods of adaptive prediction are listed in the lower part of the table. Some of the references appear more than once as these approaches combine different techniques. Most successful are those compression schemes, which incorporate any variant of least-squares approximation (LS) or least-mean-square filtering (LMS). Template matching (TM, also called texture context matching) is an alternative for repeated textural patches, which are not accessible by a local adaptation process. There are also attempts to model the non-linearity of images with neural networks.

The proposed prediction method is called context-based adaptive linear prediction (CoBaLP2). It is a variant of composite source modelling, treating the signal as an interleaved sequence generated by multiple sources, called sub-sources [24]. Each sub-source has its own model (context) and associated prediction parameters. CoBaLP2 determines suitable contexts based on the analysis of the image textures. This matches the contexts to different image characteristics and guarantees that all contexts are used in the prediction stage. The new LMS prediction is combined with template matching to form a predictor with competitive performance for images with a broad range of different characteristics.

The paper is organised as follows: Section 2 describes the new prediction method including the LMS filtering and the template-matching method. Section 3 analysis the predictors performance, compares it to state-of-the-art techniques and presents the coding results for selected test images. A summary is given in Section 4.

2. CONTEXT-BASED LINEAR PREDICTION

2.1. Determination of the prediction context

Based on the idea that an image consists of interleaved parts, the determination of the prediction contexts tries to combine pixels with similar texture. This is done via vector quantisation (k-means clustering). The proposed scheme uses a predefined (empirical) number \( n_{px} = \max(2, \sqrt{w \cdot h}/10) \) of prediction contexts, with \( w \) and \( h \) being the image width and height. Each texture vector \( \mathbf{v} = (v_1, v_2, \ldots, v_n) \) consists of five difference values and one
Fig. 1. Template of signal values which are used in the prediction stage

(modified) signal value. According to Fig. 1, the elements of vector \( \mathbf{v} \) are determined in the causal neighbourhood of position \( \mathbf{X} \) as follows:

\[ \mathbf{v} = (A - C) \ B - D \ A - E \ B - H \ \nu \cdot A \ . \quad (1) \]

The last element is the signal value \( A \) scaled down by \( \nu \) in such a manner that the resulting values are approximately in the same domain of definition as the absolute difference values. This texture vector is used for the clustering in the training process and for the classification of pixels later on. Taking differences in (1) instead of using original signal values accounts for varying illumination. The element \( \nu \cdot A \) allows a distinction in case of too large deviations in brightness. The training process utilises 150 \( \cdot n_{px} \) training vectors pseudo-randomly selected from the entire image. Considering all image positions would increase the computational costs without significant gain in the clustering process. After convergence of the clustering, the coordinates \( c_i \) of cluster \( C_i \) are set to the average of all \( N_i \) assigned training vectors \( v_j \) and are rounded to integer values

\[ c_i = \text{round} \left( \frac{1}{N_i} \sum_{v_j \in C_i} v_j \right) \quad i = 1, 2, \ldots, n_{px} . \quad (2) \]

In cases when \( N_i \) is too low, the corresponding cluster is removed and \( n_{px} \) is decremented accordingly. This reduces the overhead as \( n_{px} \) and all \( c_i \) have to be transmitted to the decoder. Based on the clusters, the prediction context for each pixel with a texture vector \( v_k \) can be determined using the Euclidean distance

\[ px = \arg \min \left( \| c_i - v_k \| \right) \quad k = 1, 2, \ldots, w \cdot h \ . \quad (3) \]

Naturally, pixels with similar texture (i.e., same context) can be spread over the entire image. Figure 2 a) visualises the 64 context numbers of the barbara image with different grey values. The numbers are sorted according the first element of the coordinates vector \( c_i \). Figure 2 b) shows all positions of a certain context, Figure 2 c) for another single context.

2.2. Prediction methods

2.2.1. Linear Prediction

The main proposed prediction technique is a variant of least-mean-square (LMS) filtering. In contrast to the commonly used approach, it does not directly predict the next signal value \( \mathbf{X} \), but the difference to its neighbour \( \mathbf{B} \). Let \( \mathbf{d} = \{d_i\} \) be a vector of eighteen differences in the causal neighbourhood of the next signal value \( \mathbf{X} \) = \( \mathbf{x} \) (Fig. 1) with

\[ d_1 = A - C , \quad d_7 = E - F , \quad d_{13} = K - L \]
\[ d_2 = C - B , \quad d_8 = D - I , \quad d_{14} = O - N \]
\[ d_3 = B - D , \quad d_9 = D - J , \quad d_{15} = J - Q \]
\[ d_4 = A - E , \quad d_{10} = E - K , \quad d_{16} = J - R \]
\[ d_5 = B - H , \quad d_{11} = H - O , \quad d_{17} = C - F \]
\[ d_6 = C - G , \quad d_{12} = F - M , \quad d_{18} = H - I \ . \]

The signal value \( x \) is then predicted by

\[ \hat{x}_B = B + \hat{d}_B = B + \sum_{i=1}^{18} w_i[px] \cdot d_i \quad (4) \]

resulting to the prediction error \( e_B = x - \hat{x}_B \). The last equation clips the estimation value to the known range of signal values \( [x_{\min}, x_{\max}] \). The weights \( w_i[px] \) depend on the current prediction context \( px \) and must be adapted after each prediction step in this context via

\[ w_i[px] \leftarrow w_i[px] + \mu_i \cdot e' \cdot \frac{d_i}{|d_i[px]| + 1} \ . \quad (5) \]

The clipped value

\[ e' = \text{sgn}(e_B) \cdot \min (|e_B|, \Theta[px]) \]

is used to slightly improve the adaptation process. The threshold \( \Theta[px] \) is not fixed as in [23], but the value is adaptively set to 2.5 times the averaged absolute prediction error \( e_{\min} \) in context \( px \).

The learning rate \( \mu_i \) is set to the values of

\[ \mu_i = \begin{cases} 0.0004 & \text{for } i \in \{1, 2, 3\} \\ 0.0002 & \text{for } i \in \{4, 5\} \\ 0.0001 & \text{for } i \in \{6, 7, \ldots, 18\} \end{cases} \]

reflecting that closer positions are more correlated than more distant values. The values \( d_i \) are normalised in (5) using the average of the absolute differences, which is recursively computed. This relates \( d_i \) in equation (5) to the difference values at former appearances of the same context. This predictor is called \( P_B \) in the sequel.

The relation of \( \mathbf{X} \) to \( \mathbf{B} \) via \( d_B \) in equation (4) implies a certain (vertical) correlation between \( \mathbf{X} \) and \( \mathbf{B} \). However, it might be that the correlation in horizontal direction is higher. This can be taken into account by computing another estimate \( \hat{x}_A = A + d_A \) in the same manner (predictor \( P_A \)). Since \( d_A \) is different to \( d_B \), the adaptation process leads to a different set of prediction coefficients \( w_i[px] \).

The differential prediction does not require a signal mean of zero for optimal weights estimation. This can be seen as its main advantage compared to conventional prediction set-ups.

2.2.2. Template matching (TM)

The linear prediction method explained above takes information from pixels into account, which can be spread over the entire image. As the contexts are determined based on the gradients in the vicinity of a certain pixel, the prediction coefficients can be adapted to different textures. Nevertheless, there are some contexts where the signal values elude from good prediction. In these cases, the method of template matching (aka pattern matching, or texture context matching, [25, 14]) can be successful. The main idea is to compare the texture at the current position \( \mathbf{X} \) with textures in the causal neighbourhood, Fig. 3. The higher the similarity between the textures, the higher the chance that \( X' \) is a good estimate for \( X \). The template used in the proposed scheme consists of the values at positions A, B, . . . , J (Fig. 1) and the search space is limited with \( S = 20 \) in order to keep the computational costs low. We use the
This is called predictor \( P \) prediction errors of predictor \( p \) at adjacent positions \( A, B, C, \) or \( D \). The weighting ensures that the influence of a predictor is higher when its error variance in the actual context \( px \) is lower and also the errors in the neighbourhood of the current position are lower compared to the errors of other predictors.

\[ \sum \text{absolute differences (SAD) as similarity measure. In total, eight estimates } \hat{x}_j \text{ with highest texture similarity found to be enough to be combined in a weighted fashion. The predictors estimate is} \]

\[ \hat{x}_{TM} = \frac{\sum_{j=1}^{8} a_j \cdot \hat{x}_j}{\sum_{j=1}^{8} a_j} \quad a_j = \frac{1}{SAD_j \cdot \delta_j + 0.01}, \quad (6) \]

while \( \delta \) is the Euclidean distance between \( X \) and \( X' \). The more similar the textures are or the closer the positions are, the higher is the influence of \( \hat{x}_j \) (larger weight \( a_j \)). The term ‘0.01’ is a safeguard in the case of identical textures (i.e., \( SAD = 0 \)). This is called predictor \( P_{TM} \).

### 2.3. Predictor blending

Having more than one method at hand, there must be a decision how to combine the individual estimates. If the predictor performance is expected to be good, the corresponding weight should be high and vice versa. In [20], the penalty is a combination of the error variance caused by each predictor in the past and the error energy in the neighbourhood of the current position. We adopt this approach in the following manner

\[ \hat{x} = \frac{\sum p \cdot w_p \cdot \hat{x}_p}{\sum p \cdot w_p} \quad p \in \{ P_A, P_B, P_{TM} \} \quad (7) \]

\[ \frac{1}{w_p} = \sigma_p^2[px] + 0.15 \cdot (|e_p[A]|^2 + |e_p[B]|^2) + 0.05 \cdot (|e_p[C]|^2 + |e_p[D]|^2) \]

with \( \sigma_p^2[px] \) equal to the variance of prediction errors generated by predictor \( p \) in the context \( px \). The values \( |e_p[| \) are the absolute prediction errors of predictor \( p \) at adjacent positions \( A, B, C, \) or \( D \). The

### 3. INVESTIGATIONS

#### 3.1. Predictor performance

Table 2 shows the prediction-error entropies for different images (taken from [28]). The image ‘barbara.y’ shows the highest difference when the predictors \( P_A \) and \( P_B \) are compared. Using a rotated version (‘barbara_rot’), it can be shown that the correlation properties are different in horizontal and vertical direction. The combination of both predictors (column \( P_{A,B,TM} \)) alleviates this characteristic in the case of the barbara image, while the combination never results to a higher entropy than the single predictors for all other images. The performance of predictor \( P_{TM} \) is comparatively low; however, its inclusion distinctly improves the results for most images (column \( P_{A,B,TM} \)). Only for images with very noisy texture (roebuck, three_flowers, xyxy_10bit), the applied blending of the predictors is not able to sufficiently down-weight the predictor \( P_{TM} \). Even though predictor \( P_{TM} \) is best for the binary image chekker_bw, its integration has adverse effects. Also for image barbara.y and its rotated version, the blending seems to not be optimal because the results for \( P_{A,TM} \) or \( P_{B,TM} \) are better. Since the predictor combination \( P_{A,B,TM} \) shows the best performance on average, we take this as proposed method in the following.

Table 3 compares the prediction performance of the proposed method with other predictive methods for which (i) software was available and (ii) it is known that they compress better than the standard JPEG-LS. The average entropies show a significant gain compared to CoBalP method [13], which is based on fixed prediction contexts. It also can be seen that the proposed approach is more flexible (with respect to the image characteristics) than the AWLS prediction used in [6]. The brute-force approach of MRP\(^1\) yields the lowest entropies for the 8bit images with the disadvantage of very long computation times being required.

#### 3.2. Coding performance

The CoBalP2 predictor is complemented with a coding scheme based on arithmetic coding using an adaptive merging of coding contexts and common techniques, such as histogram-tail truncation.

---

\(^1\)The source code of MRP had to be modified slightly as the original code processes only images with width and height being multiples of eight.


<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>barbara</td>
<td>4.506</td>
<td>4.236</td>
<td>4.140</td>
<td>3.959</td>
</tr>
<tr>
<td>kodim01</td>
<td>5.429</td>
<td>5.338</td>
<td>5.391</td>
<td>5.234</td>
</tr>
<tr>
<td>woman_G</td>
<td>4.668</td>
<td>4.540</td>
<td>4.608</td>
<td>4.497</td>
</tr>
<tr>
<td>k05_Y</td>
<td>3.342</td>
<td>3.288</td>
<td>3.270</td>
<td>3.204</td>
</tr>
<tr>
<td>roebuck</td>
<td>3.646</td>
<td>3.577</td>
<td>3.524</td>
<td>3.405</td>
</tr>
<tr>
<td>chekker</td>
<td>1.043</td>
<td>0.470</td>
<td>1.239</td>
<td>0.299</td>
</tr>
<tr>
<td>chekker_bw</td>
<td>0.048</td>
<td>0.005</td>
<td>0.158</td>
<td>0.002</td>
</tr>
<tr>
<td>tree_flowers</td>
<td>5.284</td>
<td>5.151</td>
<td>5.186</td>
<td>5.002</td>
</tr>
<tr>
<td>average</td>
<td>3.496</td>
<td>3.326</td>
<td>3.440</td>
<td>3.200</td>
</tr>
<tr>
<td>k05_U9bit</td>
<td>2.517</td>
<td>2.490</td>
<td>2.537</td>
<td>–</td>
</tr>
<tr>
<td>xray_10bit</td>
<td>3.902</td>
<td>3.499</td>
<td>3.893</td>
<td>–</td>
</tr>
<tr>
<td>RNAi_dna</td>
<td>6.369</td>
<td>6.047</td>
<td>6.112</td>
<td>–</td>
</tr>
</tbody>
</table>

The investigations have shown that the combination of different predictors generally decreases the entropy. Especially the integration of template matching benefits the performance for most images. Unfortunately, the linear blending does not lead to the best result in each case and more investigation in this direction is necessary. In principle, sub-pixel accuracy could be used to improve the template matching at the cost of higher computational complexity, similar to the approaches in video compression. There are different sizes of templates for prediction-context determination (five differences) and for LMS filtering (8 differences). This is a general problem because positions which are mapped to the same prediction context based on the only non-symmetric compressor, which determines the prediction parameters on the encoder side and transmit them to the decoder along with the compressed data. The total compression time of CoBaLP2 reduces to 84 and 67 seconds, respectively, if the template matching is switched off.

Table 4. Bitrates in bits per pixel for different approaches. (CALIC and MRP cannot process images with more than 8bpp.)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>barbara</td>
<td>3.973</td>
<td>3.915</td>
<td>4.339</td>
</tr>
<tr>
<td>kodim01</td>
<td>5.973</td>
<td>5.082</td>
<td>5.091</td>
</tr>
<tr>
<td>woman_G</td>
<td>4.147</td>
<td>4.220</td>
<td>4.295</td>
</tr>
<tr>
<td>k05_Y</td>
<td>3.188</td>
<td>3.159</td>
<td>3.318</td>
</tr>
<tr>
<td>roebuck</td>
<td>3.186</td>
<td>3.155</td>
<td>3.343</td>
</tr>
<tr>
<td>chekker</td>
<td>0.314</td>
<td>1.984</td>
<td>0.163</td>
</tr>
<tr>
<td>chekker_bw</td>
<td>0.006</td>
<td>0.397</td>
<td>0.007</td>
</tr>
<tr>
<td>tree_flowers</td>
<td>5.039</td>
<td>5.085</td>
<td>5.276</td>
</tr>
<tr>
<td>average</td>
<td>3.104</td>
<td>3.375</td>
<td>3.229</td>
</tr>
</tbody>
</table>

The proposed prediction scheme is based on least-mean-square filtering with successive adaptation of filter coefficients after processing of each single pixel. It could be shown that the performance is comparable to least-squares approaches, which use the vicinity of the current position as training data for the determination of prediction coefficients. We have used a prediction filter with fixed order of 18. The predictor could probably be improved by adapting the order to the characteristic (e.g., the noise content) of the image.
5. REFERENCES


