Models for Static and Dynamic Texture Synthesis in Image and Video Compression

Johannes Ballé, *Student Member, IEEE,* Aleksandar Stojanovic, *Student Member, IEEE,* and Jens-Rainer Ohm, *Member, IEEE*

***Abstract*—In this paper, we investigate the use of linear, parametric models of static and dynamic texture in the context of conventional transform coding of images and video. We propose a hybrid approach incorporating both conventional transform coding and texture-specific methods for improvement of coding efficiency. Regarding static (i.e. purely spatial) texture, we show that Gaussian Markov random fields (GMRF) can be used for analysis/synthesis of a certain class of texture. The properties of this model allow us to derive optimal methods for classification, analysis, quantization and synthesis. For video containing dynamic textures, a linear dynamic model can be derived from frames encoded in a conventional fashion. We show that after removing effects from camera motion, this model can be used to synthesize further frames. Beyond that, we show that using synthesized frames in an appropriate fashion for prediction leads to significant bitrate savings while preserving the same PSNR for sequences containing dynamic textures.**

***Index Terms*—perceptual coding, texture synthesis, Gaussian Markov random field, dynamic texture, video coding**

I. Introduction

Today's video coding algorithms are mostly based on the paradigm of pixel fidelity. Measures such as value deviation in terms of absolute or quadratic error (e.g., PSNR, SSE) are typically used as cost functions in the decisions of the encoder [1]. These measures, however, are only used as an approximation to the perceptual distortion that is introduced by lossy coding. Techniques to more closely model human perception could be a key to increase coding efficiency of future image and video coding methods.

The primary observation leading to our approach is the success of recent texture synthesis methods [2]–[6]. While maintaining a striking similarity of the output texture to the input example (static or dynamic), all of these methods introduce randomness, i.e., they allow to generate multiple instances of visually identical texture given the same input. This suggests that the introduced randomness is actually irrelevant to the viewer and could be exploited for coding applications. Formally, this implies that images and video should be viewed as instances of a hidden statistical process. Very similar concepts have been successfully applied in Linear Predictive Coding (LPC) of speech [7]. The main difficulty associated with this idea is to make out the exact scope of what is perceived as "texture" or "structure."

Various prior work is reported where texture is removed from an image or video representation and compressed separately. Most of these methods [8]–[11] employ non-parametric texture synthesis. A review is available in [12]. The approach presented here is – in a certain sense – more conservative. It is influenced by the observation that texture synthesis is also useful for prediction [13], [14]. This is due to the fact that texture generally exposes varying degrees of randomness. In the context of coding, it is desirable to exploit predictability as much as possible before resorting to open-loop synthesis.

In this work, we consider static and dynamic texture models separately for image and video coding, respectively. The former type is purely spatial, i.e., corresponds to texture attached to an object in a scene and exhibits spatial homogeneity, such as fabric, surfaces of stone, walls, etc. The latter are video sequences with changing texture showing temporal stationarity, e.g. water surfaces, whirlwind, clouds, or even head-and-shoulder sequences.

With respect to static texture (Section II), we examine a texture model, the Gaussian Markov random field, that is limited with respect to the class of texture that it describes. However, it is interesting due to its linearity. Since it is analytically tractable, we can show that its inherent "randomness" is defined using the mathematical concept of phase. Furthermore, we present a method to identify such texture in natural images, a comprehensive quality measure for the model, a (Maximum Likelihood) optimal method to estimate its parameters, and a synthesis algorithm that is guaranteed to produce a valid sample. Preliminary visual results are shown at the end

The authors are with Institut für Nachrichtentechnik, RWTH Aachen University, 52056 Aachen, Germany. e-mail: {balle,stojanovic,ohm}@ient.rwth-aachen.de

of the section.

Section III relates to dynamic textures. The chosen model is an adaptation of a model from the area of computer vision and allows the description of the dynamic properties by a compact vector state model, for which the parameters are derived by singular value decomposition. In contrast to the method for static texture, we show that this model can be used for prediction in video compression. An important aspect of dynamic texture is to distinguish regular (e.g. camera or object movement) motion from random motion within the texture, such as local movements of leaves, grass, water surfaces. We address this problem with a special algorithm that precedes the synthesis. We then show how the compression ratio in conventional video coders can be improved by introducing dynamic texture prediction into the coding loop in an appropriate manner. Section IV concludes the paper.

II. Static Texture

Methods to analyze and synthesize visual texture in natural images have been researched for the last 50 years [15]. An extremely useful formalization of static texture is the homogeneous Markov random field [16]. It can be characterized by its conditional probability density

$$\mathrm{p}(t(x) \mid t(x'), x' \in \mathcal{N}(x)) \tag{1}$$

where $t(x)$ represents a random field on the 2D lattice locations x and $\mathcal{N}(x)$ represents a neighborhood of pixel values around x.

Our approach with respect to random spatial texture is to investigate the Gauss–Markov random field (GMRF) model [17] for coding applications. This model is suitable for a certain class of microtexture and due to its special properties, texture can be detected using higher order statistics. This step is described in Section II-A.

In order to exploit the irrelevance inherent in texture, we must break the pixel fidelity paradigm. As a consequence, established quality assessment measures, such as the peak signal-to-noise ratio (PSNR) or the structural similarity index (SSIM) [18], lose their utility. With respect to the GMRF model, this can be seen by observing that the expected mean squared error (MSE) between two samples of the same, say, white Gaussian noise process is double the variance of the process, while the visual appearance is identical. For SSIM, a similar argument holds, as it is based on the sample cross-covariance. At this time, we cannot quantify the visual quality of our approach using an established criterion, as the available criteria are either unsuitable or un-established. Therefore, we present visual results in Section II-G.

Due to the special properties of GMRF models, it is, however, not impossible to define comprehensive quality measures that are non-negative and equal to zero if and only if the compared models are equivalent. We extend a number of concepts for estimation, quantization and coding of the model parameters known from speech coding in Sections II-B, II-D, and II-F, in order to apply them to texture images. For GMRFs, a sampling algorithm exists which is guaranteed to produce a valid sample [17]. We outline our approach in Section II-E.

A. Classification

The class of texture we are concerned with has two defining properties. Firstly, the field is homogeneous: the conditional density (1) is independent of x. Secondly, the conditional densities are Gaussian. Given an image segment, we would like to determine how likely it is a sample of such a field. Formally, this corresponds to a p-value test. Such tests have been carried out using polyspectra and the bicoherence function, which is defined as a normalization of the bispectrum [19], [20].

We propose a testing framework based on an invertible, polar separable subband transform known as the steerable pyramid [21]. Using a subband transform allows us to perform a test for homogeneity and Gaussianity not only localized in space, but also localized in frequency. We call this a space–frequency partitioning of the signal, where each partition is either classified as texture or non–texture. Spatially, the partitioning corresponds to a classic block partitioning. In the frequency domain, we allow partitioning with respect to scale, but not with respect to orientation. Furthermore, we require partitions classified as texture to be grouped in the finest scales at each spatial location, such that we can define a low-pass filter $g(x)$ for each spatial block that separates texture from non-texture. This method can be viewed as a simple structure–texture decomposition. It implies that only high-pass characteristics of texture are subjected to synthesis, while low-pass characteristics are coded in a conventional way. This solves the problem of modeling smooth variations (e.g., as mentioned in [11]) in a manner that is scale-invariant and compatible with the texture model.

The test statistic is simply the sample kurtosis in each subband. Empirically, we have found that it is suited quite well to detect homogeneous, Gaussian texture. Analytically, we can show that it is a measure for Gaussianity by considering the Gaussian Scale Mixture (GSM) model [22]. It has been found that this model

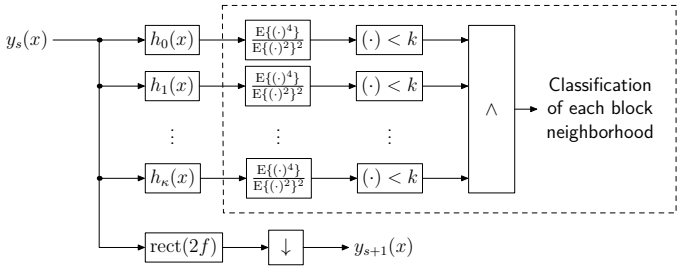


Fig. 1. Space–Frequency classification of the input signal.

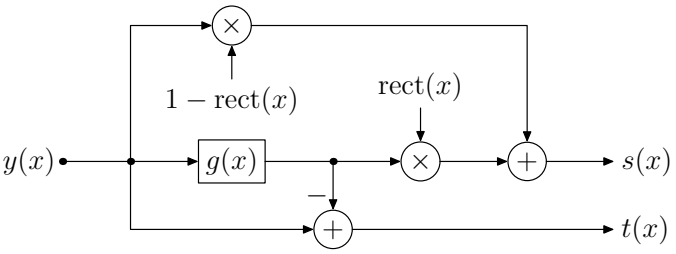


Fig. 2. Texture removal based on classification.

accounts for the conditional histograms of neighboring subband coefficients in the steerable pyramid: $q = \sqrt{z} \cdot u$, where q is the vectorized neighborhood, z is the so-called multiplier and u an i.i.d. Gaussian random vector which is independent of z. If we assume without loss of generality that $\mathrm{E}\{z\} = 1$ (where $\mathrm{E}\{\cdot\}$ denotes expectation) the kurtosis of q is equal to:

$$\frac{\mathrm{E}\left\{q^4\right\}}{(\mathrm{E}\left\{q^2\right\})^2} = \frac{\mathrm{E}\left\{z^2\right\}}{(\mathrm{E}\left\{z\right\})^2} \cdot \frac{\mathrm{E}\left\{u^4\right\}}{(\mathrm{E}\left\{u^2\right\})^2} = 3\mathrm{E}\left\{z^2\right\} \quad (2)$$

Computing sample kurtosis of the subband coefficients is therefore equivalent to estimating the variance of the multiplier, up to a constant factor. This method was also used in [23] in the context of denoising. As the variance of the multiplier approaches zero, q approaches a Gaussian vector.

To see why kurtosis is also a measure for homogeneity, we interpret it using higher-order spectrum theory [20].

$$\begin{aligned}\frac{\mathrm{E}\left\{q^4\right\}}{(\mathrm{E}\left\{q^2\right\})^2} - 3 &= \frac{\int\int\int C_4^q(f_1,f_2,f_3)df_1df_2df_3}{\left(\int C_2^q(f)df\right)^2} \\ &= \frac{\int\int\int C_4^y(f_1,f_2,f_3)M_4^h(f_1,f_2,f_3)df_1df_2df_3}{\left(\int C_2^y(f)M_2^h(f)df\right)^2} \quad (3)\end{aligned}$$

where C_n^q represents the nth-order cumulant spectrum of q; M_n^h the nth-order moment spectrum of h; and the integral limits have been omitted for brevity. If the non-negative band-pass frequency response $H(f)$ is narrow enough, this expression can be seen as an approximation to Hinich's transient detector [19]. Here, it is applied to the trispectrum of a 2D signal as opposed to the bispectrum of a 1D signal. Instead of estimating the tricoherence function, we approximate slices of it shaped by the moment spectra of $H(f)$.

The classification proceeds as follows: We start with the finest scale ($s = 0$). The input image $y_0(x)$ is fed into an array of oriented band-pass filters (Figure 1). For accuracy and computational efficiency, these filters are implemented using the DFT. Then, the image is block-partitioned, and the sample kurtosis is evaluated for each block, including a circular neighborhood around the block of a predefined size. For each block, the sample kurtosis is compared to a pre-defined threshold k which could be interpreted as a "confidence parameter." If the threshold is not exceeded for any of the oriented subbands, we classify the block and scale as texture. A higher threshold therefore implies a higher probability of classifying structure as texture. We then examine the next coarser scale by decimating $y_0(x)$ and performing the same signal processing recursively. Note that the circular neighborhood is not scaled; therefore, we process larger image areas for coarser scales. The processing ends for each block if either the current scale is classified as non-texture or we reach a pre-defined maximum scale. The maximum scale is bounded by image resolution; it does not make sense to use impulse responses $h_i(x)$ which approach the extents of the image.

The low-pass filter yielding the structure component for each block is defined in the Fourier domain by

$$G(f) = 1 - H_R(f) - \sum_{s}^{s_{\mathrm{thr}}-1} \sum_{i} H_i(2^s f) \quad (4)$$

where $H_R(f)$ is the filter corresponding to the high-pass residual subband and s_{thr} is the index of the scale where the threshold exceeded. If $s_{\mathrm{thr}} = 0$, Eq. (4) is not used and $G(f)$ is set to 1. A block diagram of the processing carried out for block-wise texture removal is shown in Figure 2. An example of texture removal is shown in Figure 4 (top right).

At this point, we may be confident of having identified homogeneous, Gaussian texture. This class of texture is related to the ones treated in [24], where texture is synthesized by randomizing Fourier phase of a texture sample. Introducing the Markov property and hence the GMRF model now leads to a dimensionality reduction: Any Gaussian random field can be approximated by a GMRF [17].

B. Texture Model

The homogeneous GMRF model is characterized by a Gaussian white noise field $w(x)$ amplified by factor σ

and filtered by an all-pole filter $a(x)$ [16]:

$$s(x) = \sigma w(x) - a(x) * s(x) \tag{5}$$

The 2-vector x denotes spatial locations; $*$ denotes 2D convolution. Like $w(x)$, $s(x)$ is a random field. $a(0)$ must be equal to zero. In the z-domain, it is described as follows:

$$\begin{aligned} S(z) &= \sigma W(z) - A(z) \cdot S(z) \\ &= \frac{\sigma}{1 + A(z)} \cdot W(z) \end{aligned} \tag{6}$$

where $z = (z_1, z_2)^{\mathrm{T}}$ represents a 2D complex-valued vector. The power spectral density of $s(x)$ is given by:

$$\phi_{ss}(e^{j2\pi f}) = \mathrm{E}\left\{\left|S(e^{j2\pi f})\right|^2\right\} = \frac{\sigma^2}{\left|1 + A(e^{j2\pi f})\right|^2} \tag{7}$$

where f is a 2D real-valued vector and $e^{(\cdot)}$ implies element-wise exponentiation.

It can be seen from Eq. (6) that the phase of the resulting process amounts to the sum of the respective phases of $W(e^{j2\pi f})$ and the frequency response of the filter. Since the phase of $W(e^{j2\pi f})$ is independent and identically distributed (i.i.d.) with a uniform distribution over all values of f on the unit square, the phase of $A(e^{j2\pi f})$ has the same distribution (any constant phase term added to an i.i.d. uniform phase will produce an i.i.d. uniform phase). Thus, the statistics of the resulting process are completely and sufficiently described by its power spectral density. The irrelevance inherent in such texture manifests itself in the mathematical concept of phase.

Furthermore, we can see from Eq. (6) that the magnitude spectrum of $S(e^{j2\pi f})$ varies with a Rayleigh noise. The authors of [24] observed that this variation seems not to influence the visual appearance of texture. With the above development, we can relate this observation to the homogeneity of the random field implied by Eq. (5).

C. Estimation

Estimation of the model parameters is not straightforward for general MRFs. However, for the unilateral GMRF model, the maximum likelihood estimate of the model parameters is linear [25]. Its parameters, $\mathbf{a}$, corresponding to the vectorized coefficients of $a(x)$, and σ can be estimated by solving the linear equation system:

$$\mathbf{V}\hat{\mathbf{a}} = (\hat{\sigma}, 0, \ldots, 0)^{\mathrm{T}} \tag{8}$$

where $\mathbf{V}$ is an appropriately constructed covariance matrix. Alternatively, we can estimate σ using the prediction error filter interpretation:

$$\hat{\sigma}^2(\mathbf{a}) = \frac{1}{\|\mathbb{D}\|} \sum_{x \in \mathbb{D}} (s(x) + a(x) * s(x))^2 \tag{9}$$

where $\mathbb{D}$ is the set of lattice locations. Details on this estimator and its relation to spectral estimation and linear prediction can be found in [25]–[27]. Although other estimators are possible, we choose the linear estimator for its computational efficiency and because it is consistent and unbiased for the unilateral model.

D. Quality Measure

Assuming that we have found the true parameters of the random field (or a very good approximation), we now consider a perturbed set of parameters (for example, by quantization), which we denote with a tilde. The Itakura measure [28] was introduced as a measure for the similarity of an autoregressive model to the speech signal it was estimated from. A 2D Itakura measure can be defined for the unilateral GMRF model as:

$$D_{\mathrm{I}}(\tilde{\mathbf{a}}, \mathbf{a}) = \ln\left(\frac{\hat{\sigma}^2(\tilde{\mathbf{a}})}{\sigma^2(\mathbf{a})}\right) \tag{10}$$

Since $\mathbf{a}$ is adapted to the data while $\tilde{\mathbf{a}}$ is predetermined, the denominator in this expression is always smaller or equal to the numerator. Hence, the measure is always non-negative and a value of 0 represents the smallest possible distance, indicating that $\tilde{\mathbf{a}} = \mathbf{a}$.

We can find a frequency domain interpretation of the Itakura measure by plugging in Eq. (9) and using Parseval's relation:

$$D_{\mathrm{I}}(\tilde{\mathbf{a}}, \mathbf{a}) = \ln\left(\int_{\square} \frac{|1 + \tilde{A}(e^{j2\pi f})|^2}{|1 + A(e^{j2\pi f})|^2} df\right) \tag{11}$$

The Itakura distance is thus equivalent to the integral of the quotient of the gain-normalized power spectra ϕ_{ss} and $\phi_{\tilde{s}\tilde{s}}$ of the true and the distorted process, respectively, over the unit square ($\square$), on a logarithmic scale. Since we showed above that the process is sufficiently described by its power spectral density, the 2D Itakura measure, along with a gain measurement, is comprehensive in the sense that it captures the characteristics of the random field. Other spectral measures could be used, but according to our experience, the 2D Itakura measure corresponds quite well to visual quality (Fig. 3) while being extremely cheap to compute.

E. Synthesis

The texture removal described in Section II-A is designed to remove block patches of texture from larger, homogeneous patches. To avoid block artifacts in the reconstruction, we must sample texture inside the blocks with respect to the borders ("inpainting"). This can be done optimally by considering the blocks as finite subsets of stationary GMRFs. We can construct the precision

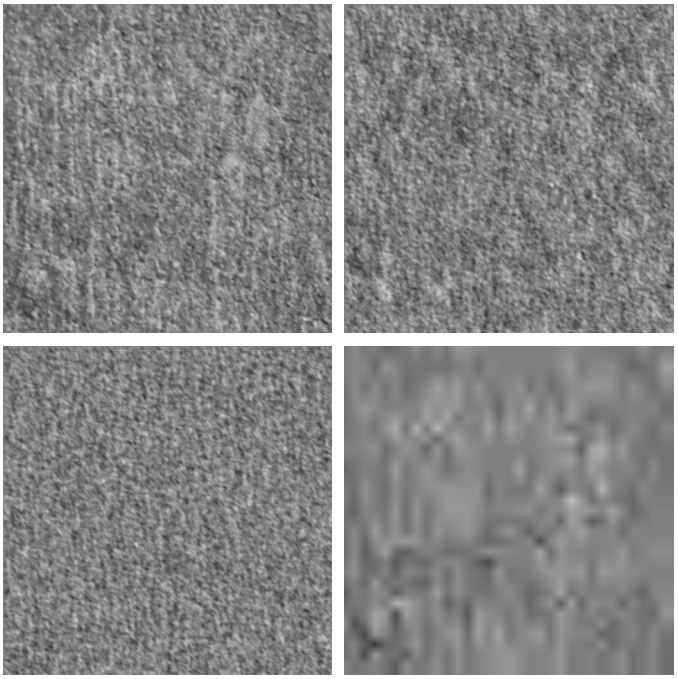


Fig. 3. Bark texture. Top left: original, 512×512 pixels. Top right: our method, 299 bytes, $D_{\mathrm{I}} \cdot 512^2 = 0.01$. Bottom left: our method, 117 bytes, $D_{\mathrm{I}} \cdot 512^2 = 807.77$. Bottom right: JPEG 2000, 325 bytes, $D_{\mathrm{I}} \cdot 512^2 = 59746.19$.

matrix $\mathbf{Q}$ [17] of such a finite GMRF using the model parameters of the stationary GMRF representation:

$$Q_{ij} = p(\alpha(i) - \alpha(j)) \tag{12}$$

where $\alpha(i)$ is any bijective mapping from matrix row (or column) indexes to the spatial locations inside the block and $p(x) = (\delta(x) + a(x)) * (\delta(x) + a(-x))$ is the autocovariance function of the prediction error filter corresponding to the unilateral GMRF. Sampling from this finite field is achieved by solving

$$\mathbf{L}^{\mathrm{T}} \hat{t}_w = \sigma w \tag{13}$$

where $\mathbf{L}$ is the Cholesky factor of $\mathbf{Q}$, w is an i.i.d. Gaussian random vector and $\hat{t}_w$ is the resulting vectorized sample. To account for the border conditions, we further solve a similar system

$$\mathbf{L}\mathbf{L}^{\mathrm{T}} \hat{t}_b = \mathbf{Q}' b \tag{14}$$

where $\mathbf{Q}'$ is an extension of $\mathbf{Q}$ constructed analogously and b are the vectorized border values. The final reconstruction of the block is $\hat{t} = \hat{t}_w + \hat{t}_b$ (α then needs to be inverted to obtain the pixel values inside the block). Both systems can be solved efficiently by back- and forward-substitution once the Cholesky factors of $\mathbf{Q}$ are known.

Fig. 6. "Barbara" (200×200 crop). Top: JPEG2000 reconstruction at 0.8 bpp. Bottom: reconstruction using our method at 0.8 bpp.

F. Quantization & Coding

Our solution to the problem of efficient coding of the filter coefficients is to use vector quantization (VQ) [7]. In previous work [29], [30], we introduced a block-adaptive version of the GMRF model and applied on-line VQ to texture blocks in order to exploit the similarity of the model parameters in similarly textured blocks. We use the same method here. The vector quantizer is optimized for the Itakura distance. Codebook vectors are signalled using sufficiently fine scalar quantization.

Unfortunately, there is no direct relationship between the Itakura distance and the scalar quantization step size. Therefore, it is possible that a finer quantization can actually lead to a worse approximated spectrum. However, with sufficiently fine quantization, the Itakura distance converges to zero and the marginal distribution of the coefficients can be exploited for entropy coding [31]. A better trade-off between rate and distortion may be possible with the use of fixed VQ codebooks.

G. Experimental Results

When compressing images composed of texture only, it appears obvious that, asymptotically, the visual appearance of texture is captured more compactly using model parameters than by conventional techniques (Figure 3). The question is how big the coding improvement will be in a practical application. Although it is possible to optimize the method presented here with respect to rate–distortion performance, this has not yet been carried out. The entropy coding is inspired by [31]. Due to the different decomposition scheme, this may not be optimal and should be regarded as preliminary.

In Figures 4 and 5, visual results are presented, where an improvement in terms of visual quality is achieved versus the reference codec. Due to the block artifacts resulting from texture removal, we use a block-based codec, the emerging High Efficiency Video Codec (HEVC), for the structure component, as well as for the

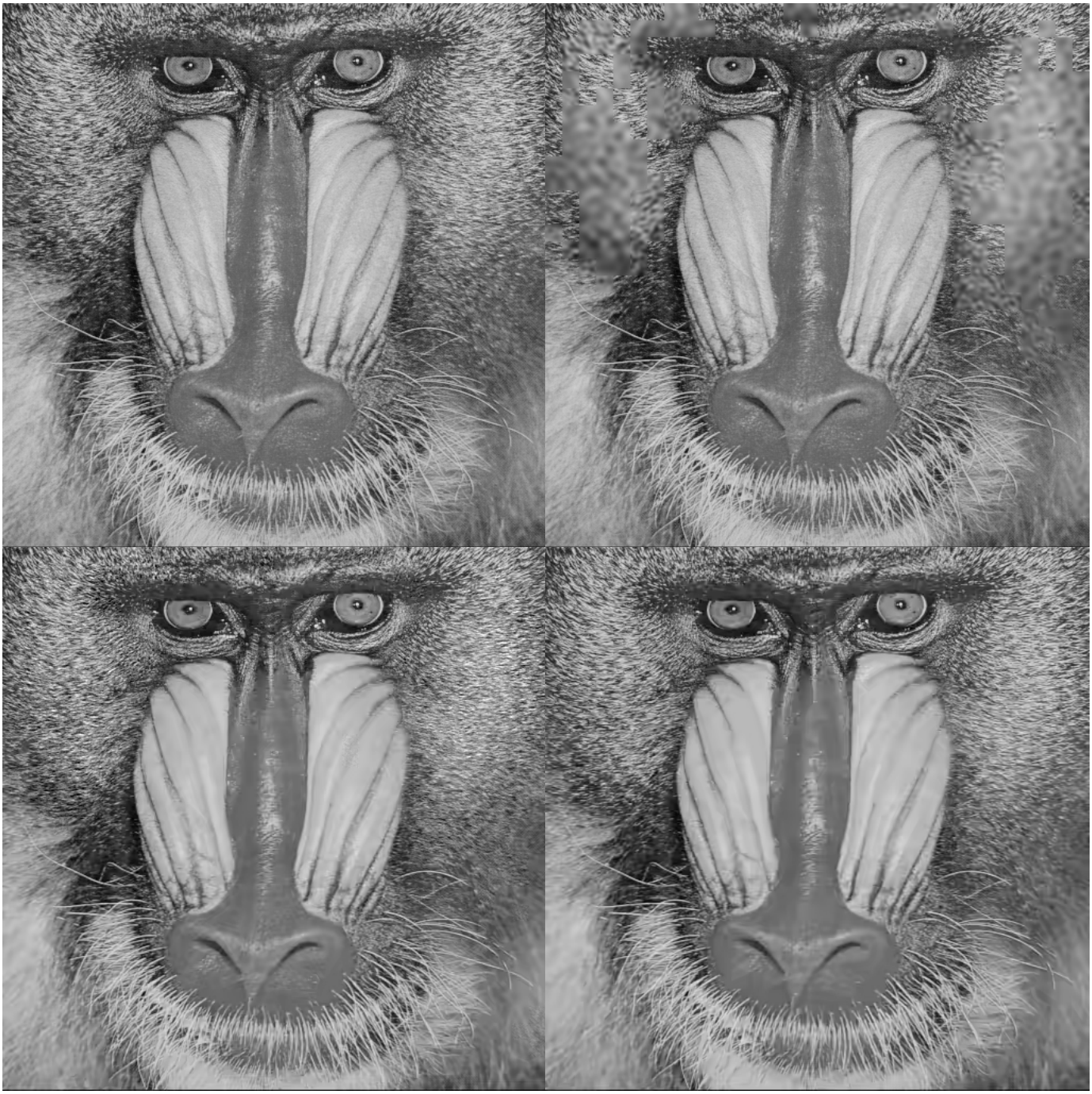


Fig. 4. Result for standard test image “Baboon”. Top left: original; top right: structure component after classification and texture removal; bottom left: reconstruction at 0.9 bpp (4.5% for texture); bottom right: reference (HEVC) at 0.9 bpp.

reference. Its transform size is aligned to the 16×16 block partitioning of the classification scheme. We make the following observations:

Removal of random texture results in improved rate–distortion performance for the structure component. In Figure 4, the structure component (top right) is compressed to 0.86 bpp at a PSNR of 31.7, while the original (top left) at 0.9 bpp yields a PSNR of 29.1. Visually, this manifests itself in increased detail in the region between mouth and eyes when comparing our method to the reference codec (bottom). This effect can be explained by the fact that random texture is, by definition, difficult to decorrelate (the most extreme case being a random field with a white spectrum).

Our method is only efficient compared to conventional coding if the area of homogeneous Gaussian texture is large enough to match the rate needed for texture parameters in an RD sense. Clearly, this depends on

Fig. 5. Result for test image "Kodim04" [32]. Left: reconstruction at 0.7 bpp (3% for texture); Right: reference (HEVC) at 0.7 bpp.

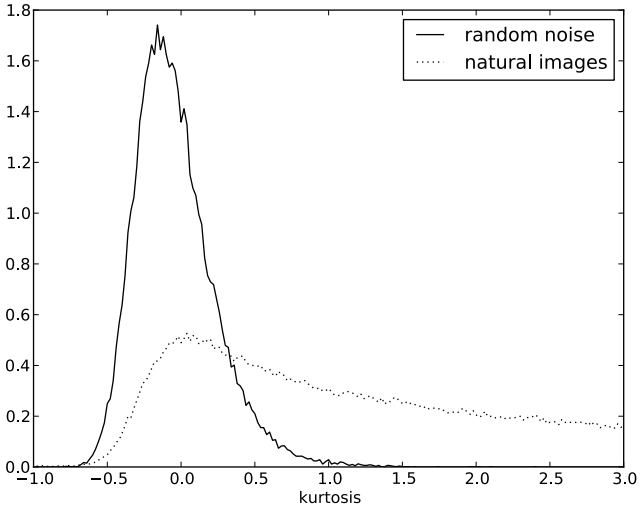


Fig. 7. Monte Carlo simulation result for kurtosis estimator. Solid line: Gaussian random noise. Dotted line: Collection of natural images ("Kodak" set [32]).

image content. It is the case in Figure 5, where the skin texture is improved. It may not be the case, for instance, in small textured areas comprising just one block. This suggests that our method should be used adaptively.

The method is reliable in the sense that the signal processing following classification is adapted to the class of texture. The estimation error introduced is minimized by using the ML optimal estimator. The error introduced by quantization is unavoidable, but can be monitored using the Itakura distance. Sampling from the MRF is done exactly, up to numerical round-off. A certain chance of misclassification is inevitable by principle, but can be controlled using the kurtosis threshold k. We used a Monte Carlo simulation to compare the kurtosis estimate for Gaussian noise against estimates obtained from a set of natural images (Figure 7). The kurtosis obtained from natural images can have values of up to the order of 100 (the figure is cropped). Note that natural images also contain homogeneous Gaussian texture, which explains the overlap of the plots. We used a threshold of 0.5 for the examples shown here. In Figure 5, it can be seen that the texture of the fabric (lower left) is too close to the threshold. Therefore, it is approximated using a GMRF such that the fabric appears slightly distorted. The variance of the kurtosis estimate can be decreased if the data size (the size of the neighborhood around each spatial block) is increased. Thus, we face a trade-off between spatial resolution and probability of error.

Computationally, the method is more complex than traditional coding methods. In the encoder, this is due to the subband transform and parameter estimation (which could be parallelized). In the decoder, the Cholesky factors corresponding to all centroid vectors must be found. Usually, this will be a number on the order of 10, depending on the image content. Subsequently, the texture blocks are reconstructed using back- and forward-substitution in two iterations. In total, images of the size 512×512 take approximately 5 seconds

to reconstruct using a (single-threaded) Python/SciPy implementation on a MacBook using an Intel Core 2 Duo processor.

A different example is shown in Figure 6, where we used a previous method based on denoising for texture removal [30]. With this method, block artifacts in the structure component are avoided. We used JPEG2000 [33] to encode the structure component. Here, texture flattening is prevented (carpet). However, with this method, it is not possible to remove high-contrast texture such as in Figure 4 without destroying image structure, due to limitations of the denoising algorithm [30].

The preliminary results shown here indicate that our method for compression of Gaussian texture has potential and should be further optimized, particularly with respect to entropy coding and rate control. This is a topic of ongoing research.

III. DYNAMIC TEXTURES

A. Introduction

While static textures undergoing motion in a video sequence can be well described with conventional motion compensation within the scope state-of-the-art video codecs like H.264/AVC, dynamic textures are difficult to encode and reproduction with pixel fidelity is often very expensive in terms of bitrate. Hence the question of separating relevant from irrelevant content naturally re-arises in this context.

In the sequel we want to show that by using dynamic textures synthesis algorithms like the one presented in [6], sequences that are visually equivalent to the original can be generated from short sample sequences, which implicitly shows that for the particular instance of dynamic textures, any frame that comes after a given number of initial frames is irrelevant. In fact after transmitting these initial frames in a classical way and deriving a dynamic model from the latter, the remaining frames can be synthesized, resulting in a sequence which is equivalent to the original for a human observer. We will address the problem that the model from [6] is only valid for sequences with static camera and without zoom.

The remaining question is how to combine synthesis with conventional video coding in a fashion that will allow for bitrate savings without jeopardizing the advantage of the guarantee not to loose semantically important content (at least as long as the quality is high enough) which comes with conventional video codecs. Our investigations show that even when we only use the model for prediction we still obtain considerable bitrate savings for some sequences. This gives a clue of the great potential inherent to the model when considering applications going beyond pixel fidelity.

B. Dynamic Texture Synthesis

Let $\{y(t)\}_{t=1\ldots\tau}$, $\{y(t)\} \in \mathbb{R}^m$ be a noisy video sequence, $y(t) = I(t) + w(t)$, where $w(t) \in \mathbb{R}^m$ is the noise in the sequence. Furthermore, let $y_\mathrm{m} \in \mathbb{R}^m$ be the temporal mean of the sequence and $y_\mathrm{d}(t) = y(t) - y_\mathrm{m}$. A dynamic texture sequence of frames $y(t)$ can then be modeled as an ARMA-process (autoregressive moving average):

$$\begin{cases} x(t) = \mathbf{A}x(t-1) + \mathbf{B}v(t) \\ y(t) = \mathbf{C}x(t) + y_\mathrm{m} + w(t), \end{cases} \tag{15}$$

with initial condition for the state vector $x(0) = x_0$, an unknown stationary distribution $q(\cdot)$ with $v(t) \overset{i.i.d.}{\sim} q(\cdot)$, and given noise $w(t) \overset{i.i.d.}{\sim} p_w(\cdot)$. In this case $y_\mathrm{d} = \mathbf{C}x(t)$. The main assumption leading to this representation is that individual frames in a sequence consisting of dynamic texture are realizations of the output of a dynamical system driven by an independent and identically distributed (i.i.d.) process.

In this model $\mathbf{A} \in \mathbb{R}^{n\times n}$ is the state transition matrix, $\mathbf{C} \in \mathbb{R}^{m\times n}$ the observation matrix, and n the order of the model and the dimension of the state vector $x(t)$.

If we write a sequence in matrix form, each column in the sequence matrix Y is one frame of the sequence. The frames are converted by writing the pixels of the frames consecutively into one vector. For YCbCr sequences the two chroma components are appended, see [6] and [34]. $\mathbf{Y}_k^l = [y(k), \ldots, y(l)] \in \mathbb{R}^{m\times(l-k+1)}$ denotes the part of the sequence containing the frames from k to l and m is the number of pixels per frame. (15) becomes

$$\begin{cases} \mathbf{X}_1^\tau = \mathbf{A}\mathbf{X}_0^{\tau-1} + \mathbf{B}\mathbf{V}_0^{\tau-1} \\ \mathbf{Y}_0^{\tau-1} = \mathbf{C}X_0^{\tau-1} + \mathbf{Y}_\mathrm{m} + \mathbf{W}_0^{\tau-1}, \end{cases} \tag{16}$$

where $\mathbf{Y}_\mathrm{m} \in \mathbb{R}^{m\times\tau}$ consists of τ equal columns y_m. A suboptimal but computationally very efficient method to determine the observation matrix $\mathbf{C}$ is computing a singular value decomposition

$$\mathbf{Y}_\mathrm{d} = \mathbf{U}\mathbf{S}\mathbf{V}^\mathrm{T} \tag{17}$$

and set $\mathbf{C} = \mathbf{U}$ and $\mathbf{X} = \mathbf{S}\mathbf{V}^\mathrm{T}$. The number of singular values is equal to the number of frames τ in the sequence. If singular values are omitted as in Doretto's approach [6], we have

$$\mathbf{Y} = \tilde{\mathbf{C}}\tilde{\mathbf{X}} + \mathbf{Y}_\mathrm{m} + \mathbf{W}. \tag{18}$$

In fact Doretto et al. [6] assume the order of the model n to be smaller than the number of training frames τ, so

only n singular values are kept. Further a least squares approximation of $\mathbf{A}$, $\widehat{\mathbf{A}}(\tau)$, can be found with:

$$\widehat{\mathbf{A}}(\tau) = arg\min_{\mathbf{A}} \|\mathbf{X}_1^{\tau} - \mathbf{A}\mathbf{X}_0^{\tau-1}\|. \qquad (19)$$

This model presented in [6] allows the extrapolation of a dynamic texture sequence to infinite length by driving the model with a noise process $v(t)$. Clearly, $x(t)$ is a low-dimensional representation of $y_{\mathrm{d}}(t)$ and hence of $y(t)$. The relation between $x(t)$ and $y(t)$ is given in the second line of (15). In the low dimensional space an AR model can be derived for $x(t)$, which is given in the first line of (15). Once all the model parameters are known, new frames are generated, which consists in finding new samples $x(t+n)$ by driving the model with random noise $v(t+n)$.

C. Camera motion compensation and synthesis results

As already mentioned, the model proposed in [6] can only be applied to sequences without camera motion and zoom. We generate a synthetic sequence without camera motion from a sequence with camera motion and/or zoom by applying a dedicated algorithm we developed for the purpose, and then use this sequence for the dynamic texture analysis. The techniques used to this end will be described in the sequel.

It is widely acknowledged that homographies can be used to warp planar surfaces from one image to another, but homographies can also be used to warp between images having a common camera center. The latter property is the reason why we can apply homographies to entire images even though the 3-D points represented are not coplanar: dynamic textures are nearly always to be found in outdoor scenes and here the translation of the camera center during a camera pan is negligible with respect to the distance of 3-D points in the scene from the camera center. We assume that for our application scenario homographies can be used for the registration, but we are fully aware that method can fail for other cases, like indoor scenes with significant camera motion and rotation.

It has previously been reported that homographies perform well in compensating camera motion and zoom, see e.g. [35]. However, the quality of the registration strongly depends on the on point correspondences used and the methods used to compute the homography. We identified three key factors that are crucial to obtain a good result that are novel in this context:

1) The homography depending on the point correspondences found in both moving foreground and the (only relevant) background, a robust method for homography computation should be used to allow for finding a good first estimation despite outliers from moving objects.
2) An estimation of the homography being available, points on moving objects can be explicitly excluded from the set by deriving a mask to discard foreground objects.
3) Point correspondences should be spatially well distributed, so that the quality of the registration is equal over the entire frame.

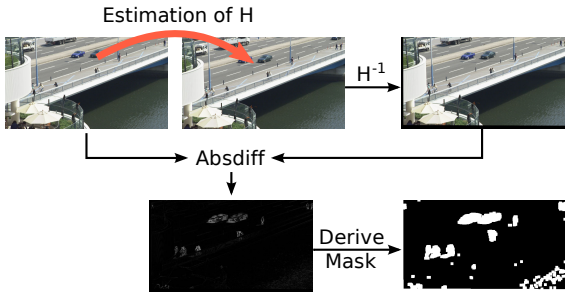
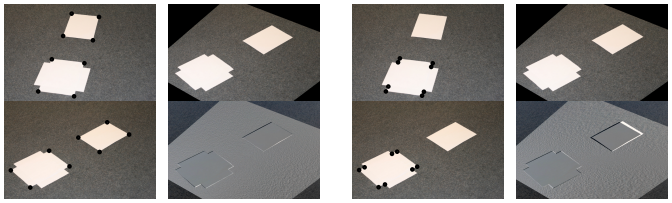


Fig. 8. Schematic representation of the difference image computation and mask generation. Moving objects and dynamic textures are detected.

The first step in the registration of videos consists in finding point correspondences in the frames of the video. To compensate the effects from camera motion and zoom only points from the static background should be used. Natural video sequences contain non-static objects (foreground) as well which renders identification of valid parts necessary. To deal with this problem we used the least median of squares algorithm as described in [36]. In case of strong motion between frames, correspondences lying on non-static objects will be classified as outliers and thereby excluded from the set of points used to compute the homography. Still, points from moderately moving objects tend to remain in the set and deteriorate the result of the registration.

For this reason we introduce another supplemental approach that detects non-static objects in the scene. Figure 8 summarizes the procedure: after an initial basic estimation of the homography done as described previously, the image from the center is warped to the view of the left image. Even though the homography was only a rough estimation, the difference image will clearly highlight the moving objects from the scene. In Figure 8 the difference image highlights the cars driving on the highway. From the latter a mask is generated by applying successively a flood-fill and a dilation algorithm.

As the mask is computed for the reference view, it can be used to discard points when the next frame from the sequence is processed. Once the algorithm has found a good mask, i.e. non-static objects are detected, a better homography from the remaining points can be computed.



(a) (b)

Fig. 9. (a): Left column shows two images of the same planar scene taken from two different viewpoints to simulate camera motion. The upper right image is the upper left image warped into the perspective of the lower left image using a homography computed from the black point correspondences. The bottom right image represents the difference between the original and the warped image. (b: Same as (a) but with other point correspondences used for the computation of the homography.

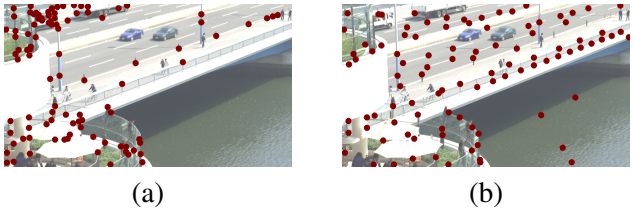
Another problem is that when just selecting point correspondences by their suitability to correspondence matching, they tend to be spatially ill-distributed. More often than not, points form clusters in small regions as can be seen in Figure 10 (a). The impact of this effect in a very simple case can be seen in Figure 9. Here we used the same number point correspondences in (a) and (b) but with different spatial distributions. While in (a) the registration is adequate over the entire image, in (b) large mismatches can be observed in the areas with no point correspondences. To enforce case (a) in general, the basic idea is is to use a large set of points, attribute a surrounding area to each point, and then try to find a subset of points where the distribution of the areas is as close as possible to a uniform distribution.

As a logical consequence, we used the Kullback-Leibler divergence of the given distribution from the uniform distribution as a criterion. We generate subsets $\Omega \backslash i$ of the current feature point set Ω by ignoring a single feature point i, respectively. We then compare the cells of each subset to a set of equally sized cells. The divergence is computed using:

$$s(i) = \sum_{j \epsilon \Omega \backslash \{i\}} \left(\frac{(1-q(j))}{N-1} - \frac{(1-q(j))\, a(j)}{a_\Sigma} \right) \cdot \log \left(\frac{\frac{1}{N-1}}{\frac{a(j)}{a_\Sigma}} \right), \quad (20)$$

where N denotes the cardinality of Ω, $a(j)$ the area corresponding to point j, a_Σ the sum of all considered areas and $q(j)$ the quality of the match ($0 < q(j) \leq 1$). The subset with the lowest score is kept which means that we simply remove point i from the set. The procedure is repeated until the number of remaining points falls below a given threshold. The final result of the correspondence selection process can be seen in Figure 10 (b).

The registration works well for the intended purpose of scenes containing dynamic textures.Even for the challenging sequence from Figure 11 where there is a large water surface changing over time and a boat crossing the scene the algorithm performs well. The implications of registration for texture synthesis can be dramatic. When a registered sequence is used for the training of the model, the synthesis result is visually appealing, whereas, when the original sequence with camera motion and zoom is used, the synthesis is flawed and useless for video compression purposes. This is can be observed in Figure 13. Another example of registration and synthesis is given Figuresee Figures 12. Here only the top part, which contains the water surface, is synthesized.



(a) (b)

Fig. 10. (a): Points only selected by quality. (b): Points selected by quality and distribution.

Fig. 11. Top row: Example frames from Sheriff sequence. Bottom row: Example frames from the sequence at the top after registration.

Fig. 12. Top row: Example frames from original BQSquare sequence. Middle row: Example frames from BQSquare sequence after registration (black border at the bottom has been cut off). Bottom row: Synthesized example frames from the sequence at the top.

Fig. 13. Example frames from Waterfall sequence. Top row: Original sequence. Middle row: Synthesis from compensated sequence using [6]. Bottom row: Synthesis using original sequence with camera motion and zoom.

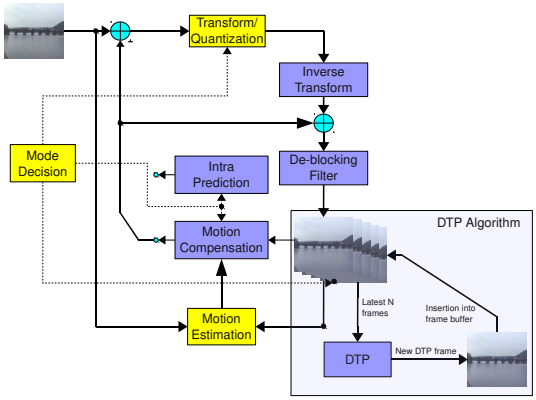


Fig. 14. The principle of the DTP prediction scheme within an H.264/AVC encoder.

D. Dynamic Texture Prediction for H.264/AVC Inter Coding

In [37] we presented a technique based on the dynamic texture model to improve the coding performance of the inter coding scheme in H.264/AVC for sequences with content considered as dynamic textures. The dynamic texture model, introduced in [6], was initially designed for dynamic texture synthesis, i.e. the generation of endless sequences from short sample sequences. We modified the synthesis algorithm so that on one hand it can deal with the impairments the original algorithm suffers when used in a video coding system. On the other hand one single prediction picture is generated instead of a random video sequence.

The principle of the Dynamic Texture Prediction (DTP) algorithm is to use frames stored in the reference picture buffer for learning and then predict the next picture to be encoded. A description of the DTP algorithm itself will be provided in Section III-E. The main modules of an H.264/AVC encoder are shown in Figure 14. Light boxes show the modules generating syntax elements that undergo entropy coding and are written into the bitstream. Clearly, the DTP algorithm is automatic and does not generate any additional information that has to be transmitted, as at the decoder side the identical DTP algorithm can be applied to the reference pictures. A rate-distortion optimization strategy to find an optimal position for the predicted frame in the reference frame buffer is described in [38].

E. Dynamic Texture Prediction Algorithm

A model similar to the one presented in Section III-B is used in the dynamic texture prediction algorithm (DTP), but the purpose is to learn the temporal behavior of a sequence from a very short learning sequence and then *predict* one single future frame. The algorithm below shows how the algorithm is implemented. In the first step the frames in the reference picture buffer are written into a reference buffer matrix $\mathbf{Y}$, using the same procedure as in Section III-B. The key idea now is to treat each column of the matrix as a separate vector and find a low-dimensional equivalent for each vector. Next we can apply a singular value decomposition on Y resulting in:

$$\mathbf{Y} = \mathbf{U} \cdot \mathbf{S} \cdot \mathbf{V}^{\mathrm{T}}. \tag{21}$$

By considering $\mathbf{CU}$ as an observation matrix, and $\mathbf{X} = \mathbf{S} \cdot \mathbf{V}^{\mathrm{T}}$ as a state vector matrix, we see that we obtained a matrix $\mathbf{X}$ of dimension $\tau \times \tau$, so that each frame is represented by a low-dimensional vector of dimension τ, i.e. the corresponding column from $\mathbf{X}$. Note here that the transform from $\mathbf{Y}$ into $\mathbf{X}$, i.e. the dimension reduction step, is not given explicitly, but implicitly using a SVD. The inverse transform however, is given explicitly by the simple multiplication with the matrix $\mathbf{C}$. This is important since we want to predict a future vector $\mathbf{X}_{pred}$ and then find its equivalent in the high dimensional space, i.e. the predicted frame. The next step is to treat the vectors from $\mathbf{X}$ as state vectors from a linear dynamical system and derive the state transition matrix $\mathbf{A}$, which is in that case:

$$\mathbf{A} = arg\min_{\mathbf{A}} \|\mathbf{X}_{n-\tau+1}^{n} - \mathbf{A} \cdot \mathbf{X}_{n-\tau}^{n-1}\|, \tag{22}$$

where $\mathbf{X}_{n-\tau+1}^{n}$ is the matrix containing the state vectors corresponding to frame number n to $n-\tau+1$ and by analogy $\mathbf{X}_{n-\tau}^{n-1}$ the matrix containing the vectors corresponding to $n-1$ to $n-\tau$. Once $\mathbf{A}$ is known, we can

predict the next vector by multiplying the last vector with $\mathbf{A}$, and find the corresponding predicted frame $\mathbf{Y}_{pred}$. In fact:

$$\mathbf{X}_{pred} = \mathbf{A}\mathbf{X}_{\tau}^{\tau}, \tag{23}$$

and finally

$$\mathbf{Y}_{pred} = \mathbf{C}\mathbf{X}_{pred}. \tag{24}$$

For more information on the DTP algorithm, we refer to [38] and [37].

F. Dynamic texture synthesis algorithm

In addition to the DTP algorithm, the dynamic texture algorithm (DTS) from [6] can be used directly for prediction as well. The main differences between DTP and Doretto's algorithm are stated below:

1) The temporal mean of the sequence is computed and the dynamical model is derived using the difference only.
2) A large number of frames is used for the analysis (e.g. 30) compared to conventional DTP.
3) The order of the model is lower than the rank of the sequence matrix, i.e. the smallest singular values are omitted.

The precise description of how to perform prediction with DTS is given below. It is important to note that $t >> \tau$, i.e. the number of learning frames for DTS is higher. Furthermore $\mathbf{left_k}(\mathbf{A})$ is the operation of taking the left k columns of the matrix $\mathbf{A}$ and similarly $\mathbf{upper_k}(\mathbf{A})$ is the operation of taking the upper k rows of the matrix $\mathbf{A}$.

Input: The decoded picture buffer matrix $\mathbf{Y}_{n-t+1}^{n}$.
Output: The value of $\mathbf{Y}_{n+1}$.

if $n >= t$ **then**
  $\mathbf{Y}_{mean} \leftarrow \mathbf{mean}(\mathbf{Y}_{n-t+1}^{n})$
  $\mathbf{Y}_{diff} \leftarrow (\mathbf{Y}_{n-t+1}^{n}) - \mathbf{Y}_{mean}$
  $\mathbf{U}, \mathbf{S}, \mathbf{V}^{T} \leftarrow SVD(\mathbf{Y}_{diff})$
  $\mathbf{C} \leftarrow \mathbf{left_k}(\mathbf{U})$
  $\mathbf{X}_{n-t+1}^{n} \leftarrow \mathbf{upper_k}(\mathbf{S} \cdot \mathbf{V}^{T})$
  $\mathbf{A} \leftarrow \mathbf{X}_{n-t+1}^{n-1} \cdot \mathbf{pinv}(\mathbf{X}_{n-t+2}^{n})$
  $\mathbf{X}_{n+1} \leftarrow \mathbf{A} \cdot \mathbf{X}_{n}$
  $\mathbf{Y}_{n+1} \leftarrow \mathbf{C} \cdot \mathbf{X}_{n+1} + \mathbf{Y}_{mean}$
end if
return $\mathbf{Y}_{n+1}$

To use DTS along with DTP we introduced a new mode called DT skip. In contrast to the conventional skip mode which signals to the decoder to automatically derive the motion vector and reference frame, and simply copy the content from there, with DT skip, the decoder doesn't need to derive a motion vector but simply copies the content from the same location in the DTP/DTS predicted picture. Details on the DT skip mode can be found in [39].

PARAMETER	VALUE
GOP STRUCTURE	IPPP
QP I	22, 27, 32, 37
QP P	23, 28, 33, 38
SEARCH RANGE	32 PIXELS
FREXT PROFILE	HIGH
RDO	ON
ENTROPY CODING MODE	CABAC
FRAME RATE	30 FRAMES/S
NUMBER REFERENCE FRAMES	5
FULL-PEL DISTORTION METRIC	SAD
HALF-PEL DISTORTION METRIC	HADAMARD SAD
QUARTER-PEL DISTORTION METRIC	HADAMARD SAD

TABLE I
JM16.1 ENCODER SETUP.

G. EXPERIMENTAL RESULTS

For the experimental investigations, we used the system described in Section III-D with various test sequences. The presented algorithm is integrated into the JM16.1 reference software [40]. For the experimental evaluation, testing conditions based upon [41] are used. Entire sequences were encoded and the detailed setup is summarized in Table I. Moreover, the comparison is done using the same number of reference frames, i.e. H.264/AVC with k reference frames is compared to the modified algorithm using $k-1$ reference frames plus an extrapolated frame using the previously described techniques. Two different scenarios were investigated: one where we use only five past frames for DTP to generate an additional reference frame as described in Section III-E respectively in [37] and [38] (which is referred to as "predicted reference frame only") and a combined method where in addition we use 30 past frames for DTS and the DT skip mode as described in Section III-F respecitively in [39] (which is referred to as "combined"). While Tables II and III show results only for the former case, Figures 15 and 16 and Table IV contain results for both scenarios.

The presented techniques aim at improving the compression performance for sequences containing dynamic textures. While remaining unchanged for sequences with other content, the performance improves with dynamic textures like e.g. water surfaces. This can be seen in the RD curves from Figure 15 and 16 for the sequences Container and Bridge-Far. The texture on the water surface in the latter sequence is very fine-grained, and only becomes visible at lower QP values. This is the reason why bitrate savings for Bridge-Far appear only for higher bitrates and explains the unusual shape of

Sequence	Source	ΔPSNR[dB]	ΔRate[%]
BRIDGE-CLOSE	ORIG.	0.062	-2.264
CLAIRE	ORIG.	0.105	-1.895
CONTAINER	ORIG.	0.377	-7.987
FOREMAN	ORIG.	0.064	-1.163
MISS-AMERICA	ORIG.	0.140	-3.236
MOTHER	ORIG.	0.073	-1.535
PREAKNESS	ORIG.	0.091	-2.188
SEAN	ORIG.	0.099	-1.800
SUZIE	ORIG.	0.045	-1.018
RUSHHOUR	CROP.	0.139	-3.557
SHUTTLESTART	CROP.	0.142	-3.344
Average		0.136	-3.090

TABLE II
BD-PSNR RESULTS FOR JM16.1 COMPARED TO JM16.1 WITH ADAPTIVE DTP FOR SELECTED QCIF SEQUENCES.

Sequence	Source	ΔPSNR [dB]	ΔRate[%]
CONTAINER	ORIG.	0.161	-4.334
SEA	ORIG.	0.093	-1.957
RUSHHOUR 1	CROP.	0.074	-3.174
RUSHHOUR 2	CROP.	0.061	-2.130
SHUTTLESTART	CROP.	0.137	-3.618
Average		0.105	-3.043

TABLE III
BD-PSNR RESULTS FOR JM16.1 AND COMPARED TO JM16.1 WITH ADAPTIVE DTP FOR SELECTED CIF SEQUENCES.

the curve (which in turn makes BD-PSNR computation impossible).

For typical dynamic texture sequences, e.g. Shuttlestart (steam appearing at start), or Preakness (chaotically moving crowd shown from far) and Sea (water surface) we provide the BD-PSNR values in Tables II and III. Moreover, sequences showing mainly faces of talking people, turned out to show some noticeable improvements. Talking faces, due to the complex motion, are only regarded as dynamic texture in a wide sense, but still some bitrate savings can be observed with Sean, Foreman, Claire, and Miss-America.

Sequences where the "combined" method performs better than the "predicted reference frame only" method are listed in Table IV, but the difference is only considerable for the sequence Bridge-Far (see Figure 16), where the bitrate saving at $Qp = 23$ amounts to 30% compared to conventional JM. Yet a disadvantage of the "combined" method is the increased computational complexity. In fact, the presented algorithms have to be performed at both encoder and decoder, and while DTP can can be performed in real-time with a frame rate of 30 frames/sec, the DTS method requires much more computation time (approx. factor 6). As a result real-time decoding cannot be guaranteed in our implementation for the "combined" method, which constitutes a drawback.

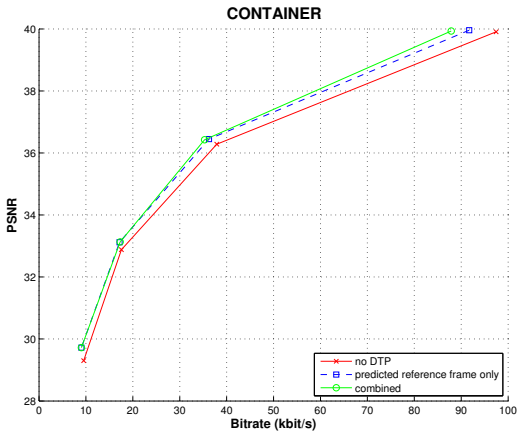


Fig. 15. RD curve for Container qcif

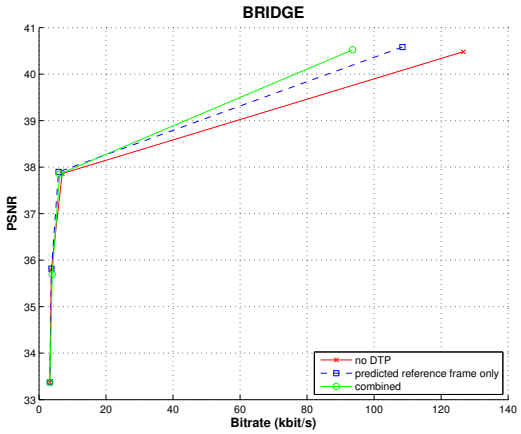


Fig. 16. For the Bridge-far sequence BD-PSNR [42] computation was not possible, but the rate savings for $QP = 23$ amounts to 30% for the combined algorithm.

Although the algorithms presented here are inspired from the texture synthesis algorithm from [6], we want to dispel the misconception that our algorithm yields a lower PSNR at equal subjective visual quality, as can be clearly seen in the RD curves in and the BD-PSNR Tables. In particular, we did not change the RD decision in the first place, so the techniques under consideration turn out to be simple additional predictors that do not change any of the constituent parts of the encoding process.

IV. CONCLUSIONS AND FUTURE WORK

Though it is not claimed that the solutions for static and dynamic texture synthesis introduced in this paper are generally competitive yet with conventional video coding algorithms for all types of content, we believe that our work shows some promising advances in that:

- We derive a complete framework for encoding microtextures using the GMRF model, including a method to reliably identify texture and a novel

Sequence	Source	Predicted Reference Frame		Combined	
		Δ PSNR [dB]	Δ Rate [%]	ΔPSNR [dB]	ΔRate[%]
BRIDGE-CLOSE	ORIG.	0.058	-2.192	0.097	-3.647
CONTAINER	ORIG.	0.385	-7.922	0.438	-8.866
PREAKNESS	ORIG.	0.094	-2.232	0.131	-3.044
RUSHHOUR	CROP.	0.084	-2.590	0.109	-3.003
SEAN	ORIG.	0.095	-1.729	0.102	-1.849
AVERAGE		0.1432	-3.333	0.1754	-4.082

TABLE IV
BD-PSNR [42] RESULTS FOR JM16.1 COMPARED TO JM16.1 WITH ADAPTIVE DTP AND COMPARED TO THE NEW COMBINED ALGORITHM FOR SELECTED SEQUENCES.

extension of the Itakura distance which is designed for this class of texture. We provide preliminary results indicating that the method can provide an improvement in visual quality at equal bit-rate and should be further optimized.

- For dynamic textures, we use a model previously introduced in computer graphics which allows plausible synthesis. We have shown that after registration visually appealing textures can be synthesized. Beyond that, in terms of classical video compression, we found that it is possible to use synthesized frames for prediction and thereby achieve bitrate savings using the conventional rate-distortion decision.

We are confident that static and dynamic texture synthesis have high potential to complement state of the art image and video coding algorithms, achieving improved quality in areas that are otherwise expensive in data rate. The main challenge in achieving this is the development of quantitative measures for subjective synthesis quality, which must then be combined with rate–distortion optimization for proper encoder decisions.

ACKNOWLEDGMENT

The present projects are supported by the National Research Fund, Luxembourg under grant 795405 and DFG under grant OH 50/13, respectively.

REFERENCES

[1] *ITU-T Rec. H.264 & ISO/IEC 14496-10 AVC: Advanced Video Coding for Generic Audiovisual Services*.

[2] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *Proc. of IEEE International Conference on Computer Vision ICCV*, Sep. 1999, pp. 1033–1038.

[3] L.-Y. Wei and M. Levoy, "Fast texture synthesis using tree-structured vector quantization," in *Proc. of International Conference on Computer Graphics and Interactive Techniques SIGGRAPH*, Jul. 2000, pp. 479–488.

[4] V. Kwatra, I. Essa, A. Bobick, and N. Kwatra, "Texture optimization for example-based synthesis," in *Proc. of International Conference on Computer Graphics and Interactive Techniques SIGGRAPH*, 2005, pp. 795–802.

[5] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: Image and video synthesis using graph cuts," in *Proc. of International Conference on Computer Graphics and Interactive Techniques SIGGRAPH*, Jul. 2003, pp. 277–286.

[6] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, "Dynamic Textures," *International Journal of Computer Vision*, vol. 51, no. 2, pp. 91–109, 2003.

[7] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Kluwer, 1992.

[8] A. Dumitras and B. Haskell, "A texture replacement method at the encoder for bit-rate reduction of compressed video," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 2, pp. 163 – 175, Feb. 2003.

[9] C. Zhu, X. Sun, F. Wu, and H. Li, "Video coding with spatio-temporal texture synthesis," in *Multimedia and Expo, 2007 IEEE International Conference on*, Jul. 2007, pp. 112 –115.

[10] P. Ndjiki-Nya, T. Hinz, and T. Wiegand, "Generic and robust video coding with texture analysis and synthesis," in *Proc. of IEEE International Conference on Multimedia and Expo ICME*, Jul. 2007, pp. 1447–1450.

[11] J. Byrne, S. Ierodiaconou, D. Bull, D. Redmill, and P. Hill, "Unsupervised image compression-by-synthesis within a JPEG framework," in *Proc. of IEEE International Conference on Image Processing ICIP '08*, 2008.

[12] P. Ndjiki-Nya, D. R. Bull, and T. Wiegand, "Perception-oriented video coding based on texture analysis and synthesis," in *Proc. of IEEE International Conference on Image Processing ICIP '09*, 2009, pp. 2273–2276.

[13] T. K. Tan, C. S. Boon, and Y. Suzuki, "Intra prediction by template matching," in *Proc. of IEEE International Conference on Image Processing ICIP*, Oct. 2006, pp. 1693–1696.

[14] J. Ballé and M. Wien, "Extended texture prediction for H.264/AVC intra coding," in *Proc. of IEEE International Conference on Image Processing ICIP '07*, vol. VI. San Antonio, TX, USA: IEEE, Piscataway, Sep. 2007, pp. 93–96.

[15] B. Julesz, "Early vision and focal attention," *Reviews of Modern Physics*, vol. 63, no. 3, pp. 735–775, Jul. 1991.

[16] H. Derin and P. A. Kelly, "Discrete-index Markov-type random processes," *Proceedings of the IEEE*, vol. 77, no. 10, pp. 1485–1510, Oct. 1989.

[17] H. Rue and L. Held, *Gaussian Markov Random Fields*, ser. Monographs on Statistics and Applied Probability. Chapman & Hall/CRC, 2005, no. 104.

[18] Z. Wang, A. C. Bovik, H. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[19] M. J. Hinich, "Detecting a transient signal by bispectral analysis," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 7, pp. 1277–1283, Jul. 1990.

[20] C. L. Nikias and A. P. Petropulu, *Higher-Order Spectra Analysis*. Prentice-Hall, Englewood Cliffs, 1993.

[21] E. P. Simoncelli, W. T. Freeman, E. H. Adelson, and D. J. Heeger, "Shiftable multiscale transforms," *IEEE Transactions on Information Theory*, vol. 38, no. 2, pp. 587–607, Mar. 1992.

[22] M. J. Wainwright and E. P. Simoncelli, "Scale mixtures of gaussians and the statistics of natural images," *Advances in Neural Information Processing Systems*, no. 12, pp. 855–861, 2000.

[23] J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli, "Adaptive Wiener denoising using a Gaussian scale mixture model in the Wavelet domain," in *Proc. of IEEE International Conference on Image Processing ICIP '01*, Oct. 2001.

[24] B. Galerne, Y. Gousseau, and J. M. Morel, "Random phase textures: Theory and synthesis," *IEEE Transactions on Image Processing*, vol. 20, no. 1, pp. 257–267, Jan. 2011.

[25] A. K. Jain, *Fundamentals of Digital Image Processing*. Prentice-Hall, Englewood Cliffs, 1988.

[26] S. L. Marple, Jr., *Digital Spectral Analysis*. Prentice-Hall, Englewood Cliffs, 1987.

[27] D. E. Dudgeon and R. M. Mersereau, *Multidimensional Digital Signal Processing*. Prentice-Hall, Englewood Cliffs, 1984.

[28] F. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 23, no. 1, pp. 67–72, Feb. 1975.

[29] J. Ballé and M. Wien, "A quantization scheme for modeling and coding of noisy texture in natural images," in *Proc. of IASTED Conference on Signal and Image Processing SIP '09*. Honolulu, HI, USA: ACTA Press, Calgary, Aug. 2009.

[30] J. Ballé, B. Jurczyk, and A. Stojanovic, "Component-based image coding using non-local means filtering and an autoregressive texture model," in *Proc. of IEEE International Conference on Image Processing ICIP '09*. Cairo, Egypt: IEEE, Piscataway, Nov. 2009, pp. 1937–1940.

[31] C. Feldmann and J. Ballé, "Improved entropy coding for component-based image coding," in *Proc. of IEEE International Conference on Image Processing ICIP '11*, Bruxelles, Belgium, Sep. 2011, accepted for publication.

[32] "CIPR still images: Kodak," Downloaded from http://www.cipr.rpi.edu/resource/stills/kodak.html, Jun. 2009.

[33] *ITU-T Rec. T.800 & ISO/IEC 15444-1: JPEG 2000 Image Coding System: Core Coding System*.

[34] R. Costantini, L. Sbaiz, and S. Süsstrunk, "Higher Order SVD Analysis for Dynamic Texture Synthesis," *IEEE Transactions on Image Processing*, vol. 17, no. 1, pp. 42–52, 2008.

[35] M. Bosch, F. Zhu, and E. Delp, "Video coding using motion classification," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, oct. 2008, pp. 1588 –1591.

[36] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. New York, NY, USA: Cambridge University Press, 2003.

[37] A. Stojanovic, M. Wien, and J.-R. Ohm, "Dynamic Texture Synthesis for H.264/AVC Inter Coding," in *Proceedings of the IEEE International Conference on Image Processing*, October 2008, pp. 1608–1612.

[38] A. Stojanovic, M. Wien, and T. Tan, "Synthesis-in-the-Loop for Video Texture Coding," in *Proceedings of the IEEE International Conference on Image Processing*, November 2009.

[39] A. Stojanovic and P. Kosse, "Extended dynamic texture prediction for H.264/AVC inter coding," in *Proceedings of the IEEE International Conference on Image Processing*. Hong Kong, People's Republic of China: IEEE, Piscataway, 2010.

[40] JM16.1: http://iphome.hhi.de/suehring/tml.

[41] T. Tan, G. Sullivan, and T. Wedi, "Recommended simulation common conditions for coding efficiency experiments," in *ITU-T / Study Group 16 / Question 6 - VCEG*. Document VCEG-AH10r3, January 2008.

[42] G. Bjøntegaard, "Calculation of average PSNR differences between RD curves," in *ITU-T / Study Group 16 / Question 6 - VCEG*. Document VCEG-M33, April 2001.

Johannes Ballé received the Dipl.-Ing. degree in Computer Engineering from RWTH Aachen University, Aachen, Germany, in 2007. Currently, he is pursuing the Ph.D. degree at Institut für Nachrichtentechnik (IENT), RWTH Aachen University, focusing on image processing, signal theory, and data compression.

Aleksandar Stojanovic received his diploma degree in Electrical Engineering from RWTH Aachen Universtity in 2007 where he is currently a PhD student at Institut für Nachrichtentechnik (IENT). His research interests include video coding with emphasis on texture compression and motion analysis.

Jens-Rainer Ohm received the Dipl.-Ing. degree in 1985, the Dr.-Ing. degree in 1990, and the habil. degree in 1997, all from Technical University of Berlin (TUB), Germany. From 1985 to 1995, he was a research and teaching assistant with the Institute of Telecommunications at TUB. Between 1992 and 2000, he has also served as lecturer on topics of digital image processing, coding and transmission at TUB. From 1996 to 2000, he was project manager/coordinator at the Image Processing Department of Heinrich Hertz Institute (HHI) in Berlin. In 2000, he was appointed full professor and since then holds the chair position of Institut für Nachrichtentechnik (IENT) at RWTH Aachen University, Germany. His research and teaching activities cover the areas of motion-compensated, stereoscopic and 3-D image processing, multimedia signal coding and content description, transmission of video signals over mobile networks, as well as general topics of signal processing and digital communication systems. Since 1998, he participates in the work of the Moving Pictures Experts Group (MPEG), where he has been contributing to the development of MPEG-4 (Video and AVC) and MPEG-7 standards. He is chair of the ISO/IEC WG 11 (MPEG) Video Subgroup since May 2002. From January 2005 until November 2009, he was also co-chairing the Joint Video Team (JVT) of MPEG and ITU-T SG 16 VCEG. Currently, he is co-chairing the Joint Collaborative Team on Video Coding (JCT-VC) of ISO and ITU-T, with intended mandate of developing the next generation of high-efficiency video coding technology. Prof. Ohm has authored textbooks on multimedia signal processing, analysis and coding, on communications engineering and signal transmission, as well as numerous papers in the various fields mentioned above. He is member of various professional organizations including IEEE, VDE/ITG, EURASIP and AES.