# Development of an Entropy Code for Component Based Image Compression

*Christian FELDMANN*[1], *Johannes BALLÉ*[1]

[1]Institut für Nachrichtentechnik
RWTH Aachen University
Aachen, Germany

{feldmann,balle}@ient.rwth-aachen.de

**Abstract.** *A major problem in image compression is the coding of noise and noisy texture in an image. In a classic image compression approach, a transparent representation of these components is only possible with a very high bit rate. However, a pixel exact representation is not necessary in order to achieve visually perceived transparency because the human viewer is not able to distinguish two noise signals with the same statistical properties.*

*Ballé [1] proposes a system that allows for a reconstruction of noise without pixel wise exactness. At first, the image is decomposed into a structural part and another part that contains noise and noise like texture. The structure part is encoded using conventional techniques (e.g. JPEG 2000) where the noise component is modeled by an extended autoregressive (ARX) model. At the receiver side, we then use the ARX model coefficients to reconstruct a noise component with similar characteristics.*

*In this paper, we propose and evaluate different coding schemes in order to design a fast and efficient entropy code for these ARX model coefficients. We also show a first comparison to a conventional image compression technique.*

## Keywords

video coding, noise, texture, statistical model, autoregressive process, arithmetic coding, CABAC

## 1. Introduction

A classical approach for image compression is to use different decorrelation and transformation techniques in order to create the best possible reconstruction with respect to the PSNR. However, the PSNR is not a very good measure for the actual perceived image quality. Especially in noise signals, decorrelation techniques are inefficient and require a high bit rate to yield a good reconstruction with respect to the PSNR. Moreover, studies of the human perception have shown that it is sufficient to reconstruct noise with a similar statistic in order to be indistinguishable from the original noise signal. The same applies to noise like textures in an image (e.g. carpet or wallpaper).

Ballé proposes a system that handles the noise component of an image independently from its structure. In a first step, we split the image into a noise and a structure component. The structure component contains edges and homogenous areas and is encoded using JPEG2000 as a classical compression approach. In a second step, we use the noise component to estimate the model parameters of an extended autoregressive model. The ARX parameters are then quantized and entropy coded.

Since this paper is based on the component based concepts proposed by Ballé in [1], a short overview is given before the entropy coding of the model parameters is discussed and evaluated. Finally, a brief comparison to the established compression standard JPEG2000 is given.

## 2. Image Decomposition

The decomposition of the image into a structure and a noise component still poses a challenging problem with room for improvement. On the one hand, we would like the structure component to be as noise free as possible in order to make the classical image compression highly effective. On the other hand, we want the noise component to contain only semantically irrelevant noise and noisy texture but no structural information. If there is structural information like edges in the noise component, it will be reconstructed by the statistical model, which leads to a high loss in image quality.

Especially the identification of areas that are semantically irrelevant to the human viewer is largely based on prior knowledge and thus very hard to implement. [2] However, it can be shown that the Non-Local Means algorithm provides reasonably good results for the image decomposition.

The decomposition algorithm is based on the assumption that the image ($u$) is composed of a noise free image and additive noise. It then tries to estimate the noise free pixels ($\hat{u}(x)$) from an average value of the noisy pixels $u$ in a window $\mathcal{W}$ around the pixel position $x$.

$$\hat{u}(x) = \frac{1}{C(x)} \sum_{y_i \in \mathcal{W}} w(x, y_i) \cdot u(y_i) \tag{1}$$

Each value in the average is weighted by a factor $w$ that depends on the similarity of the region around $x$ compared to the region around $y_i$. If the similarity is high $w$ is high because we assume that the noisy pixel $y_i$ carries a lot of information about the pixel $\hat{u}$ that we are trying to estimate. [3]

## 3. The Texture Model

In a first approach, we can approximate the noisy texture by a Gaussian white noise process. In this case the process can be described by its standard deviation $\sigma$. However, noise like texture carries a lot of pixel dependencies and thus can better be approximated by an autoregressive model:

$$n(x) = \sigma(x)\omega - a(x) * n(x) \tag{2}$$

In this model the Gaussian white noise process $\omega(x)$ is amplified by $\sigma$ and filtered by an all-pole filter $a(x)$.

Since we will quantize the filter coefficients we need a way of measuring the quality of the noise signal that is generated by the autoregressive model. For this, classical measures like the mean squared error or the peak signal to noise ratio are not applicable since we do not intend to generate a pixel exact reconstruction of the noise component.

In audio compression, there exists a very similar problem for linear predictive coding (LPC). Based on a maximum likelihood approach, Itakura introduced a way of measuring the probability that a given signal has been created by a system with certain LPC coefficients [8]. If we transfer this measure to our autoregressive model we get:

$$D(\boldsymbol{a}) = \left(\frac{\hat{\sigma}(\hat{\boldsymbol{a}})}{\hat{\sigma}(\boldsymbol{a})}\right) \tag{3}$$

where $\hat{\sigma}$ represents the maximum likelihood estimate of $\sigma$ for the distorted model coefficients $\hat{\boldsymbol{a}}$ and for the optimal model parameters $\boldsymbol{a}$. [9] Although we use this distance function to measure the distortion that is introduced by quantization of the AR model parameters, a connection between the measure and the visual quality still has to be empirically proven.

Another observation concerning the autoregressive model is that after the decomposition step parts of noisy texture can still be found in the structure component $s(x)$ creating dependencies of the model to the structure component. To compensate for these dependencies we add an additional term to the autoregressive model in equation (2):

$$n(x) = \sigma(x)\omega - a(x) * n(x) - c(x) * s(x) \tag{4}$$

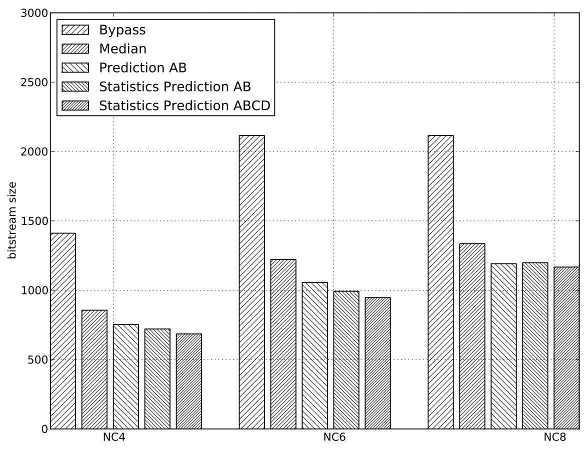

**Fig. 1.** Cluster map coding results averaged over the test set for three different numbers of centroid vectors (4, 6, and 8).

where $c(x)$ represents the filter that models the linear dependency between the two components. Since noisy texture in natural images is usually attached to objects in a scene, we allow the parameters $\sigma$, $a$ and $c$ to depend on the position in the image ($x$).

If we apply this model to our image, we get a standard deviation $\sigma$ and the dependency vectors $a$ and $c$ for every pixel in the image. Since this will not lead to a data reduction, we need to do a quantization of the parameters. In a first step, we split the image into blocks and assume that the ARX model parameters are constant within these blocks reducing the parameters to $\sigma$, $a$, and $c$ per block. In a second step, we assume that noisy texture in an image is associated with objects and stretches over a range of blocks in an image. Because of this, we merge blocks with similar $a$ and $c$ vectors into clusters and only save one set of $a$ and $c$ vectors per cluster. [7]

## 4. Coding Schemes

Overall, the quantization results in three components that need to be encoded for the ARX model. In this section, we present different approaches for an efficient encoding of these model parameters. For all methods we use an independent implementation of the Context-Adaptive Binary Arithmetic Coding (CABAC) engine. [6]

### 4.1. Cluster map

The cluster map maps every block in an image to a cluster. The cluster then contains the ARX model parameters for this block. Since texture in natural images usually attaches to objects that span over multiple adjacent blocks, the cluster map tends to form areas of blocks that are assigned to the same cluster. In addition, we force the cluster map to form homogenous areas by applying a Gaussian kernel to the mapping in every iteration of the clustering algorithm. Especially for small block sizes, this is necessary for the algorithm to converge. Choosing the wrong cluster for a block would result in a noise reconstruction with a very different characteristic than the original noise signal. Due to this fact, all presented coding methods allow a perfect reconstruction of the cluster map.

We evaluated several predictive methods that try to generate bits with a high probability of being 1 so they can be efficiently coded in a CABAC context. The mapping is scanned in a raster scan order, and the current block X is predicted using the neighboring blocks A (left), B (top), C (top right), and D (top left). If a prediction fails or no surrounding blocks are avaliable the index, X is encoded in bypass mode using a fixed length binarization.

*Bypass Mode:* All indices are encoded using a fixed length binarization and the CABAC bypass engine.

*Median Prediction:* The median value of A, B, and C is used for the prediction. If C is not available, D replaces it. The idea is that the median will act as a majority decision favoring the value that occurs more than once. One bit then encodes if the prediction is correct.

*Prediction AB:* This method, that was already proposed in [1], uses the blocks A and B for a prediction of the mapping of X. The first bit encodes if either A or B yields the correct prediction. The context for this bit is chosen according to the condition if A and B share the same index or not. If the first bit is true but A and B do not have the same index, a second bit, encoded in bypass mode, signals if A or B yields the correct value for X.

*Statistic Prediction AB:* In order to further improve the prediction efficiency we measure the relative frequencies of the neighboring blocks A, B, C, and D yielding the correct prediction depending on the neighborhood constellation (e.g. A, B, and C share the same index, D has a different index). The prediction works exactly like the previous method (prediction AB) but the bits are encoded in contexts depending on the measured relative frequencies. We use 10 contexts to bundle bits with approximately the same probability.

*Statistic Prediction ABCD:* The relative frequencies can also be used to extend the prediction to the blocks C and D. For this, the statistic cannot only provide a most probable prediction but also a second most probable prediction for X. In the first bit, we encode if either of these is correct. A second bit then signals if it actually is the most probable or the second most probable prediction that yields the correct prediction. We then encode both bits in contexts that are chosen according to their relative frequencies.

## 4.2. ARX coefficients

Analysis of the value distribution in the $a$ and $c$ vectors show that there is a strong concentration at zero that can be exploited for entropy coding of the values (Fig. 2). Since positive and negative values are equiprobable, the sign for all non-zero values is encoded using the bypass engine.

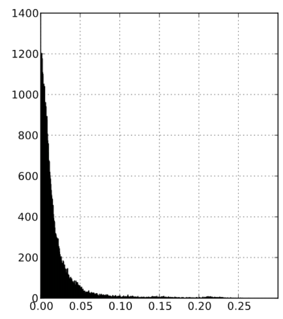

**Fig. 2.** Histogram over the absolute value of ARX model coefficients.

*Exp-Golomb k=0:* The values are encoded using an Exp-Golomb code for k=0 and the CABAC bypass engine. This is the scheme utilized in [1].

*Exp-Golomb, adaptive k:* Since k=0 is usually not a very good approximation of the distribution of the model parameters we determine a better value for k by testing values from 0 to 25 and choosing the value that yields the smallest bit stream size. The optimal k value is signaled to the decoder at the beginning of the bit stream with an Exp-Golomb (k=0) code.

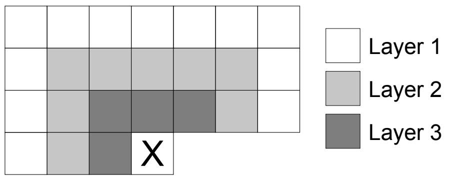

**Fig. 3.** Layering of the neighborhood around X depending on the distance to the center.

*Layer based:* The ARX coefficient vector represents a neighborhood of dependencies. Because of this, large values in the vector tend to concentrate at the center so we can also exploit the position in the neighborhood mask. The values in the neighborhood are divided into layers (see Fig. 3). Each layer is then encoded like before with an Exp-Golomb code with adaptive k. The necessary signalling for the k values of each layer is minimized by only encoding the difference to the last layers k value with an Exp-Golomb (k=0) code.

*Bit-wise arithmetic coding:* Here we adopt a method proposed in [4] that uses a context based binary arithmetic code for values that have a concentration at zero. The quantized value is represented by its binary representation. Every bit level is now encoded in a different CABAC context because we assume that the most significant bit has a high probability of being zero which decreases with decreasing significance. The CABAC engine then exploits these sorted probabilities by using the CABAC binarization.

## 4.3. ARX model standard deviation

The last component we need to encode is the ARX model standard deviation $\sigma$ for each block. In our experi-

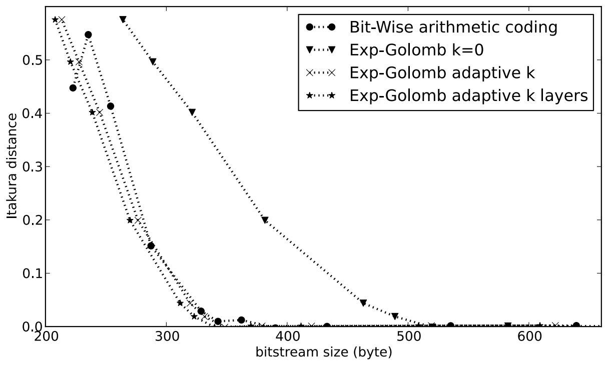

**Fig. 4.** Bit stream size vs. Itakura distance for image "'Lena"' and a varying scalar quantization of A. The depicted quantization step size ranges from 0.01 to 0.00001.

ments, the $\sigma$ values presented certain properties of natural images. This can be explained by the fact that $\sigma^2$ is directly proportional to the noise energy in the corresponding block. In the reconstruction, $\sigma$ corresponds to the local contrast of texture multiplied by the filter gain of $a(x)$. An example for the image *Lena* can be seen in Fig. 5. An obvious way to measure the reconstruction quality is therefore to calculate the mean squared error for the reconstructed $\sigma$ values. Due to this similarity to natural images, we also evaluated the use of transform-based codes for the $\sigma$ values.

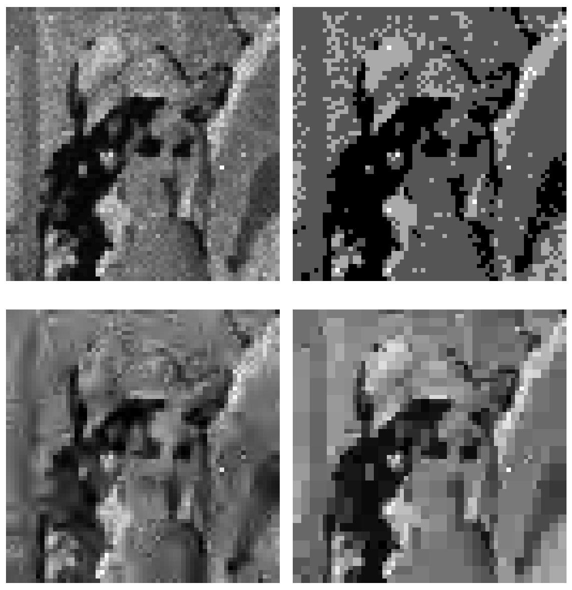

**Fig. 5.** Example of a standard deviation map for "'Lena"'. Top left: estimated $\sigma$ values. Top right: lossy scalar quantization. Bottom left: lossy Daubechies wavelet. Bottom right: lossy Haar wavelet. All lossy methods are compared at the same bit rate (1 bit per $\sigma$ value).

*CABAC binarization:* Each value is encoded using a unary code up to a quantization step index $g$. If the value is higher than the limit the difference between the value and the limit is encoded with an Exp-Golomb code with k=0. The bits of the unary code are encoded in seperate CABAC contexts in order to let CABAC approximate the distribution.

*Context from cluster mapping:* Blocks that are associated to the same cluster share a similar texture and thus tend to have a similar distribution of $\sigma$ values. The coding works like in the method above but with a seperate range of CABAC contexts for every cluster. This method is used in [1].

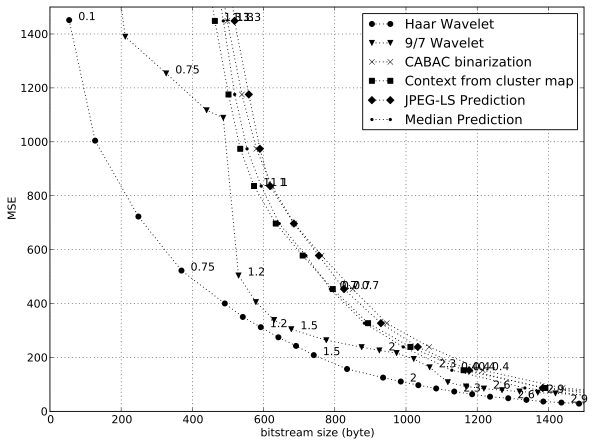

**Fig. 6.** MSE vs. bitrate size of $\sigma$ for Kodim02.

*JPEG-LS Prediction:* JPEG also defines a lossless encoding scheme that uses a prediction based on an edge heuristic employing the blocks A, B, and D to predict the value of X [5]. The resulting error signal is then encoded using the already presented CABAC unary/Exp-Golomb code.
*Median prediction:* Another practical prediction can be made by calculating the median value of A, B, and C (if not available D). The resulting prediction error is then again encoded using the CABAC unary/Exp-Golomb code.
*Wavelet methods:* Due to the fact that the map shows certain characteristics of natural images, we also considered the use of a transform coder. As a readily available method, we implemented the SPITH algorithm in combination with the Haar and the Daubechies 9/7 wavelet.

## 5. Results

The coding methods were evaluated on a test set comprising the Kodak set and the well-known test images Baboon, Barbara, Boat, Clown, Elaine, Lena, Peppers and Plane. Unless otherwise noted we used an 11x6 support for $a(x)$, a 1x1 support for $c(x)$, block size 8x8 pixels, and a quantization step size of 0.00001 for the ARX model parameters. These settings were also used in [1] so that results can be compared.
*Cluster mapping:* All predictive methods provide comparable results (Fig. 1). It can be seen that the median prediction performs slightly worse than the method proposed in [1] (Prediction AB). Another observation is, that the use of a statistic results in a slight improvement of the efficiency at the additional cost of creating the statistic from a representative image set and saving it. In addition, this statistic has to be recreated every time the decomposition filter or the

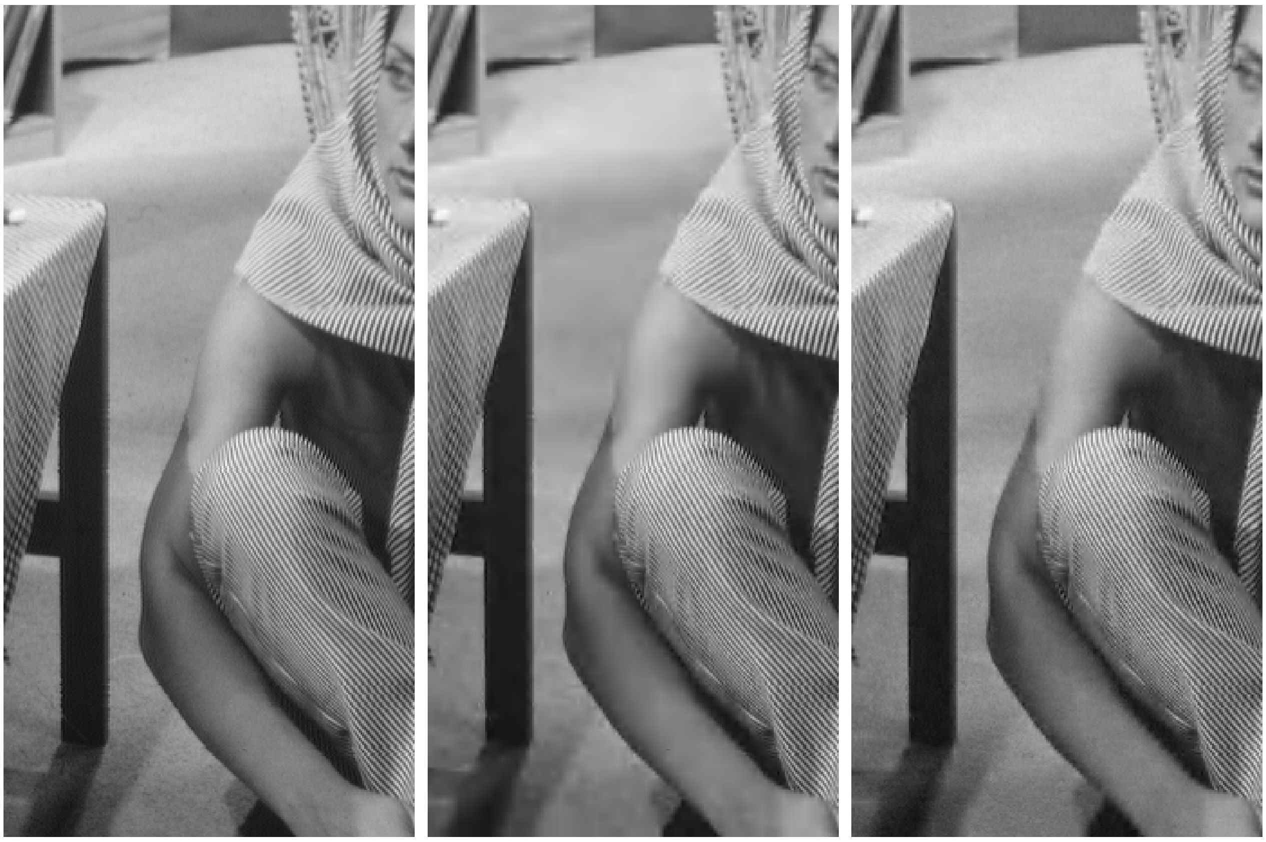

**Fig.7.** Example (crop) of the component based image compression for the image *barbara*. Left: Original image. Center: JPEG2000 compression. Right: Component based image compression. The compression ratio for both compressions is set to 0.8bpp.

quantization scheme for AR vectors is changed what might render it impractical.

*ARX coefficients:* The results show that the bit-wise arithmetic coding as well as the adaptive Exp-Golomb methods perform significantly better than the Exp-Golomb code for k=0 (Fig 4) as used in [1]. In addition, one can observe that the encoding of the vectors in layers yields a slight performance gain at the very low additional cost of sorting the values in layers. These results are very similar for all images in the test set.

*ARX model standard deviation:* Fig. 6 shows the MSE to bit rate ratio for the different $\sigma$ coding methods. We can observe that the wavelet methods and especially the Haar wavelet outperform the predictive methods with respect to the MSE. Similar rate-distortion results can be observed for the whole test set. Since scalar and wavelet methods yield different reconstructions, we assigned the rate for the wavelet methods so that the reconstruction yields approximately the same MSE as the reconstruction of the scalar methods with a fixed quantization step size. This way, we created comparable results for both methods.

Fig. 8 shows the results averaged over the test set for each component. Here we can see that, for a similar MSE, the wavelet methods yield substantially better performance than the scalar methods. Similar to the Itakura distance, it has to be considered that the actual relationship between MSE and perceived visual quality of the noise reconstruction has yet to be evaluated.

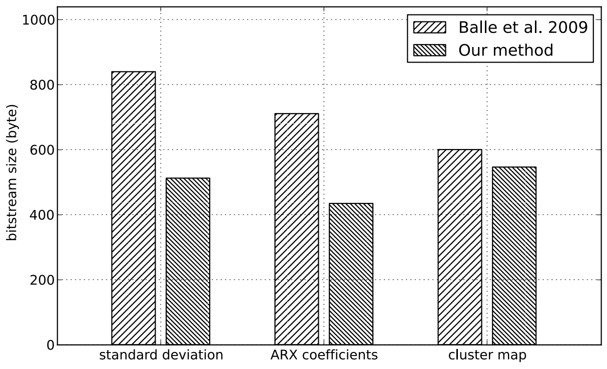

**Fig. 8.** Overall rate improvement compared to [1] at same settings, averaged over the test set.

The previously used method ([1]) consists of Prediction AB for the cluster mapping, Exp-Golomb with k=0 for the ARX coefficients, and the Cluster mapping method for the $\sigma$ values. Our method combines statistic prediction ABCD for the cluster mapping, the layer based method for the ARX coefficients, and the Haar wavelet method for the $\sigma$ values.

Overall, we were able to create an efficient code for the ARX model parameters. With this component in place the component based image compression can now be compared to classical image compression methods like JPEG2000. Fig. 4.3 shows an example for the proposed coding scheme. The original (left) has been encoded with JPEG2000 (middle) and the component based image compression (right). In the JPEG reconstruction strong compression artifacts are visible; The carpet and the arm lost almost all of its texture. However, our method is able to create a convincing reconstruction of these noise like textures at the same bit rate (0.8bpp).

## Acknowledgements

Research described in this paper was supervised by Prof. Dr.-Ing. J.R. Ohm and Dipl.-Ing. Johannes Ballé, Institut für Nachrichtentechnik, RWTH Aachen University.

## References

[1] J. BALLÉ, M. WIEN, A Quantization Scheme for Modeling and Coding of Noisy Texture in Natural Images, 2009

[2] J.S. HARE, P.H. LEWIS, P.G.B. ENSER, and C.J. SANDOM, Mind the gap: Another look at the problem of the semantic gap in image retrieval, Proceedings of SPIE, volume 6073, pages 75-86, 2006

[3] BUADES, A., COLL, B., MOREL, J.M., On image denoising methods, *tech. rep., Centre de Mathématiques et Leurs Applications*, Cachan, France, 2004.

[4] A.B. KIELY, Bit-wise arithmetic coding for data compression, in Proc. of IEEE International Symposium on Information Theroy, Sep 1995, p 394

[5] D.S. TAUBMAN and M.W. MARCELLIN, JPEG2000 - Image Compression Fundamentals, Standards and Practice. Kluwer, Amsterdam, 2002.

[6] J. BALLÉ, libcabac, 2010. A flexible, portable and efficient C++ implementation of the CABAC (context-adaptive binary arithmetic coding) framework. Download at: http://developer.berlios.de/projects/libcabac/.

[7] F. JÄGER, Development of a Texture Model for Component Based Video Coding, Diplomarbeit Institut für Nachrichtentechnik, 2009.

[8] F. ITAKURA, Minimum prediction residual principle applied to speech recognition. IEEE Transactions on Acoustics, Speech and Signal Processing, 23(1):67-72, 1975

[9] J. BALLÉ, A texture model including a quality measure for perceptual image coding, in Proc. of IEEE International Conference on Image Processing ICIP '11, 2011, submitted.

## About Authors...

### Christian FELDMANN

was born in Aachen, Germany in 1985. He completed his degree in Media Engineering in 2010 and is currently working as a Ph.D. student at the Institut für Nachrichtentechnik, RWTH Aachen University focusing on video coding technologies.

### Johannes BALLÉ

was born in Dortmund, Germany in 1980. He 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 the Institut für Nachrichtentechnik, RWTH Aachen University, focusing on image processing, signal theory, and data compression.