Modeling of 2D+1 Texture Movies for Video Coding

S. Valaeys [a], G. Menegaz [b,*], F. Ziliani [a], J. Reichel [a]

[a] *VisioWave S.A., Route de la Pierre, CH-1024 Ecublens, Switzerland*

[b] *Audio Visual Communications Laboratory, Swiss Federal Institute of Technology, CH-1015 Lausanne, Switzerland*

Abstract

We propose a novel model-based coding system for video. Model-based coding aims at improving compression gain by replacing the non-informative image elements with some perceptually equivalent models. Images enclosing large textured regions are ideal candidates. Texture movies are obtained by filming a static texture with a moving camera. The integration of the motion information within the generative texture process allows to replace the "real" texture with a "visually equivalent" synthetic one, while preserving the correct motion perception. Global motion estimation is used to determine the movement of the camera and to identify the overlapping region between two successive frames. Such an information is then exploited for the generation of the texture movies. The proposed method for synthesizing *2D+1* texture movies is able to emulate any piece-wise linear trajectory. Compression performances are very encouraging. On this kind of video sequences, the proposed method improves the compression rate of an MPEG4 state-of-the-art video coder of an order of magnitude while providing a sensibly better perceptual quality. Importantly, the current implementation is real-time on Intel PIII processors.

Key words: Model-based Coding, Dynamic textures, Dynamic Coding

1 Introduction

Classical coding techniques for still images and videos are reaching their full potential. A new perspective must be taken to improve coding performances. In this paper, we propose a new *model-based* approach to coding, based on the redefinition of the notion of relevance. The so-called model-based approach to coding

* Corresponding author.
Email addresses: gloria@ieee.org (G. Menegaz), Francesco.Ziliani@visiowave.com (F. Ziliani), Julien.Reichel@visiowave.com (J. Reichel).

exploits the semantics of the scene for improving compression performances. For those regions that are not “informative”, it is assumed that the relevance resides in the “visual appearance”. The real information can thus be replaced by some “visually equivalent” synthetic representation generated by an adequate model. The practical use of such a model within a coding system adds some constraints to the modeling task. The model must be concise (i.e. require a relatively small number of parameters) to be competitive towards classical coding techniques. Furthermore, the generative process must be fast to enable real-time applications.

Images and videos containing large amounts of texture are appropriate candidates for model-based coding. Textures are difficult to code by classical schemes based on the exploitation of the spatial, temporal and statistical redundancy. Often they create a bottleneck in the coding chain. Moreover, they usually constitute a non informative part of the scene, so that their replacement by a visually equivalent pattern would not degrade the global *perceived* quality of the resulting image. Another factor that makes textures particularly appealing for model-based coding is that in the last decade a great research effort has been devoted the field of texture modeling, resulting in a wide number of possible solutions. The strict connections with the investigation of human vision makes it a powerful tool for the determination of the parameters, or perceptual primaries, which are extracted by the visual system and the way they are translated to higher level perceptual units.

Besides the pioneering work of Heeger and Bergen [1], particularly relevant in this respect are the contributions of Portilla and Simoncelli [2] and Zhu and Mumford [3]. These probabilistic texture models have grown on the insights coming from neurosciences. The main guidelines are in the assumption that the visual system responds to the statistics of the stimuli to which it is exposed and processes the visual information in order to maximize the “efficiency” of the representation [4]. The good match between the derived bases functions with the receptive fields in primary visual cortex [5] provides a justification for such works on a neurophysiological basis. An alternative approach aiming primarily at texture replication, as opposed to the identification of the perceptual features, has been followed by other authors. Some noteworthy examples are the solutions proposed in [6–12].

Conversely, the field of “dynamic” texture modeling is relatively under-investigated. Dynamic textures are usually meant as multi-dimensional stochastic processes exhibiting some stationarity over time [13]. Some examples are smoke, waves and foliage. This can be regarded as a generalization of the bi-dimensional case, where temporal evolution is a feature of the global stochastic process [13–15].

The novelty of our contribution is that we address the problem of modeling a different class of dynamic textures, for which the motion is not an intrinsic property of the considered process, but the result of a continuous change of the point of view. We aim at modeling the motion features as perceived by a moving observer. To make the distinction with respect of the 3D dynamic processes mentioned above,

we call the considered class *2D+1 Texture Movies* (2D+1 TM). In this case, the key point is the preservation of the temporal correlation between subsequent images, or frames. More specifically, we consider here the case of a static texture - the grass - shot by a moving camera. This situation is indeed representative of the typical set of applications we are considering. The synthesis of 2D+1 textures is integrated in a wavelet-based coding system and combined with classical compression of the remaining parts of the images.

This paper is organized as follows. Sec. 2 provides a formalization of the problem. Sec. 3 revisits the modeling technique for static textures proposed in [8] and details the specific current implementation. Sec. 4 describes the core of the proposed system, namely the movement-simulating algorithm. Sec. 5 illustrates some results for all the steps involved, that is static and dynamic texture synthesis as well as video coding performances, and Sec. 6 derives conclusions.

2 Modeling 2D+1 Texture Movies

Probabilistic modeling of static texture aims at generating a new image from a sample texture, such that it is *sufficiently different* from the original yet appears to be generated by the *same underlying stochastic process*.

The goal of the proposed algorithm is to generalize such an idea to the generation of a progressively "growing" texture, where the direction and speed of growth is given *a-priori* by a predefined motion model. More specifically, we focus here on piece-wise linear trajectories of the viewpoint. In this case, the main issue is the preservation of the perception of motion, namely the preservation of those visual features which determine the sensation of continuity in the texture flow.

It is worth pointing out that the trivial juxtaposition of temporally subsequent patches respectively sampled from successive frames is not a solution. The aliasing phenomena due to the sampling as well as the mismatch between the sampling grids associated to two successive frames would result in discontinuities along the boundaries. Importantly, such sampling artifacts are the origin of the failure of prediction-based coders when applied to texture sequences.

In what follows, we provide a simple formalization of the proposed solution.

Let Ω be the infinite lattice, and let Ω_t be the domain which is observed at time t, namely the spatial support associated with the observation at a given instant in time, as illustrated in fig. 1. Let then $I(\Omega_t)$ be the observation at time t and $\tilde{I}(\Omega_t)$ be the synthetic counterpart. Clearly:

$$\Omega_t \subset \Omega \qquad \forall t \tag{1}$$

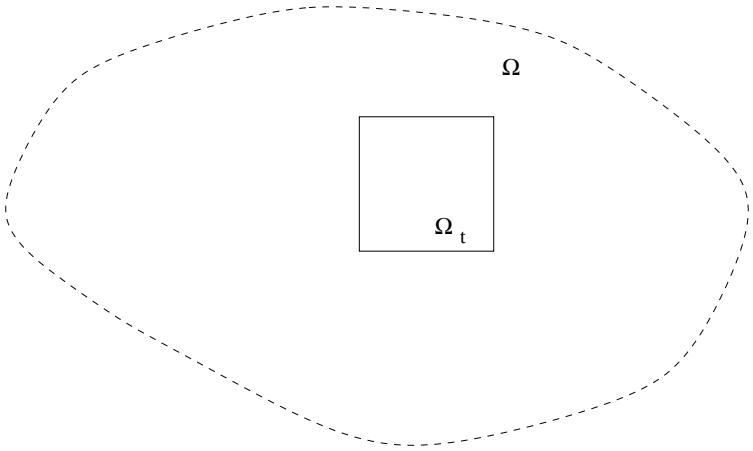


Fig. 1. Sub-lattice covered by the observation at time t.

Accordingly, Ω_{t_1} and Ω_{t_2} denote the domains covered by the observations at times t_1 and t_2, respectively. The specificity of the proposed approach is that it provides a solution for the following problem

Problem 1 *Given two sub-lattices Ω_{t_1} and Ω_{t_2} such that:*

$$\Omega_{t_1} \cap \Omega_{t_2} = \Omega_{\Delta t} \neq \emptyset, \tag{2}$$

generate a synthetic texture over Ω_{t_2} by "growing" it from the seed already present on $\Omega_{\Delta t}$ such that the impression of visual continuity is preserved.

If the two sets were disjoint, then the independent generation of the texture over the two domains would have been adequate. Conversely, when there is an overlap between the two domains, the independent generation of the texture would produce an apparent edge at the boundary or, equivalently, a flickering on the representation as a temporal sequence which destroys the continuity of the visual flow.

The key feature of the proposed model is the ability to synthesize a texture $\tilde{I}(\Omega_{t_2})$ over the domain Ω_{t_2} by growing the missing portion over $\overline{\Omega}_{\Delta t} = \Omega_{t_2} \backslash \Omega_{\Delta t}$ while keeping unchanged the texture already present over $\Omega_{\Delta t}$ and avoiding discontinuities along the boundary.

The previous discussion can be generalized to the case where the observations are themselves realizations of the stochastic process represented by the considered model for static textures. This is indeed the case when focusing on video coding. In such a framework, the idea is to transmit the model of the texture to be reproduced, and use it to generate the texture all along the sequence. [1]

[1] This holds under the hypothesis of homogeneity of the texture all along the sequence. A simple generalization would be to assume that a model for an homogeneous texture would be suitable as long as the texture parameters are contained within a bounded region of the feature space. We leave this subject for future investigation.

In this case, the following relation holds:

$$\tilde{I}(\Omega_{t+\Delta t}) = \tilde{I}(\Omega_t) \oplus \tilde{I}(\overline{\Omega}_{\Delta t}) \tag{3}$$

where $\tilde{I}(\Omega_{t+\Delta t})$ is the synthetic texture simulating the observation at time $t+\Delta t$, $\tilde{I}(\Omega_{\Delta t})$ is the texture seed and the operator $\oplus$ indicates the juxtaposition of the textures stated.

Given the geometry of the scene, the spatial position of $\Omega_{t+\Delta t}$ can be easily recovered from the underlying motion model. Let $x, y \in \mathbb{R}$ be the spatial coordinates of the upper left corner of Ω_t and let h and w, with $h, w \in \mathbb{R}_*^+$, be the height, respectively the width, of the spatial domain Ω_t, assumed to be of rectangular shape. Given the estimated speed $\overrightarrow{v} = (v_x, v_y)$ at which the point of view moves, it is straightforward to derive the position of the domain $\Omega_{t+\Delta t}$ of the observation at time $t+\Delta t$ as the one whose upper left corner has coordinates:

$$\begin{aligned} x + \Delta x &= x + v_x \cdot \Delta t \\ y + \Delta y &= y + v_y \cdot \Delta t \end{aligned} \tag{4}$$

Therefore, one can easily identify $\Omega_{\Delta t}$ and $\overline{\Omega}_{\Delta t}$. The procedure followed for growing $\tilde{I}(\overline{\Omega}_{\Delta t})$ from the seed covering $\Omega_{\Delta t}$ is detailed in Sec. 4.

3 DWT-based Multiresolution Probabilistic Texture Modeling

Static texture synthesis is achieved using the Discrete Wavelet Transform-based Multiresolution Probabilistic Texture Modeling (DWT-MPTM) method proposed in [8]. The model consists in an implicit (non-parametric) representation and reproduction of intra- and inter-scale dependencies among DWT wavelet coefficients. These dependencies are captured and regenerated using the multiresolution conditional sampling technique which can be seen as a direct sampling from a distribution via a Parzen estimator [16]. Since the theoretical link between the size and shape of the Parzen window and the quality of the perceptual features of the synthesized texture is still to be established (apart from the well-known impact of the window width), the size of the window is set empirically and is allowed to vary to ensure it that it encloses a “sufficient” amount of samples (we refer to the Appendix for further discussion).

Linking model parameters to perceptual features is a very difficult problem. Even though many statistical parameters have been proposed by different authors [1–3] as those relevant in terms of ”visual appearance”, the way each of these parameters maps to perception and to higher visual cues is still unknown. Namely, the gap between low-level parameters and mid- to high-level visual features is still unsolved.

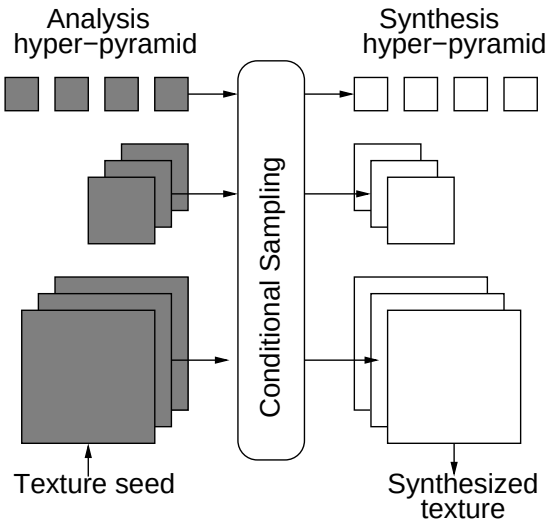


Fig. 2. Principles of the DWT-MPTM. Three levels of decomposition for the DWT are represented.

The fact that non-parametric models, such as the DWT-MPTM, which are based on a decomposition too simple to be representative of visual processes, provide results comparable in quality to other parametric state-of-the-art techniques suggests some redundancy in the more complex representations at least as far as perceptual features are concerned.

The algorithm that inspired the DWT-MPTM [6] basically generates a new texture image by shuffling coefficients of the original texture at different spatial resolutions. This shuffling is constrained in two ways:

- The local characteristics inside a frequency band must be preserved
- Coefficients in a frequency band depend on the corresponding ones at lower resolution

First, a Laplacian *analysis pyramid* is built. Then, a *conditioning pyramid* is obtained by applying a set of orientation selective filters to each level of the analysis pyramid. The conditioning pyramid describes the local properties of each frequency band. A *synthesis pyramid*, the counterpart of the analysis pyramid for the synthetic texture, is then filled with coefficients obtained by conditional sampling on the corresponding levels of the analysis pyramid.

The distinguishing feature of the approach proposed in [8] is that a single *hyper-pyramid* consisting of the DWT subbands combines the functions of both the analysis and the conditioning pyramids for both analysis and synthesis. The algorithm is illustrated in fig. 2. The three detail subbands of the DWT are used as the local descriptors of the frequency bands.

In order to reproduce the characteristic structure of a texture, the statistical intra and inter-scale relationships observed in the original sample between coefficients

must be preserved [2]. In that regard, we define the terms of *parent*, *tree* and *parent vector* as follows. Given the total number N of levels in the pyramid, let $F_i^j(x,y)$ be the coefficient of coordinates (x,y) of the feature image F_i^j of level $0 < i < N$ and orientation $0 < j \leq 3$, $i, j \in \mathbb{N}$. Then, the *parent* of this coefficient is the sample $F_{i+1}^j(\lfloor \frac{x}{2}, \frac{y}{2} \rfloor)$. By extension, we say that the former coefficient is the *child* of the latter. Inspired by the wavelet quad-tree, we then call *tree* any group of coefficients in a pyramid linked by a parent-children relationship throughout the levels. The parent vector of the coefficient $F_i^j(x,y)$ is the vector:

$$\begin{aligned}\vec{V}_i^j(x,y) = \; & \big[F_{i+1}^1(\lfloor \tfrac{x}{2}, \tfrac{y}{2} \rfloor), F_{i+1}^2(\lfloor \tfrac{x}{2}, \tfrac{y}{2} \rfloor), F_{i+1}^3(\lfloor \tfrac{x}{2}, \tfrac{y}{2} \rfloor), \ldots, \\ & F_N^1(\lfloor \tfrac{x}{2^{N-i}}, \tfrac{y}{2^{N-i}} \rfloor), F_N^2(\lfloor \tfrac{x}{2^{N-i}}, \tfrac{y}{2^{N-i}} \rfloor), F_N^3(\lfloor \tfrac{x}{2^{N-i}}, \tfrac{y}{2^{N-i}} \rfloor)\big]\end{aligned} \tag{5}$$

where the notation is the same as above. Such a vector holds all the information necessary for conditioning: the intra and inter-scale dependencies among DWT coefficients are accounted for and modeled by a chain across scales which is implicitly represented by the parent vector.

Conditional sampling consists of two successive steps:

(1) Build a candidate set $C_i^j(x,y)$ for the coefficient $F_i^j(x,y)$
(2) Randomly sample from $C_i^j(x,y)$

The candidate set $C_i^j(x,y)$ identifies the positions of all the coefficients $F_i^{j,a}(x',y')$ of the corresponding analysis pyramid subband which have "similar" parent vectors $\vec{S}_i^j(x',y')$:

$$C_i^j(x,y) = \{(x',y') : D(\vec{V}_i^j(x,y), \vec{S}_i^j(x',y')) \leq \vec{T}_i\} \tag{6}$$

In eq. (6), $\vec{T}_i$ is a threshold vector:

$$\vec{T}_i = [T_{i+1}^1, T_{i+1}^2, T_{i+1}^3, \ldots, T_N^1, T_N^2, T_N^3] \tag{7}$$

and $D(\cdot)$ represents the component-wise absolute difference operator (information on the selection of the threshold vector can be found in the Appendix). Accordingly, the condition given in eq. (7) can equivalently be written in the form:

$$|V_{i,k}^j(x,y) - S_{i,k}^j(x',y')| \leq T_{i,k} \tag{8}$$

where the subscript k indicates the k^{th} component of the respective vectors. Once the candidate set $C_i^j(x,y)$ is determined, one of its members is randomly chosen and its value is assigned to the coefficient $F_i^j(x,y)$ of the synthesis pyramid.

The synthesis process starts by filling the highest level of the synthesis hyper-pyramid, namely the three detail subbands of level N and the low frequency subband, by replicating the coefficients of the corresponding subbands of the analysis pyramid. Successive finer levels are then filled by conditional sampling.

Results show that in many cases the structure of the texture is better preserved by limiting the sampling to the higher levels of the pyramid. In this particular implementation of the DWT-MPTM, conditional sampling is limited to the level $N-1$. Hence, subbands of the level N of the synthesis pyramid are copied from the analysis one, those of level $N-1$ are filled through conditional sampling and the remaining levels are filled by retaining the entire trees emerging from the last synthesized samples.

As explained in the Appendix, groups of coefficients are sampled at once to improve the efficiency of the algorithm while better preserving texture characteristics. Because they share the same parent vector, coefficients of coordinates $(x,y),(x+1,y),(x,y+1)$ and $(x+1,y+1)$, where x and y are even numbers, of all three subbands of level $N-1$ are sampled together.

The two ideas explained above, which are crucial to making the DWT-MPTM fast enough for real-time applications, influence the degree randomness introduced by algorithm. Since in the current implementation it basically consists in a controlled shuffling of trees of wavelet coefficients, we call it *Conditional Tiling in Transform Space* (CTTS). The advantages of CTTS over conventional tiling are twofold. Because it takes place in transform space, and because tiles are conditionally positioned next to one another, feature mismatches and block effects are greatly reduced. This enables the use of smaller tiles than would be possible at image level, which reduces the chance of an apparent replica of the original sample.

3.2 Generating Pictures of Any Size

The guideline of the DWT-MPTM [8] algorithm, which is the ground of the proposed method, is to reproduce intra and inter-scale dependencies among subband samples.

The generative process starts by filling the highest level of the synthesis pyramid with DWT coefficients in the corresponding subbands of the analysis pyramid. In case the texture to be synthesized has the same size as the original sample, such coefficients are simply copied from one pyramid to the other. When the synthetic texture is larger than the original one, there is obviously a size mismatch between the respective subbands which makes that the DWT coefficients of the coarsest level of the analysis pyramid are not numerous enough to fill completely the same level of the synthesis pyramid. In [8] and [6], either pixel replication (i.e. zero-order interpolation) or tiling of the entire subbands are used. The main drawback of tiling is that it would generate blocking artifacts unless the concerned subbands are strongly homogeneous. For example, if a texture illuminant were not constant

but fading from right to left, tiling would be visible and the resulting output image would be vitiated. If, in principle, such a condition could always be satisfied with a sufficiently deep subband decomposition, in practical applications this is usually not the case.

Instead, the solution adopted here consists in progressively filling the empty positions by randomly picking one of the DWT coefficients which are within a predefined threshold from the one lying in the just-filled position. This implies the assumption of smoothness of the concerned subbands. Such method is particularly suitable for our framework because no additional computation is needed: the set of coefficients that are eligible for sampling the coarsest level is a by-product of the procedure leading to the candidate set for conditional sampling of the just-filled positions.

4 Incorporation of the DTW-MPTM in a movement-simulating algorithm

As stated in Sec. 2, the specificity of the proposed approach is to integrate the DWT-MPTM in a solution to the more general problem of growing a synthetic texture over a domain Ω_{t_2} from a seed already present on the sub-lattice $\Omega_{\Delta t}$ such that the impression of visual continuity is preserved (**Problem 1**).

The propose method consists in synthesizing a texture covering an area larger than the video frame size while preserving the portion over $\Omega_{\Delta t}$. A new texture is generated only when necessary, to fill the region of support $\overline{\Omega}_{\Delta t}$ without creating apparent discontinuities.

It is worth pointing out that the straightforward solution of synthesizing each frame independently with the DWT-MPTM is not suitable because it creates a disjointed succession of rapid texture changes that fails to generate an impression of movement. One also quickly comes to the conclusion that a cut-and-paste approach at image level, in which the common section is correctly displaced and remaining empty parts of the frame are filled with newly synthesized patches of texture, creates unacceptable discontinuities.

Another trivial solution would be to synthesize a much larger texture area than the frame size and to select the covered domain to be part of the frame according to the camera movement. This method is however subject to application concerns. First, the required size of the synthetic texture should be known *a-priori*. Moreover, large amounts of texture could be produced without ever being needed.

A way to answer those concerns is to work in feature space. Although the DWT used for compression purposes is in general not shift-covariant, covariance properties hold for translations in transform space which correspond to translations at

image level that can be broken down in horizontal and vertical shifts of $k \cdot 2^N$ and $h \cdot 2^N$ pixels respectively, where $k, h \in \mathbb{Z}$ and N is the number of decomposition levels of the DWT. Working in feature space, we are consequently able to generate from a synthetic image S the following set:

$$\Gamma = \{S_{(k,h)}, k, h \in \mathbb{Z} | S_{(k,h)} = T_{(k,h)} S\} \tag{9}$$

where $T_{(k,h)}$ is the translation operator that applied to S produces a shift of $k \cdot 2^N$ and $h \cdot 2^N$ units in the horizontal and vertical directions, respectively.

As the translation from S to $S_{(k,l)}$ takes in fact place in feature space, the remaining empty parts of $S_{(k,l)}$ can be filled in feature space by applying the DWT-MPTM algorithm locally without creating discontinuities.

More in general, any random translation can be obtained by extending the size of S by adding a border of 2^N pixels on all sides. Therefore, simulating a random translation is a two-step process: obtaining the correct $S_{(k,h)}$; selecting the correct area of $S_{(k,h)}$ which is to make up the video frame. An example is shown in fig. 3. Let p and q, with $p, q \in \{0, 2 \cdot 2^N\}$ be the width in pixels of the border zone respectively in the horizontal and vertical direction of movement. To generate a horizontal, respectively vertical, movement of m, respectively n, pixels at image level, with $m \geq p$ corresponding to Δx and $n \geq q$ corresponding to Δy in Sec. 2, the correct $S_{(k,h)}$ is chosen so that:

$$\begin{aligned} k &= \min_{\tilde{k}} \{(p + \tilde{k} \cdot 2^N) \geq m\} \\ h &= \min_{\tilde{h}} \{(q + \tilde{h} \cdot 2^N) \geq n\} \end{aligned} \tag{10}$$

The window of visibility is then correctly positioned inside $S_{(k,h)}$, namely:

$$\begin{aligned} p_{new} &= p + k \cdot 2^N - m \\ q_{new} &= q + h \cdot 2^N - n \end{aligned} \tag{11}$$

Proceeding in such a way, we avoid both having to know beforehand the amount of texture needed and generating large useless amounts of it.

Another aspect of the procedure is that, for obvious memory concerns, synthesized textures are not saved once they are not needed anymore. This leads to the regeneration of texture in the case of movement in one direction then in the opposite one. However, because we deal with highly stochastic textures, subsequent changes are in our opinion very difficult to discern.

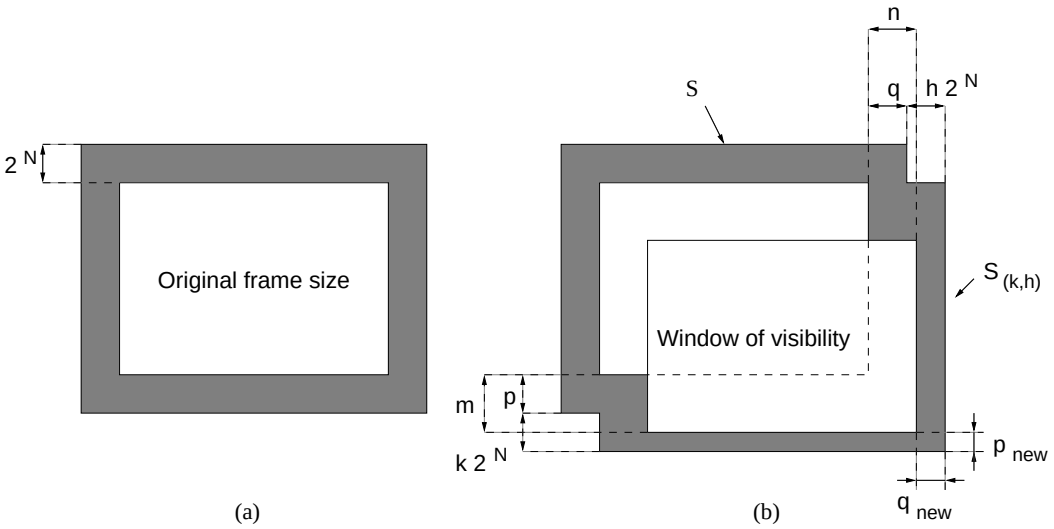


Fig. 3. Simulation of movement. The size of S is bigger than that of the original. When the added border is not enough to simulate the required movement, a shifted version of S is created. The window of visibility is then moved inside $S_{(k,h)}$ to reproduce the correct movement.

5 Results and Discussion

The performance of the proposed system has been characterized according to three criteria: the model ability to preserve the perceptual features of the original sample, in the static as well as the dynamic case; the coding gain; and the computational efficiency.

5.1 Assessment of the perceptual features

Before tackling this subject, it is important to mention that despite the great amount of research devoted to the identification of the *perceptual features* which determine texture perception, the problem is still unsolved. Two main guidelines can be identified.

The first follows a synthesis-by-analysis approach and is based on the assumption that there exists a set of statistics which is *necessary and sufficient* to identify a *texture class*. The way the resulting model parameters translate to perception is still unknown.

The second aims at identifying the *visual mechanisms* which are responsible for texture perception on a psychophysical or, more in general, neuro-physiological basis [17–21]. The focus is on the *local* parameters that are relevant for the analysis of the visual stimulus with regard to the considered task.

The problem is very complex and further investigation is needed to understand and model the involved visual processes. In this contribution, we do not put forward a general theory for texture perception neither a golden rule for the evaluation of a modeling technique. In our opinion, the visual mechanisms which subserve these phenomena need to be investigated further before being able to formulate a general theory. Instead, we have focused on a particular case - driven by an application - and we have faced it in an empirical way leading to what can be considered a "first order" solution. The identification of the features which determine the classification of the texture as belonging to a given class as well as the impression of motion will be an essential part of our future research.

The evaluation of the ability of any texture model, either static or dynamic, to reproduce the perceptual features of the original texture implies *ad-hoc* subjective tests respecting the paradigm set by psychophysics. However, as mentioned above, such a task was beyond the scope of this contribution. Nevertheless, some informal subjective tests involving untrained people of the laboratory revealed that the majority of the subjects were not able to distinguish between the original and synthetic samples.

Fig. 4 gives an example of the performance of the DWT-MPTM on two natural textures - `bark` and `fabric` - from the Brodatz album [22]. The original and synthesized textures are shown in the left and right column, respectively. In general, results show that the method performs well on a wide variety of random and structured textures. We refer to [8] for further discussion.

The ability of the proposed algorithm to reproduce the correct impression of motion between successive frames or, equivalently, the temporal *correlation* present in the original sequence, has been qualitatively tested by comparing the motion vector field estimated on the original (fig. 5(a)) with that obtained by applying the motion estimation algorithm to the synthesized sequence (fig. 5(b)). The similarity between fig. 5(a) and (b), as opposed to the random distribution of the vectors that would result from the estimation of the motion vector field on frames that were generated independently from each other, proves that the proposed method is able to preserve the perceived temporal flow between the frames of the sequence and, consequently, to induce the same motion *perception* in the observer. The randomness of the spatial directions of the subset of vectors in the rightmost part of fig. 5(b) is an example of the performance of the motion estimation technique on a synthetic texture. The lack of point-wise correspondences between the current and reference frames prevents from establishing a block-wise matching and results in a random spatial distribution of the motion vectors. As previously mentioned, the preservation of the correct impression of motion has been informally tested on some non-trained subjects. All of them could correctly identify the situation and provided the same description of the dynamic scene for both the original and the synthesized test sequences.

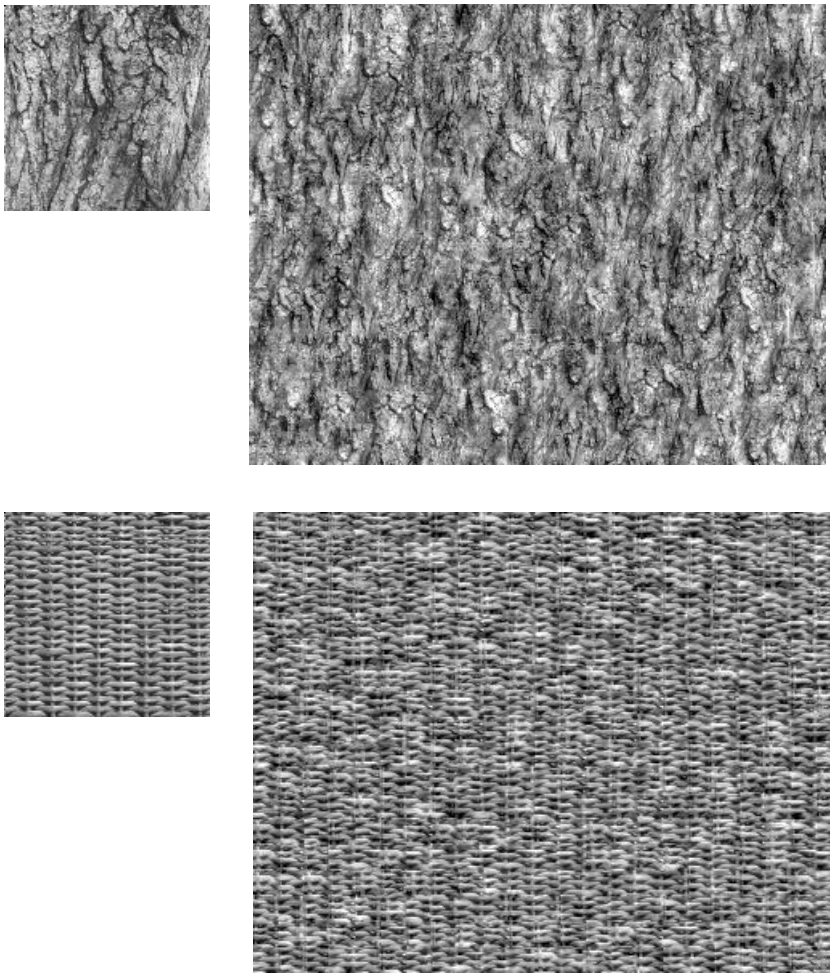
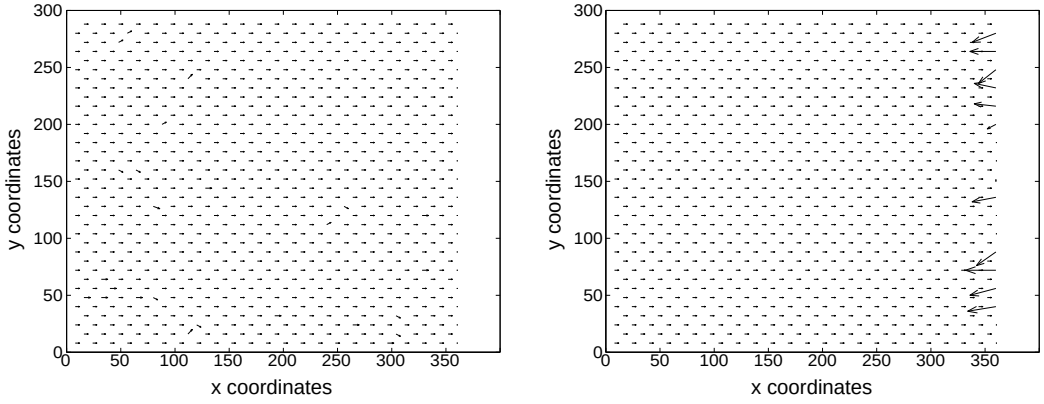


Fig. 4. Original (left column) and synthesized (right column) samples for the `bark` and `fabric` textures from the Brodatz album.



(a) (b)

Fig. 5. Motion vector fields as estimated on (a) the original sequence (b) the synthesized sequence.

The proposed generative model for 2D+1 TM is particularly suitable for the integration within a coding system. First, it is based on typical tools exploited for video coding: the DWT and the classical motion estimation techniques [23]. Second, the use of the lifting steps scheme [24] for the wavelet decomposition has a number of features that make it particularly adapted in view of the implementation on a device: it enables the in-place processing as well as the use of integer arithmetic [25]; it simplifies the management of border conditions; and it reduces the complexity by up to a factor 4 [26]. Furthermore, the model is very concise and the synthesis algorithm is computationally efficient (single-step).

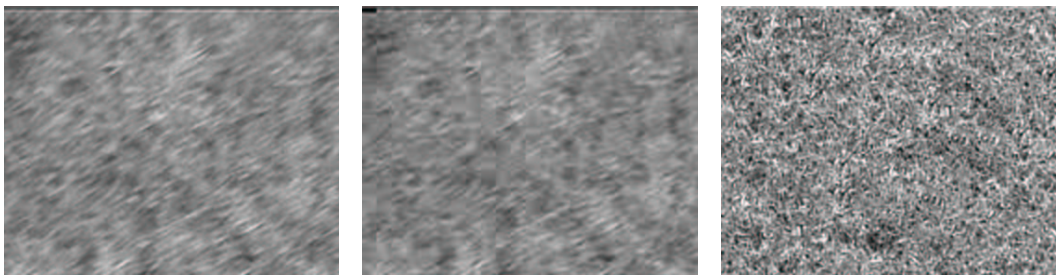
The suitability of the 2D+1 texture generative model for compression have been tested by integrating it within a coding system for video sequences. The idea is to first pre-process the scene by separating the objects (the football players) from the background (the grass) and then encode each element wit the most suitable engine, according to some predefined optimality criteria [27].

Our claim is that the compression performances can be significantly improved by replacing the the background information by a synthetic counterpart. The test sequences consist of homogeneous 2D+1 TM. It is worth to point out that compression rates are not affected by the size of the synthetic picture to be generated, because the information to be transmitted is restricted to the model itself, which is the same as long as the hypothesis of homogeneity holds.

First results are quite promising [2]. The total rate corresponding to the proposed model-based system, i.e. the bitrate needed to encode the model itself and the approximation subbands of the chromatic channels for a 2D+1 grass texture, is about 104 kb/s. It is worth pointing out that the decoding process works in real-time (40 frames/s) on Pentium III processors. By comparison, the minimum rate attainable by a state-of-the-art MPEG4 [28] encoder on the same sequence was more than ten times higher, that is about 1408 kb/s [3]. Even though the movement of the camera is reduced to simple translations, predictive coders like MPEG4 struggle with such texture sequences. The analog texture is captured and sampled by the digital recording device. The high frequency content of the texture usually exceeds the bandwidth of the input chain and aliasing occurs. Moreover, the limited acquisition speed of the camera introduces additional frequency smoothing. Successive frames in the original movie are therefore not an exact translated version of one another. For the same reason, it is not possible to encode with a lossless method the extra amount of texture needed from one frame to another and ”paste” it alongside

[2] A wide set of synthesized as well as MPEG4 decoded sequences is available at `http://www.visiowave.com/GMBV/`.

[3] The public domain DivX implementation of the MPEG4 standard is available at `http://www.divx.com/`.



(a) (b) (c)

Fig. 6. Performance of the MPEG4 algorithm. (a) Original frame; (b) Minimum rate MPEG4 decoded frame (c) Corresponding synthesized frame.

the previous frame without creating discontinuities. This leads us to believe that our system fills a gap in the range of coding techniques. Importantly, even though some frames of the video sequence are blurred - because of both the motion and the limitations of the acquisition device which smooth spatial frequencies - the visual quality of the decoded frames at low rates is still unsatisfying. Fig. 6 shows one example. The upper limit to the visual quality of the decoded frame is set by the original frame. At low rates, some coding artifacts are added to the blurring (see fig. 6 (b)) further reducing the quality of the image available at the decoder side. Conversely, the proposed model-based approach implicitly leads to *quality enhancement* of the decoded frames, as long as the model is properly selected. The perceptual quality of the frames is the same all over the sequence, for a given rate, namely, without an additional cost in terms or resources. This has a direct impact on the applications, enabling for example the `pause` function with same quality on any image.

5.3 Computational efficiency

As mentioned throughout this paper, care has been devoted to attaining high computational efficiency. It is worth pointing out that entire frames of texture rarely need to be generated. In fact, for most frames, no new texture needs to be synthesized at all. In most of the cases, for the considered type of sequences the estimated motion corresponds to a displacement of a few pixels. Using an enlarged sample as reference, as described in Sec. 2, allows to reproduce such small displacements by simply moving the visibility window within the image that is already available. On the same line, when the available residual border is not enough to simulate the movement, only a small percentage of pixels needs to be generated, keeping the global computational cost quite low.

Even though the analysis of the complexity of the algorithm has not been performed, a concrete indication of the efficiency can be deduced from the processing time. The time needed to generate an entire new frame of 432×352 pixels starting from a 128×128 texture seed amounts to about 45 ms. When the movement

is such that only a portion of new texture must be generated, the synthesis process takes about 15 ms. Finally, the process is extremely fast (about 0.002 ms) for most frames where the window of visibility must simply be adjusted.

A further improvement would result from the exploitation of the inability of the visual system to perceive high spatial frequencies in presence of motion. The possibility to reduce the set of subbands to account for would further simplify the synthesis process. The price to pay would be the loss of the frame-wise constant resolution.

A major shortcoming of the proposed model is the inability to render perspective. This of course limits its applicability in real world situations.

Even though these first results are very encouraging, a number of issues are still to be solved to be able to cope with real world situation. Among the main open issues are the generalization to other types of camera movements, like zooms and rotations (when these cannot be adequately approximated by piece-wise linear trajectories) and the rendering of perspective.

6 Conclusions

We propose a novel generative model for 2D+1 TM, suitable for model-based coding of video. The integration of the motion information within the DWT-MPTM algorithm for static textures results in a dynamic generative model able to synthesize any 2D+1 TM featuring any piece-wise linear trajectory. A texture seed is extracted from a frame of the original sequence and is used as model for synthesizing the other frames. The motion vector field is estimated at any frame and is used to constrain the generative process so that the correct temporal correlation between the images is preserved. The global model-based video coding system is highly computationally efficient. The current implementation is real-time on PIII processors. Furthermore, first results show that it outperforms a reference implementation of the MPEG4 coder in both coding gain and quality of the decoded information. This holds for the comparison being performed in both static (frame-wise) and dynamic (on the entire video) regime. Among the issues deserving further investigation are the emulation of other camera functions, like zooming and rotation, the rendering of perspective and the integration of the synthesis procedure within an object-based coding framework.

A Determining the Threshold Vector Values

The threshold values making up the vector $\vec{T}_i$ represent one of the most sensitive parameter of the algorithm. Roughly, large values do not hold enough discerning power making the resulting candidate set is too vast to be representative of the texture structure. Conversely, small values over-constrain the sampling generating some replica of the model.

The difficulty in finding the optimal threshold values is increased by the fact that they are texture-dependent. Our solution consists in fixing the minimum number C_{min} of candidates and progressively updating the threshold values until such a condition is met. The initial values $\vec{T}_{i,0}$ used for $\vec{T}_i$ are the estimated standard deviations of the corresponding analysis subbands:

$$T_{i,0}^j = \sqrt{\frac{1}{n-1}\sum_x\sum_y (F_i^j(x,y))^2} \tag{A.1}$$

where n is the number of coefficients in the subband F_i^j. A wide range of values has been tested for both T_i^j and C_{min} (between 0.2 and 1.5 times the standard deviation for the first, and 4 to 32 candidates for the second) but no sensible variations has been observed.

B Improving the Speed of the DWT-MPTM

One specificity of the current implementation is that the computational efficiency is sensibly improved with respect to [8]. This is due to the strategy followed for designing and implementing the multiscale conditional sampling.

First, the synthesis pyramid is filled *tree by tree* instead of level by level, that is by proceeding "vertically" instead of "horizontally" through the pyramid. This is based on the following property:

$$F_i^{j,a}(x',y') \in C_i^j(x,y) \rightarrow F_{i+1}^{j,a}(\lfloor\frac{x'}{2},\frac{y'}{2}\rfloor) \in C_{i+1}^j(x,y) \tag{B.1}$$

namely, a coefficient can only belong to the candidate set $C_i^j(x,y)$ if its parent belongs to $C_{i+1}^j(\lfloor\frac{x}{2},\frac{y}{2}\rfloor)$. However, this is a necessary but not sufficient condition. The candidates for $F_i^j(x,y)$ are the subset of descendants of the elements $C_{i+1}^j(x,y)$, $F_{i+1}^j(\frac{x'}{2},\frac{y'}{2})$ satisfying the additional condition:

$$|F_{i+1}^j(\lfloor\frac{x}{2},\frac{y}{2}\rfloor) - F_{i+1}^{j,a}(\lfloor\frac{x'}{2},\frac{y'}{2}\rfloor)| \leq T_{i+1}^j \quad \forall j = 1,\ldots,3 \tag{B.2}$$

Basically, the candidate set of the "just filled" sample brings all the *a-priori* information needed to build the candidate set for its descendants. The gain in efficiency is due to the (progressive) filling of all the positions within the current tree before switching to other positions of the current level. Therefore, the synthesis tree is built branch by branch.

Second, the definition of the parent structure allows to sample *groups* of coefficients at once. More specifically: samples of coordinates (x,y) of their respective subbands in a common level share the same parent vector. Consequently, if a coefficient of the analysis pyramid $F_i^{1,a}(x',y')$ belongs to the candidate set $C_i^1(x,y)$, then $F_i^{2,a}(x',y')$ and $F_i^{3,a}(x',y')$ belong to $C_i^2(x,y)$ and $C_i^3(x,y)$, respectively. We exploit this property by *linking* such coefficients for sampling the subbands of a common level. Instead of randomly choosing three different coefficients for $F_i^1(x,y)$, $F_i^2(x,y)$ and $F_i^3(x,y)$, the three coefficients $F_i^{1,a}(x',y')$, $F_i^{2,a}(x',y')$ and $F_i^{3,a}(x',y')$ are chosen at once and assigned to their respective position. Furthermore, because of the subsampling implied by the DWT, four pixels per subband share the same parent vector. If x and y are both even numbers, then the coefficients of coordinates $(x,\ y), (x,y+\ 1), (x+1,y)$ and $(x+1,y+1)$ of all detail subbands of a common level share the same parent vector. We also choose to link the sampling of those four coefficients. Accordingly, if $F_i^{j,a}(x',y')$ is chosen for $F_i^j(x,y)$, then the three other children of the same father are assigned the following values:

$$\begin{aligned} F_i^j(x+1,y) &= F_i^{j,a}(x'+1,y') \\ F_i^j(x,y+1) &= F_i^{j,a}(x',y'+1) \\ F_i^j(x+1,y+1) &= F_i^{j,a}(x'+1,y'+1) \end{aligned} \tag{B.3}$$

For this to be correct, x, y, x' and y' must all be multiples of two.

In all, twelve coefficients are linked together. It is worth mentioning that besides the gain in complexity (a single candidate set is built for 12 coefficients, and the random function is called only once), the linking tends to strengthen the relationship between coefficients both within and across scales, improving the representation of the spatial structure.

Finally, to keep the complexity low, the size of both the original and synthetic images is constrained to be a power of two.

C Processing Boundary Samples

In our implementations of the DWT and IDWT, the subband boundaries are extended by symmetry. A virtual neighborhood is therefore created to perform the

convolution. While this does not have an impact for static texture synthesis, problems arise when movement comes into play. When coefficients are shifted, some of them which lied at the border of their respective subbands are now surrounded by other pixels. Their neighborhoods have changed. Hence the result of convolution will change, which in turn will affect the pixel values at image level.

To minimize the artifacts, we use the lifting scheme implementation of the $(5,3)$ [25] interpolating wavelet. The corresponding lifting steps representation consists of only two steps involving the current and either the previous or following samples. The size of the local neighborhood is then one. Consequently, only the border coefficient of each subband is concerned by changing neighborhoods. But since a coefficient at highest level of decomposition corresponds to 2^N pixels at image level, a border of 2^N pixels at image level is affected by changing values from one frame to another.

To avoid this problem we extend once again the size of the synthesized texture image, so that changing pixels are contained within a border of 2^N pixels, which is never visible. Artifacts due to the border effect are always kept out of the actual video frame.

References

[1] D. Heeger, J. Bergen, Pyramid-based Texture Analysis/Synthesis, in: Proc. of SIGGRAPH '95, 1995, pp. 229–238.

[2] J. Portilla, E. Simoncelli, A parametric texture model based on joint statistics of wavelet coefficients, International Journal of Computer Vision 40 (1) (2000) 49–71.

[3] S. Zhu, Y. Wu, M. Mumford, Filters, Random Fields and Maximum Entropy (FRAME) - Towards the unified theory for texture modeling, in: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1996.

[4] E. Simoncelli, B. Olshausen, Natural image statistics and neural representation, Annual Review of Neuroscience 24 (2001) 1193–1215.

[5] B. Olshausen, D. Field, Emergence of simple-cell receptive field properties by learning a sparse code for natural images, Nature 381 (1996) 607–609.

[6] J. S. De Bonet, Multiresolution sampling procedure for analysis and synthesis of texture images, in: Computer Graphics, ACM SIGGRAPH, 1997, pp. 361–368.

[7] J. De Bonet, P. Viola, A non-parametric multi-scale statistical model for natural images, Advances in Neural Information Processing 10.

[8] G. Menegaz, DWT-based non-parametric texture modeling, in: Proc. of the International Conference on Image Processing (ICIP), Vol. 1, Thessaloniki, Greece, 2001, pp. 173–176.

[9] A. Efros, T. Leung, Texture synthesis by non-parametric sampling, in: Proc. of the International Conference on Computer Vision (ICCV), 1999.

[10] L. Wei, M. Levoy, Fast texture synthesis using tree-structured vector quantization, in: Proc. of SIGGRAPH 00, 2000, pp. 479–488.

[11] Y. Xu, B. Guo, H. Shum, Chaos mosaic: Fast and memory efficient texture synthesis, Tech. Rep. MSR-TR-2000-32, Microsoft (April 2000).

[12] M. Ashikhmin, Synthesizing natural textures, in: Symposium on Interactive 3D Graphics, 2001.

[13] S. Soatto, G. Doretto, Y. Wu, Dynamic textures, in: Proc. of the International Conference on Computer Vision (ICCV), 2001.

[14] Z. Bar-Joseph, R. El-Yaniv, D. Lischinski, M. Werman, Texture mixing and texture movie synthesis using statistical learning, Proc. IEEE Visualization (2001) 403–410.

[15] M. Szummer, R. Picard, Temporal texture modeling, in: Proc. of the International Conference on Image Processing (ICIP), Lausanne, Switzerland, 1996.

[16] R. Duda, P. Hart, Pattern classification and scene analysis, John Wiley and sons, 1973.

[17] M. Landy, Texture perception, Encyclopedia of Neuroscience, Elsevier., Amsterdam, The Neatherlans, 1996.

[18] S. Wolfson, M. Landy, Examining edge- and region-based texture analysis mechanisms, Vision Research 38 (3) (1998) 439–446.

[19] M. Landy, H. Kojima, Ideal cue combination for localizing texture-defined edges, J. Opt. Soc. Am. A 18 (9) (2001) 2307–2320.

[20] Z. Li, Modeling pre-attentive stereo grouping by intracortical interactions in early visual cortex, Journal of Vision 1 (3).

[21] Z. Li, A saliency map in the primary visual cortex, Trends in Cognitive Sciences 6 (1) (2002) 9–16.

[22] P. Brodatz, Textures : a photographic album for artists and designers, Dover, New York, 1966.

[23] T. Aach, A. Kaup, R. Mester, Statistical model-based change detection in moving video, IEEE Trans. on Signal Processing 31 (2) (1993) 165–180.

[24] I. Daubechies, W. Sweldens, Factoring wavelet transforms into lifting steps, J. Fourier Anal. Appl. 4 (3) (1998) 247–269.

[25] A. Calderbank, I. Daubechies, W. Sweldens, B. Yeo, Wavelet transforms that map integers to integers, Appl. Comput. Harmon. Anal. 5 (3) (1998) 332–369.

[26] J. Reichel, Complexity-related aspects of image compression, Ph.D. thesis, Swiss Federal Institute of Technology (EPFL) (Feb. 2001).

[27] M. Kunt, A. Ikonomopoulos, M. Kocher, Second generation image coding techniques, Proceedings of the IEEE 73 (4).

[28] ISO/IEC JTC 1/SC 29/WG11, Working Document N3156 (Dec. 1999).