Binary Pseudowavelets and Applications to Bilevel Image Processing

Steven Pigeon
Université de Montréal

pigeon@iro.umontreal.ca

Yoshua Bengio
Université de Montréal

bengioy@iro.umontreal.ca

ABSTRACT

This paper shows the existance of binary pseudowavelets, bases on the binary domain that exhibit some of the properties of wavelets, such as multiresolution reconstruction and compact support. The binary pseudowavelets are defined on $\mathbb{B}^n$ (binary vectors of length *n*) and are operated upon with the binary operators *logical and* and *exclusive or*. The forward transform, or analysis, is the decomposition of a binary vector into its constituant binary pseudowavelets. Binary pseudowavelets allow multiresolution, progressive reconstruction of binary vectors by using progressively more coefficients in the inverse transform. Binary pseudowavelets bases, being sparse matrices, also provide for fast transforms; moreover pseudowavelets rely on hardware-friendly operations for efficient software and hardware implementation.

Introduction

The motivation of this work is to provide a new tool for the compression and multiresolution analysis of bilevel images. Bilevel images are usually coded by mechanisms akin to CCITT G4 standard fax encoding [1], where entropy coding is achieved by a modified Huffman code, or some other device using context modeling and arithmetic coding [3,4], or even pattern matching combined with context modeling and arithmetic coding [5]. These methods compress the image in its original domain. The proposed method decomposes the image into its constituent binary pseudowavelets and compression can be done in the transformed domain. It also provides means to lossy compression with graceful degradation of the image. Binary pseudowavelets exhibit some of the properties of ordinary wavelets, like localization in space and multiresolution reconstruction. One major difference between ordinary wavelets and binary pseudowavelets is that binary pseudowavelets are computed using inherently discrete transforms. Furthermore, binary pseudowavelets are computed using only logical operations (**and** and **xor**) whereas classical wavelets require more or less complex arithmetic operations to compute the transforms.

Binary pseudowavelets exhibit properties that are wavelet-like. Localization in space, or compact support, takes a slightly different meaning with binary pseudowavelets. Orthogonality of basis vectors relative to integer translates of themselves is also present, as is the nesting of reconstruction spaces (multiresolution). However, individual binary pseudowavelets do not integrate to zero, since they only use zeroes and ones (therefore no negative values). Basis vectors are the rows of the pseudowavelet basis matrix.

Binary pseudowavelets are localized in space: no row in a pseudowavelet basis matrix (except possibly for the vector associated with zero frequency, associated with the so-called DC term) is filled with ones. Ideally, all rows will only contain a number of ones proportional to its associated wavelength (inverse of frequency) and these ones will be contiguous to one another, surrounded by zeros. Orthogonality of basis vectors of same associated frequency in relation with their own integer translates must be verified. This implies in practice that two vectors of same associated frequency do not have any bit set to one at the same position at the same time. That is, (1,1,0,0) and (0,0,1,1) could be two vectors of an order 4 pseudowavelet basis, while (1,1,0,0) and (0,1,1,0) could not since they overlap[1]. This means that the integer translates requirement in pseudowavelets is changed in a way that makes the minimal translate proportional to the size of the run of ones in the basis vectors considered.

The multiresolution property requires that rows in the basis matrix contain less and less ones as their associated frequency increases. It also requires that a reconstruction of a transformed vector gets more and more precise with respect to the original as the number of reconstruction vectors used in the inverse transform increases (thus the nesting of vector spaces). Essentially, a basis that satisfies all of these conditions is a binary pseudowavelet.

Section 1. Definition of Binary Pseudowavlets

Let us now give the formal definitions pertaining to binary pseudowavelets. The basic set $\mathbb{B}$, is the binary alphabet, $\{0,1\}$. Let us first define operations on elements of $\mathbb{B}$ and $\mathbb{B}^n$.

Def. 1a. For $a,b \in \mathbb{B}$, $a \,\&\, b$ is the logical and of *a* and *b*.

Def. 1b. For $a,b \in \mathbb{B}$, $a \oplus b$ is the logical exclusive or of *a* and *b*.

Def. 1c. For $\boldsymbol{v} \in \mathbb{B}^n$, $\boldsymbol{v} = (v_0, v_1, \ldots, v_{n-1})$, $v_i \in \mathbb{B}$, $i = 0,\ldots,n-1$ is a **binary vector**.

Def. 1d. For $\boldsymbol{v} \in \mathbb{B}^n$, $|\boldsymbol{v}|_1$ is the number of ones in $\boldsymbol{v}$, $|\boldsymbol{v}|_0$ the number of zeroes, and $|\boldsymbol{v}|_1 + |\boldsymbol{v}|_0 = n$.

Def. 1e. For $\boldsymbol{v} \in \mathbb{B}^n$, $|\boldsymbol{v}|_T$ is the number of transistions in $\boldsymbol{v}$. The number of transistion is the number of times the elements of the vector go from 0 to 1 or *vice versa*.

Def. 1f. For $W \in \mathbb{B}^{n\times n}$, $W = (\boldsymbol{w}_0, \boldsymbol{w}_1, \ldots, \boldsymbol{w}_{n-1})$, $\boldsymbol{w}_i \in \mathbb{B}^n$ the i^{th} row of *W*, $i = 0,\ldots,n-1$, is a **binary matrix**.

Def. 2a. For $\boldsymbol{a} \in \mathbb{B}^n, b \in \mathbb{B}$, $\boldsymbol{a} \,\&\, b = (a_0 \,\&\, b, a_1 \,\&\, b, \ldots, a_{n-1} \,\&\, b)$.

For $\boldsymbol{a} \in \mathbb{B}^n, b \in \mathbb{B}$, $\boldsymbol{a} \oplus b = (a_0 \oplus b, a_1 \oplus b, \ldots, a_{n-1} \oplus b)$.

Def. 2b. For $\boldsymbol{a} \in \mathbb{B}^n, \boldsymbol{b} \in \mathbb{B}^n$, $\boldsymbol{a} \oplus \boldsymbol{b} = (a_0 \oplus b_0, a_1 \oplus b_1, \ldots, a_{n-1} \oplus b_{n-1})$.

For $\boldsymbol{a} \in \mathbb{B}^n, \boldsymbol{b} \in \mathbb{B}^n$, $\boldsymbol{a} \,\&\, \boldsymbol{b} = (a_0 \,\&\, b_0, a_1 \,\&\, b_1, \ldots, a_{n-1} \,\&\, b_{n-1})$.

[1] These rows, together, might, however, still form a basis.

Def. 3a. For $V \in \mathbb{B}^{n\times n}, \boldsymbol{v} \in \mathbb{B}^n$, $V{*}\boldsymbol{v} = (V_0 \,\&\, v_0) \oplus (V_1 \,\&\, v_1) \oplus \ldots \oplus (V_{n-1} \,\&\, v_{n-1})$.

Def. 3b. For $A \in \mathbb{B}^{n\times n}, B \in \mathbb{B}^{n\times n}$, $A{*}B = \left(A{*}B_o, A{*}B_1, \ldots, A{*}B_{n-1} \right)$.

With these basic operators on single bits and bit vectors, let us show the existence of binary pseudowavelet bases on $\mathbb{B}^n$ under the $*$ operator. We have to show that there are such bases and that their inverses exist.

Theorem 1. Let $V \in \mathbb{B}^{n\times n} = \left(\boldsymbol{v}_0, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_{n-1} \right)$, $\boldsymbol{v}_i \in \mathbb{B}^n$, $i = 0, 1, \ldots, n-1$. V is a **basis** on $\mathbb{B}^n$ under the $*$ operator, iff $\forall\, \boldsymbol{x} \in \mathbb{B}^n, \exists!\, \boldsymbol{x}' \in \mathbb{B}^n \,|\, V{*}\boldsymbol{x}' = \boldsymbol{x}$.

Theorem 2. If $V \in \mathbb{B}^{n\times n}$ is a basis on $\mathbb{B}^n$, then its **inverse** is the binary matrix $V^{-1} \in \mathbb{B}^{n\times n} = \left(\boldsymbol{v}_0^{-1}, \boldsymbol{v}_1^{-1}, \ldots, \boldsymbol{v}_{n-1}^{-1} \right)$, such that, for $i = 0, 1, \ldots, n-1$, we have $V{*}\boldsymbol{v}_i^{-1} = \mathrm{I}_i$ (where I_i is the i^{th} row of the identity matrix), and V^{-1} is also a basis on $\mathbb{B}^n$ under the $*$ operator. V^{-1} is the inverse of V, since $V^{-1}{*}V = \mathrm{I}$ and $\forall\, \boldsymbol{x} \in \mathbb{B}^n$, $V^{-1}{*}(V{*}\boldsymbol{x}) = \boldsymbol{x}$.

Corollary 2. $V \in \mathbb{B}^{n\times n}$ is a **basis** iff $\exists V^{-1} \in \mathbb{B}^{n\times n}$ such that $V^{-1}{*}V = \mathrm{I}$.

Def 4. Two vectors $\boldsymbol{w}_i$ and $\boldsymbol{w}_j$ of a basis have the same associated **frequency** iff $|\boldsymbol{w}_i|_1 = |\boldsymbol{w}_j|_1$ and $|\boldsymbol{w}_i|_T = |\boldsymbol{w}_j|_T$.

Def 5. A **binary pseudowavelet basis**, $W \in \mathbb{B}^{n\times n}$, is a basis on $\mathbb{B}^n$, such that for its inverse W^{-1}:

1) All of W^{-1} row vectors are **localized**, except possibly for the zero frequency basis vector: this means that $\forall\, \boldsymbol{w}_i^{-1} \in W^{-1}$, $\left|\boldsymbol{w}_i^{-1}\right|_T \le 2$, and the number of ones in the basis vectors is proportional to the associated frequencies, i.e. $\left|\boldsymbol{w}_i^{-1}\right|_1 \propto n/_{i+1}$. Therefore $\left|\boldsymbol{w}_0^{-1}\right|_1 \ge \left|\boldsymbol{w}_1^{-1}\right|_1 \ge \cdots \ge \left|\boldsymbol{w}_{n-1}^{-1}\right|_1$.

2) All basis vectors of same frequency are non-overlapping under the & operator, that is, if $\forall\, \boldsymbol{w}_i^{-1}, \boldsymbol{w}_j^{-1} \in W^{-1}$, we have that $\left|\boldsymbol{w}_i^{-1}\right|_1 = \left|\boldsymbol{w}_j^{-1}\right|_1 \to \boldsymbol{w}_i^{-1} \,\&\, \boldsymbol{w}_j^{-1} = \left(0, 0, \ldots, 0 \right)$ (all zeroes).

3) The basis exhibits the **multiresolution property**. Let $\mathrm{H}(\boldsymbol{v},\boldsymbol{w}) = |\, \boldsymbol{v} \oplus \boldsymbol{w} \,|_1$ be the number of places in which $\boldsymbol{v}$ and $\boldsymbol{w}$ differ, the Hamming distance. Let $\boldsymbol{l}_i$ be the vector that contains *i* ones and then *n-i* zeroes. Let $\boldsymbol{x}' = W{*}\boldsymbol{x}$, the transform of $\boldsymbol{x}$. A basis W is multiresolution iff , $\forall\, \boldsymbol{x}$, we have that
$n \ge \mathrm{H}\left(W^{-1}{*}(\boldsymbol{x}' \,\&\, \boldsymbol{l}_1), \boldsymbol{x} \right) \ge \mathrm{H}\left(W^{-1}{*}(\boldsymbol{x}' \,\&\, \boldsymbol{l}_2), \boldsymbol{x} \right) \ge \cdots \ge \mathrm{H}\left(W^{-1}{*}(\boldsymbol{x}' \,\&\, \boldsymbol{l}_n), \boldsymbol{x} \right) = 0$.

Def. 6. A pseudowavelet basis is **regular** with respect to its inverse if each time we double the number of coefficients used in the reconstruction, the reconstruction resolution also doubles: $\forall \boldsymbol{x}'$, $\mathrm{H}\left(W^{-1}{*}(\boldsymbol{x}' \,\&\, \boldsymbol{l}_i), \boldsymbol{x} \right) \ge 2\,\mathrm{H}\left(W^{-1}{*}(\boldsymbol{x}' \,\&\, \boldsymbol{l}_{2i}), \boldsymbol{x} \right)$.

The last conditions of Def. 5 and Def. 6 can be modified depending on exactly what type of progressive reconstruction we want because various types of reconstruction corresponds to different notions of error. We will use some function F(.) that is more representative of the error in a given context than H(.), but we will require that the relation $cn \geq \mathrm{F}\left(W^{-1} * (\boldsymbol{x}' \,\&\, \boldsymbol{l}_i)\right) \geq \mathrm{F}\left(W^{-1} * (\boldsymbol{x}' \,\&\, \boldsymbol{l}_j)\right) \geq 0$ holds for $0 \leq i \leq j \leq n$ (right side equality must hold if $j = n$) and for some constant *c* if we are to say that the basis *W* exhibits the multiresolution property. This condition can be further relaxed to become $cn \geq \mathrm{E}\left[\mathrm{F}\left(W^{-1} * (\boldsymbol{x}' \,\&\, \boldsymbol{l}_i)\right)\right] \geq \mathrm{E}\left[\mathrm{F}\left(W^{-1} * (\boldsymbol{x}' \,\&\, \boldsymbol{l}_j)\right)\right] \geq 0$, for $0 \leq i \leq j \leq n$ and where $\mathrm{E}[.]$ stands for the expectation, assuming a distribution of the vectors $\boldsymbol{x}$, if multiresolution is needed on the average rather than on every vector of $\mathbb{B}^n$.

To compute the inverse of a binary pseudowavelet basis, we usually proceed backwards. Since it is the characteristics of the multiresolution reconstruction that are the most important, we choose a set of reconstruction vectors, that is, the inverse. We construct W^{-1} so that it computes the type of reconstruction we want, then we inverse it to get *W*. The naïf algorithm suggested by the definition of the inverse (see theorem 2) is $O\left(2^n\right)$ which is prohibitively complex except for trivially small *n*. A modified version of the classical Gauss-Jordan elimination algorithm for matrix inversion provides an algorithm in $O\left(n^3\right)$. The modifications to the Gauss-Jordan algorithm are minimal. Subtraction is replaced by the exclusive or (**xor**) and the pivot is chosen at time *t* so that it is the row with the greatest number of ones but with its first one in position *t*. If another row has a one in position *t*, we **xor** it against the pivot row (and so with the identity matrix side). The algorithm stops when the basis side is the identity (or a permutation of it). The side initialized with the identity is now the inverse. When the algorithm stops, we reorder the resulting matrix to get the identity properly sorted on the left side. The right side contains the inverse of W^{-1}, i.e., *W*.

Section 2. Examples of Analysis and Reconstruction using Binary Pseudowavelets.

Let us present an example of a binary pseudowavelet basis. For a given vector length, there are numerous possible binary pseudowavelet bases. For the sake of simplicity, we picked $n = 8$ as the transform size and a regular basis. As the choice of basis should depend on its target application, we may have one or the other of the possible binary pseudowavelet bases. Here is the basis *W* and its inverse W^{-1}:

$$W = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad W^{-1} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

The pseudowavelet basis *W* and its inverse W^{-1}.

Since it is convenient to think of all the operations involved in the binary pseudowavelet transforms as operations on binary numbers, the indexing convention is different from the usual matrix indexing convention, as is shown here:

$$\begin{bmatrix} (0,n-1) & (0,n-1) & \cdots & (0,1) & (0,0) \\ (1,n-1) & (1,n-2) & \cdots & (1,1) & (1,0) \\ \vdots & \vdots & \therefore & \vdots & \vdots \\ (n-2,n-1) & (n-2,n-2) & \cdots & (n-2,1) & (n-2,0) \\ (n-1,n-1) & (n-1,n-2) & \cdots & (n-1,1) & (n-1,0) \end{bmatrix}$$

Ordering convention in the pseudowavelet basis matrix

Using W and W^{-1}, let us compute the forward transform of the vector $\boldsymbol{x} = [1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0]$. In the vector notation, also, the indexing is reversed, that is, $\boldsymbol{w} = [w_7 \quad w_6 \quad w_5 \quad w_4 \quad w_3 \quad w_2 \quad w_1 \quad w_0]$. To avoid confusion, in the examples, the indexes are written beside the vectors. To compute the forward transform:

$$\boldsymbol{x} = \begin{bmatrix}0\\1\\0\\0\\0\\1\\1\\1\end{bmatrix}\begin{matrix}0\\ \vdots\\ \\ \\ \\ \\ \vdots\\ 7\end{matrix} \qquad W * \boldsymbol{x} = \begin{bmatrix} 1&0&0&0&1&0&1&1\\ 1&0&0&0&0&0&0&0\\ 0&1&0&0&1&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&1&1&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&1&0&0\\ 0&0&0&1&0&0&0&0 \end{bmatrix} * \begin{bmatrix}0\\1\\0\\0\\0\\1\\1\\1\end{bmatrix}\begin{matrix}0\\ \vdots\\ \\ \\ \\ \\ \vdots\\ 7\end{matrix} = \begin{array}{c cccccccc} & 1&0&0&0&0&0&0&0\\ \oplus & 0&0&1&0&0&0&0&0\\ \oplus & 0&0&0&1&0&1&0&0\\ \oplus & 0&0&0&1&0&0&0&0\\ \hline = & 1&0&1&0&0&1&0&0 \end{array} = \begin{bmatrix}0\\0\\1\\0\\0\\1\\0\\1\end{bmatrix}\begin{matrix}0\\ \vdots\\ \\ \\ \\ \\ \vdots\\ 7\end{matrix} = \boldsymbol{x}'$$

The forward transform of vector *x* with transform *W*.

The inverse transform, or reconstruction is the same operation as the forward transform, except that we use W^{-1} instead of W.

$$\boldsymbol{x}' = \begin{bmatrix}0\\0\\1\\0\\0\\1\\0\\1\end{bmatrix}\begin{matrix}0\\ \vdots\\ \\ \\ \\ \\ \vdots\\ 7\end{matrix} \qquad W^{-1} * \boldsymbol{x}' = \begin{bmatrix} 1&1&1&1&1&1&1&1\\ 1&1&1&1&0&0&0&0\\ 1&1&0&0&0&0&0&0\\ 0&0&0&0&1&1&0&0\\ 1&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&1&0 \end{bmatrix} * \begin{bmatrix}0\\0\\1\\0\\0\\1\\0\\1\end{bmatrix}\begin{matrix}0\\ \vdots\\ \\ \\ \\ \\ \vdots\\ 7\end{matrix} = \begin{array}{c cccccccc} & 1&1&0&0&0&0&0&0\\ \oplus & 0&0&1&0&0&0&0&0\\ \oplus & 0&0&0&0&0&0&1&0\\ \hline = & 1&1&1&0&0&0&1&0 \end{array} = \begin{bmatrix}0\\1\\0\\0\\0\\1\\1\\1\end{bmatrix}\begin{matrix}0\\ \vdots\\ \\ \\ \\ \\ \vdots\\ 7\end{matrix} = \boldsymbol{x}$$

The inverse transform of *x'* with W^{-1}.

Multiresolution reconstruction is best understood using progression in time. Suppose we devise an experiment where both sender and receiver have agreed upon a binary pseudowavelet basis, W. The sender computes the binary pseudowavelet transform of $\boldsymbol{x}$ according to W, giving the vector $\boldsymbol{x}'$. Starting at time $t = 0$, the bits of $\boldsymbol{x}'$ are sent one by one to the receiver, beginning with x'_o. As the receiver gets the bits, he computes the inverse transform with W^{-1} and $\boldsymbol{x}^t$, his approximation of x' at time t. The vector $\boldsymbol{x}^t$ will have its first t bits set as the first t bits of $\boldsymbol{x}'$ (that is, $\boldsymbol{x}^t = \boldsymbol{x}' \,\&\, \boldsymbol{l}_t$, cf. def. 5), and the remaining bits set to zero. As each new bit gets in, the decoder computes $W^{-1} * \boldsymbol{x}^t = \tilde{\boldsymbol{x}}^t$, where $\tilde{x}^t$ is the approximation of $\boldsymbol{x}$ at time t. When t reaches n-1, $\tilde{\boldsymbol{x}}^n = \boldsymbol{x}$.

The pseudowavelet basis can be chosen to exhibit some special properties, such as, if we are considering analysis and reconstruction of bilevel images, that the multiresolution reconstruction is visually pleasant. We used the binary pseudowavelet basis U for the analysis and reconstruction of bilevel images. In the case of image processing, we will need 2D binary pseudowavelet transforms. One way of obtaining 2D transforms is simply to use separable transforms, that is, to apply the transform on the rows of the image and then on the (transformed) columns. The other would be to use full 2D binary pseudowavelets bases (in this case, the basis would be represented by a tensor).

$$U = \begin{bmatrix} 1&0&0&0&1&0&1&0\\ 1&0&0&0&0&0&0&0\\ 0&1&0&0&1&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&1&0&1\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&1&0&0\\ 0&0&0&1&0&0&0&0 \end{bmatrix} \qquad U^{-1} = \begin{bmatrix} 1&1&1&1&0&0&0&0\\ 0&0&0&0&1&1&1&1\\ 1&1&0&0&0&0&0&0\\ 0&0&0&0&1&1&0&0\\ 1&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&1&0 \end{bmatrix}$$

The separable binary pseudowavelet transform is simply, for a section of a picture $P \in \mathbb{B}^{n \times n}$, given by $U * (U * P)^T$. Fig 1. shows the multiresolution reconstruction of the "vitruvia" image with the separable 2D transform based on the binary pseudowavelet basis *U* (shown above). In Figs. 1 and 2, the pictures are, from left to right, reconstructed with progressively more bits, to simulate a progressive transmission environment (or a lossy compression scheme where coefficients are discarded).

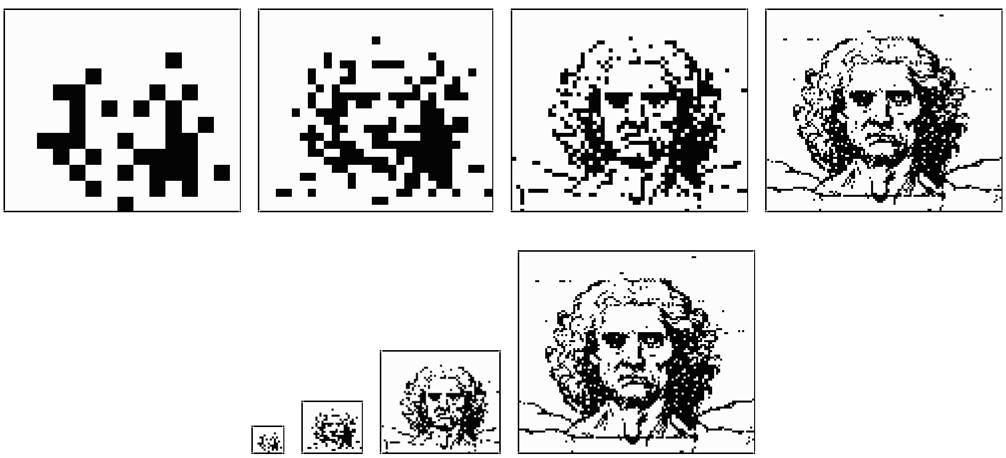

Fig 1. "Vitruvia" picture at different resolutions, using basis U_8. Top, from left to right, the number of coefficients used in the inverse transform is increased fourfold for each frame. The first frame contains only 1/64 of the coefficients, the second 1/16, the third ¼ and the last all of the coefficients. Below, the same pictures but displayed at corresponding scales.

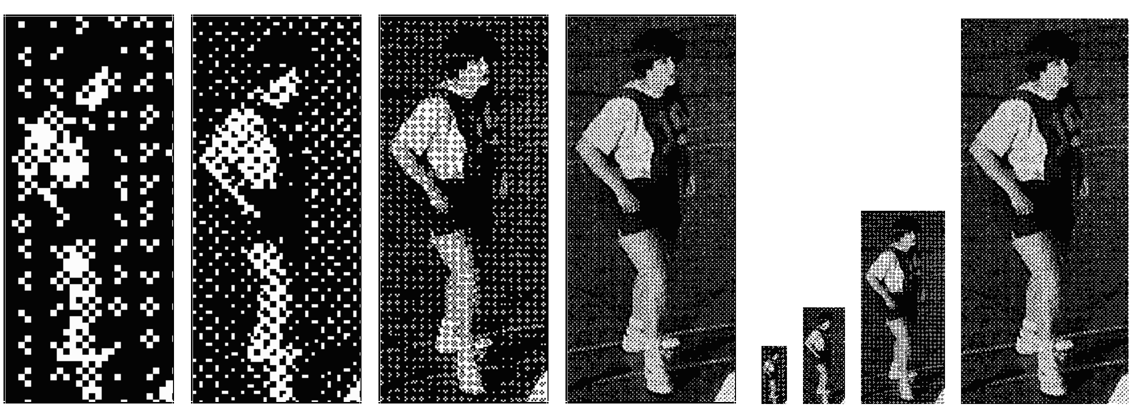

Fig 2. Same as fig. 1, with halftoned picture "basketball".

Basis *U* also behaves well on halftoned images, as shown in Fig. 2. Since there are numerous bases on $\mathbb{B}^n$ under $*$ that are also binary pseudowavelets, one can choose the basis that fits best his needs. For halftoning applications, the user must find a wavelet that matches the pace and scale of the halftone dots [2], so that during the reconstruction process, it takes relatively few coefficients to approximate the dots with relative accuracy.

Section 3. Generalization and Finding New Binary Pseudowavelet Bases.

As binary pseudowavelet bases are sparse and regular (for regular binary pseudowavelets), it is often possible to derive generating functions for bases associated to different dimensions*n*. For basis *U*, for example, we find that one possible expression is:

$$U_n(\ f,t\ ) = \begin{cases} 1 & \text{if } (\ n-1-t<2\ ) \wedge f \bmod n = \frac{(2-n+t)n}{2} \\ 1 & \text{if } (\ t \bmod 2^{\alpha(n,n-1)-\alpha(n,t)} = f \operatorname{div} 2^{\alpha(n,t)+1}\ ) \wedge (\ f \bmod 2^{\alpha(n,t)} = 0\ ) \\ 0 & \text{otherwise} \end{cases}$$

$$U_n^{-1}(\ f,t\ ) = U_n^{-1}\Big(\ (\ p,q\ ),t\Big) = \begin{cases} 1 & \text{if } (f=1) \wedge t \geq \frac{n}{2} \\ 1 & \text{if } (f \neq 1) \wedge \frac{nq}{2^p} \leq t < \frac{n(q+1)}{2^p} \\ 0 & \text{otherwise} \end{cases}$$

with *p* and *q* non-negative integers such that $2^p + q = f$, and *p* the largest non-negative integer satisfying the equality, but with the following exceptions: $f = 0 \rightarrow p = 0, q = 0$ and $f = 1 \rightarrow p = 0, q = 1$. $\alpha(\ n,t\ )$ is

$$\alpha(\ n,t\ ) = \begin{cases} 0 & \text{if } 0 \leq t < \frac{n}{2} \\ 1 & \text{if } \frac{n}{2} \leq t < \frac{3n}{4} \\ 2 & \text{if } \frac{3n}{4} \leq t < \frac{7n}{8} \\ \vdots & \vdots \\ \lceil \lg n \rceil - 1 & \text{if } \frac{(n/2-1)n}{n/2} \leq t < \frac{(n-1)n}{n} \end{cases}$$

This formulation strongly resembles the formulation of Haar wavelets, although it is not as elegant. It will allow us to construct generalized transforms *U* for vectors of length $n = 2^m$, not just vectors of length 8.

Another interesting question is: how do we obtain new binary pseudowavelet bases? One way is to place manually bits in a matrix, and making sure that it has an inverse (the forward transform) and that it satisfies all of the criteria of a binary pseudowavelet basis. This can easily be done with an interactive program that, each time a bit changes state (by the action of the user), computes the inverse and prompts the user in some way when a basis is reached, and possibly indicate the column that causes the computation of the inverse to fail. While with this method one can test if one's idea corresponds indeed to a binary pseudowavelet basis for a small *n*, one still has to determine the general generation formulae, which are not always obtained easily.

Another possible approach is to use combinatorial search. While brute force combinatorial search is practical only for small *n*, combinatorial search can be restricted by preselecting a limited number of candidate vectors. One can generate a restricted set of vectors according to one's specifications, such as, for example, the number of ones must be a power of two ($\forall\, \boldsymbol{v}, |\boldsymbol{v}|_1 = 2^k$ for $k = \{0, 1, \ldots, \lg n\}$), the number of transitions is less than z ($\forall\, \boldsymbol{v}, |\boldsymbol{v}|_T \le z$), etc. Let m be the number of the vectors of length n thus selected (or generated). While an exhaustive search for all bases on all vectors of length n would be $O\left(\binom{2^n}{n}\right)$, the restricted search is now only $O\left(\binom{m}{n}\right)$, and since m is usually much smaller than 2^n, we can consider the generation of all bases using n of the m candidate vectors and the selection of the ones that fit best our needs. The resulting algorithm is now $O\left(\binom{m}{n} n^3\right)$, since it is enough to check for inversibility to validate a set of vectors as a basis. It is possible to further reduce the complexity of the generation of bases, as we generate the possible combinations, by pruning those which violate one of the properties of binary pseudowavelet bases as soon as we can determine that they do. For example, while we are constructing a candidate basis, we will make sure that two vectors of same associated frequency do not overlap. Applying diverse pruning techniques will speed up the combinatorial search significantly. When a combinatorial search has produced a number of binary pseudowavelet bases, the user picks one, or a subset, that satisfies a set of user- and target-defined constraints.

This algorithm also suggests a protocol to exchange information about bases between a sender and a receiver. Suppose that both sender and receiver agree beforehand on a set of binary pseudowavelet bases. Rather than explicitely store the k different bases, as they can be very numerous, both sender and receiver will use a set of m vectors, all the distinct vectors contained in all agreed upon binary pseudowavelet bases. As mn can be small compared to kn^2, it can be more efficient to store only this table. To identify which basis to use, out of all possible bases, the sender can send m bits to the receiver. The decoder then builds its own basis for decoding. (Although a more efficient approach would be to use exactly $O\left(\lg \binom{m}{n}\right)$ bits by basis identifier; this encoding would be more efficient since $\forall\, m, \forall\, n \le m, \lg \binom{m}{n} \le m$.)

Section 4. Efficient Software and Hardware Implementations of Binary Pseudowavelets Transforms

The sparseness and regularity (for regular binary pseudowavelets) of binary pseudowavelet bases suggests that fast transform algorithms exist. In software, if the vectors to transform are not too long, say 16 bits long or less, the forward and inverse transforms can be implemented as look-up tables, since a table size of 2^{16} entries is reasonable. If the vectors to transform are not of an amenable length, fast transforms algorithms can be derived for the chosen binary pseudowavelet basis. While the optimal fast transform algorithm will ultimately depend on the basis, since a binary pseudowavelet basis has between $O(n)$ and $O(n \lg n)$ non zero coefficients (this can be shown using property (1) of def. 5), it will be computed by an algorithm with a complexity ranging from $O(n)$ to $O(n \lg n)$. The multiplicative constant can be made quite small, since general purpose microprocessors offer binary operations on 'words' rather than on individual bits. So rather than computing **and** and **xor** for each individual bit, we can process 2^p bits at the same time (usually p ranges from 3 to 6), giving complexities from $O\left(\frac{1}{2^p}n\right)$ to $O\left(\frac{1}{2^p}n\lg n\right)$.

As for hardware implementation, the circuit required for the forward and inverse transforms have few gates. The fact that we are dealing with binary data instead of integers, or even floating point numbers, will allow each node in the circuit to be constructed with only one gate. Moreover, if there are k bits to 1 in a column of the transform, the number of gates needed to compute the result for this column will be $O(k)$ and the depth of the circuit will be $O(\lg k)$ — a series of the type of $x_0 \oplus x_1 \oplus x_2 \oplus x_3 \oplus \ldots \oplus x_{n-2} \oplus x_{n-1}$ can be evaluated as $\left(\ldots\left((x_0 \oplus x_1) \oplus (x_2 \oplus x_3)\right) \oplus \ldots \oplus (x_{n-2} \oplus x_{n-1})\ldots\right)$ which leads to a $O(\lg k)$ depth circuit. As k is $O(\lg n)$ on the average (cf. property (1) of def. 5), here again, the circuit will have a complexity of at most $O(n\lg n)$ gates, with an average $O(\lg\lg n)$ depth for a transform on vectors of length *n*. However, some transforms will reduce to a circuit with $O(n)$ gates, as does basis *U*, since not all columns contain the same number of ones. Columns with only one one are only pass-through wires.

The great simplicity of the circuit will allow realization of many parallel transforms or of a single but wider transform on a single chip. For example, a fax image has a horizontal resolution of 1728 pixels [1]. For such a length, we would have a circuit with about 10000 gates, which is modest by today's standards. It would be easily feasible to construct such a chip, or even include the transform circuitry as a part of an extended ALU of a dedicated processor. In that case it would be even highly likely that the circuitry needed to feed the transform and to store back the results would be as large, if not larger.

Future work

This paper covers only generation of bases, analysis and reconstruction of binary vectors using binary pseudowavelets, but applications to data compression and document analysis are clear. The multiresolution property of binary pseudowavelets allows for progressive reconstruction, which, in terms of data compression, corresponds to the possibility of using lossy compression schemes. A document image could be transformed onto one of the binary pseudowavelet domains, with resulting bits either kept or discarded (we can hardly speak of quantization in this case) in function of their relative importance (according to some criterion), and remaining bits coded efficiently using context modeling and arithmetic coding. This is the spirit of the coders that are found in current wavelet-based compression schemes [6,7]. Binary pseudowavelet bases also have applications in bandwidth-constrained reconstruction of bilevel images and documents. Future work would require exploration of "quantization" techniques applicable to the transformed domain, and what types of encoding are sensible.

Conclusion

Binary pseudowavelet are simple bases defined on $\mathbb{B}^n$ under the $*$ operator, which is defined with the **and** and **xor** logic functions, that exhibit the properties of locality and multiresolution, as do classical wavelets. Binary pseudowavelet are tools for the multiresolution analysis of bilevel images, can be used in lossless or lossy compression schemes of bilevel images, and are computed efficiently either in software by fast transform algorithms, or in hardware by relatively simple circuitry.

References

[1] RFC-804: Draft Recommendation t.4 — Standardization of Group 3 Facsimile Apparatus for Document Transmission.

[2] K. Denecker, P. De Neve, and I. Lemahieu — Improved lossless halftone image coding using a fast adaptive context template scheme — in Proceedings of the Data Compression Conference, March 1998, Snowbird, Utah, USA, p. 541 (J. A. Storer and M. Cohn, eds. , IEEE Computer Society).

[3] K. Denecker, S. Van Assche, and I. Lemahieu — Binary tree context modeling of halftone images using a fast adaptive template scheme — in Signal Processing Symposium, (Leuven), pp. 75-78, IEEE Benelux, Mar. 1998.

[4] UPI: ITU-11508 : Recommendation T.82 — Information technology - coded representation of picture and audio information - progressive bi-level image compression.

[5] JBIG Committee, Working Draft WD14492 — JBIG 2 Working Draft —, http://www.jpeg.org/public/wd14492.pdf

[6] Patrick Haffner, Léon Bottou, Paul G. Howard, Patrice Simard, Yoshua Bengio, and Yann Le Cun — Browsing through High Quality Document Images with DjVu — In Proceedings of IEEE Advances in Digital Libraries '98.

[7] Léon Bottou, Steven Pigeon — Lossy Compression of Partially Masked Still Images — James A. Storer, Martin Cohn, eds, *Data Compression Conference* 1998, p. 528.