Depth and Depth-Color Coding using Shape-Adaptive Wavelets

Matthieu Maitre and Minh N. Do

Abstract

We present a novel depth and depth-color codec aimed at free-viewpoint 3D-TV. The proposed codec uses a shape-adaptive wavelet transform and an explicit encoding of the locations of major depth edges. Unlike the standard wavelet transform, the shape-adaptive transform generates small wavelet coefficients along depth edges, which greatly reduces the bits required to represent the data. The wavelet transform is implemented by shape-adaptive lifting, which enables fast computations and perfect reconstruction. We derive a simple extension of typical boundary extrapolation methods for lifting schemes to obtain as many vanishing moments near boundaries as away from them. We also develop a novel rate-constrained edge detection algorithm, which integrates the idea of significance bitplanes into the Canny edge detector. Together with a simple chain code, it provides an efficient way to extract and encode edges. Experimental results on synthetic and real data confirm the effectiveness of the proposed codec, with PSNR gains of more than 5 dB for depth images and significantly better visual quality for synthesized novel view images.

Index Terms

3D-TV, free-viewpoint video, depth map, depth-image, shape-adaptive wavelet transform, lifting, boundary wavelets, edge coding, rate-distortion optimization

I. Introduction

Free-viewpoint three-dimensional (3D-TV) video provides an enhanced viewing experience in which users can perceive the third spatial dimension (via stereo vision) and freely move inside the 3D viewing

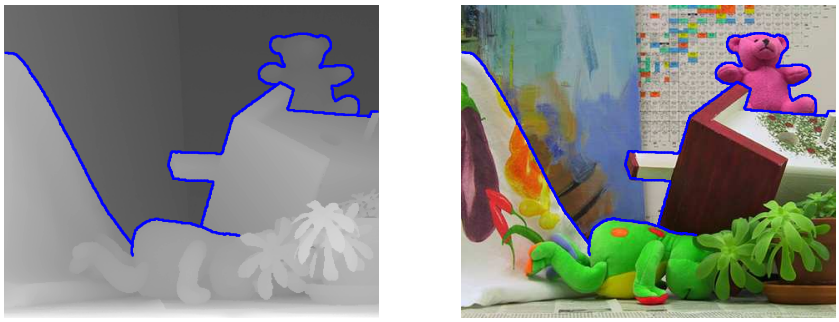
M. Maitre is with the Windows Experience Group, Microsoft, Redmond, WA (email: mmaitre@microsoft.com). The work was performed while at the University of Illinois at Urbana-Champaign, Urbana IL.

M. N. Do is with the Department of Electrical and Computer Engineering, the Coordinated Science Laboratory, and the Beckman Institute, University of Illinois at Urbana-Champaign, Urbana IL (email: minhdo@uiuc.edu).

This work was supported by the National Science Foundation under Grant ITR-0312432.

space [1]. With the advent of multi-view autostereoscopic displays, 3D-TV is expected to be the next evolution of television after high definition. Three-dimensional television poses new technological challenges, which include recording, encoding, and displaying 3D videos. At the core of these challenges lies the massive amount of data required to represent the set of all possible views – the plenoptic function [2] – or at least a realistic approximation.

The Depth-Image-Based Representation (DIBR) has recently emerged as an effective approach [3], which allows both compact data representation and realistic view synthesis. A DIBR is made of pairs of and *depth* and *color* images (see an example in Figure 1), each of which provides a local approximation of the plenoptic function. At the receiver, arbitrary views are synthesized from the DIBR using image-based rendering with depth information [4].



(a) Depth image and edges (b) Color image and edges

Fig. 1. Input data of the DIBR representation with shared edges superimposed over a depth image (a) and color image (b).

Depth information can be obtained by stereo match or depth estimation algorithms [5]. However, these algorithms are usually complicated, inaccurate and inapplicable for real-time applications. Thanks to the recent developments of lower-priced, fast and robust range sensors [6] which measure time delay between transmission of a light pulse and detection of the reflected signal on an entire frame at once, depth information can be directly obtained in real-time from depth cameras. This makes the DIBR problem less computationally intense and more robust than other techniques. Furthermore, depth information helps to significantly reduce the required number of cameras and transmitting data for 3D-TV. Thus, depth measurements will likely become ubiquitous in visual communication as they provide a perfect complementary information to the traditional color measurements in capturing 3D scenes.

Depth inputs can be considered as monochrome images and therefore encoded using classical codecs like MPEG-2, H.264/AVC, or JPEG-2000 with only minor modifications [1], [3]. However, the transforms used to decorrelate the data, for instance Discrete Wavelet Transforms (DWT) or Discrete Cosine Trans-

forms (DCT), loose their decorrelation power along edges. This issue is paramount with depth images, which tend to exhibit much sharper edges than regular images. Context-adaptive entropy coding [7] can limit the Rate-Distortion (RD) impact of poor data decorrelation, but our experimental results shall show that designing better decorrelation transforms can still lead to significant RD gains.

A large research effort has been spent to design more efficient image transforms. Wavelet footprints [8] reduced scale correlation by representing edges explicitly, through their locations and a set of wavelet coefficients. Geometric wavelets [9]–[13] reduced space correlation by taking into account the 2D nature of images using non-separable wavelets. In [14] for example, major RD gains over JPEG-2000 were reported on depth images using a representation based on platelets [11].

Several issues limit the appeal of geometric wavelets and wavelet footprints for coding of depth images. These methods tend to have large computational requirements due to their reliance on complex RD optimizations [10], [11], [13], [14] instead of established tools like fast filter banks, quantizers, and entropy coders in common codecs. Some of these methods also tend to rely on redundant data representations [8], [9], [12], which reduces RD performances.

Unlike typical color or grayscale images, depth images do not contain texture. In fact, as seen in Figure 1(a), depth images are piecewise smooth with distinct edge around object boundaries that give sharp discontinuity in depth measurements. Therefore, unlike for traditional color and grayscale images, edges in depth images can be detected robustly and accurately (assuming that depth images have good quality). Moreover, for many image-based rendering algorithms with depth information, the accuracy of input depth image around object boundaries is very critical for the quality of the synthesized novel views [15]. A slight shift of an edge point in the depth image would lead to a large distance change in 3D of the corresponding point on the object boundary.

These observations lead us to consider *explicitly encoding locations of edges in depth images*. With encoded edge information, we can use the Shape-Adaptive Discrete Wavelet Transform (SA-DWT) [16] to obtain invertible representation with small coefficients in both smooth regions and along encoded edges. The result is a simple, yet highly effective codec for depth images or depth-color image pairs.

The remainder of the article is organized as follows. Section II presents an overview of our proposed codec. Section III presents the SA-DWT based on lifting with a novel border extension scheme. Section IV details the encoding of edges with a novel rate-constrained edge detection. Section V presents experimental results. Some preliminary results have been presented in [17], [18].

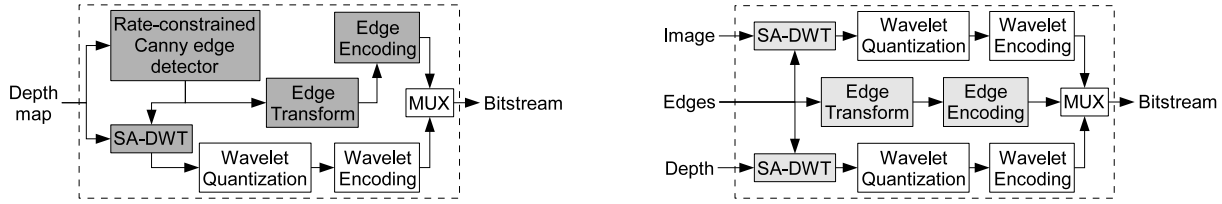


Fig. 2. Overview of the proposed encoder. It explicitly encodes the locations of the main edges and applies a Shape-Adaptive Discrete Wavelet Transform (SA-DWT) to the depth and color images (gray boxes). (a) Depth only. (b) Depth and Color.

II. Proposed Codec

Figure 2(a) shows our proposed codec for depth images. The proposed codec takes advantage of the Shape-Adaptive Discrete Wavelet Transform (SA-DWT) [16] implemented by lifting [7], [19]. Lifting provides a procedure which is fast, in place, simple to implement, and trivially invertible. The locations of the main edges are encoded explicitly and the regions on opposite sides of these edges are processed independently, effectively preventing wavelet bases from crossing edges. As a result, the transform generates small wavelet coefficients both in smooth regions and along the encoded edges.

In order to handle signals with finite-domain, SA-DWT schemes extend the signals using usually zero-padding, periodic extension, or symmetric extension [20]. As a consequence, the transform looses vanishing moments near edges and tends to generate larger high-pass coefficients there. Boundary wavelets have been proposed to address this issue [21]. Here, we introduce a novel scheme which only adds trivial computations to the lifting scheme: a short linear filter is applied along edges, which provides a polynomial extrapolation of the signal with arbitrary order and results in transforms with as many vanishing moments near edges as aways from them.

In our proposed codec, depth edges are detected using a novel rate-constrained version of the Canny edge detector [22]. The proposed edge detector replaces the two hysteresis thresholds of the original detector by a series of decreasing thresholds and associated edge significance bitplanes. These biplanes serve the same goal as wavelet significance bitplanes in codecs based on SPIHT and EBCOT [7]: they allow the most important edges to be encoded first, which leads to higher RD performances. Edges are encoded using a simple differential Freeman chain code [23].

Figure 2(b) shows our extended codec for the case of depth and color images. This is the a typical setting of DIBR representation [3] with an example given in Figure 1. Looking at the example pair of depth-color images, we notice a key source of redundancy is the correlation between *locations of edges*

in these two images. Indeed, 3D scenes are usually made of objects with well-defined surfaces, which, by projection onto the camera image plane, create edges at the same locations in the depth map and the image. The explicit edge coding allows the codec to share the edge information between depth and color images, which reduces their joint redundancy. With the encoded edges, the SA-DWT is applied to both the depth and color images.

Our proposed codecs amount to simple modifications of existing algorithms. As a result, they benefits from the large body of existing work on wavelet-based codecs and edge detectors. Compared to the standard wavelet codec, the increase in complexity is only due to the additional edge detection and coding. Moreover, data redundancy is limited to the locations of edges encoded explicitly. As experimental results shall show, the proposed modifications lead to significant PSNR gain of more than 5 dB for depth images and significant better visual quality for synthesized novel-view images on the Middlebury dataset [24].

III. Lifing-based Shape-Adaptive Wavelet Transform

A. SA-DWT

SA-DWT [16] was originally developed to handle images with arbitrary shapes, not limited to rectangles like the regular DWT. Away from the shape boundary, SA-DWT and DWT are equivalent: they apply similar separable filtering and downsampling, as shown in Figure 3(a) to obtain a multi-resolution representation. SA-DWT differs from DWT along the shape boundary: pixels outside the shape are considered missing and are extrapolated from pixels inside the shape, using symmetric extension for instance. This amounts to modifying the wavelet bases so that they do not cross the shape boundary. This way, SA-DWT does not create large wavelet coefficients along the shape boundary.

In this context, boundaries are defined as curves separating the inside from the outside of a shape. Here, we propose to extend this definition to any curve, be it at the border of the shape or inside it. This extended definition shall allow us to process opposite sides of depth edges independently, in the same way classical SA-DWT processes the inside and outside of a shape independently.

Figure 5 shows how processing independently opposite sides of edges can be beneficial. A piecewise-polynomial signal, akin to a 1D depth map, is transformed by the DWT and the SA-DWT developed in this section. The SA-DWT clearly generates much fewer non-zero wavelet coefficients around edges than the DWT, which leads to significant bitrate savings. The SA-DWT has the disadvantage of requiring an overhead to code the edge location. However, experiments shall show that this overhead is more than compensated by the bitrate savings.

B. Conventional Lifting

We rely on separable lifting [7], [19] to implement the SA-DWT. Since the 2D transform is obtained by applying two 1D transforms, we only study the latter. The lifting scheme, as shown in Figure 3(b) splits the samples into two cosets, one with samples at odd locations and the other with samples at even locations. Each coset is modified alternatively using a series of "lifting steps" called predicts and updates. In general, any wavelet filter bank with FIR filters can be *algebraically* factorized [by applying the Euclidean algorithm to the polyphase components of filter pair $H_0(z)$ and $H_1(z)$] into a finite sequence of lifting steps, each uses a one or two taps filters [25].

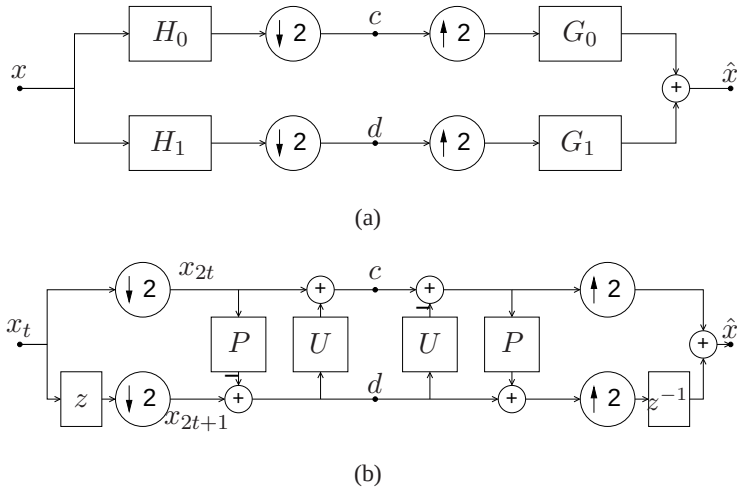


Fig. 3. (a) The two-channel filter bank that is the building block of the discrete wavelet transform. The same filter bank is applied iteratively at the lowpass output c. (b) The lifting scheme with one pair of predict (P) and update (U) steps.

For simplicity, we only consider liftings modifying each sample based on its two neighbors. This case is sufficient to describe the two main wavelets in image processing, namely the 5/3 and 9/7 [7]. More specifically, these liftings begin by modifying the odd coset by a so-called "predict" step, which transforms a signal x into a signal y such that

$$\begin{cases} y_{2t} = x_{2t}, \\ y_{2t+1} = x_{2t+1} + \lambda_{2k}(x_{2t} + x_{2t+2}), \end{cases} \tag{1}$$

where t denotes the sample locations, $2k$ the step number and λ_{2k} a weight. The even coset is modified by a so-called "update" step, which transforms a signal x into a signal y such that

$$\begin{cases} y_{2t} = x_{2t} + \lambda_{2k+1}(x_{2t-1} + x_{2t+1}), \\ y_{2t+1} = x_{2t+1}. \end{cases} \tag{2}$$

The above process can be extended into multiple alternations of "predict" and "update" steps [19]. Lifting ends by a scaling which transforms a signal x into a signal y such that

$$\begin{cases} y_{2t} = K_0 x_{2t}, \\ y_{2t+1} = K_1 x_{2t+1}, \end{cases} \tag{3}$$

where K_0 and K_1 are two weights. At this point, the odd coset contains the high-pass coefficients and the even coset the low-pass ones. Multiresolution lifting is obtained by repeating the process on the low-pass coefficients.

Figure 4 shows a graphical example of these lifting steps. After the final update step, the "odd" coset contains the high-pass coefficients and the "even" coset the low-pass ones. Weights λ for the 5/3 and 9/7 wavelets are given in Table I(a). A key advantage of lifting is the trivial invertibility of its equations.

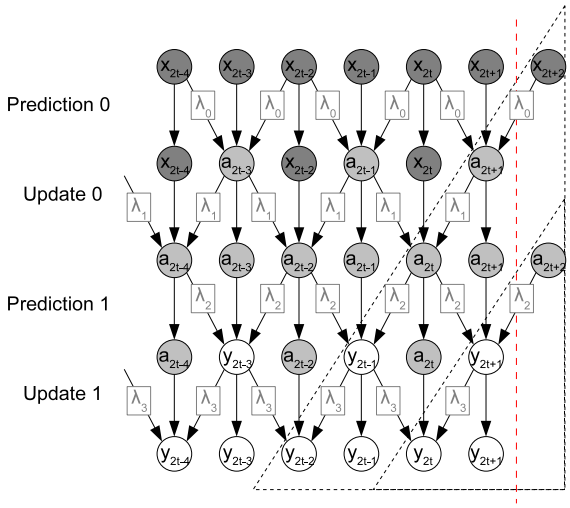


Fig. 4. The four lifting steps associated with a 9/7 wavelet, which transform the signal x first into a and then into y. The values x_{2t+2} and a_{2t+2} on the other side of the edge (dashed red line) are extrapolated. They have dependencies with the values inside the two dashed triangles.

C. Lifting with Polynomial Extrapolation

The key reason that wavelets can efficiently represent piecewise-smooth signals is because of their *vanishing moment* property [26]. Specifically, a wavelet transform is said to have N vanishing moments if the analysis highpass filter $H_1(z)$ in Figure 3(a) can be factored as $H_1(z) = (1 - z^{-1})^N R(z)$. The factor $(1 - z^{-1})^N$ means that any polynomial signal of degree less than N will be zeroed out by filtering with $H_1(z)$ in the highpass channel. For a piecewise-smooth signal, away from the discontinuities it can

(a) Examples of lifting weights [7].

	5/3	9/7
λ_1	-0.50	-1.586134342
λ_2	0.25	-0.052980118
λ_3	–	0.882911075
λ_4	–	0.443506852
K_0	1.224744871	1.139764010
K_1	0.847791247	0.887276732

(b) Examples of extrapolation filters.

N	c	ext.
0	–	zero
1	[1]	sym.
2	[2 -1]	linear
3	[3 -3 1]	quad.
4	[4 -6 4 -1]	cubic

TABLE I

LIFTING WEIGHTS (A) AND EXTRAPOLATION FILTERS (B) FOR THE 5/3 AND 9/7 WAVELETS.

be *locally approximated* by a polynomial signal and thus compactly supported wavelets with enough vanishing moments would result in small wavelet coefficients. The 9/7 and 5/3 wavelets have 4 and 2 vanishing moments, respectively.

Therefore, in the lifting scheme (1) and (2) can be applied when the three used consecutive input samples – re-denoted by x_{t-1}, x_t, and x_{t+1} – are not separated by edges. When edges are present, samples belonging to the sides of the edges not including x_t are considered missing and their values need to be extrapolated.

Let us consider the case where sample x_{t+1} needs to be extrapolated; a similar reasoning applies to x_{t-1}. To maintain simple invertibility, the sample x_{t+1} is extrapolated by a linear combination of nearby samples which are on the same side as x_t and belong to the same coset as x_{t+1}. In other words, we simply modify the predict or update operators P and U in Figure 3(b) around the edges to avoid mixing samples across the edges.

More precisely, we consider extrapolated filters of the form

$$x_{t+1} = \sum_{n=0}^{N-1} c_n x_{t-1-2n} \tag{4}$$

where c is a real vector of length N. The order N needs to be less than the distance to the closest edge on the left side of sample x_t and thus should be chosen as small as possible. In a special case where sample x_t is surrounded by two edges, then we set $N = 0$, $x_{t+1} = x_{t-1} = 0$, and the value of x_t is left unchanged by (1) or (2). Higher values of N allow extrapolations whose associated boundary wavelets have higher moments. The following result provides a closed form solution for the desired extrapolation

filter (4).

Proposition 1: Suppose that the input signal up to the sample at t is a polynomial signal of degree less than N. Also suppose that the used lifting scheme corresponds to an infinite-domain wavelet transform with N vanishing moments. Let the extrapolation filter for computing required samples at $t+1$ during the lifting steps as in (4) be

$$C(z) = z - z(1 - z^{-1})^N. \tag{5}$$

Then the resulting boundary wavelet transform (up to the sample at t) generates all zero wavelet coefficients.

Proof: First, observe that uniform sampling and linear combination of a polynomial of degree less than N also result in polynomials of degree less than N. It follows that for the given input signal, the output of each lifting step (in both the even and odd cosets) up to the sample at t are polynomials of degree less than N. In particular, the sequence $\{x_{t-1-2k}\}_{k\geq 0}$ is a polynomial signal of degree less than N. Thus this polynomial is uniquely defined by N samples $\{x_{t-1-2k}\}_{k=0,1,\ldots,N-1}$. Furthermore, the extrapolated sample x_{t+1} would belong to this polynomial signal if and only if $N+1$ samples $\{x_{t-1-2k}\}_{k=-1,0,1,\ldots,N-1}$ belong to a polynomial of degree less than N.

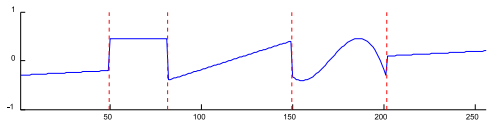
Indeed, the extrapolation in (4) implies

$$x_{t+1} - \sum_{n=0}^{N-1} c_n x_{t-1-2n} = 0. \tag{6}$$

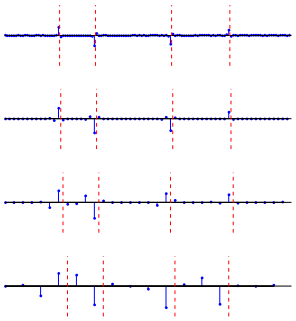
The left hand side of (6) amounts to filtering the sequence $\{x_{t-1-2k}\}_{k=-1,0,1,\ldots,N-1}$ with the filter $A(z) = 1 - z^{-1}C(z)$. With $C(z)$ given in (5) we have $A(z) = 1 - z^{-1}[z - z(1-z^{-1})^N] = (1-z^{-1})^N$. And thus $A(z)$ annihilates the sequence $\{x_{t-1-2k}\}_{k=-1,0,1,\ldots,N-1}$ only if it belongs to a polynomial of degree less than N.

Therefore, the extrapolated sample x_{t+1} by (4) at each required lifting step is exactly equal to the output as if the original input signal was extended beyond sample t as a polynomial of degree less than N. Since the infinite-domain DWT has N vanishing moments, all the high-pass coefficients of the SA-DWT are zero. ■

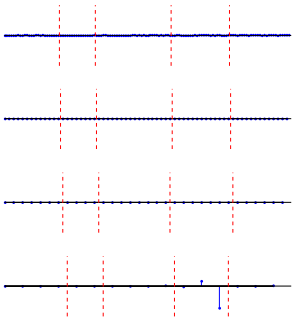
Examples of extrapolation filters c are given in Table I(b). Figure 5 shows an example of a piecewise-cubic polynomial signal transformed by the 9/7 SA-DWT with the cubic extrapolation given in Proposition 1. We see that the resulting SA-DWT gives exactly zero wavelet coefficients everywhere.



(a) Signal made of constant, linear and cubic pieces. Dashed red lines indicate edges.



(b) High-pass bands of a standard 9/7 DWT with symmetric extension



(c) High-pass bands of a 9/7 SA-DWT with cubic extension

Fig. 5. Comparison of standard DWT and SA-DWT. By processing independently opposite sides of edges, the SA-DWT generates null high-pass coefficients everywhere, except where the extrapolation filter does not fit between edges.

IV. Edge Coding and Detection

A. Edge Representation and Encoding

We now turn to the representation of edges. As shown in Figure 6, edge elements, or edgels for short, are either horizontal or vertical. They separate either columns or rows of samples into independent intervals. Edgels, their end-points, and depth samples form three interlaced lattices. Samples are assumed to fall on integer locations. As a consequence, edgel end-points fall on half-integer locations. We represent edgels by two binary edge maps, denoted $e^{(h)}_{s+1/2,t}$ and $e^{(v)}_{s,t+1/2}$, where s and t denote integer spatial locations, and h and v respectively horizontal and vertical directions.

Since the SA-DWT is a multi-resolution transform, each low-pass band must be associated with a pair of edge maps. Let us denote by $e^{(h)}_{s+1/2,t,j}$ and $e^{(v)}_{s,t+1/2,j}$ the two edge maps at resolution j. The pyramid of edge maps is obtained by iteratively downsampling the two edge maps at the finest resolution $j = 0$ using the equations

$$\begin{cases} e^{(h)}_{s+1/2,t,j} = \max\left(e^{(h)}_{2s+1/2,2t,j-1}, e^{(h)}_{2s+1+1/2,2t,j-1}\right) \\ e^{(v)}_{s,t+1/2,j} = \max\left(e^{(v)}_{2s,2t+1/2,j-1}, e^{(v)}_{2s,2t+1+1/2,j-1}\right) \end{cases} \tag{7}$$

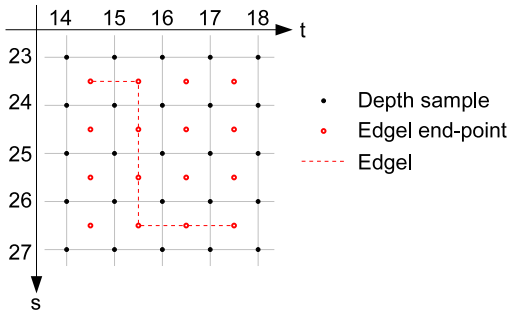
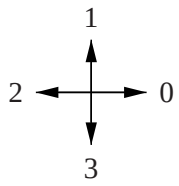


Fig. 6. Example of the lattices of depth samples, edgels, and edgel end-points. Each edgel indicates the statistical independence of the two half rows or half columns of samples it separates.



(a) Edgel directions d_l

	d_0
00	0
01	1
10	2
11	3

(b) Codebook for edgel direction d_0

	Δd_l
00	end of chain
01	+1
10	-1
11	0

(c) Codebook for differential edgel directions Δd_l

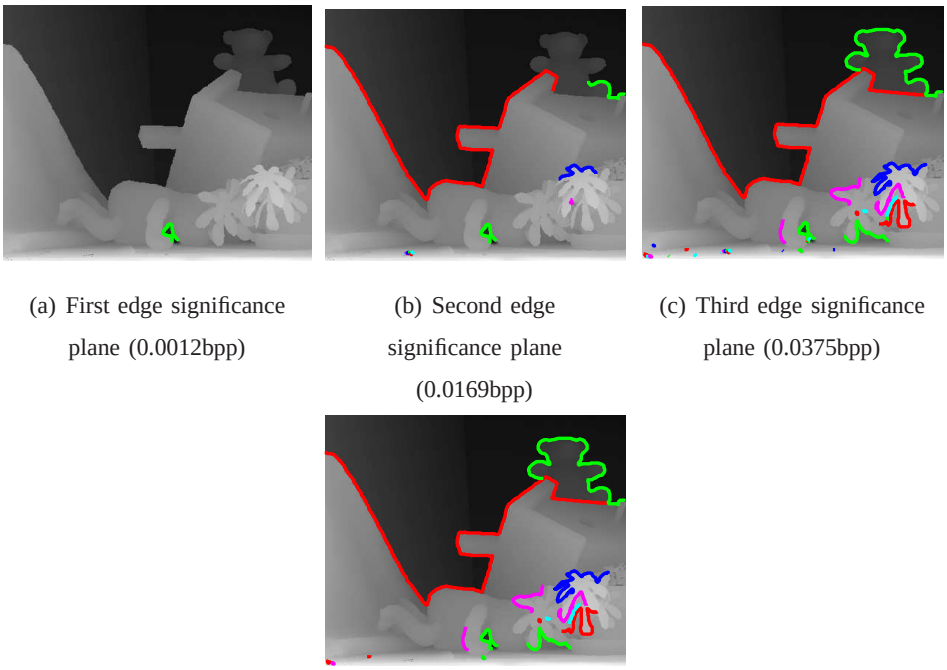
Fig. 7. Edge-chain codebooks. They provide a compact representations, which is simple both to encode and to decode.

The edge maps are encoded at the finest resolution $j = 0$ using a differential Freeman chain code inspired from [27]. Let us assume for now that the edge maps have been estimated and that a set of edge chains has been extracted from these edge maps. Section IV-B shall address this issue. Each edge chain is represented by the location $(s + 1/2, t + 1/2)$ of an initial edgel end-point, along with a series of edgel directions $d_l, l \in [0, L-1]$, where L is the length of the chain. Each direction d_l takes one of the four values shown in Figure 7(a).

First, all but the first edge directions are transformed into differential edge directions Δd_l, such that

$$d_{l+1} = (d_l + \Delta d_l) \mod 4. \tag{8}$$

Differential edge directions take the values -1, 0, or +1, depending on whether the direction remains the same or undergoes a right-angle rotation. The chain is then encoded. For simplicity and compactness, the fixed-length codes shown in Figure 7(b) and (c) are used. The chain code begins with a header containing



(a) First edge significance plane (0.0012bpp)

(b) Second edge significance plane (0.0169bpp)

(c) Third edge significance plane (0.0375bpp)

(d) Final edge chains (0.0347bpp)

Fig. 8. Edge detection with a target bitrate of 0.0375bpp. The multiple edge significance planes let the encoder select the edgels with the most impact on the wavelet coefficients first.

- $\lceil \log_2 h \rceil + \lceil \log_2 w \rceil$ bits for the location of the first edgel end-point,
- 2 bits for the edgel direction d_0, and
- 1 bit for a flag signaling the end of the chain set,

where $\lceil . \rceil$ denotes the ceiling operation, and w and h are respectively the width and height of the edgel end-point lattice. The chain code continues with a series of pairs of bits representing the differential edgel directions Δd_l, and ends with two bits signaling the end of chain. Each edge chain therefore requires

$$R_{chain}(L) = \lceil \log_2 h \rceil + \lceil \log_2 w \rceil + 2L + 3 \text{ bits.} \tag{9}$$

B. Edge Detection

Since edge maps are unknown, the encoder must estimate them from the depth maps. The most popular edge detector is probably the Canny edge detector [22], which proceeds as follows:

- Gaussian smoothing is applied to the image,
- image gradients are computed,

- edgels whose gradient magnitudes are not local maxima are discarded, and
- edgels whose gradient magnitudes fail a two-threshold hysteresis test are discarded.

The two hysteresis thresholds control the number of edgels: decreasing them increases the number of edgels, which tends to increase the edge bitrate and decrease the wavelet bitrate. The objective is therefore to find the optimal thresholds which maximize bitrate savings. However, there does not seem to be an obvious relation between the thresholds and the two bitrates. For instance, although lowering the thresholds tend to increase the edge bitrate, the opposite can happen when chains merge. A brute force search being too computationally expensive, we formulate several approximations and follow a greedy approach to obtain an efficient algorithm.

First, we assume that the bitrate allocation between edges and wavelets is given. We also ignore multi-resolution dependencies between edgels and wavelet coefficients. We only detect edgels at the finest resolution, which is sufficient for depth maps since the edges with the most impact on the total bitrate appear as large C^0 discontinuities at the finest resolution.

Classical wavelet codecs based on EBCOT or SPIHT [7] order wavelet coefficients by decreasing magnitudes, so that the coefficients encoded first are those with the largest impact on RD. Coefficient ordering is achieved by defining a series of decreasing thresholds and a set of associated significance planes. From this point of view, the two thresholds of the Canny edge detector can be seen as the beginning of such a series, applied to image gradients instead of wavelets.

Therefore, we define a series of strictly decreasing thresholds $T_n, n \geq 0$ and associate each threshold T_n with an edge significance plane, which includes all the edgels whose gradient magnitude is larger or equal to T_n. The algorithm then proceeds by extracting edgels from these significance planes in order, until the target bitrate is reached.

The hysteresis test of the Canny edge detector consists in imposing a stronger constraint on the creation of new chains than on the propagation of existing ones, which improves the robustness of the algorithm against noise. We preserve this feature by creating new chains using edgels from one significance plane and propagating chains using edgels from the next significance plane.

The Canny edge detector relies on Gaussian smoothing to reduce the effects of Gaussian noise and optical blur found in images. Since depth maps do not suffer from this kind of noise, we remove the Gaussian smoothing stage from the algorithm and rely on the kernel $[-1\ 1]$ to compute derivatives. This reduces computational complexity and increases the precision of edge locations, which is required to avoid large wavelet coefficients. Depth maps tend however to contain spurious noise. Its impact is reduced by removing short edge chains during a post-processing step.

The proposed algorithm proceeds as follows. First, depth-map derivatives $\delta^{(h)}$ and $\delta^{(v)}$ are computed using

$$\begin{cases} \delta^{(h)}_{s+1/2,t} = x_{s+1,t} - x_{s,t}, \\ \delta^{(v)}_{s,t+1/2} = x_{s,t+1} - x_{s,t}. \end{cases} \tag{10}$$

Edgels whose derivatives are not local maxima are then discarded. Non-maximality is tested by comparing the derivative of each edgel with those of its two neighbors sharing the same direction:

$$\begin{cases} \delta^{(h)2}_{s+1/2,t} \geq \max\left(\delta^{(h)}_{s+1/2,t}\delta^{(h)}_{s+1/2-1,t}, \delta^{(h)}_{s+1/2,t}\delta^{(h)}_{s+1/2+1,t}\right), \\ \delta^{(v)2}_{s,t+1/2} \geq \max\left(\delta^{(v)}_{s,t+1/2}\delta^{(v)}_{s,t+1/2-1}, \delta^{(v)}_{s,t+1/2}\delta^{(v)}_{s,t+1/2+1}\right). \end{cases} \tag{11}$$

The initial significance threshold T_0 is set to

$$T_0 = \max\left(\max_{\substack{0\leq s<h \\ 0\leq t\leq w}} \left|\delta^{(h)}_{s+1/2,t}\right|, \max_{\substack{0\leq s\leq h \\ 0\leq t<w}} \left|\delta^{(v)}_{s,t+1/2}\right|\right), \tag{12}$$

and the following steps are iterated until the target bitrate is reached. During the n^{th} iteration,

- existing chains are propagated by adding connected edgels with $\delta \geq T_n/2$,
- existing chains are merged when they become connected,
- new chains are created from still-unconnected edgels with $\delta \geq T_n$,
- new chains are propagated and merged like existing chains, and
- the significance threshold is halved, that is $T_{n+1} = T_n/2$.

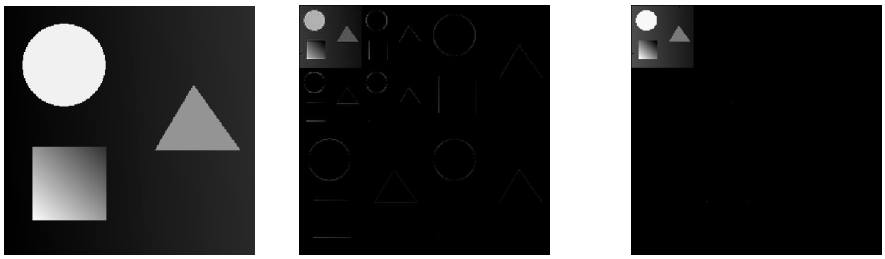
Finally, small chains are discarded.

V. Experimental Results

A. Synthetic Piecewise-Smooth Images

We begin by comparing the behavior of standard and SA DWT on the synthetic image shown in Figure 9(a). Figures 9(b) and 9(c) represent the coefficients generated by respectively 5/3 standard DWT with symmetric extension and 5/3 SA-DWT with linear extension. Since the image is piecewise linear, the high-pass coefficients of the standard DWT are zero everywhere, except around edges where they take large values. Moreover, in the low-pass band the standard DWT tends to create ringing artifacts around edges.

SA-DWT, on the other hand, does not suffer from such issues. High-pass coefficients are zero both in smooth regions and along edges. Larger values are only generated around the corners of the triangle, where parallel edges are only one pixel apart. Furthermore, the low-pass band is free of ringing artifacts and sharp edges can be observed, which underlines the absence of over-smoothing.



(a) Synthetic image (b) Standard DWT (c) SA-DWT

Fig. 9. A synthetic image transformed by standard and SA DWT. SA-DWT generates virtually no high-pass coefficients and preserves sharp discontinuities free of ringing artifacts in the low-pass band.

B. Depth Images Only

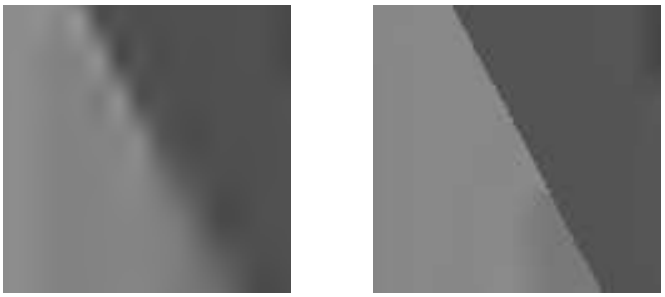
We also present experimental results on the four depth images of the Middlebury dataset [24], shown in Figure 10. In-painting was used to estimate the depth in regions with missing data. The codec relies on a five-level decomposition based on 9/7 SA-DWT with symmetric extension [7]. Wavelet coefficients are quantized and encoded using the implementation of SPIHT provided by the QccPack library [28]. The bitrate allocation between edges and wavelets coefficients can be varied, the case of zero bitrate allocated to edges corresponding to a codec based on standard DWT.

(a) Teddy (b) Cones (c) Tsukuba (d) Venus

Fig. 10. The Middlebury dataset encoded at 0.05bpp with 30% allocated to edges. The proposed codec reconstructs sharp edges even at very low bitrates.

Figure 11 displays a zoom on low-bitrate reconstructions of the Teddy depth image. The rate constraint prevents enough non-zero wavelet coefficients from being encoded to accurately represent the discontinuity. This leads the standard DWT to generate ringing artifacts and over-smoothing. The SA-DWT, on the other hand, even at this low bitrate can reconstruct sharps and avoids Gibbs artifacts along edges.

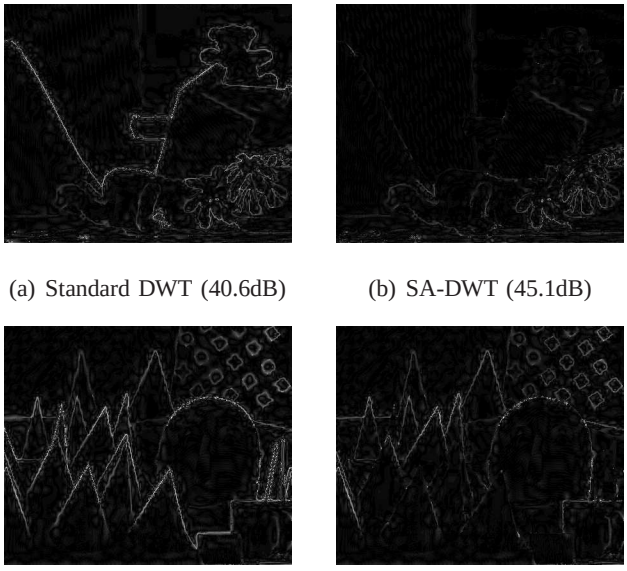
The ability of SA-DWT to efficiently represent sharp discontinuities is confirmed in Figure 12, which compares the reconstruction errors of standard DWT and SA-DWT on Teddy and Cones with a total



(a) Standard DWT (b) SA-DWT

Fig. 11. Zoom on low-bitrate reconstructions of the Teddy depth image. The edge-location information lets SA-DWT reconstruct sharp edges (b), which cannot be obtained by standard DWT (a).

bitrate of 0.1bpp. SA-DWT generates much smaller errors along explicitly encoded edges. SA-DWT and standard DWT have similar errors along edges whose locations were not explicitly encoded.



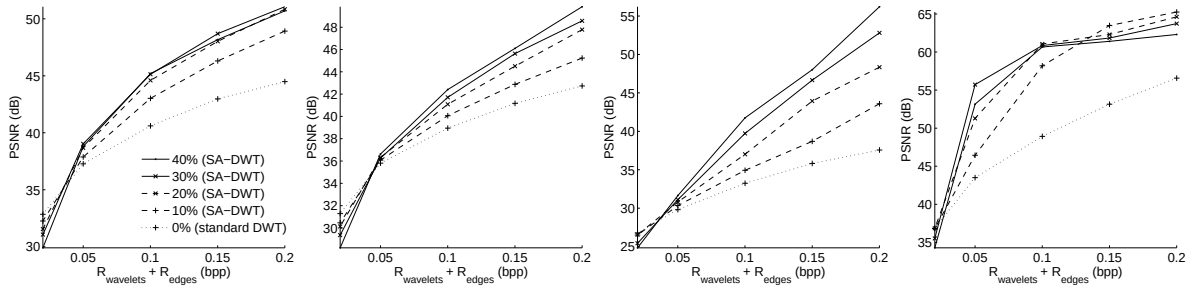
(a) Standard DWT (40.6dB) (b) SA-DWT (45.1dB)

(c) Standard DWT (38.9dB) (d) SA-DWT (41.7dB)

Fig. 12. Reconstruction errors for Cones [(a) and (b)] and Teddy [(c) and (d)] at 0.1bpp. SA-DWT [(b) and (d)] generates much less errors than standard DWT [(a) and (c)] along explicitly encoded edges.

Figure 13 presents the RD performances of the codec, with between 0% (standard DWT) and 40% of the total bitrate allocated to the edge information. RD performance increases with the amount of bitrate allocated to edges. On Teddy and Venus a saturation is observed in the 30%-40% range, where performance starts decreasing at higher bitrates. SA-DWT proves to be superior to standard DWT over

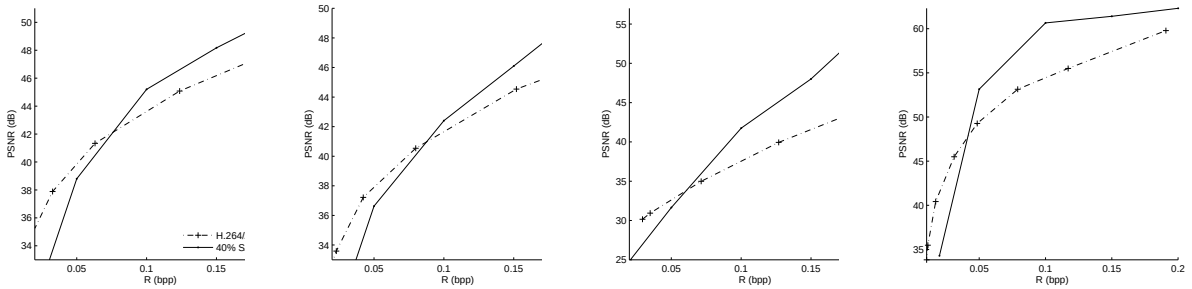
the vast majority of the range of bitrates considered and provides major PSNR gains. For instance, SA-DWT with a 30% bitrate allocation provides up-to 6.58 dB on Teddy, 5.84 dB on Cones, 15.2 dB on Tsukuba, and 7.16 dB on Venus.



(a) Teddy (b) Cones (c) Tsukuba (d) Venus

Fig. 13. RD performances with between 0% and 40% of the total rate allocated to edges. The 0% case corresponds to classical DWT. SA-DWT provides large performance gains over the entire dataset.

Figure 14 compares the RD performances of the proposed codec with those of the H.264/AVC codec JM 16.1 in intra mode. Bitrate allocation was set to 40% for edges in the proposed codec. The proposed codec provides significant gains over H.264/AVC on all four images for bitrates over 0.07bpp.



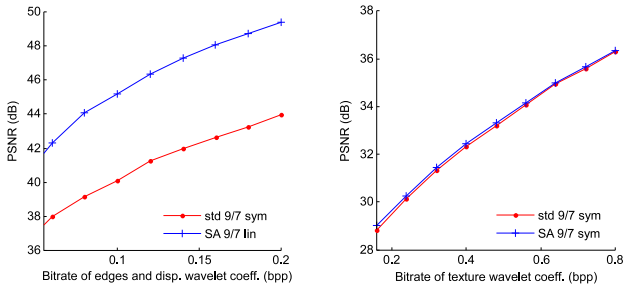
(a) Teddy (b) Cones (c) Tsukuba (d) Venus

Fig. 14. RD performances of the proposed codec (solid lines) and the H.264/AVC codec in the intra mode (dash lines). The proposed codec significantly outperforms the H.264/AVC codec at birates greater than 0.07bpp.

C. Depth and Color Images

We present experimental results on the Teddy set with both depth and color images. For simplicity, only the luma channel of the color image is considered. We again compare the performances of two codecs: one based on the DWT and the other on the proposed SA-DWT with explicit edge coding. Both

codecs perform a five-level decomposition and rely on the same quantizer and entropy coder, provided by the SPIHT implementation of the QccPack library. They also rely on the same 9/7 wavelet for the transform, which is the main wavelet in JPEG2000 [7]. Following [29], both codecs allocate 20% of the bitrate to the depth.



(a) Depth (b) Color

Fig. 15. Rate-distortion performances of standard and shape-adaptive wavelets. The latter gives PSNR gains of up to 5.46dB on the depth map and 0.19dB on the image.

The two codecs differ in their handling of edges: the DWT-based codec uses the standard 9/7 wavelet with symmetric extension, denoted “std9/7sym,” while the SA-DWT-based codec uses the shape-adaptive 9/7 wavelet with linear extension, denoted “SA9/7lin,” for the depth map and the shape-adaptive 9/7 wavelet with symmetric extension, denoted “SA9/7sym,” for the image. The SA-DWT-based codec also includes an edge codec, as shown in Figure 2. The experiments are based on the edge map shown in Figure 1, which has a bitrate overhead of 0.015bpp. To give a general idea of the speed of the proposed codec, on a Intel Pentium D at 2.8GHz, our mixed Matlab/C implementation takes 0.02s to perform the SA-DWT and 0.31s to perform the chain encoding.

Figure 15(a) compares the RD performances of the two codecs for the depth image. The bitrate consists in the wavelet coefficients, along with the edges in the shape-adaptive case. The figure shows that the edge overhead is more than compensated by the reduced entropy of the wavelet coefficients. This leads to PSNR gains over the whole bitrate range, achieving up to 5.46 dB.

Figure 15(b) compares the RD performances of the two codecs for the color image. The bitrate is made of the wavelet coefficients in both cases, the edge overhead having been accounted for in the depth bitrate. The figure shows PSNR gains over the whole bitrate range, achieving up to 0.19 dB.

Finally, Figure 16 compares the synthesized novel view images from decoded DIBR representation using coding parameters as in Figure 10 (0.05 bpp for depth images, with 30% allocated to edges).

(a) Standard DWT (b) SA DWT

Fig. 16. Synthesized novel view from decoded DIBR using the standard DWT codec and the proposed SA DWT codec. Notice the visual improvement along edges in the novel view synthesis with the proposed codec.

VI. Conclusion

We have presented a novel depth image codec for free-viewpoint 3D-TV. The codec begins by extracting and encoding depth edges. It relies on an ordered set of edge significance planes to extract edges with the largest impact on bitrate first. Thanks to SA-DWT, the codec then takes advantage of this information by treating opposite sides of edges as independent. This prevents wavelet bases from crossing edges, which leads to small wavelet coefficients both inside smooth regions and along explicitly encoded edges. SA-DWT is implemented using lifting to obtain an efficient algorithm. PSNR increases of 5 dB and more were observed over the entire Middlebury dataset.

The proposed scheme is also extended to jointly encode depth and color images for DIBR by sharing the same edge information to reduce redundancies between the two. The gain for depth image is again significant, while the gain for color image is modest. This is partly because color edges tend to be less sharp than deph edges, due for instance to optical blur, which makes SA-DWT less effective. Nevertheless, the shared edge locations in effect register encoded depth and color images around the critical areas of object boundaries. This alignment ensures high quality synthesizing of novel views for free-viewpoint 3D-TV.

This paper has aimed at improving the compression of single images. Although the proposed scheme could be applied as-is to videos, better RD performances are likely to be achieved by taking into account temporal data dependencies in addition to spatial ones.

REFERENCES

[1] C. Fehn, R. Barre, and R. S. Pastoor, "Interactive 3-D TV – concepts and key technologies," *Proc. of the IEEE*, vol. 94, no. 3, pp. 524–538, 2006.

[2] E. Adelson and J. Bergen, "The plenoptic function and the elements of early vision," in *Computational Models of Visual Processing*, M. S. Landy and J. A. Movshon, Eds. Cambridge, MA: MIT Press, 1991, pp. 3–20.

[3] A. Smolic and P. Kauff, "Interactive 3-D video representation and coding," *Proc. of the IEEE*, vol. 93, no. 1, pp. 98–110, 2005.

[4] H.-Y. Shum, S.-C. Chan, and S. B. Kang, *Image-Based Rendering*. New York, NY: Springer-Verlag, 2007.

[5] C. L. Zitnick, S. B. Kang, M. Uyttendaele, and R. Szeliski, "High-quality video view interpolation using a layered representation," in *Proc. SIGGRAPH*, 2004, pp. 600–608.

[6] A. Kolb, E. Barth, R. Koch, and R. Larsen, "Time-of-Flight Sensors in Computer Graphics," in *Proc. Eurographics (State-of-the-Art Report)*, 2009.

[7] D. Taubman and M. Marcellin, *JPEG2000: Image Compression Fundamentals, Standards and Practice*. New York, NY: Springer-Verlag, 2001.

[8] P. Dragotti and M. Vetterli, "Wavelet footprints: Theory, algorithms and applications," *IEEE Trans. on Signal Proc.*, vol. 51, pp. 1306–1323, 2003.

[9] M. N. Do and M. Vetterli, "The contourlet transform: an efficient directional multiresolution image representation," *IEEE Trans. on Im. Proc.*, vol. 14, pp. 2091–2106, 2005.

[10] G. Peyre and S. Mallat, "Surface compression with geometric bandelets," in *Proc. SIGGRAPH*, 2005, pp. 601–608.

[11] R. M. Willett and R. Nowak, "Platelets: a multiscale approach for recovering edges and surfaces in photon-limited medical imaging," *IEEE Trans. on Medical Imaging*, vol. 22, pp. 332–350, 2003.

[12] M. B. Wakin, J. K. Romberg, H. Choi, and R. G. Baraniuk, "Wavelet-domain approximation and compression of piecewise smooth images," *IEEE Trans. Image Proc.*, vol. 15, pp. 1071–1087, May 2006.

[13] R. Shukla, P. L. Dragotti, M. N. Do, and M. Vetterli, "Rate-distortion optimized tree-structured compression algorithms for piecewise polynomial images," *IEEE Trans. on Im. Proc.*, vol. 14, pp. 343–359, 2005.

[14] Y. Morvan, D. Farin, and P. H. N. de With, "Depth-image compression based on an R-D optimized quadtree decomposition for the transmission of multiview images," in *Proc. ICIP*, 2007.

[15] H. T. Nguyen and M. N. Do, "Error analysis for image-based rendering with depth information," *IEEE Trans. Image Proc.*, vol. 18, pp. 703–716, Apr. 2009.

[16] S. Li and W. Li, "Shape-adaptive discrete wavelet transforms for arbitrarily shaped visual object coding," *IEEE Trans. on Circuits and Sys. for Video Tech.*, vol. 10, pp. 725–743, 2000.

[17] M. Maitre and M. N. Do, "Joint encoding of the depth image based reprensentation using shape-adaptive wavelets," in *Proc. ICIP*, 2008.

[18] ——, "Shape-adaptive wavelet encoding of depth maps," in *Proc. Picture Coding Symposium*, 2009.

[19] W. Sweldens, "The lifting scheme: A construction of second generation wavelets," *SIAM Journal on Mathematical Analysis*, vol. 29, pp. 511–546, 1997.

[20] S. Mallat, *A Wavelet Tour of Signal Processing*. Burlington, MA: Academic Press, 1999.

[21] A. Cohen, I. Daubechies, and P. Vial, "Wavelet bases on the interval and fast algorithms," *J. of Appl. and Comput. Harmonic Analysis*, vol. 1, pp. 54–81, 1993.

[22] J. Canny, "A computational approach to edge detection," *IEEE Trans. on PAMI*, vol. 8, no. 6, pp. 679–698, November 1986.

[23] H. Freeman, "On the encoding of arbitrary geometric configurations," *IRE Trans. of Electronic Computers*, vol. 10, no. 2, pp. 260–268, 1961.

[24] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Int. J. of Comp. Vis.*, vol. 47, no. 1–3, pp. 7–42, 2002.

[25] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *J. Fourier Anal. Appl.*, vol. 4, no. 3, pp. 245–267, 1998.

[26] M. Vetterli, "Wavelets, approximation and compression," *IEEE Signal Proc. Mag.*, pp. 59–73, Sep. 2001.

[27] Y. K. Liua and B. alik, "An efficient chain code with huffman coding," *Pattern Recog.*, vol. 38, pp. 553–557, 2005.

[28] J. E. Fowler, "QccPack: an open-source software library for quantization, compression, and coding," in *Proc. of SPIE Appli. of Digital Im. Proc.*, 2000, pp. 294–301.

[29] M. Maitre, Y. Shinagawa, and M. N. Do, "Wavelet-based joint estimation and encoding of depth-image-based representations for free-viewpoint rendering," *IEEE Trans. Image Proc.*, vol. 17, pp. 946–957, June 2008.