Normal Mesh based Geometrical Image Compression

W. Van Aerschot [a,*], M. Jansen [b,a], and A. Bultheel [a]

[a] *K.U.Leuven, Department of Computer Science, Celestijnenlaan 200A, 3000 Leuven, Belgium*

[b] *K.U.Leuven, Department of Mathematics, Celestijnenlaan 200B, 3000 Leuven, Belgium*

Abstract

Recently the performance of nonlinear transforms have been given a lot of attention to overcome the suboptimal n-terms approximation power of tensor product wavelet methods on higher dimensions. The suboptimal performance prevails when the latter are used for a sparse representation of functions consisting of smoothly varying areas separated by smooth contours. This paper introduces a method creating normal meshes with *non*subdivision connectivity to approximate the nonsmoothness of such images efficiently. From a domain decomposition viewpoint, the method is a triangulation refinement method preserving contours. The method is nonlinear as it depends on the actual image. This paper proposes a normal offset based compression algorithm for digital images. The discretisation causes the transform to become redundant. We further develop a model to encode the obtained coefficients. We show promising rate distortion curves and compare the results with the JPEG2000 encoder.

Key words: normal multiresolution mesh, image compression, piecewise smooth, wavelets

1 Introduction

Over the last decade normal meshes have been studied as a method for approximating smooth curves [5] and surfaces [2, 13, 15, 17]. A normal mesh

* Corresponding author.

Email addresses: ward.vanaerschot@cs.kuleuven.be (W. Van Aerschot), maarten.jansen@wis.kuleuven.be (M. Jansen), adhemar.bultheel@cs.kuleuven.be (A. Bultheel).

is created by successive refinements of an initial mesh, where each refinement step uses information from the target function to create a new mesh and a better approximant. Each refinement step adds an additional vertex (one for each edge) that can be represented by one coefficient, called "normal offset", in a local frame instead of its usual three coordinates.

This paper investigates the application of normal meshes to compress *piecewise* smooth images whose content is dominated by contours. This kind of images will be referred to as *geometrical* images. It is well known that, for this type of images, transform coders using nonredundant bases fall short when creating compact high resolution representations. Let f_n be an element of subspaces $(S_n)_{n\geq 0}$ of a normed space X which can be described by n parameters. We define the best approximation error as $\sigma_X(n) := \inf_{g\in S_n} \|f-g\|_X$. The highest achievable n-term approximation rate using a wavelet transform combined with a nonlinear thresholding equals $\sigma^2_{L_2}(n) = O(n^{-1})$ [8]. Therefore, transform coders like JPEG2000 will perform suboptimally compared to recently developed schemes capable of sparsely representing line discontinuities.

Candès and Donoho [4] presented Curvelets as tight directional frames combined with a nonlinear n-term selection which are able to achieve $\sigma^2_{L_2}(n) = O(n^{-2})$. Contourlets, a filterbank approach was presented in [9] and allow for fast implementations on digital images. Donoho [10] presented Wedgelets as an extension of recursive dyadic partitioning methods where terminal nodes are decorated with so-called wedges. Wedgelets achieve $\sigma^2_{L_2}(n) = O(n^{-2}) + \delta$ on piecewise *constant* objects with boundaries of Hölder[2] regularity, where δ represents the angular resolution of the wedges. On the other hand, we have *adaptive* schemes where the construction of the approximants is driven by the target function. Le Pennec and Mallat [16] proposed bandelets which are based on tensor product of wavelet bases combined with local warping operators (based on geometrical flow) adapted to the edge of the image. Dekel and Leviatan [6] proposed geometric wavelets based on binary partitioning algorithms. The computational complexity, however, makes the scheme less suitable for practical algorithms. Gray-scale images can be also be seen as elevation models. In the area of terrain modelling elevation models are approximated by triangular irregular networks. For piecewise smooth images Demaret et al. [7] present an adaptive thinning strategy capable of preserving edges. The common goal of all these schemes is to partition the image domain into parts of different sizes and shapes. The small parts correspond to image regions that contain much information (e.g. edges, texture) and larger parts correspond to smooth image regions that can be approximated by few parameters given a certain error bound. The normal offset scheme proposed in this paper is in that respect no different from the previous approaches. In contrast to previous methods, however, we do not track boundaries actively but rely on the implicit boundary locating property of normal meshes. This way our algorithm performs significantly faster compared to coarse-to-fine partitioning

algorithms which perform an exhaustive search looking for the best domain dissection over a large set of allowed dissections.

There are many papers involving normal mesh compression for smooth manifold surfaces [11, 12, 13, 15, 17]. Despite the similarities between surface and image compression we cannot take 'off the shelf' normal mesh encoders and apply them to the surface equivalents of a geometrical image. The first major difference concerns the smoothness properties of the target surfaces. While surfaces in computer graphics are smooth, geometrical images are non-smooth: the edges are the information carrying feature. A second major difference – and the actual reason for using a normal offset transform in an image compression algorithm– is the exploitation of normal approximation properties in order to compress. The reason for using normal offsets in surface rendering applications is that they contain almost all geometrical information [13]. Thanks to the minor contribution of the parametrical information, semi-regular meshes – demanding almost no connectivity information – can be utilized. As such, the surface can be approximated by a mesh where each point is defined using a single scalar. This leads to a bonus compression factor of three. In contrast, image grey levels are not triple coordinates in $\mathbb{R}^3$, but single values on a given regular lattice, so there is no bonus to gain in the image case. Futhermore, for piecewise smooth images, irregular meshes have to be used to preserve contours. In this respect, the fundamental reason for using normal offsets in image processing is the edge locating property of the normal search direction.

A first attempt to use normal mesh techniques for image approximation was made by Jansen et al. [14]. Gray scale images are treated as two dimensional functions dominated by geometric structures comparable with terrain models used in geographical information systems. The authors achieve an n-term approximation rate of $\sigma^2_{L_2}(n) = O(n^{-2})$ that is twice as good as wavelet approximations on the studied images. Since approximation and compression are tightly related their results indicate that the normal offset method should be considered for the development of rate-distortion efficient piecewise-smooth image encoders. The focus of this paper is more practical oriented where we focus on rather small n, i.e. far from values where asymptotical theoretical behaviour can be observed. Also results given in an earlier paper [21] indicate that for images of geometrical nature the normal offset decomposition is a promising compression technique.

Unfortunately, the authors of [14] do not address compression. For instance, the multi-resolution model used is non nested and additional side information needs to be stored in order to encode the complex dependency relations between consecutive resolution levels. In addition, the authors do not incorporate a contour preserving triangulation method which explains the rather poor visual results in practice.

This paper focusses on compression issues. In this context, we use nested triangulations for which we have to redefine the normal direction. Additionally we utilize *local* mesh refinement operations, i.e. triangle splits, to create piecewise linear approximations of the smooth trajectories of the contours by triangle edges. The use of local mesh refinements results in tree-structured dependency relations between successive approximations that can be encoded in a straightforward manner.

The main contribution of this paper is the development of a model for the offsets such that they can be encoded efficiently by an entropy coder.

The paper is outlined as follows. Section 2 introduces the class of piecewise smooth images. In Section 3 the proposed normal mesh algorithm is given in detail. This nonlinear transform forms the main component of the encoder. It produces sparse representations of geometrical images. The efficient encoding of the wavelet coefficients (normal offsets) is done by a model based entropy coder and will be explained in Section 4. Finally Section 5 shows rate-distortion curves of the proposed encoder and compares the results with the state of the art JPEG2000 encoder.

2 Piecewise smooth images and the Horizon Class

Since we aim to extract geometrical information, we mainly focus on horizon class $\mathcal{H}$, introduced by Donoho [10], which contains objects are constant except for a smooth boundary (with Hölder regularity $\in (1, 2]$) over the unit square $[0, 1]^2$. Let us first define the Hölder conditions.

- Let $0 < \alpha \leq 1$ we say that $c \in \text{Hölder}^{\alpha}(C_\alpha)$ if

$$|c(x) - c(x')| \leq C_\alpha |x - x'|^\alpha, 0 \leq x, x' \leq 1 \tag{2.1}$$

- Let $1 < \alpha \leq 2$ we say that $c \in \text{Hölder}^{\alpha}(C_\alpha)$ if

$$|c'(x) - c'(x')| \leq C_\alpha |x - x'|^{(\alpha-1)}, 0 \leq x, x' \leq 1 \tag{2.2}$$

  with $c'(x)$ the derivative of $c(x)$.

Definition 2.1 *We define the Horizon class $\mathcal{H}^\alpha$ as follows:*
$\mathcal{H}^\alpha := \{H\}$ *with* $H(x_1, x_2) : [0, 1]^2 \rightarrow \{0, h\}$:

$$\begin{cases} h \text{ if } x_2 \leq c(x_1) \\ 0 \text{ if } x_2 > c(x_1) \end{cases},$$

with $c(x_1) \in$ *Hölder*$^\alpha (C_\alpha) \cap$ *Hölder*$^1 (C_1)$, $\alpha \in (1, 2]$.

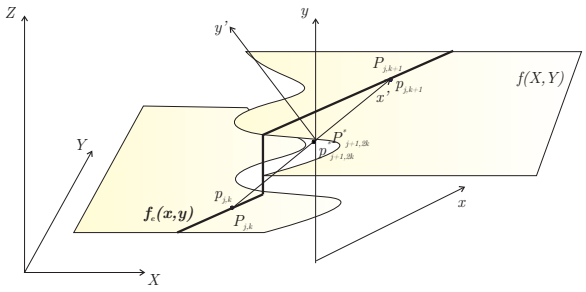


Fig. 1. Global coordinate system XYZ together with local coordinate frames xy and $x'y'$.

Up to a sign value, all image information is located in $c(x)$. For $\mathcal{H}^2$, we will drop the exponent and note $\mathcal{H}$.

3 The normal offset scheme

In this section we state the algorithm that produces successive normal mesh approximations of a given function f defined on $\Omega \in [0,1]^2$. We use the following notation $\mathcal{M}_j := (\mathcal{V}_j, \mathcal{E}_j, \mathcal{F}_j)$ to denote a mesh built of vertices $\mathcal{V}$, edges $\mathcal{E}$ and faces $\mathcal{F}$ at resolution level j. We note Δ_j for the triangulation obtained by projecting $\mathcal{M}_j$ on Ω.

Initially the algorithm starts from an interpolating triangular mesh $\mathcal{M}_0$. From then on the normal offset scheme consists of an iterative application of three steps. The first step, the *prediction* step, constructs additional vertices – prediction points – as linear combinations of surrounding mesh points. The second step, the *correction* step, constructs piercing points as the intersection of rays normal to the coarser mesh, going through those prediction points and the image surface. The third step, the *interconnection* step, adds all piercing points to the set of vertices $\mathcal{V}_j$ forming $\mathcal{V}_{j+1}$. The corresponding mesh $\mathcal{M}_{j+1}$ with edges $\mathcal{E}_{j+1}$ and triangles Δ_{j+1} is constructed by a triangulation of $\mathcal{V}_{j+1}$. These steps are repeated, creating meshes at different resolution levels j until a certain stopping criterion is met.

3.1 The algorithm

We now give a specific implementation of each step the normal offset *refinement* scheme:

(a) **Prediction**: In what follows, we use linear interpolation for the prediction step. Suppose the image f is given as a surface $S_f := (X, Y, f(X,Y))$. In order to obtain an hierarchical edge refinement scheme we consider the

subspace defined as the vertical plane ($\perp XY$) containing edge $e_{j,k} \in \mathcal{E}_j$ with end points $P_{j,k}, P_{j,k+1} \in \mathcal{V}_j$. The part of the image contained in the vertical plane is denoted by f_e. We attach a $2d$ coordinate system to this plane with the x-axis in the XY-plane and the y-axis parallel w.r.t. the Z-axis (see Figure 1). The prediction point $p^*_{j+1,2k+1} = (x^*_{j+1,2k+1}, y^*_{j+1,2k+1})$ on location $2k+1$ and resolution level $j+1$ is defined as:

$$p^*_{j+1,2k+1} := \frac{p_{j,k} + p_{j,k+1}}{2}$$

where $p_{j,k}, p_{j,k+1}$ are the projections of $P_{j,k}, P_{j,k+1}$.

(b) **Correction**: The correction step adds a vector in the direction normal to the coarse mesh to the points predicted in the previous step. The normals $\overrightarrow{n}_{j,k}$, on the coarse mesh such that $\|\overrightarrow{n}_{j,k}\| = 1$ are expressed as

$$\overrightarrow{n}_{j,k} = \begin{bmatrix} n_{[x]j,k} \\ n_{[y]j,k} \end{bmatrix} := [-(y_{j,k+1} - y_{j,k}), x_{j,k+1} - x_{j,k}]/\|p_{j,k+1} - p_{j,k}\|,$$

and represent the direction of the perpendicular bisector on $e_{(p_{j,k+1}, p_{j,k})}$. The normal ray $r_{j+1,2k+1}$ going through $p^*_{j+1,2k+1}$ is defined as:

$$r_{j+1,2k+1}(\gamma) := p^*_{j+1,2k+1} + \gamma\, \overrightarrow{n}_{j,k}$$

For $f_e \in \mathcal{C}$, with $\mathcal{C}$ the class of continuous functions, the correction step calculates

$$\gamma_{j+1,2k+1} := \min \{\gamma \;\; | \;\; r_{j+1,2k+1}(\gamma) = [x, f_e(x)]\}\,. \tag{3.1}$$

The minimum value of γ over a set is taken, since in general the normal ray can pierce f_e more than once.

For $f_e \notin \mathcal{C}$ and r parameterized as $r_{j+1,2k+1}(\gamma) = (x_{j+1,2k+1}(\gamma), y_{j+1,2k+1}(\gamma))$, Eq. (3.1) takes the following form:

$$\begin{aligned} \gamma_{j+1,2k+1} := \min \Bigg\{ \gamma | \quad & \operatorname{sign}\left(\lim_{x_{j+1,2k+1}(\gamma) \to x^+} y_{j+1,2k+1}(\gamma) - f_e(x) \right) \\ & = -\operatorname{sign}\left(\lim_{x_{j+1,2k+1}(\gamma) \to x^-} y_{j+1,2k+1}(\gamma) - f_e(x) \right) \Bigg\}. \end{aligned} \tag{3.2}$$

As such we always obtain a subdivision scheme i.e., $x_{j,k} \leq x_{j+1,2k+1} \leq x_{j,k+1}$. Setting $x_{j+1,2k} := x_{j,k}$ we obtain a monotonically increasing sequence $\mathbf{x}_j = \{x_{j,k}, x_{j,k+1}, x_{j,k+2}, \ldots\}$ on each edge. A new piercing point $p_{j+1,2k+1}$ is inserted between $p_{j,k}$ and $p_{j,k+1}$ in order to form the sequence

$\mathbf{p}_{j+1}$:

$$\begin{aligned} p_{j+1,2k} &\leftarrow p_{j,k} \\ p_{j+1,2k+1} &\leftarrow r_{j+1,2k+1}\left(\gamma_{j+1,2k+1}\right). \\ p_{j+1,2k+2} &\leftarrow p_{j,k+1} \end{aligned}$$

Each slice f_e created by the previous step is gradually approximated by a polyline (a continuous curve composed of several line segments) through the sequence $\mathbf{p}_j = \left\{p_{j,0}, \ldots, p_{j,k} \ldots, p_{j,2^j+1}\right\}$.

(c) **Interconnection**: Each coarse triangle is split into 4 disjunct subtriangles whose projection on the XY-plane form a planar graph. The inner edges of the tessellation are formed from couples from the set of vertices of the coarse triangle and the piercing points. The choice of these inner edges yields four possible triangle splits. The split minimising an error metric is included into the finer resolution mesh. We will use the L_2-error metric between the original surface S_f and the approximating mesh $\mathcal{M}_j$ in the remainder of the paper.

The recursive partitioning, as constructed by the interconnection step, can be represented by a quadtree $\mathcal{T}_j$ where each node t corresponds to a triangle $\in \cup_{k=0}^{j}\Delta_k$. In this respect the refinement can be seen as tree growing algorithm. In a later section we will prune this tree to find almost optimal tilings of Ω.

Remark 3.1 *We emphasize some major differences where our method distinguishes itself from the method proposed in [14]. We restrict the rays emanating from the prediction point to lie in a plane perpendicular to the XY plane, still forming a right angle ($\angle = \frac{\pi}{2}$) with an edge of the coarser mesh. Also the choice of the* triangulation method *is crucial for approximation and compression performance. Despite the theoretical approximation rate of $O(n^{-1})$ using a Delaunay retriangulation it is not particulary suited for compression. The edges defining the local frames for the piercing points can disappear in higher resolution levels. This requires a lot of bookkeeping when we want to reconstruct the compressed image from thresholded and quantised coefficients. Even vertex connectivity can change when the triangulation method is sensitive to perturbations in the vertex positions due to a quantisation step on the normal coefficients. This further complicates the compression stage. Furthermore a lot of exception handling has to be done to avoid triangle fold overs since the normal direction does not guaranty a proper parametrisation. We therefore opt for a local triangulation scheme equipped with a certain amount of flexibility to preserve contours. The level-to-level mesh refinement relations can be captured by tree structures.*

Consider a digital image obtained from sampling an analogue grayscale image. A *digital* grayscale image is defined as a regular tessellation of *pixels*, disjunct rectangular areas having a gray value. The gray values attached to a pixel take on a discrete value. We use the term *discrete* edge for the set of adjacent pixels connecting two pixels. The collection of pixels is generated by a line rasterisation algorithm $\mathcal{B}$ [1] taking the two end pixel locations as input giving back an array of in between pixel locations. The number of pixels generated by $\mathcal{B}(P_{j,k}, P_{j,k+1})$ is represented by $L_{j,k}$. We use the notation $\mathcal{B}_{[X]}(P_{j,k}, P_{j,k+1})_i, \quad i \in [0 \ldots L_{j,k} - 1]$ to indicate the X-value of the ith edge pixel. The pixels at both ends of a digital edge are the digital counterparts of a vertex. As such discrete edges have a nonzero area, just as discrete points (namely pixels). Accordingly three discrete edges – having pairwise an end point in common– together with the pixels residing in the interior define a *discrete* triangle.

In this paper we approximate a *digital* image by a *discrete mesh* which is defined as the triple of the set of discrete points, edges and triangles. For the sake of brevity we will drop the word 'discrete' in the remainder of this paper.

3.2.1 Definitions and notations

The definitions used in Section 3.1 have to be adjusted towards the digital setting. In the digital setting where the pixels $(X, Y, f(X, Y)) \in \mathbb{N} \times \mathbb{N} \times \mathbb{R}$, the prediction point expressed in the global coordinate system is defined as:

$$P^*_{j+1,k+1} = \begin{bmatrix} \mathcal{B}_{[X]}(P_{j,k}, P_{j,k+1})_{\lfloor L_{j,k}/2 \rfloor} \\ \mathcal{B}_{[Y]}(P_{j,k}, P_{j,k+1})_{\lfloor L_{j,k}/2 \rfloor} \\ \dfrac{y_{j,k} + y_{j,k+1}}{2} \end{bmatrix}$$

such that also $P^*_{j,k} \in \mathbb{N} \times \mathbb{N} \times \mathbb{R}$. The definition of the normal ray in the previous section is slightly adapted for the digital setting.

Definition 3.1 (normal ray) *The digital normal ray* $\bar{r}_{j+1,2k+1}(\gamma) = (\bar{r}^x_{j+1,2k+1}(\gamma), \bar{r}^y_{j+1,2k+1}(\gamma))$ *expressed in the coordinate system xy through* $e_{j,k}$ *is defined as:*

[1] In this paper we take for $\mathcal{B}$ Bresenham's line rasterisation algorithm.

$$\bar{r}_{j+1,2k+1}(\gamma) = \left(\begin{bmatrix} 0 \\ \frac{y_{j,k}+y_{j,k+1}}{2} \end{bmatrix} + d_{j,k}(\gamma) \begin{bmatrix} 1 \\ \frac{n_{[y]_{j,k}}}{n_{[x]_{j,k}}} \end{bmatrix} \right) \tag{3.3}$$

for

$$\gamma = 0 \dots L_{j,k} - 1$$

and $d_{j,k}(\gamma) = \operatorname{sign}(\gamma - \lfloor L_{j,k}/2 \rfloor) \left\| \left(\mathcal{B}(P_{j,k}, P_{j,k+1})_\gamma, \mathcal{B}(P_{j,k}, P_{j,k+1})_{\lfloor L_{j,k}/2 \rfloor} \right) \right\|_{L_2}$ *the signed distance between the midpixel of* $e_{j,k}$ *and the pixel with index* γ*.*

Definition 3.2 (normal index) *We define the* normal index $i_{j,k}$ *as:*

$$\begin{aligned} i_{j,k} := \min \Bigg\{ \gamma - \lfloor L_{j,k}/2 \rfloor \quad | \quad & \operatorname{sign}\left(\bar{r}^y_{j,k}(\gamma) - f_e(\bar{r}^x_{j,k}(\gamma)) \right) \\ & = -\operatorname{sign}\left(\bar{r}^y_{j,k}(\gamma+1) - f_e(\bar{r}^x_{j,k}(\gamma+1)) \right) \Bigg\}. \end{aligned} \tag{3.4}$$

Contrary to the continuous setting the normal offset algorithm for the digital setting is unlikely to exactly pinpoint a sample. Indeed, due to discretisation in most of the cases $y_{j,k}$, the y-value of the piercing point, will differ from the exact function value $f_e(x_{j,k})$ as $\bar{r}^y_{j,k}(\gamma+1) - \bar{r}^y_{j,k}(\gamma) \approx s^{-1}_{j,k}$, with $s_{j,k} \in \mathbb{R}$ the slope of the normal ray $\bar{r}_{j,k}(\gamma)$. To ensure perfect reconstruction a vertical offset $v_{j,k}$ is introduced.

Definition 3.3 (vertical offset) *The vertical offset is the difference between the exact function value and the value of the digital ray (Definition* (3.1)*) at* $i_{j,k}$*:*

$$v_{j,k} = \bar{r}^x_{j,k}(i_{j,k}) - \bar{r}^y_{j,k}(i_{j,k}). \tag{3.5}$$

This extra offset caused by the discretisation process makes the transform redundant.

4 Compression

The previous section stated the algorithm that adaptively partitions a given image, where the possible triangle splits are dictated by the generated piercing points. The output of the algorithm consists of both topological data, i.e. the representation of the tree $\mathcal{T}_j$, and geometrical data, i.e. the sequence of

normal and vertical offsets. A previous paper [21] discusses the encoding of the hierarchical mesh structure to a serialized bitstream and explains data structures used by the normal mesh algorithm. This section focusses on how the geometric information can be encoded efficiently for images $\in \mathcal{H}$. We introduce a probability model for the normal indices $i_{j,k}$ and vertical offsets $v_{j,k}$ in order to reduce the overall bitrate given a target signal to noise ratio. Apart from the local function behavior, the PMF (Probability Mass Function) of $i_{j,k}$ depends on orientation of the edge $e_{j,k}$ w.r.t. the XY-plane. Once we have a sound model for the PMF in terms of $H_{j,k} := |Z_{j,k+1} - Z_{j,k}|$ and $l_{j,k} := \|(X_{j,k+1}, Y_{j,k+1}), (X_{j,k}, Y_{j,k})\|$ for those indices $i_{j,k}$, we apply an entropy-coder to reduce the expected bitrate.

4.1 Model for the vertical offsets

Vertical offsets will have a high probability to have significant values when piercing points are found on the trajectory of $c(x)$. The detail information or the high-frequency part of the input signal is gathered into vertical offsets. According to [18] and confirmed by the experiments shown in Figure 2, the vertical offsets v, can be modelled by a two-parametric zero inflated geometrical distribution ZID(p, λ) (see [1] and references therein) given by:

$$v \sim f_V(v) = p\delta(v) + (1-p)\lambda/2(1-\lambda)^v. \tag{4.1}$$

In the next paragraph we discuss how to obtain the parameters p and λ using a most-likelihood estimate (MLE) method.

4.1.1 Zero inflated distribution: parameter estimation

Given a parametric distribution $\pi(v|\lambda)$. We define the associated zero-inflated distribution to have:

$$P(V = 0|p, \lambda) = p + (1-p)\pi(0|\lambda) \tag{4.2}$$
$$P(V = v|p, \lambda) = (1-p)\pi(v|\lambda) \tag{4.3}$$

with p the chance that the hidden state Z finds itself in the δ distribution. We have the following properties:

$$E(V|p, \lambda) = (1-p)E_\pi(V|\lambda)$$
$$\mathrm{Var}(V|p, \lambda) = p(1-p)E_\pi(V|\lambda)^2 + (1-p)Var_\pi(V|\lambda)$$

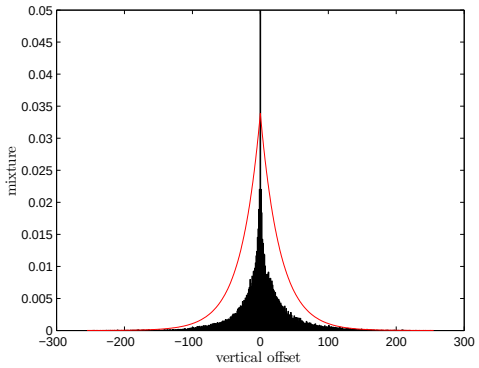


Fig. 2. Histogram of the vertical offsets coming from a large test set of geometrical images. The figure suggests a zero inflated double geometrical distribution to model the statistics of the vertical offsets. The parameters produced by the MLE procedure are $p = 0.3955$ and $\lambda = 0.0339$.

For a sample of size n (v_i given) the full likelihood is given by:

$$\begin{aligned} L(p,\lambda|V,Z) &= \prod_{i=1}^{n} P(V=v_i|Z_i=z_i)P(Z_i=z_i) \\ &= \prod_{i=1}^{n} p^{z_i}((1-p)\pi(v|\lambda))^{1-z_i} \\ &= \prod_{v_i>0}((1-p)\pi(v_i|\lambda)) \prod_{v_i=0} p^{z_i}((1-p)\pi(0|\lambda))^{1-z_i} \end{aligned} \tag{4.4}$$

The log-likelihood is given by:

$$\begin{aligned} \log L(p,\lambda|V) =& N_0 \log(p+(1-p)\pi(0|\lambda)) \\ &+ (n-N_0)\log(1-p) + \sum_{v_i>0}\log(\pi(v_i|\lambda)) \end{aligned} \tag{4.5}$$

where N_0 is the number of zero occurrences.

Suppose we model the samples with a zero inflated geometrical distribution, i.e., $\pi(v|\lambda) \sim \lambda(1-\lambda)^v$, $v \in \mathbb{N}$. Equation (4.5) becomes:

$$\begin{aligned} \log L(p,\lambda|V) =& N_0 \log(p+(1-p)\lambda) \\ &+ (n-N_0)\left(\log(1-p)+\log\lambda\right) + \log(1-\lambda)\bar{V} \end{aligned} \tag{4.6}$$

with $\overline{V} = \mathrm{E}_{v_i>0}[v_i]$, the mean of the values $v_i > 0$. We have to maximize Eq. (4.6) under the constraints $0 \le p \le 1$ and $0 \le \lambda \le 1$. Note that Eq. (4.6) is concave within the feasible region. This can be translated into an ICP (inequality constrained problem), which can be solved by a Newton minimization procedure. The derivatives of $\log L(p,\lambda|V)$ are:

$$\frac{\partial \log L(p,\lambda|V)}{\partial p} = N_0 \frac{(1-\pi(0|\lambda))}{p+(1-p)\pi(0|\lambda)} - (n-N_0)\frac{1}{1-p} \tag{4.7}$$
$$\frac{\partial \log L(p,\lambda|V)}{\partial \lambda} = N_0 \frac{\pi'(0|\lambda)(1-p)}{p+(1-p)\pi(0|\lambda)} + \sum_{v_i>0} \frac{\pi'(v_i|\lambda)}{\pi(v_i|\lambda)}$$

with $\pi'(v|\lambda) = \dfrac{\partial \pi(v|\lambda)}{\partial \lambda}$. For the case of a zero inflated geometrical distribution we have $\pi(v|\lambda) = \lambda(1-\lambda)^v$ and $\pi'(v|\lambda) = (1-\lambda)^{v-1}(1-\lambda(1+v))$. Figure 2 shows the parameter outcome of the minimization procedure applied to experimental data; that is, the vertical offsets resulting from the proposed normal offset transform applied to a large test set of simple geometrical images.

4.2 Model for the normal indices i

The normal indices can be encoded directly with codewords of $\lceil \log_2 L_{j,k} \rceil$ bits, with $L_{j,k}$ the number of pixels in the edge $e_{j,k}$ decaying like $O(2^{-j})$. However, this way of encoding the normal indices presupposes that those indices are distributed uniformly, maximising the entropy.

Previous section derived a model for the vertical offsets. When the class of target functions is the Horizon class $\mathcal{H}$ given in section 2, we are now able to give a model that allows us to efficiently encode normal indices.

For now we *assume* the trajectory of the contour runs in between the two end points of $e_{j,k}$. Remaining scenarios (state) will be discussed at the end of this section. An algorithm to detect in which scenario an edge finds itself in will be given in Section 4.2.1.

To lower the entropy we derive an appropriate PMF associated with i for the class of Horizon images which allows an entropy encoder such as an Huffman encoder to lower the expected bitrate. The following lemma connects the PMF to the orientation of an edge somewhere during the decomposition stage assuming the function f_e above the edge behaves like a step function, which is true for images belonging to $\mathcal{H}$ and the edge endpoints are located at both sides of the contour. Vertical offsets show up in the derivation since in the discrete setting normal offsets are incapable of exactly pinpointing a function value.

Lemma 4.1 *Let the image be an instance of $\mathcal{H}$. Assume $c(x) = y$ runs in between the endpoints of edge e as depicted in Figure 3. Further assume that the horizontal distance d between the begin point of the edge and the discontinuity is uniformly distributed on the interval $[0, l]$, with l the euclidian distance in*

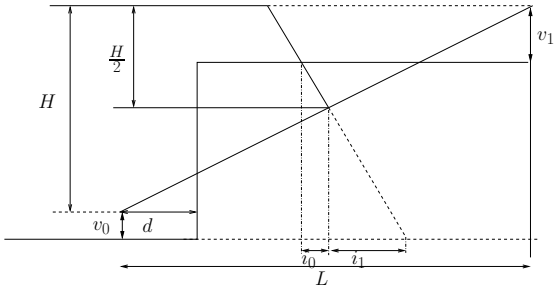


Fig. 3. Situation where piercing points do not take on the exact function value. In the example $v_0 > 0$ while $v_1 < 0$.

the XY-plane between the end points of e:

$$f_D(d) = \frac{1}{l}\chi_A(d) \text{ with } \chi_A(x) = \begin{cases} 1, \, x \in A \\ 0, \text{ else} \end{cases}, A = [0, l] \tag{4.8}$$

the signed magnitude of the vertical offsets associated with the begin point and end point of e are respectively $v_0, v_1 \in \mathbb{N}$ having a zero inflated geometrical distribution (see Eq. (4.1)) and independently distributed with parameters $\lambda_0 = \lambda_1 = \lambda$ and $p_0 = p_1 = p$. Define i_0 and i_1 as:

$$\begin{aligned} i_0 &= -H/l\,(H/2 + v_1) \\ i_1 &= \quad H/l\,(H/2 + v_0) \end{aligned} \tag{4.9}$$

then, for $0 < i < l/2$, the PDF is given by:

$$\begin{aligned} f_I(i) =& \frac{\frac{l}{2} - i_1}{l} \left[(1-p)\frac{\lambda_0}{2}(1-\lambda_0)^{\left|\frac{l}{H}i - \frac{H}{2}\right|} + p\delta\left(\left|\frac{l}{H}i - \frac{H}{2}\right|\right)\right] \\ &+ \sum_{v_0} f_{V_0}(v_0) \sum_{v_1} \frac{1 - (f_I(i_0(v_1)) + f_I(i_1(v_0)))}{i_1 - i_0} f_{V_1}(v_1). \end{aligned} \tag{4.10}$$

Analogously for $-\frac{l}{2} < i < 0$.

Proof 4.1 *Assuming v_0, v_1 independently distributed, the probability mass function $f_I(i)$ is given by:*

$$
\begin{aligned}
f_I(i) &= \sum_{-\infty}^{\infty}\sum_{-\infty}^{\infty} f_I(i|V_0=v_0,V_1=v_1) f_{V_0,V_1}(v_0,v_1)dv_0dv_1 \\
&= \sum_{-\infty}^{\infty} f_{V_0}(v_0) \sum_{-\infty}^{\infty} f_I(i|V_0=v_0,V_1=v_1) f_{V_1}(v_1)dv_1dv_0
\end{aligned}
\tag{4.11}
$$

with

$$
\begin{aligned}
f_I(i|V_0=v_0,V_1=v_1) := \quad & f_I(i_0(v_1))\delta_{i_0}(i) \\
+ \quad & \frac{(1-[f_I(i_0(v_1))+f_I(i_1(v_0))])}{i_1-i_0}\chi_{[i_0,i_1]} \\
+ \quad & f_I(i_1(v_0))\delta_{i_1}(i) \\
f_I(i_0(v_1)) := \quad & \frac{\frac{l}{2}+i_0(v_1)}{l} \\
f_I(i_1(v_0)) := \quad & \frac{\frac{l}{2}-i_1(v_0)}{l}
\end{aligned}
$$

The top right picture of Figure 4 shows an instance of the set of distributions given by Eq. (4.11), together with the other possible states an edge can find itself into when it comes into contact with a contour. Lemma 4.2 gives normal index PDF for each of the states. The proof of Lemma 4.2 is similar to the proof of Lemma 4.1.

Lemma 4.2

(1) Assume that a discontinuity is present between both edge points then the PDF is given by:

$$
\begin{aligned}
&a) && \text{Equation (4.10)} && ; H < l && (4.12)\\
&b) p(i) = && 1/l, && i \in [-l/2, l/2] && ; H \geq l && (4.13)\\
& && 0, && \text{else} && && (4.14)
\end{aligned}
$$

(2) Assume that one of the endpoints of the edge is located at the discontinuity then:

$$
a) p(i) = \frac{1}{2}\left[(1-p)\frac{\lambda_0}{2}(1-\lambda_0)^{|\frac{l}{H}i-\frac{H}{2}|} + p\delta\left(\left|\frac{l}{H}i-\frac{H}{2}\right|\right)\right] \quad ; H < l \tag{4.15}
$$

$$
b) p(i) = \begin{cases} 1/2,\ i = \pm l/2 & \\ 0, \qquad \text{else} & \end{cases} \quad ; H \geq l \tag{4.16}
$$

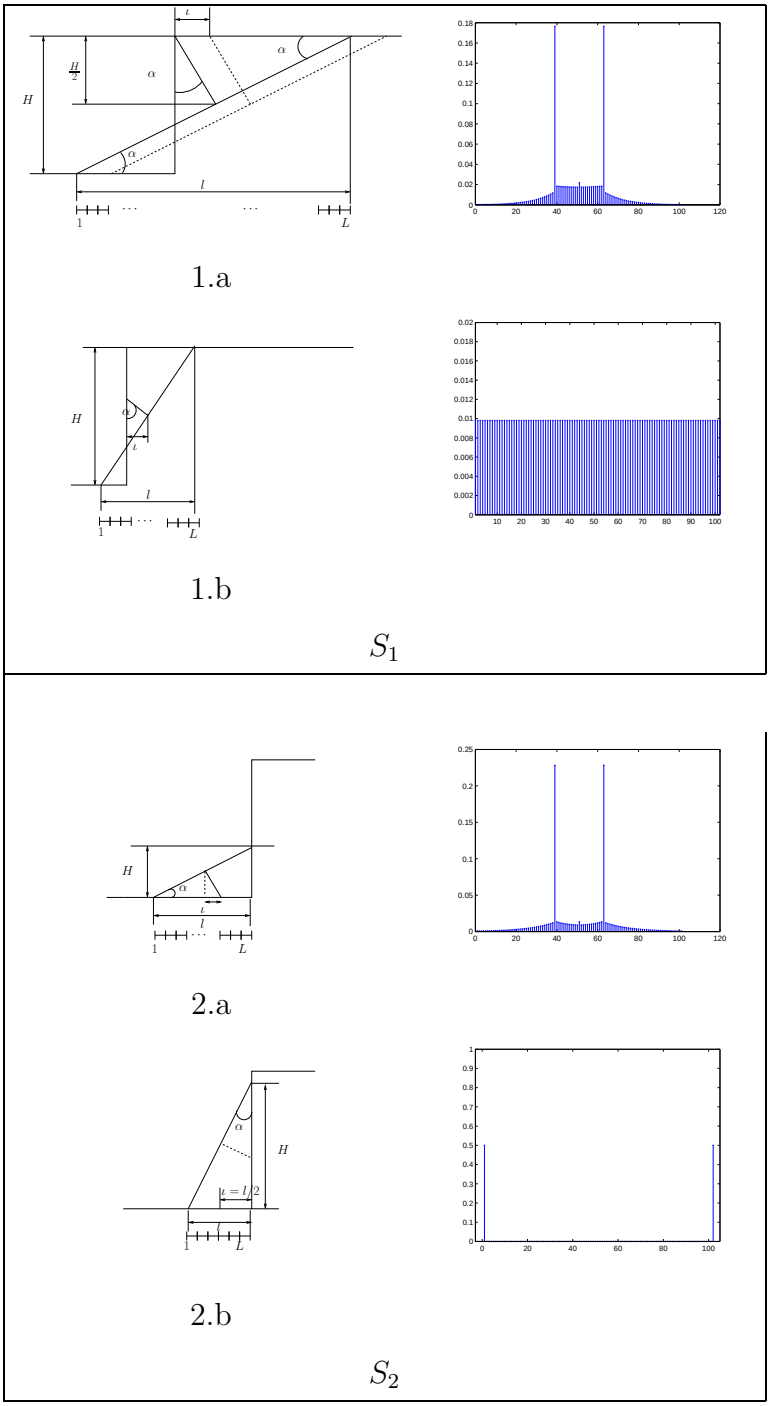


Fig. 4. Four edge vs. step function situations together with their associated PMF for the normal indices given by Eq.(4.13)-(4.16). The two situations at the top of the figure assume a contour runs through the vertical plane containing edge e. The two situations at the bottom involve an edge where one of the end points is located on the contour.The a-part of each group concerns an edge where $H < l$ where the b-part concerns edges where $H \geq l$.

4.2.1 Finer scale model selection from given coarse scale situation

Previous section enumerated the different PMF to model the normal indices, given the state of the edge corresponding to the piercing point. Although we could estimate the state from the image at the encoder side, this data would additionally have to be transmitted to the decoder. We therefore present a heuristic procedure to predict the state solely on the offset behaviour known at a particular time instance.

We differentiate the cases in three groups.

(1) S_0: An edge can be *approximately* flat (S_0). From Equation (3.5) we see that magnitude of the vertical offset v is greatly influenced by the slope s of the normal ray r. We therefore say that an edge e is flat ($\in S_0$) is the slope $s = n_{[x]_{j,k}}/n_{[y]_{j,k}}$

$$|s| < \frac{1}{\delta}, \tag{4.17}$$

with δ a tunable threshold parameter.
(2) S_1: S_1 contains both states 1.a and 1.b
(3) S_2: S_2 contains states 2.a and 2.b

The last two groups fall into the states mentioned in previous section, once the group of an edge e is known, the particular instance of the distribution depicted by Figure (4) can be easily determined by $H_{j,k}$ and $l_{j,k}$.

The main difficulty is the separation of the 3 groups only at what will be known at the decoder side, i.e., $H_{j,k}, l_{j,k}, i_{j,k}$. The flow chart given in Figure 5 gives a rough outline of where the decisions are based upon. The procedure

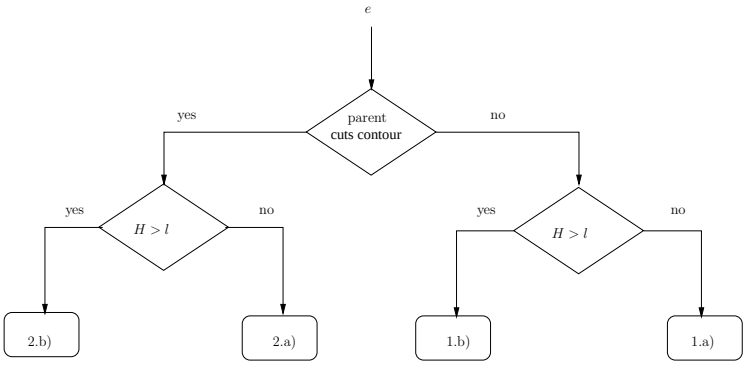


Fig. 5. Decision tree to estimate the state of an edge.

to determine whether or not the edge is crossed by a contour is given by Algorithm (1). Since our coder is a symmetric coder, Algorithm (1) will be traversed in exactly the same way at both the encoder and decoder side. The data needed for a certain decision at the encoder will also be available at the decoder side.

Algorithm 1 Group selection procedure

procedure PARENT STATUS($e_{j,k}$)
  if e has no parents **then**
    goto: NO_PARENT($e_{j,k}$)
  else if The normal ray of the parent of edge $e_{j,k}$ has not pierced a contour **then**
    goto: PARENT_DID_NOT_PIERCED_CONTOUR($e_{j,k}$)
  else
    goto: PARENT_PIERCED_CONTOUR($e_{j,k}$)
  end if
end procedure

procedure NO_PARENT($e_{j,k}$)
  goto: S_0_OR_S_1($e_{j,k}$)
end procedure

procedure PARENT_DID_NOT_PIERCED_CONTOUR($e_{j,k}$)
  if the parent edge belongs to S_0 **then**
    goto: S_0_or_S_1($e_{j,k}$)
  else if $H_{j-1} < l_{j-1}$ **then**
    if $i < H_{j-1}^2/(2l_{j-1}) \times L_{j-1}/l_{j-1}$ **then**
      goto: S_0_OR_S_2($e_{j,k}$)
    else
      goto: S_0_OR_S_1($e_{j,k}$)
    end if
  else
    $e \in S_2$
  end if
end procedure

procedure PARENT_PIERCED_CONTOUR($e_{j,k}$)
  goto: S_0_OR_S_2($e_{j,k}$)
end procedure

procedure S_0_OR_S_1($e_{j,k}$)
  if slope of r satisfies Equation (4.17) **then**
    $e_{j,k} \in S_0$
  else
    $e_{j,k} \in S_1$
  end if
end procedure

procedure S_0_OR_S_2($e_{j,k}$)
  if slope of r satisfies Equation (4.17) **then**
    $e_{j,k} \in S_0$
  else
    $e_{j,k} \in S_2$
  end if
end procedure

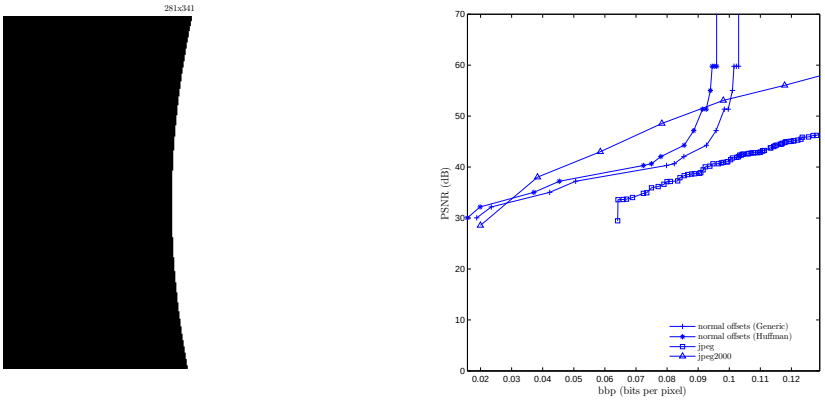
For more natural images there could be more than one contour passing through the vertical plane in between the end points of the edge. In that case it will be possible for an edge to form a merely flat 'bridge' spanning over a canyon-like region, although its normal offset can be assigned a large value. These images can be tiled making sure that each tile contains no more than one contour. This can be obtained by a preprocessing procedure, like presented in [20], which constructs the initial coarse mesh adapted to the main features of the function to be processed.

5 Results and Further research

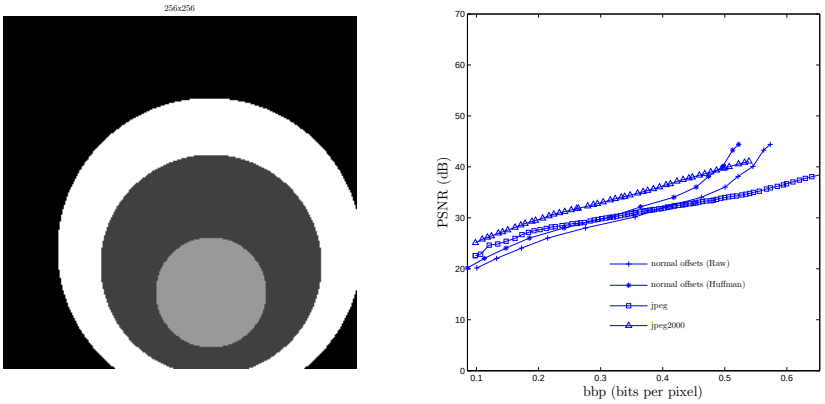
Recently much research has been done to solve the problem of sparsely representing two dimensional functions having line discontinuities. Since many images consist of large smooth areas separated by smooth contours, they can be interpreted as a regular sampling of such functions. Current transform coders based on wavelet decompositions have a suboptimal n-terms approximation rate for such images. They fail to produce sparse representations of smooth line discontinuities.
We proposed a nonlinear refinement procedure based on normal mesh techniques to sparsely represent contours. We adapted the nonlinear transform towards the digital setting.
We proposed a model for the wavelet coefficients coming from our encoder and added an entropy encoder to reduce the bit-rate. Here we show several results on the performance of the proposed encoder.

5.1 Experiments

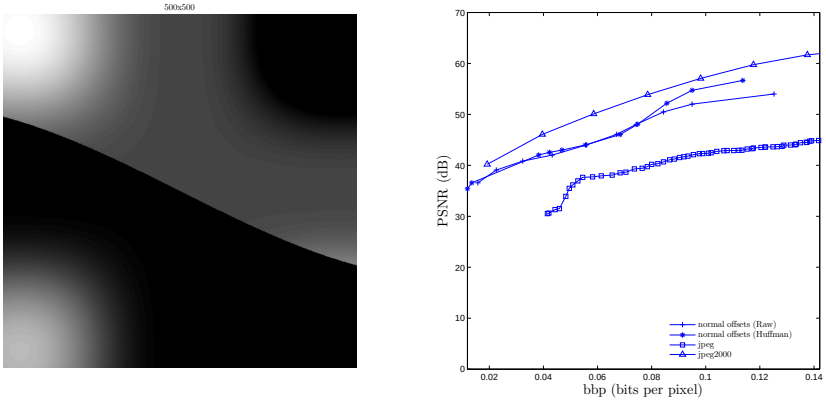
For the experiments depicted in Figure 6 we used the L_2-norm (MSE). Using the L_1-norm, for which the transform performs best, the outcome of our experiments would be more favorable, since in contrast to L_1, the L_2-norm prefers a lot of little errors over a few big errors. The experiments shown in Figure 6 indicate that the current normal offset encoder as described here does not yet outperform the JPEG2000 encoder over the entire range. At low rates the strong aliasing effect of the scheme being a refinement scheme plays the dominant role. The justification for this is that over the entire range (even for low resolution triangulations) the vertical offsets are stored losslessly in our current implementation while they are not the main contributors of the image quality. Once the mesh is fine enough the distortion decays rapidly. Including a rate distortion optimized quantizer ([19]) on the vertical offsets starting from the lossless encoded image will improve the performance at lower rates.



'Horizon Image'



'Circles Image'



'Piecewise Smooth'

Fig. 6. The rate distortion curve of a JPEG encoder and a JPEG2000 encoder together with the rate distortion curve of a normal offset encoder without using offset modelling (Raw) and a normal offset encoder using the proposed models as input for an Huffman entropy encoder (Huffman). Note that the gain of using entropy coding of the non Horizon images kicks in at higher bit rates, when the fine representation of the contour becomes more and more important.

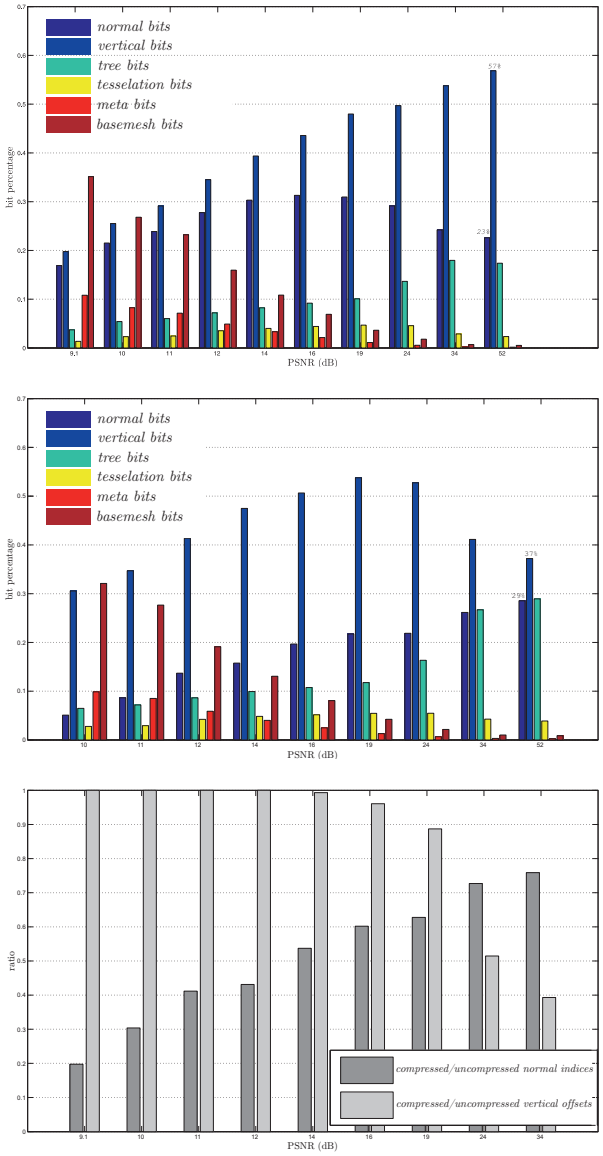


Fig. 7. Percentage of the bits allocated to different coefficients for the bottom (‘`circles`’) image of Figure 6. (Top) The vertical offsets where *not* entropy coded. The normal offsets where coded with a number of bits decaying like $\log_2(\mathcal{O}(2^{-j}))$. (Bottom) The bottom Figure shows the reduction in both vertical and normal offsets when entropy coding is used as explained in Section 4.

Around the lossless rate a strong convergence is seen. For simple geometrical images, the rate at which a lossless compression is achieved is much lower than the JPEG2000 as shown in Table 1.

Figure 7 illustrates the gain in the relative bit budget consumed by both normal and vertical offsets using the proposed model. Figure 8 points out where most of the distortion is put by both wavelet and normal mesh based image coders. The comparison on piecewise smooth images (in contrast to piecewise

Table 1
File size in kilo bytes of *lossless* encoded test images (Figure 6) using different encoders. We note that for 'simple' geometrical images the normal offset based algorithms outperform both jpeg standards. For more complex images, e.g. 'cameraman' the jpeg standards still perform better, since our encoder has not yet incorporated a segmentation procedure to create an appropriate initial mesh where the rest of the refinement procedure is based upon.

Image	JPG	JPEG2000	Raw	Huffman
horizon	2.42	1.70	1.47	**1.33**
circles	9.97	7.28	4.58	**4.17**
cameraman	**22.5**	32	93	75.8

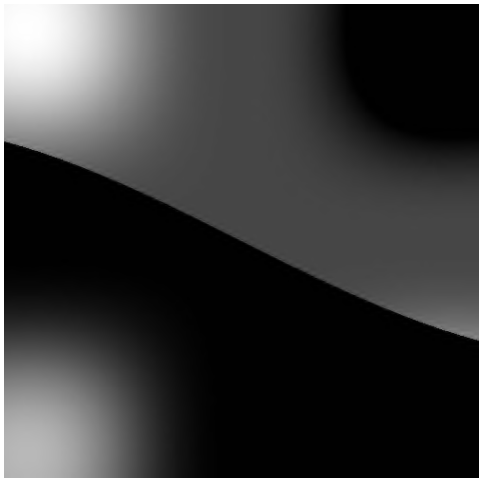
constant images) favours the JPEG2000 as it is equipped with enough vanishing moments to efficiently capture the smooth regions. However, lots of error is introduced along the trajectory of the smooth contours due to the inability of tensor product wavelet to efficiently capture geometric smoothness. The normal offset scheme does quite the opposite, i.e., it does a good job approximating the contour but lacks higher order polynomials to approximate the smooth evolving regions.

Further research has to be done to finetune different parameters on the level of the transform as well as on the bitcoding level. For instance more suitable and image specific basemeshes can be used instead of the trivial diagonal split of the domain. Parameters p, λ described in Section 4 can be adjusted for each image and could be made 'resolution level dependent'. The 'ad hoc' algorithm described in Section 4.2.1 to select the appropriate probability model to encode a certain offset has to be made more robust. Those measures can still further improve the performance of the proposed encoder.

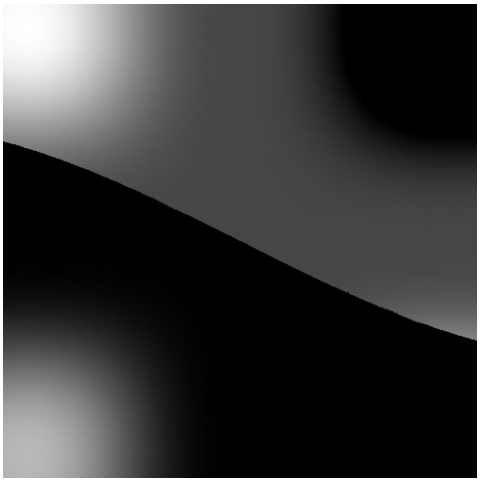
5.2 Real-life images

The method presented in this paper is highly sensitive to noise. In order to reduce this sensitivity, we could define the normal offset between the predicted pixel value and a smoothed version of the true image. The vertical offset takes care of the difference between the smoothed and observed values, thereby preserving the perfect reconstruction property (see Fig. 9). In a post-processing step these vertical offsets can even be adapted to increase the signal to noise ratio.

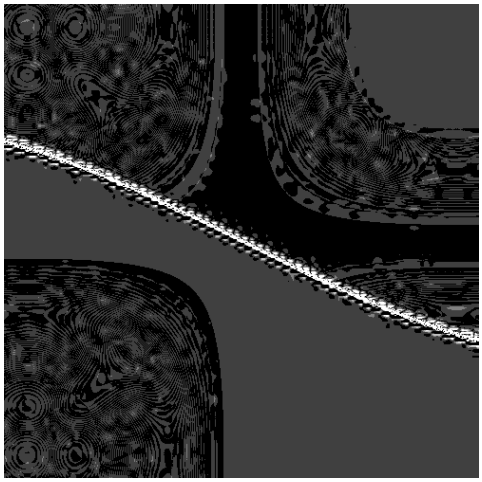
Several tools can be included to alleviate some bottlenecks in the performance of the current algorithm w.r.t. a broader class of images. We can, for instance, incorporate a segmentation procedure to create an appropriate initial mesh where the rest of the refinement procedure is based upon. This segmen-



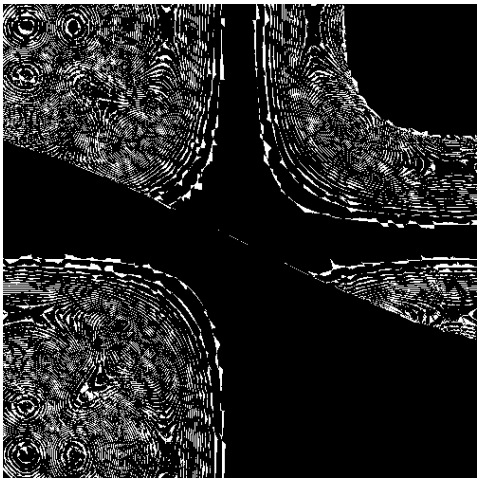
(a) JPEG2000 encoded piecewise smooth image (PSNR=45dB, 0.12bpp).



(b) Normal offset encoded piecewise smooth image. (PSNR=47dB, 0.13bpp)



(c) Difference image of Fig. 8(a) with the original.



(d) Difference image of Fig. 8(b) with the original.

Fig. 8. JPEG2000 and normal offset compressed piecewise smooth image. We see that the wavelet-based JPEG2000 image coder introduces significantly more error when approximating the contour.

tation procedure should make sure that each tile contains no more than one contour. This can be obtained by a preprocessing procedure, like presented in [20], which constructs the initial coarse mesh adapted to the main features of the function to be processed. Currently, our implementation encodes the vertical offsets losslessly while they are not the main contributors of the image quality. The losslessly encoded vertical offsets account for more than 50% of the total bit budget (see Fig. 10). The encoding of the vertical offsets should be done in a rate-distortion framework. That is, only spending as much bits to lossy encode vertical offsets as is justified by the overall distortion. Another extension for real-life images is the use of a tree-pruning strategy, similar to the

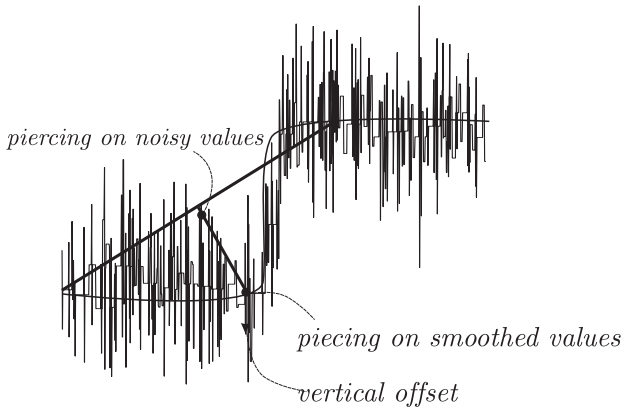


Fig. 9. For noisy images we define the normal offset between the predicted pixel value and a smoothed version of the true image. The vertical offset takes care of the difference between the smoothed and observed values, thereby preserving the perfect reconstruction property.

optimal tree pruning algorithm in the CART book [3]. This would avoid the currently greedy tree-growing algorithm spending too much effort (in terms of parameters or bits) in approximating textured regions.

6 Acknowledgments

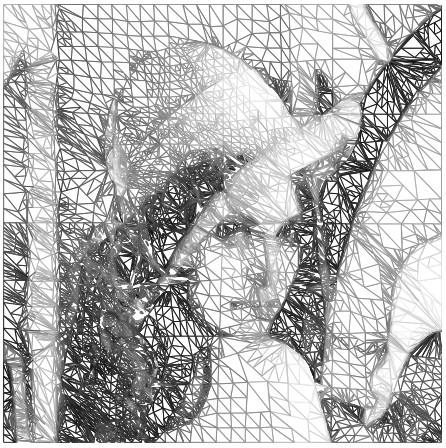
The work is supported by the Fund for Scientific Research (FWO) project SMID: Stability of Multiscale Transforms on Irregular Data, grant #G.0431.05 and the Belgian Programme on Interuniversity Poles of Attraction, initiated by the Belgian Federal Science Policy Office and by the Center Of Excellence on Optimisation in Engineering of the K.U.Leuven.

References

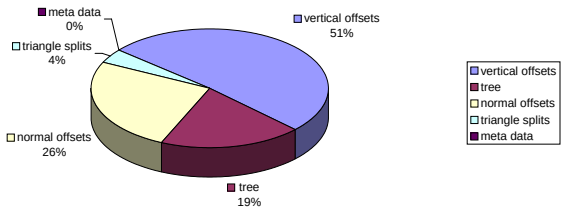
[1] D.K. Agarwal, A.E. Gelfand, and S. Citron-Pousty. Zero-inflated models with application to spatial count data. *Environmental and Ecological Statistics*, 9(4):pp. 341–355, December 2002.

[2] P. Schröder A.Khodakovsky and Wim Sweldens. Progressive geometry compression. In Kurt Akeley, editor, *Siggraph 2000, Computer Graphics Proceedings*, pages 271–278. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.

(a) Compressed 'Lena' image at a Psnr of 30dB. The output size is 38 KBytes or 1.1bpp.



(b) Triangulation of the 'Lena' image corresponding to Figure 10(a)



(c) Bit budget allocated to each part of the algorithms output.

Fig. 10. 'Lena' image.

[3] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and regression trees (CART)*. Wadsworth, Monterey, CA, USA, 1984.

[4] E. J. Candès and D. L. Donoho. Curvelets - a surprisingly effective nonadaptive representation for objects with edges. Technical report, Department of Statistics, Stanford University, 2000.

[5] I. Daubechies, O. Runborg, and W. Sweldens. Normal polyline approximation. *Constructive Approximation*, 20(3):399–463, May 2004.

[6] S. Dekel and D. Leviatan. Adaptive multivariate approximation using binary space partitions and geometric wavelets. *SIAM J. Numer. Anal.*, 43(2):707–732, 2005. ISSN 0036-1429. doi: http://dx.doi.org/10.1137/040604649.

[7] Laurent Demaret, Nira Dyn, and Armin Iske. Image compression by linear splines over adaptive triangulations. *Signal Process.*, 86(7):1604–1616, 2006. ISSN 0165-1684. doi: http://dx.doi.org/10.1016/j.sigpro.2005.09.003.

[8] R. A. DeVore. Nonlinear approximation. *Acta Numerica*, 7:51–150, 1998.

[9] M. N. Do and M. Vetterli. Contourlets. In G. Welland, editor, *Beyond*

Wavelets, pages 83–107. Academic Press, 2003.

[10] D. L. Donoho. Wedgelets: Nearly minimax estimation of edges. *Annals of Statistics*, 27(3):859–897, 1999.

[11] M. Antonini F. Payan. An efficient bit allocation for compressing normal meshes with an error-driven quantization. *Computer Aided Geometric Design*, 22:466?486, 2005.

[12] Ilja Friedel, Peter Schröder, and Andrei Khodakovsky. Variational normal meshes. *ACM Trans. Graph.*, 23(4):1061–1073, 2004. ISSN 0730-0301. doi: http://doi.acm.org/10.1145/1027411.1027418.

[13] I. Guskov, K. Vidimče, W. Sweldens, and P. Schröder. Normal meshes. In *SIGGRAPH 2000 Conference Proceedings*, 2000.

[14] M. Jansen, R. Baraniuk, and S. Lavu. Multiscale approximation of piecewise smooth two-dimensional functions using normal triangulated meshes. *Appl. Comp. Harm. Anal.*, 2005.

[15] A. Khodakovsky and I. Guskov. Compression of normal meshes. In *Geometric Modeling for Scientific Visualization*, pages 189–207. Springer Verlag, 2003.

[16] E. Le Pennec and S. Mallat. Bandelet image approximation and compression. *Society for Industrial and Applied Mathematics*, 4:992–1039, 2005. doi: 10.1137/040619454.

[17] H. Choi S. Lavu and R. Baraniuk. Geometry compression of normal meshes using rate-distortion algorithms. In *SGP '03: Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 52–61, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association. ISBN 1-58113-687-0.

[18] E. Simoncelli. Modeling the joint statistics of images in the wavelet domain. In M. A. Unser, A. Aldroubi, and Laine A. F., editors, *Wavelet Applications in Signal and Image Processing VII*, volume 3813 of *SPIE Proceedings*, pages 206–214, July 1999.

[19] Gary J. Sullivan and Shijun Sun. On dead-zone plus uniform threshold scalar quantization. In *Visual Communications and Image Processing 2005, Proc. of SPIE*, volume 5960.

[20] W. Van Aerschot, Vanreas E., and Bultheel A. Preprocessing for the decomposition of images with normal offsets. Technical Report TW 475, Department of Computer Science, K.U.Leuven, 2006.

[21] W. Van Aerschot, M. Jansen, E. Vanraes, and A. Bultheel. Image compression using normal mesh techniques. In A. Cohen and L. Schumaker, editors, *Curves and Surfaces, Avignon 2006*, Brentwood, 2006. Nashboro Press.