Image Compression by Linear Splines over Adaptive Triangulations

Laurent Demaret, Nira Dyn, and Armin Iske

Abstract. This paper proposes a new method for image compression. The method is based on the approximation of an image, regarded as a function, by a linear spline over an adapted triangulation, $\mathcal{D}(Y)$, which is the Delaunay triangulation of a small set Y of significant pixels. The linear spline minimizes the distance to the image, measured by the mean square error, among all linear splines over $\mathcal{D}(Y)$. The significant pixels in Y are selected by an adaptive thinning algorithm, which recursively removes less significant pixels in a greedy way, using a sophisticated criterion for measuring the significance of a pixel. The proposed compression method combines the approximation scheme with a customized scattered data coding scheme. We compare our compression method with JPEG2000 on two geometric images and on three popular test cases of real images.

Zusammenfassung. Dieser Artikel stellt eine neue Methode zur Bildkompression vor. Diese Methode beinhaltet die Approximation eines gegebenen Bildes, hier aufgefasst als eine Funktion über den diskreten Bildpunkten, durch eine lineare Splinefunktion über einer adaptiven Triangulierung, $\mathcal{D}(Y)$, wobei $\mathcal{D}(Y)$ die Delaunay-Triangulierung einer kleinen Teilmenge Y signifikanter Bildpunkte bezeichnet. Diese lineare Splinefunktion minimiert dabei unter allen linearen Splinefunktionen über $\mathcal{D}(Y)$ die Distanz zu dem vorliegenden Bild, die durch die gemittelte Summe der Fehlerquadrate gemessen wird. Die signifikanten Bildpunkte werden unter Verwendung eines adaptiven Thinning-Algorithmus ausgewählt, der weniger signifikante Bildpunkte aus der gegebenen Menge aller Bildpunkte rekursiv entfernt. Zum Entfernen der Bildpunkte wird dabei ein geeignetes Bewertungskriterium verwendet, mit dem Signifikanzen einzelner Bildpunkte gemessen werden können. Die hier vorgestellte Kompressionsmethode kombiniert das verwendete Approximationsschema mit einer passenden Kodierungsmethode für unstrukturierte planare Punktmengen. Unsere Kompressionsmethode wird schliesslich unter Verwendung von zwei geometrischen Testbeispielen und drei populären realen Testbildern mit JPEG2000 verglichen.

Résumé. Cet article propose une nouvelle méthode de compression d'images. Cette méthode est basée sur l'approximation d'une image, vue comme une fonction, par une fonction spline linéaire sur une triangulation adaptée, $\mathcal{D}(Y)$, qui est la triangulation de Delaunay d'un ensemble réduit Y de pixels significatifs. Cette fonction spline linéaire minimise la distance à l'image originale, mesurée par l'erreur quadratique moyenne, parmi la classe de toutes les fonctions splines linéaires sur la triangulation $\mathcal{D}(Y)$. Les pixels significatifs de Y sont sélectionnés par un algorithme d'Adaptive Thinning, qui élimine récursivement les pixels les moins significatifs. La suppression de ces pixels est effectuée selon un critère mesurant la significativité relative de ce pixel. Le schéma de compression proposé combine la méthode d'approximation précédente avec un codage adapté d'un ensemble de points du plan non structurés. Finalement, nous comparons notre méthode de compression avec le standard de compression JPEG2000 sur deux images géométriques et trois images naturelles classiques.

Keywords: Image compression, adaptive thinning, linear splines, Delaunay triangulations, scattered data coding.

1 Introduction

Many of the well-established methods for image compression, including EBCOT [24, 25], are based on wavelets and related techniques [22, 23, 26], see [6] for a survey on wavelet-based image coding. When working with a dyadic wavelet decomposition [18], digital images are represented by wavelet coefficients. This representation is a linear decomposition over a fixed orthogonal basis. The non-linearity in the approximation of images by wavelets is introduced by the thresholding of the wavelet coefficients [4, 12]. This type of approximation can be viewed as *mildly* nonlinear. Recently, several *highly* nonlinear methods for capturing the geometry of images were developed, such as bandelets [17], curvelets [2], contourlets [13], wedgelets [14, 21], surflets [3], as well as edge-adapted nonlinear multiresolution [19] and geometric spline approximation [7].

This paper proposes a conceptually new highly nonlinear image compression method. The image, viewed as a function, is approximated by a linear spline over the Delaunay triangulation of a small adaptively chosen set of significant pixels, such that these pixels capture the geometry of the image. Since in general the significant pixels are scattered in the rectangular image domain, their Delaunay triangulation is anisotropic. All linear splines over this adaptive triangulation constitute a suitable approximation space for the

image, from which we take the best approximation to the image, minimizing the mean square error. This linear spline is a *continuous* function, which can be evaluated at any point in the rectangular image domain, in particular at the *discrete* set of pixels. Indeed, the compressed image is reconstructed from this linear spline. Moreover, our specific representation of the image (by a continuous function) allows us to display the reconstructed image on any subset of the (continuous) image domain. This option is especially relevant for applications such as zooming, rescaling and conversion between different image representations.

The idea to approximate an image by first identifying significant pixels is not new (see e.g. [16]). In this paper, we go further and obtain a complete compression method. Our compression method combines an efficient algorithm for the selection of a set of significant pixels, adaptive thinning, with a customized scattered data coding scheme.

The utilized adaptive thinning algorithm recursively removes *least significant* pixels from the image, one at a time. The selection of a *good* set of significant pixels, i.e., whose corresponding linear spline over their Delaunay triangulation approximates the image *well*, requires a suitable measure for the significance of pixels. In our previous survey [10] a first *prototype* of an image compression method, here referred to as $\mathbf{AT}^-$, was presented, where some ideas from our earlier papers [8, 9] were included. For the prototype $\mathbf{AT}^-$ a suitable measure for a pixel significance, based on the error incurred by the removal of *one* pixel, was developed and studied.

In this paper, the compression method $\mathbf{AT}^-$ of [10] is substantially improved. This is accomplished by a more sophisticated one-pixel removal strategy and by an improved scattered data coding scheme, yielding a new compression method, called $\mathbf{AT}^*$, whose performance is considerably better than that of $\mathbf{AT}^-$. More detailed arguments in favour of $\mathbf{AT}^*$, when compared with $\mathbf{AT}^-$, are provided later in this paper. Be it sufficient for the moment to say that the new removal strategy of $\mathbf{AT}^*$ uses, unlike $\mathbf{AT}^-$, a significance measure which considers both significances of pixels and of pixel pairs. Moreover, due to the improved coding scheme of this paper, the compression method $\mathbf{AT}^*$ is effective for both low and high bitrates, wheras $\mathbf{AT}^-$ is only effective for low bitrates.

The good performance of the compression method $\mathbf{AT}^*$ is further supported by several comparisons between $\mathbf{AT}^*$ and JPEG2000. The comparisons are performed at both low and high compression rates on five test cases, including two geometric images and three popular real images.

The outline of the paper is as follows. In Section 2, the image approximation scheme is presented. This includes a discussion of our adaptive thinning

algorithm along with its basic ingredients. Then, in Section 3, we explain the use of the approximation scheme for image compression, and we present the coding and decoding of the compressed image. Finally, in Section 4 we compare the performance of $\mathbf{AT}^*$ with those of $\mathbf{AT}^-$ and JPEG2000.

2 Image Approximation

This section provides a detailed description of our image approximation scheme. We introduce the adaptive thinning algorithm for the selection of a set of significant pixels Y. This includes a discussion of its significance measure and of linear splines over Delaunay triangulations. Moreover, we describe the final step of the image approximation scheme, where we construct the best approximation to the image, minimizing the mean square error among all linear splines over the Delaunay triangulation of Y. Finally, we show how to control the mean square error of the image approximation.

2.1 Image Representation

A digital image is a rectangular grid of *pixels*, where each pixel bears a color value or a greyscale *luminance*. We restrict the following discussion to greyscale images. The digital image can be viewed as an element $I \in \{0, 1, \ldots, 2^r - 1\}^X$, where X is the set of pixels, and r is the number of bits in the representation of the luminance values. In this paper, we regard images as functions over the convex hull $[X]$ of the set of pixels X, so that $[X]$ constitutes the rectangular image domain. Each pixel in X is corresponding to a planar grid point, with integer coordinates, lying in $[X]$.

2.2 Adaptive Thinning Algorithm

To obtain a set X_n of n significant pixels, our adaptive thinning algorithm constructs a sequence of nested subsets of pixels

$$X_n \subset X_{n+1} \subset \cdots \subset X_{N-1} \subset X_N = X, \tag{1}$$

where the size $|X_p|$ of any subset X_p in (1) is p, and so $N = |X|$ is the number of pixels in X.

The algorithm recursively removes one pixel from the current set of pixels in a greedy way, which depends on the luminance values attached to the pixels of the image. In each step, the removed pixel is a *least significant* pixel in a sense to be discussed later in this section. Let us first formulate our adaptive thinning algorithm.

Algorithm 1 (Adaptive Thinning).

(1) *Let* $X_N = X$*;*

(2) *For* $k = 1, \ldots, N - n$

(2a) *Find a least significant pixel* $x \in X_{N-k+1}$*;*

(2b) *Let* $X_{N-k} = X_{N-k+1} \setminus x$*.*

In order to describe our specific thinning strategy, it remains to give a definition for a *least significant* pixel in step **(2a)** above, or more generally, what is called in the language of thinning algorithms to determine a *removal criterion* [10]. This requires a further discussion of the image approximation scheme used during the performance of Algorithm 1. For a subset $Y = X_p$ in (1), the approximation of the image is the linear spline over the Delaunay triangulation $\mathcal{D}(Y)$ of Y, which takes the value $I(y)$ at y, for all $y \in Y$.

2.3 Delaunay Triangulations

In order to explain some relevant properties of Delaunay triangulations, let Y denote a finite planar point set. First recall that a triangulation $\mathcal{T}(Y)$ of Y is a collection of triangles, whose vertex set is Y, whose union is $[Y]$, and for which any pair of two distinct triangles in $\mathcal{T}(Y)$ intersect at most at one common vertex or along one common edge.

- A Delaunay triangulation $\mathcal{D}(Y)$ of Y is a triangulation of Y, such that for any triangle in $\mathcal{D}(Y)$, the interior of its circumcircle does not contain any point from Y. This property is termed the *Delaunay property.*

- The Delaunay triangulation $\mathcal{D}(Y)$ of Y is unique, provided that no four points in Y are co-circular.

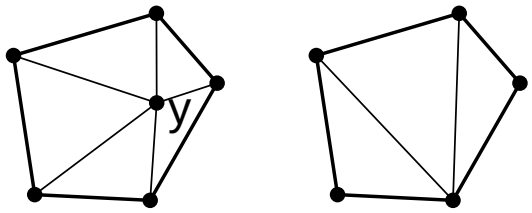
  Since neither the set X of pixels nor its subsets satisfy this condition, we initially perturb the pixel positions in order to guarantee unicity of the Delaunay triangulations of X and of its subsets. Each perturbed pixel corresponds to one unique unperturbed pixel. From now on, we denote the set of perturbed pixels by X, and the set of unperturbed pixels by $\tilde{X}$.

- For any $y \in Y$, $\mathcal{D}(Y \setminus y)$ can be computed from $\mathcal{D}(Y)$ by a *local* update. This follows from the Delaunay property, which implies that only the

cell $C(y)$ of y in $\mathcal{D}(Y)$ needs to be retriangulated. Recall that the cell $C(y)$ of y is the domain consisting of all triangles in $\mathcal{D}(Y)$ which contain y as a vertex. Figure 1 shows a vertex $y \in \mathcal{D}(Y)$ and the Delaunay triangulation of its cell $C(y)$.

- $\mathcal{D}(Y)$ provides a partitioning of the convex hull $[Y]$ of Y.

For further details on Delaunay triangulations, see the textbook [20].



(a) (b)

Figure 1: Removal of the vertex $y \in \mathcal{D}(Y)$, and the Delaunay triangulation of its cell $C(y)$. The five triangles of the cell $C(y)$ in (a) are replaced by the three triangles in (b).

2.4 Linear Splines over Delaunay Triangulations

Let Π_1 denote the space of linear bivariate polynomials. For any $Y \subset X$, we define the linear spline space $\mathcal{S}_Y$, containing all continuous functions over $[Y]$, whose restriction to any triangle $T \in \mathcal{D}(Y)$ is in Π_1, namely

$$\mathcal{S}_Y = \left\{ s \,:\, s \in C([Y]), s\big|_T \in \Pi_1 \text{ for all } T \in \mathcal{D}(Y) \right\}.$$

Any element in $\mathcal{S}_Y$ is referred to as a *linear spline* over $\mathcal{D}(Y)$. For given luminance values at the pixels of Y, $\{I(y) \,:\, y \in Y\}$, there is a unique linear spline interpolant $L(Y, I) \in \mathcal{S}_Y$ satisfying

$$L(Y, I)(y) = I(y), \qquad \text{for all } y \in Y.$$

For a fixed subset $Y \subset X$, we can take $\mathcal{S}_Y$ as an approximation space for the image I, defined over the domain $[X]$, provided that the convex hull $[Y]$ of Y coincides with $[X]$. Therefore, the initial perturbation of the pixels is such that the four corners of $\tilde{X}$ are unperturbed, and the other boundary pixels in $\tilde{X}$ are perturbed along the boundaries. Moreover, Y is required to contain the four corner pixels of $\tilde{X}$.

2.5 Significance Measures

The quality of image compression schemes is usually measured in `dB` (decibels) by the *Peak Signal to Noise Ratio*,

$$\mathrm{PSNR} = 10 * \log_{10}\left(\frac{2^r \times 2^r}{\bar{\eta}^2(Y,X)}\right).$$

The PSNR is equivalent to the reciprocal of the *mean square error* (MSE),

$$\bar{\eta}^2(Y,X) = \eta^2(Y,X)/|X|, \tag{2}$$

where

$$\eta(Y,X) = \sqrt{\sum_{x\in X} |L(Y,I)(x) - I(x)|^2}.$$

Therefore, to approximate the image, we wish to construct a subset $Y \subset X$, such that the *approximation error* $\eta(Y,X)$ is small.

The construction of a suitable such subset $Y \subset X$ is accomplished by Algorithm 1, with an appropriate definition for a least significant pixel. The most natural notion of a least significant pixel is given by the following definition.

Definition 1 *For $Y \subset X$, a pixel $y^* \in Y$ is said to be* least significant *in Y, iff*

$$\eta(y^*) = \min_{y\in Y} \eta(y),$$

where for any $y \in Y$,

$$\eta(y) = \eta(Y \setminus y, X)$$

is the significance *of the pixel y in Y.*

In a previous paper [10], we have already used the above definition for pixel removal. In this paper, we also consider least significant pixel pairs.

Definition 2 *For $Y \subset X$, a pair $\{y_1^*, y_2^*\} \subset Y$ of two pixels in Y is said to be* least significant *in Y, iff*

$$\eta(y_1^*, y_2^*) = \min_{\{y_1,y_2\}\subset Y} \eta(y_1, y_2),$$

where for any pixel pair $\{y_1, y_2\} \subset Y$, we denote by

$$\eta(y_1, y_2) = \eta(Y \setminus \{y_1, y_2\}, X),$$

its significance *in Y.*

As supported by our comparisons in Section 4, the following significance measure improves the pixel removal criterion of [10] considerably.

Definition 3 *For $Y \subset X$, a pixel $y^* \in Y$ is said to be* least significant *in Y, iff it belongs to a least significant pixel pair in Y, $\{y^*, y\} \subset Y$, and satisfies $\eta(y^*) \leq \eta(y)$.*

2.6 Equivalent Significance Measures

We introduce significance measures e_δ, equivalent to the significance measures η, in order to reduce the computational costs of Algorithm 1. To this end, we establish a useful relation between the significance measure for a pixel $y \in Y$,

$$e_\delta(y) = \eta^2(y) - \eta^2(Y, X),$$

and the significance measure for a pixel pair $\{y_1, y_2\} \subset Y$,

$$e_\delta(y_1, y_2) = \eta^2(y_1, y_2) - \eta^2(Y, X).$$

Any pixel pair $\{y_1, y_2\} \subset Y$ is either an edge of $\mathcal{D}(Y)$, $[y_1, y_2] \in \mathcal{D}(Y)$, or the two pixels y_1, y_2 are not connected in $\mathcal{D}(Y)$. In the latter case, the two cells $C(y_1)$ and $C(y_2)$ have no common triangle. Therefore, we have

$$\eta^2(y_2) - \eta^2(Y, X) = \eta^2(y_1, y_2) - \eta^2(y_1),$$

which implies

$$\begin{aligned} e_\delta(y_1, y_2) &= \eta^2(y_1, y_2) - \eta^2(Y, X) \\ &= \eta^2(y_1, y_2) - \eta^2(y_1) + \eta^2(y_1) - \eta^2(Y, X) \\ &= \eta^2(y_2) - \eta^2(Y, X) + \eta^2(y_1) - \eta^2(Y, X). \end{aligned}$$

This shows that for any pixel pair $\{y_1, y_2\} \subset Y$, with $[y_1, y_2] \notin \mathcal{D}(Y)$,

$$e_\delta(y_1, y_2) = e_\delta(y_1) + e_\delta(y_2). \tag{3}$$

Due to the simple representation (3), the maintenance of the significances $\{e_\delta(y_1, y_2) : \{y_1, y_2\} \subset Y\}$ can be reduced to the maintenance of $\{e_\delta(y_1, y_2) : [y_1, y_2] \in \mathcal{D}(Y)\}$ and $\{e_\delta(y) : y \in Y\}$.

Indeed, for the efficient implementation of Algorithm 1, we use two different priority queues, one for the significances e_δ of pixels, and one for the significances e_δ of edges in $\mathcal{D}(Y)$. Each priority queue has a least significant element (pixel or pixel pair) at its head, and is updated after each pixel removal. The resulting algorithm has complexity $\mathcal{O}(N \log N)$ [15]. For more details concerning the efficient maintenance of such priority queues, we refer to our paper [15].

2.7 Minimization of the Mean Square Error in $\mathcal{S}_Y$

In a post-processing to adaptive thinning, we further reduce the mean square error (2) by *least squares approximation* [1]. More precisely, we compute from the set $Y \subset X$ of significant pixels, output by Algorithm 1, and from the luminance values at the pixels in X the unique *best approximation* $L^*(Y, I) \in \mathcal{S}_Y$ satisfying

$$\sum_{x \in X} |L^*(Y, I)(x) - I(x)|^2 = \min_{s \in \mathcal{S}_Y} \sum_{x \in X} |s(x) - I(x)|^2.$$

Such a best approximation exists and is unique, since $\mathcal{S}_Y$ is a finite dimensional linear space and since $Y \subset X$.

The final approximation to the image I is the linear spline $L^*(Y, I) \in \mathcal{S}_Y$, determined by the set Y and the corresponding *optimal* luminances $I^*(y) = L^*(Y, I)(y)$, $y \in Y$.

2.8 Controlling the Mean Square Error

The image approximation scheme allows us to control the MSE (2) corresponding to the image approximation. This can be done during the performance of Algorithm 1 as follows.

For a given MSE value, $\bar{\eta}^*$, Algorithm 1 can be changed in order to terminate when for the first time, the MSE value corresponding to the current linear spline $L(X_p, I)$ is above $\bar{\eta}^*$, for some X_p in (1) (in this version of Algorithm 1, $n = p$ a posteriori). We take as the final approximation to the image the linear spline $L^*(X_{p+1}, I)$. Observe that $L^*(X_{p+1}, I)$ satisfies

$$\sum_{x \in X} |L^*(X_{p+1}, I)(x) - I(x)|^2 / |X_{p+1}| \leq \bar{\eta}^*,$$

as desired.

3 Image Compression

Our compression method replaces an image by its linear spline approximation $L^*(Y, I)$, corresponding to a set of significant pixels Y. The number of parameters, which determine the linear spline approximation, depends on the number $|Y|$ of significant pixels.

To code the approximated image, given by $L^*(Y, I)$, we code the information $\{(\tilde{y}, I^*(y)) : y \in Y\}$, where $\tilde{y}$ denotes the unperturbed *integer* pixel

corresponding to y. For this purpose, we have developed a customized scattered data coding scheme, which improves the one in our papers [8, 9]. We remark that the coding scheme of this paper is similar to the one in [11], but there are some subtle differences concerning effective coding of *unconnected* scattered data. Indeed, the coding scheme in [11] is primarily aiming at mesh compression, whereas the coding scheme of this paper is concerned with the compression of a Delaunay triangulation of scattered planar points, the latter being a task which requires coding the points only, but not their connectivities. This important detail, as well as other main ingredients of our coding scheme and further differences to the coding scheme in [11], is explained in the remainder of this section.

3.1 Theoretical Coding Costs

To code the information $\{(\tilde{y}, I^*(y)) : y \in Y\}$, we first apply a uniform quantization to the optimal luminances $\{I^*(y) : y \in Y\}$. This yields quantized symbols $\{Q(I^*(y)) : y \in Y\}$, corresponding to quantized luminance values $\{\tilde{I}(y) : y \in Y\}$, where, for some $s < r$, we use $\{0, 1, \ldots, 2^s - 1\}$ for the range of the quantized symbols.

Note that due to the uniqueness of the Delaunay triangulation $\mathcal{D}(Y)$, we do not need to code any connectivity information. We are only concerned with the coding of the elements of the set $\{(\tilde{y}, Q(I^*(y))) : y \in Y\} \in \mathcal{I}_n^s$, where

$$\mathcal{I}_n^s = \left\{ \{0, 1, \ldots, 2^s - 1\}^{\tilde{Z}} : \tilde{Z} \subset \tilde{X} \text{ and } |\tilde{Z}| = n \right\},$$

with $n = |Y|$.

The number of elements in $\mathcal{I}_n^s$ is $\binom{|X|}{n} \times 2^{s \times n}$. If we assume that every element of $\mathcal{I}_n^s$ has the same probability of occurrence, then the theoretical coding cost is

$$\log_2 \left(\binom{|X|}{n} \right) + s \times n. \tag{4}$$

This cost can be reduced by taking advantage of the geometric structure of the image. Indeed, the coding scheme of our compression method leads to lower costs for real images, as is observed in Table 1.

3.2 Scattered Data Coding Scheme

Our coding scheme comprises two consecutive steps: the coding of the scattered integer pixels in $\tilde{Y} = \{\tilde{y} : y \in Y\}$, followed by the coding of the quantized symbols $Q_Y = \{Q(I^*(y)) : y \in Y\}$.

Let us first explain the coding of the pixels in $\tilde{Y}$. This relies on a recursive splitting of the pixel domain $\Omega = [\tilde{X}]$. For the sake of simplicity, let us assume that Ω is a square domain of the form $\Omega = [0, 2^q - 1] \times [0, 2^q - 1]$.

A square subdomain $\omega \subset \Omega$ (initially $\omega = \Omega$) is split horizontally into two rectangular subdomains of equal size. A rectangular subdomain is split vertically into two square subdomains of equal size. The splitting terminates at subdomains which are either *empty*, i.e., not containing any pixel from $\tilde{Y}$, or *atomic*, i.e., of size 1×1.

This splitting process can be represented by a binary tree, whose nodes correspond to the subdomains. The root of the tree corresponds to Ω, and its leaves correspond to empty or atomic subdomains.

In each node of the tree, with a corresponding subdomain ω, we store the number $|\omega|$ of pixels from $\tilde{Y}$ contained in ω, i.e., $|\omega| = |\tilde{Y} \cap \omega|$. Note that for a parent node ω, and its two children nodes, ω_1 and ω_2, we have the relation $|\omega| = |\omega_1| + |\omega_2|$. This relation allows a non-redundant representation of the binary tree. The bitstream, representing the tree, is constructed by a Huffman code. This is in contrast to the coding scheme in [11], where an optimal adaptive arithmetic encoder is employed, with assuming a uniform probability distribution.

Now let us turn to the coding of the quantized symbols in Q_Y. We first split the image domain Ω into a small number of square subdomains of equal size. For each subdomain, the pixels from $\tilde{Y}$ contained in it are ordered linearly, such that close pixels in the image domain are close in this ordering.

The quantized symbol of any pixel in this ordering is coded relative to the quantized symbol of its predecessor, except for that of the first pixel. The coding is done by a Huffman code.

Case	Coding costs (in bits)		
(see Figure 4)	$n = 7,200$	$n = 15,000$	$n = 30,000$
Theoretical (4)	83,576	157,911	284,515
Fruits	77,280	139,864	241,584
Peppers	78,168	141,800	243,576
Lena	77,136	139,136	237,880

Table 1: Coding costs for $s = 5$.

Unlike the coding scheme in [11], our coding scheme does not assume equal probabilities for the possible numbers of significant pixels in a cell. In Table 1, the observed coding costs obtained by using our method are compared with the theoretical coding costs in (4). Observe that the ratio between the cost of our coding scheme and the theoretical cost in (4) is decreasing with increasing n. This is due to a higher correlation between luminance values at significant pixels for larger sets Y.

3.3 Reconstruction at the Decoder

At the decoder, the reconstruction of the compressed image from the information $\{(\tilde{y}, Q(I^*(y))) : y \in Y\}$ is accomplished by four steps.

- The set $\{\tilde{y} : y \in Y\}$ is perturbed by the same perturbation rules used at the encoder to yield the set of perturbed pixels Y.
- The *unique* Delaunay triangulation $\mathcal{D}(Y)$ of Y is computed.
- The unique linear spline $L(Y, \tilde{I}) \in \mathcal{S}_Y$ satisfying

  $$L(Y, \tilde{I})(y) = \tilde{I}(y), \quad \text{for all } y \in Y,$$

  is constructed from the quantized luminance values $\{\tilde{I}(y) : y \in Y\}$.
- The reconstructed image is given by

  $$\tilde{I} = \{(\tilde{x}, L(Y, \tilde{I})(\tilde{x})) : \tilde{x} \in \tilde{X}\}.$$

4 Comparison with JPEG2000

We compare the performance of our compression method $\mathbf{AT}^*$ with that of the powerful method EBCOT [24], which is the basic algorithm in the standard JPEG2000 [25], using the Kakadu implementation.

In the test examples below, we let $s = 5$, and so the range for the quantized symbols is $\{0, 1, \ldots, 31\}$. In each comparison, the compression rate, measured in *bits per pixel* (`bpp`), is fixed. The quality of the resulting reconstructions is measured by their PSNR values, and for a small set of comparisons it is further evaluated by their visual quality. We provide rate-distortion curves for comparing the performance of the two methods on three real images.

4.1 Geometric Images

We first consider two artificial test images, *Chessboard* and *Reflex*, each of small size 128×128. These test images are displayed in Figures 2 and 3.

The purpose of the comparisons with these two test images is two-fold. Firstly, we indicate why the compression method of this paper, $\mathbf{AT}^*$, is superior to that of [10], here referred to as $\mathbf{AT}^-$. Secondly, we show the good performance of $\mathbf{AT}^*$ on texture-free images with sharp edges.

A comparison between JPEG2000, $\mathbf{AT}^*$ and $\mathbf{AT}^-$ is done on the test image Chessboard. In this example, 299 significant pixels were selected by $\mathbf{AT}^*$ and $\mathbf{AT}^-$. The compression method $\mathbf{AT}^*$ selects an *optimal* subset Y of 299 most significant pixels, so that $L(Y, I)(x) = I(x)$ for all $x \in X$. In consequence, the corresponding mean square error (2) is zero, but due to the quantization of the luminance values, the mean square error of the compressed image $\tilde{I}$ is slightly increased, yielding a PSNR of 45.15 dB. The almost exact reconstruction provided by $\mathbf{AT}^*$ is shown in Figure 2. JPEG2000 leads to an inferior PSNR of only 18.68 dB, and our previous method $\mathbf{AT}^-$ leads to a PSNR of 15.24 dB. Moreover, the *visual quality* of the resulting reconstructions by JPEG2000 and $\mathbf{AT}^-$ is rather poor, as depicted in Figure 2.

We can explain the superiority of $\mathbf{AT}^*$ over $\mathbf{AT}^-$ in this case by comparing their removal strategies. $\mathbf{AT}^*$ allows the removal of a pixel from an edge $[y_1, y_2] \in \mathcal{D}(Y)$, whose corresponding significance $e_\delta(y_1, y_2)$ is small, even if the significances $e_\delta(y_1)$ and $e_\delta(y_2)$ are large. This is typically the case for pixel pairs $\{y_1, y_2\}$ whose corresponding edge $[y_1, y_2]$ crosses the boundary between two squares of the chessboard in nearly perpendicular direction, away from the corners. In this case, $\mathbf{AT}^*$ removes a pixel, either y_1 or y_2, from the edge $[y_1, y_2] \in \mathcal{D}(Y)$, whereas $\mathbf{AT}^-$ is too short-sighted to make such removals, and keeps both y_1 and y_2, but removes pixels near the corners of the chessboard squares. In contrast, $\mathbf{AT}^*$ keeps the pixels near such corners.

We remark that the compression method $\mathbf{AT}^*$ of this paper shows a much better performance than our previous compression method $\mathbf{AT}^-$ of [10] for *all* test images which were ever considered in our comparisons, in particular for all test images presented in this paper, see the results in Table 2. Therefore, for the following test cases, we prefer to focus on the comparison between our improved compression method $\mathbf{AT}^*$ and JPEG2000.

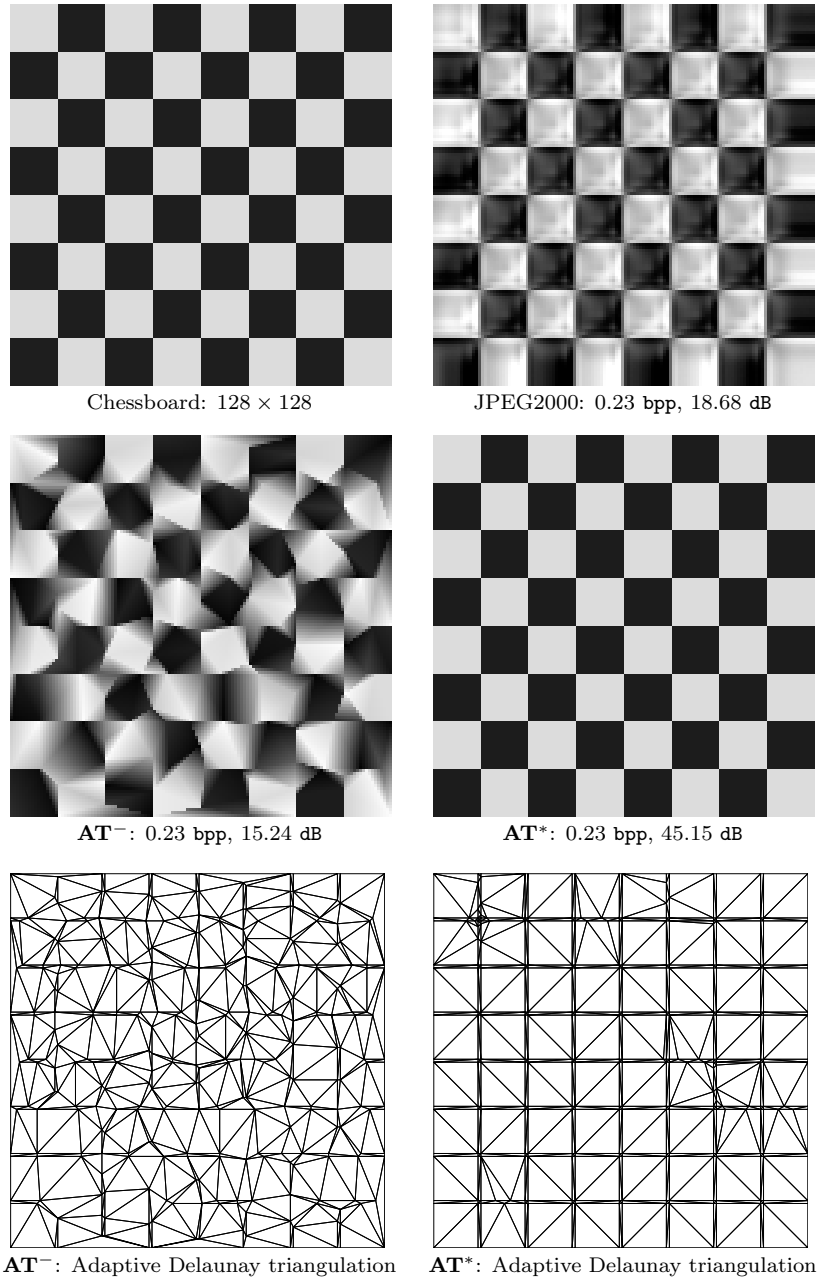
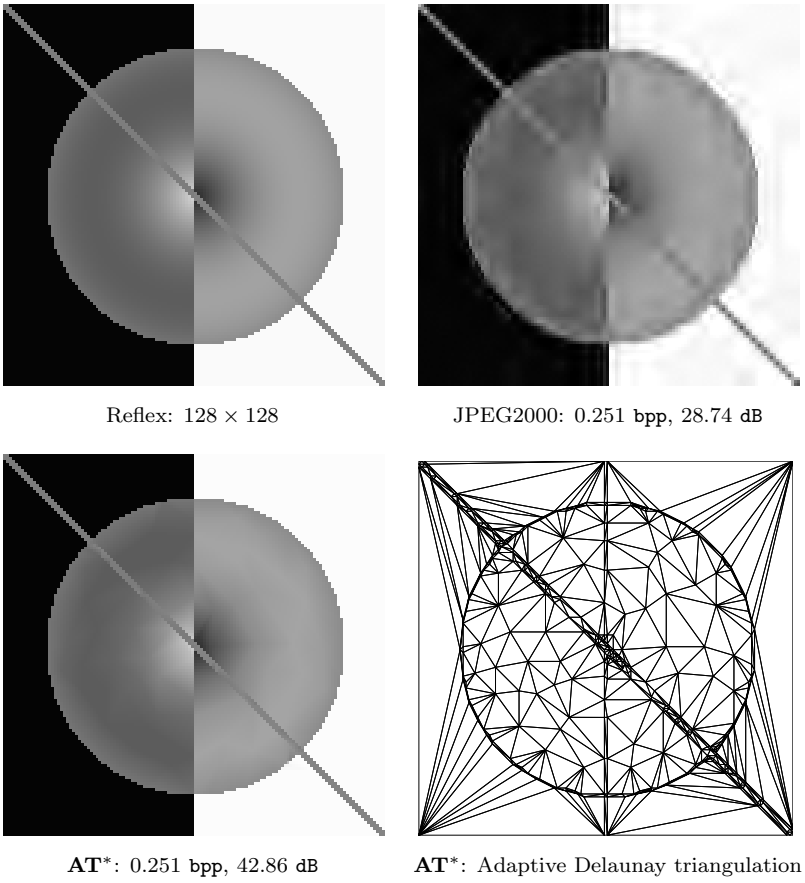


Figure 2: Geometric test image Chessboard.



Reflex: 128×128 | JPEG2000: 0.251 bpp, 28.74 dB

AT*: 0.251 bpp, 42.86 dB | **AT***: Adaptive Delaunay triangulation

Figure 3: Geometric test image Reflex.

For the other geometric test image, Reflex, we fix the compression rate to 0.251 bpp. The resulting reconstructions by JPEG2000 and **AT*** are displayed in Figure 3. **AT*** yields the PSNR value 42.86 dB, whereas JPEG2000 provides the inferior PSNR value 28.74 dB. Hence, with respect to this quality measure, **AT*** is much better. Moreover, the reconstruction by **AT*** provides a superior *visual quality* to that of the reconstructed image by JPEG2000 (see Figure 3). Indeed, **AT*** manages to localize the sharp edges of the test image Reflex. Moreover, it avoids undesired oscillations near the edges, unlike JPEG2000.

The fact that **AT*** outperforms JPEG2000 on the test images Chessboard and Reflex at low bit rates is not too surprising, insofar as **AT*** was particularly designed to capture geometric feature lines. But a somewhat

fairer comparison between $\mathbf{AT}^*$ and JPEG2000 is presented in the following subsection, where three popular real images are used. We have recorded the results of the two geometric examples of this subsection, along with those of four comparisons on the three real images in Table 2.

		PSNR (in dB)			
Test Case	bitrate (bpp)	JPEG2000	$\mathbf{AT}^-$	$\mathbf{AT}^*$	$\lvert Y\rvert$
Chessboard	0.230	18.68	15.24	*45.15*	299
Reflex	0.251	28.74	41.94	*42.86*	384
Fruits	0.180	31.88	31.83	*32.38*	4,044
	0.500	*36.44*	35.81	36.23	13,800
Peppers	0.154	31.94	31.78	*32.33*	3,244
Lena	0.150	*32.04*	30.95	31.48	3,244

Table 2: Comparison between JPEG2000, $\mathbf{AT}^-$, and $\mathbf{AT}^*$.

4.2 Popular Real Images

We consider three popular real images of size 512×512. The first image, called *Fruits*, is a part of the standard test case *Bike*, used in [25]. The two other images are the standard test cases *Peppers* and *Lena*. The three images are displayed in Figure 4.

For each image, we compare the PSNR values provided by the two compression methods at various bitrates. These comparisons are summarized by the rate-distortion curves in Figure 4. Moreover, the results of four comparisons are shown in Table 2, and the corresponding reconstructions in Figures 5 and 6, together with the adaptive Delaunay triangulations output by $\mathbf{AT}^*$.

From the rate-distortion curves in Figure 4 and from Table 2 we conclude that for the images Fruits and Peppers the PSNR values obtained by $\mathbf{AT}^*$ are slightly larger than those obtained by JPEG2000 at low bitrates, and are slightly smaller at higher bitrates. For the image Lena the PSNR values obtained by $\mathbf{AT}^*$ are slightly smaller than those obtained by JPEG2000 at low bitrates, with the difference between the PSNR values growing with increasing bitrate.

The comparisons in Figures 5 and 6 exhibit *visual* differences between the compressed images output by the two methods. Observe that $\mathbf{AT}^*$ man-

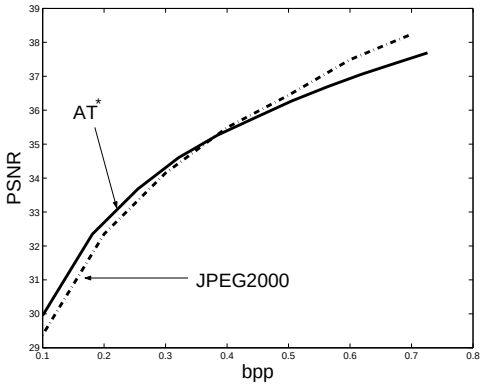
ages to denoise the test images quite successfully and to capture important features of the images, such as sharp edges and silhouettes. This is because the Delaunay triangulations, output by $\mathbf{AT}^*$, are well-adapted to the geometry of the images (see Figures 5 and 6). This, however, leads to higher coding costs in textured regions of the images, which partly explains the inferior performance of $\mathbf{AT}^*$ (in comparison with JPEG2000) for the test image Lena.

Our final conclusion is that the compression method $\mathbf{AT}^*$ recovers the image geometry very well, and it provides a fairly good alternative to existing standard methods, including JPEG2000, especially for texture-free images with distinctive geometric features.

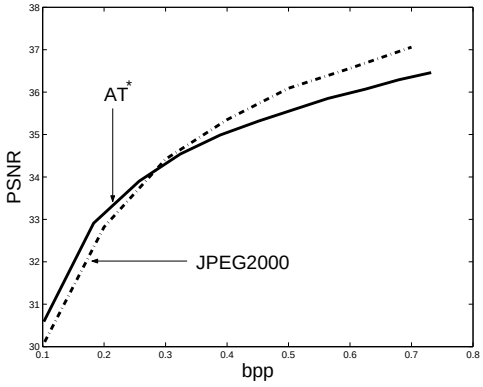
Acknowledgment

We wish to thank the contributors to our previous work on image compression, which constitutes the basis for the compression method of this paper. We also wish to thank Shai Dekel for his useful support concerning the numerical tests. The authors were partly supported by the European Union within the project MINGLE (Multiresolution in Geometric Modelling), HPRN-CT-1999-00117.

Fruits: 512×512



Peppers: 512×512



Lena: 512×512

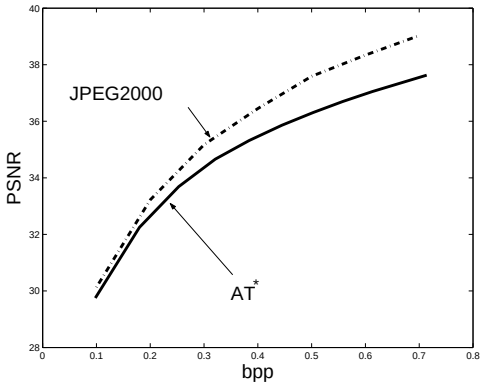


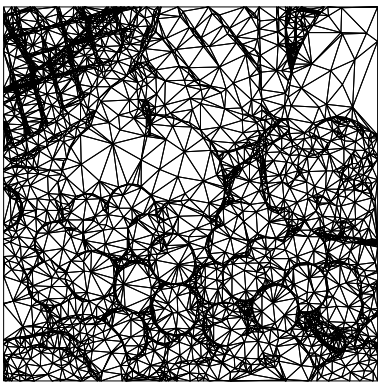
Figure 4: Images Fruits, Peppers, and Lena with rate-distortion curves for **AT*** and JPEG2000.

JPEG2000: 0.18 bpp, 31.88 dB

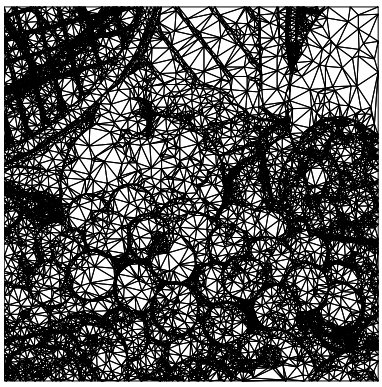
JPEG2000: 0.5 bpp, 36.44 dB

AT*: 0.18 bpp, 32.38 dB

AT*: 0.5 bpp, 36.23 dB



AT*: Adaptive Delaunay triangulation



AT*: Adaptive Delaunay triangulation

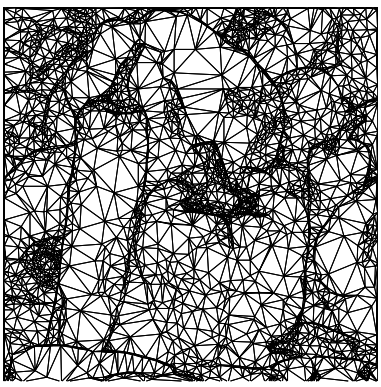
Figure 5: Comparisons with Fruits (low and high bitrate).

JPEG2000: 0.154 bpp, 31.94 dB

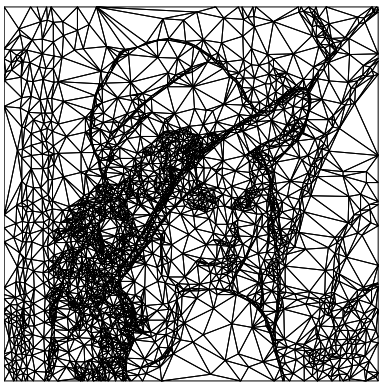
JPEG2000: 0.15 bpp, 32.04 dB

AT*: 0.154 bpp, 32.33 dB

AT*: 0.15 bpp, 31.48 dB



AT*: Adaptive Delaunay triangulation



AT*: Adaptive Delaunay triangulation

Figure 6: Comparisons with Peppers and Lena (low bitrate).

References

[1] Å. Björck, Numerical Methods for Least Squares Problems, SIAM, Philadelphia, 1996.

[2] E. Candès and D. Donoho, "Curvelets and curvilinear integrals", J. Approx. Theory **113**, pp. 59-90, 2001.

[3] V. Chandrasekaran, M. Wakin, D. Baron, and R. Baraniuk, "Compression of higher dimensional functions containing smooth discontinuities", Conference on Information Sciences and Systems, Princeton, NJ, March 2004.

[4] A. Cohen, "Applied and computational aspects of nonlinear wavelet approximation", *Multivariate Approximation and Applications*, N. Dyn, D. Leviatan, D. Levin, and A. Pinkus (eds.), Cambridge University Press, Cambridge, pp. 188–212, 2001.

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, 2nd edition, MIT Press, Cambridge, Massachusetts, 2001.

[6] G. M. Davis and A. Nosratinia, "Wavelet-based image coding: an overview", in *Appl. Comp. Control, Signal & Circuits*, B. N. Datta (ed.), Birkhauser, pp. 205–269, 1999.

[7] S. Dekel, N. Dyn, and R. Kazinnik, "Low bit-rate image coding using adaptive geometric splines approximation", in preparation.

[8] L. Demaret and A. Iske, "Scattered data coding in digital image compression", in *Curve and Surface Fitting: Saint-Malo 2002*, A. Cohen, J.-L. Merrien, and L. L. Schumaker (eds.), Nashboro Press, Brentwood, 107–117, 2003.

[9] L. Demaret and A. Iske, "Advances in digital image compression by adaptive thinning", *Annals of the MCFA* **3**, 105–109.

[10] L. Demaret, N. Dyn, M. S. Floater, and A. Iske, "Adaptive thinning for terrain modelling and image compression", in *Advances in Multiresolution for Geometric Modelling*, N. A. Dodgson, M. S. Floater, and M. A. Sabin (eds.), Springer-Verlag, Heidelberg, pp. 321–340, 2004.

[11] O. Devillers and P.-M. Gandoin, "Geometric compression for interactive transmissions", Proc. of IEEE Visualization 2000, 319–326.

[12] R. DeVore, "Nonlinear approximation", Acta Numerica **7**, pp. 51–150, 1998.

[13] M. N. Do and M. Vetterli, "The contourlet transform: an efficient directional multiresolution image representation", to appear in *IEEE Transactions on Image Processing*.

[14] D. Donoho, "Wedgelets: nearly-minimax estimation of edges", Annals of Stat., vol. 27, pp. 859–897, 1999.

[15] N. Dyn, M. S. Floater, and A. Iske, "Adaptive thinning for bivariate scattered data", J. Comput. Appl. Math. **145**(2), pp. 505–517, 2002.

[16] Y. Eldar, M. Lindenbaum, M. Porat, and Y.Y. Zeevi, "The farthest point strategy for progressive image sampling", IEEE Trans. Image Processing **6**(9), pp. 1305–1315, Sep. 1997.

[17] E. LePennec and S. Mallat, "Sparse geometric image representation with bandelets", to appear in *IEEE Transactions on Image Processing*.

[18] S. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation", IEEE Trans. Pattern Anal. Machine Intell., vol. 11, pp. 674–693, July 1989.

[19] B. Matei and A. Cohen, "Compact representation of images by edge adapted multiscale transforms", Proceedings of IEEE International Conference on Image Processing, Tessaloniki, October 2001.

[20] F. P. Preparata and M. I. Shamos, Computational Geometry, Springer, New York, 1988.

[21] J. K. Romberg, M. Wakin, and R. Baraniuk, "Multiscale wedgelet image analysis: fast decompositions and modeling", IEEE International Conference on Image Processing, September 2002.

[22] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees", IEEE Trans. Circuits and Systems for Video Technology **6**, pp. 243–250, 1996.

[23] M. Shapiro, "An embedded hierarchical image coder using zerotrees of wavelet coefficients", IEEE Trans. on Signal Processing **41**, pp. 3445–3462, 1993.

[24] D. Taubman, "High performance scalable image compression with EBCOT", IEEE Trans. on Image Processing, pp. 1158–1170, July 2000.

[25] D. Taubman, M. W. Marcellin, JPEG2000: Image Compression Fundamentals, Standards and Practice, Kluwer, Boston, 2002.

[26] Z. Xiong, K. Ramchandran, M. Orchard, and K. Asai, "Wavelet packet image coding using space frequency quantization", IEEE Trans. Image Processing, vol. 7, pp. 892–898, June 1998.

Authors' addresses:

Laurent Demaret
Forschungszentrum für Umwelt und Gesundheit (GSF)
Institut für Biomathematik und Biometrie (IBB)
D-85764 Neuherberg, GERMANY
laurent.demaret@gsf.de

Nira Dyn
Tel-Aviv University
School of Mathematical Sciences
Tel Aviv 69978, ISRAEL
niradyn@post.tau.ac.il

Armin Iske
University of Leicester
Department of Mathematics
Leicester LE1 7RH, UK
iske@mcs.le.ac.uk