Low bit-rate image coding using adaptive geometric piecewise polynomial approximation

Roman Kazinnik, Shai Dekel, and Nira Dyn

***Abstract*— Today, the state-of-the art image coding algorithms are transform-based coders. A good example is Taubman's EBCOT wavelet coding algorithm, which is the basis of the JPEG2000 standard which provides excellent results in terms of rate-distortion performance. However, there is a significant research activity of 'geometric' multiscale sparse representation and coding methods, i.e., methods that try to exploit the geometry of the edge singularities in an image (if exist).**

In this work, we depart from transform-based methods and, in some sense, from the multi-scale framework and choose to draw on recent developments in the theory of approximation with adaptive multivariate piecewise polynomials. We give details of a surprisingly simple image coding algorithm and show examples where our algorithm of *Geometric Piecewise Polynomials* (GPP) outperforms the state-of-the-art wavelet coding in the low bit-rate range. Employing piecewise polynomials for image coding is not a new idea, in the past such algorithms were known as *Second generation* coding. To the best of our knowledge, those methods did not report competitive compression results. In this work we will highlight some new key ideas that allowed us to improve upon these results.

***Index Terms*— Image coding, non-linear approximation.**

Shai Dekel works at GE Healthcare, Israel.

N. Dyn and R. Kazinnik are with Tel-Aviv University, School of Mathematical Sciences.

I. INTRODUCTION

From the mid 80s, there have been many attempts to design 'Second Generation' image coding techniques that exploit the geometry of edge singularities of an image. The reader may consult the survey [21]. To this day, almost all of the proposed 'Second Generation' algorithms are not competitive with state of the art (dyadic) wavelet coding [22], [23], [27]. In one of the outstanding 'Second Generation' methods [14], Froment and Mallat constructed multi-scale wavelet-like edge detectors and showed how to reconstruct a function from the responses of a sparse collection of these detectors. They reported good coding results at low bit-rates. There are coding algorithms that are geometric enhancements of existing wavelet transformed-based methods, where wavelet coefficients are coded using geometric context modelling [28]. In a recent work [23], the authors enhance classical wavelet coding by detecting and coding the strong edges separately and then using wavelets to code a residual image. Candès and Donoho [1] constructed Curvelets, a bivariate transform designed to capture local multi-scale directional information. Do and Vetterli's construction of Contourlets [12], is similar but is a purely discrete construction. Cohen and Matei [4] also showed a discrete construction of an edge-adapted transform that is closely related to nonlinear Lifting [3]. All of these constructions are redundant, i.e., the output of the discrete transform implementations produces more coefficients than the original input data. The possibility to use these new transforms to outperform wavelet coding is still an on-going research. LePennec and Mallat [18] recently applied their 'Bandelets' algorithm to image coding, where a warped-wavelet transform is computed to align with the geometric flow in the image and the edge singularities are coded using one-dimensional wavelet-type approximations. Previous work that we find to be the closest to ours are the papers by Shukla, Dragotti, Do and Vetterli [24] (see also [20]), Dekel and Leviatan [6], and Demaret, Dyn and Iske [8]. The authors in [24] show how to encode a very geometrically adaptive piecewise polynomial approximation on the one hand while using a nested dyadic square data-structure to avoid significant over-head in bit allocation of the 'side information' required to encode such highly adaptive data.

Our approach departs from the framework of harmonic analysis, which is the theoretical basis for transform based methods and even from the more general framework of multi-scale geometric processing, and is based on the *Geometric Piecewise Polynomials*(GPP) method introduced in [17]. Our prototype approximation scheme is the conceptually simpler piecewise polynomial approximation. For $f \in L^p([0,1]^d)$, the degree of piecewise polynomials approximation is

$$\sigma_{n,r}(f)_p = \inf_{S \in \sum_{n,r}} \|f - S\|_{L^p([0,1]^d)}, \qquad (1)$$

where $\sum_{n,r}$ is the collection

$$\sum_{k=1}^{n} 1_{\Delta_k} \mathrm{P}_k, \qquad (2)$$

and Δ_k are simplices (segments for $d = 1$, triangles for $d = 2$, etc.) with disjoint interiors, such that $\bigcup_{k=1}^{n} \Delta_k = [0,1]^d$, and P_k, $1 \le k \le n$, are polynomials of total degree $r - 1$. There are computational variants of the prototype where the domains are polytopes and may satisfy an inclusion relation (see e.g. [6]).

It is known that in the univariate case, wavelets and piecewise polynomials have the same (theoretical) performance since their corresponding approximation spaces are identical [11]. However, in the multivariate case this is no longer true and at least theoretically, piecewise polynomials outperform wavelets whenever the approximated function has some 'structure', i.e., edge singularities that are smooth in some weak sense [7], [16]. Indeed, in any dimension d, an n-term sum of compactly supported spline-wavelets can be thought of as a special case of (2), where the simplices $\{\Delta_k\}$ triangulate the 'dyadic ring' structure of the supports of the spline-wavelets in the sum (see [7] for details in the bivariate case. The case of arbitrary dimension is similar).

We now highlight three key concepts of our GPP algorithm. First, we model an image as a piecewise smooth bivariate function with curve singularities of weak type smoothness. We apply a segmentation algorithm derived from the $\widetilde{K}$-functional of [7].We then approximate the detected edge singularities, where the distortion is resolved with *bands* (Section III-D). Finally, we address the inefficiency of polynomial approximation of smooth functions over nonconvex domains (Section III-B).

II. Outline of the GPP algorithm

The steps in our coding algorithm are derived from the theory detailed in [7]. Given an image we first apply the segmentation algorithm of Section III-A that captures the significant edges in the images. Given a target bit-rate, the edges are then pruned using the $\widetilde{K}$-functional model. The pruned edges are then encoded using the lossy adaptive algorithm of Section III-C, which is closely related to adaptive polygonal curve approximation. To better fit the $\widetilde{K}$-functional model, we apply further partitioning of the segmentation domains. This step, detailed in Section III-B, is almost a 'pure-geometric' domain recursive algorithm, that the decoder can apply without requiring too much information from the encoder. In Section III-D we describe how the segmentation curves are allocated some width, so that they become 'bands'. In Section III-E we give details of the last step of the algorithm where polynomial approximation, quantization and coding is performed over each of the remaining subdomains. Finally, in Section III-G we provide experimental results and compare performance of the GPP algorithm with Kakadu implementation [26] of the EBCOT algorithm [25].

III. GPP Algorithm

A. Segmentation into subdomains of smoothness

The first step of the GPP algorithm is a segmentation procedure whose goal is to approximate the solution of the $\widetilde{K}$-functional introduced in [7].

For the sake of completeness we recall a simple form of the $\widetilde{K}$-functional. Let $f \in L^2([0,1]^2)$ and $t > 0$. Take any segmentation Λ of $[0,1]^2$, as in Figure 1(b), defined by continuous curves $b_j : [0,1] \to [0,1]^2$, $1 \le j \le n_E(\Lambda)$, each of finite length, denoted by $len(b_j)$. The curves may intersect only at endpoints and a subset of the curves should compose the boundary of $[0,1]^2$. Thus, the curves partition $[0,1]^2$ into open domains, Ω_k , $1 \le k \le n_F(\Lambda)$.

We now attach to each curve b_j a weight t_j and we say that the partition Λ is in $\Lambda(t)$ if $\sum_{j=1}^{n_E(\Lambda)} t_j^{-1} < t^{-1}$. The $\widetilde{K}$-functional of order $r \in \mathbf{N}$ is defined as

$$\widetilde{K}_r(f,t)_2 := \inf_{\Lambda\in\Lambda(t)} \{ \sum_{j=1}^{n_E(\Lambda)} len(b_j) K_2(b_j, t_j^2)_{\infty,1} + \sum_{k=1}^{n_F(\Lambda)} K_r(f,\Omega_k)_2^2 \}^{1/2} , \tag{3}$$

where

$$K_r(b,t)_{\infty,1} = \inf_{g\in C^2[0,1]} \|b-g\|_\infty + t\|g''\|_{L_1} ,$$

measures the (weak-type) smoothness of curves, and

$$K_r(f,\Omega)_2 = \inf_{g\in H^r(\Omega)} \|f-g\|_{L_2(\Omega)} + diam(\Omega)^r |g|_{H^r(\Omega)} , \tag{4}$$

measures the (weak-type) smoothness of a surface piece, where $H^r(\Omega)$ is the Sobolev space equipped with the seminorm

$$|g|_{H^r(\Omega)} = \sum_{|\alpha|=r} \|D^\alpha g\|_{L^2(\Omega)} \tag{5}$$

For a curve $b : [0,1] \to [0,1]^2$, the quantity $K_2(b_j,\cdot)_{\infty,1}$ is small if the curve is smooth, for example, if the L^1-norm of its second derivative is small. It is also small if the curve is only piecewise smooth, with a 'small' number of pieces and it is identically zero for a line segment. The quantity $K_r(f,\Omega)_2$ ([10]) is small if f is smooth in Ω and is identically zero if the function is a bivariate polynomial of degree $< r$ in Ω. The reader should have in mind the case where for a sufficiently small t, the curves b_j, of a near-optimal partition $\Lambda \in \Lambda(t)$, align on the curve singularities of the function, and where over the segmentation domains, Ω_k, the function is smooth. The novelty of the $\widetilde{K}$-functional is the way it combines the smoothness gauges of the curve and surface to give a geometric generalization of the classical K-functional. More importantly, in [7] it is shown that, roughly speaking, the quantity $\widetilde{K}_r(f,n^{-1})_2$ is equivalent to the approximation error $\sigma_{n,r}(f)_2$ defined in (1). Lastly, it is also important to note the close relationship between the $\widetilde{K}$-functional and the well-known Mumford-Shah functional [7].

Therefore, since a segmentation which is a near-minimizer of (3), leads to a construction of a good piecewise polynomial approximation, we may conclude that designing an algorithm that tries to find such a segmentation is the key to good performance of our coding algorithm. This task is still an on-going research project, however, for the purpose of low-bit rate coding, we found out that the following simple and heuristic algorithm works sufficiently well. First we apply the well known Gaussian zero-crossing segmentation algorithm, also known as the 'Laplacian-Gaussian' (LoG), with a relatively small width, ϵ_G, of the Gaussian kernel (see e.g. [15], Section 9.4). The idea here is to pick out the edge singularities of the image that are not noise or high-frequency texture, which indeed, the Gaussian kernel, smoothes out sufficiently well. One of the main properties of the zero-crossing algorithm is connectivity, i.e., the segmentation pixels combine to form continuous segmentation curves. Observe, that in correlation with the $\widetilde{K}$-functional the LoG segmentation produces smooth segmentation curves that approximate wiggly edge singularities. In Figure 1(b), we see the segmentation produced at this initial step.

With the minimization of the $\tilde{K}$-functional (3) in mind, we now prune the collection of segmentation curves. First we prune away the curves where the norm of the image gradient is below some threshold, since these are curves that do not represent significant edges of the image. Then, we sort the remaining curves based on their smoothness, using the following formula

$$len\,(b)\,\|b''\|_1\,,$$

which is a simplification of a 'curve smoothness' term appearing in (3). Thus, we prune away the curves whose relative higher curvature impact the $\tilde{K}$-functional the most and equivalently, whose encoding requires a higher bit allocation budget. In Figure 1(b) we see the initial segmentation of the Cameraman image and in Figure 1(c) the pruned segmentation.

Obviously, the amount of pruning relates to the target bit-rate we wish to achieve in our coding algorithm. The pruned segmentation can serve as a good basis for coding of geometric images, for example, graphic-art images that are simple piecewise constant with spline edge singularities. Until now we have not fully taken into account the 'surface smoothness' over the domain. Further partitioning needs to take place as explained in the next section.

B. *Convex driven binary tree partitioning of the segmentation domains*

The purpose of this step is to improve the segmentation of step III-A by further partitioning of the domains into 'almost convex' subdomains. In principal, this step complements stage III-A that did not fully take into account the functional (3).

We now review the basics of the underlying theory that motivates this step. It is known from the theory that polynomial approximation over multivariate domains is determined by both the smoothness of the function and domain's geometry. Evidently, it is difficult to approximate a non-smooth function by low-order polynomials, but it is also difficult to approximate a smooth function over highly non-convex domain.

Let $\Pi_{r-1} = \Pi_{r-1}(\mathcal{R}^d)$ denote the multivariate polynomials of total degree $r-1$ (order r). Given a bounded domain $\Omega \subset \mathcal{R}^d$, we define the degree of approximation by piecewise polynomials of a function $f \in L^2(\Omega)$

$$E_{r-1}(f,\Omega)_2 = \inf_{P\in\Pi_{r-1}} \|f-P\|_{L_2(\Omega)}. \qquad (6)$$

As the next result shows, polynomial approximation over convex domains is well characterized by the classical K-functional (4).

Theorem 3.1: [5] For all convex domains $\Omega \in \mathrm{R}^d$ and functions $f \in L^2(\Omega)$,

$$K_r(f,\Omega)_2 \le E_{r-1}(f,\Omega)_2 \le C(r,d)K_r(f,\Omega)_2 \qquad (7)$$

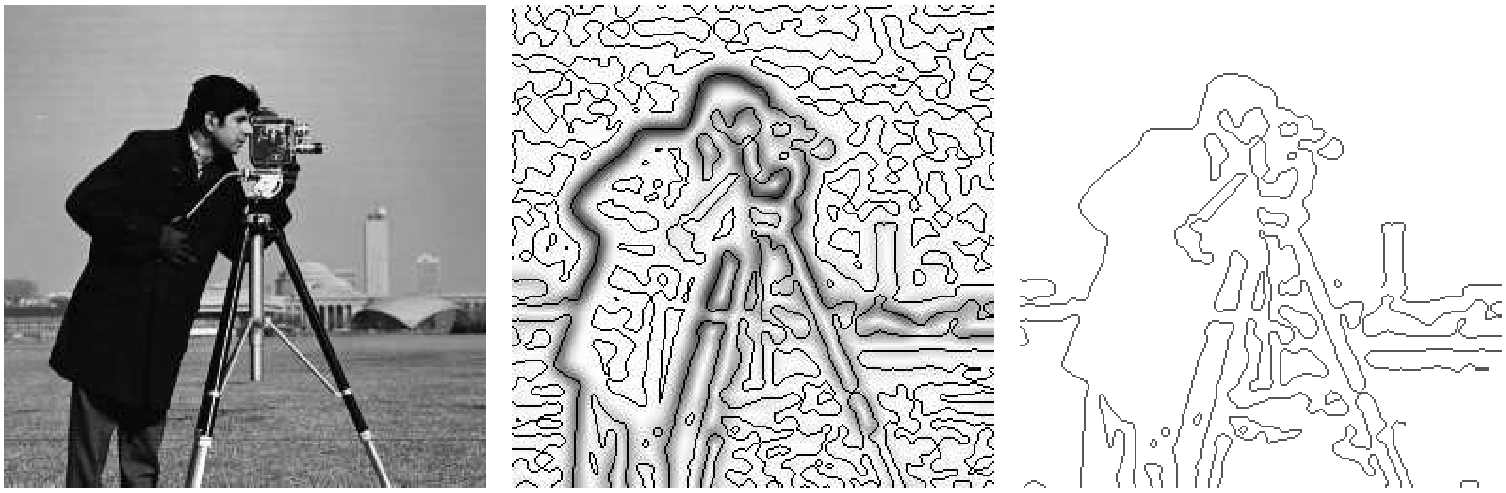

(a) Cameraman (b) Overlay of the image gradient and the zero-crossing (c) Pruned segmentation

Fig. 1. Pruning of a small width Gaussian zero-crossing of the Cameraman image

Whenever the domain is not convex, (7) becomes

$$K_r(f,\Omega)_2 \le E_{r-1}(f,\Omega)_2 \le C(r,d,p,\Omega)K_r(f,\Omega)_2 \ , \tag{8}$$

where the constant on the right hand side inequality further depends on the geometry of the domain. Indeed, in Figure 2(a) we see a function that is very smooth in an open ring-shaped domain. However, for this domain the right hand side constant in (8) is large. Indeed, as we see in Figure 2(b), a quadratic polynomial approximation of this function is of poor quality. When we partition the domain into two 'more-convex' domains, the approximation significantly improves (Figure 2(c)). This is well understood, since polynomials are smooth function over the entire plane.

Thus, equipped with the understanding that 'convex is good' when the approximation tool at hand are the classical polynomials, we apply a convex driven binary partitioning algorithm. The algorithm recursively subdivides the segmentation domains we get from step III-A, until they are partitioned into 'more convex' subdomains with a satisfactory polynomial approximation.

Since the binary partitions need to be encoded, it is advantageous that the partitioning algorithm be pure geometric. Then, once the decoder receives a 'subdivide' bit from the encoder, for a given domain, it uses the decoded approximated segmentation, and applies a pure-geometric subdivision algorithm without any further information from the encoder. This is closely related to the dyadic square partitioning, that is widely used in many works. For example in [24] once the decoder is 'informed' by the encoder that a dyadic square must be subdivided, it immediately knows that it must be subdivided into four smaller dyadic squares.

Here is the algorithm description. Given a planar domain Ω, we denote its convex hull as H and the complement to the convex hull as $C = H - \Omega$. Notice that the subset C contains essentially a number of disjoint connected components, which we denote as $\mathcal{C}_i$: $C = \bigcup \mathcal{C}_i$. For each connected component let

$$\mu_i = \max_{P_1,P_2 \in \partial\mathcal{C}_i} \frac{\rho(P_1,P_2)}{\|P_1 - P_2\|}, \ \textit{where}$$

$$\rho(P_1,P_2) = \inf_{\gamma \in C([0,1]\to\Omega)} \{length(\gamma);\ \gamma(0)=P_1,\ \gamma(1)=P_2\},$$

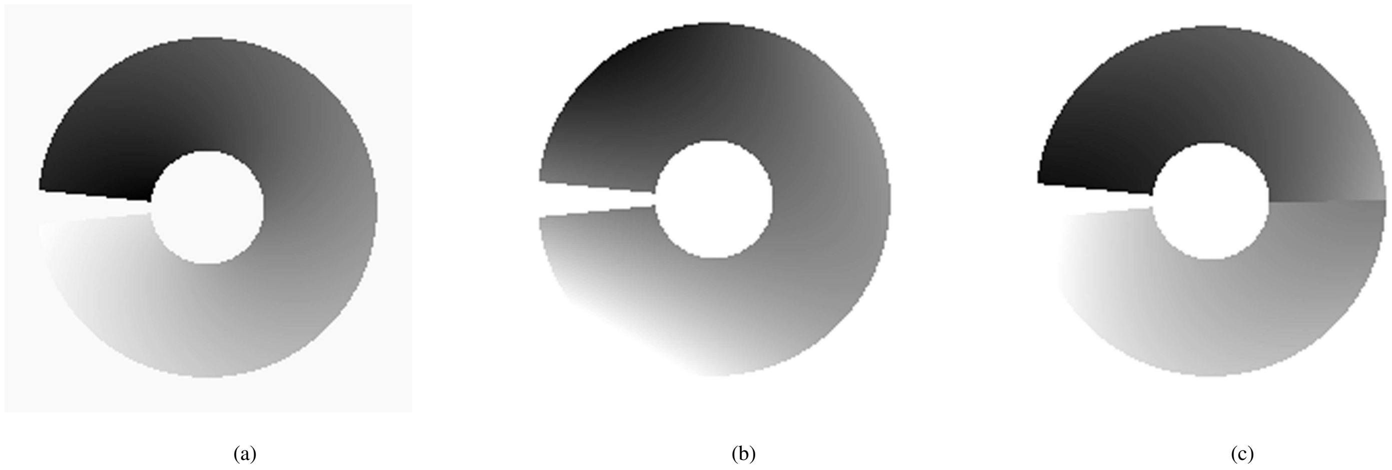


Fig. 2. (a) A smooth function over a non-convex domain, (b) best L^2 quadratic polynomial approximant (PSNR=21.5dB), (c) approximation significantly improves after partitioning (PSNR=33dB)

and let $P_i^1,\ P_i^2 \in \partial\mathcal{C}_i$ be the two points where this maximum is attained.

The algorithm consists of two steps. First, select the component $\mathcal{C}_i$ having the largest value μ_i. Secondly, subdivide the domain Ω by a ray cast from a point P_i at the boundary $\partial\mathcal{C}_i$ such that: $\rho(P_i, P_i^1) = \rho(P_i, P_i^2)$. For each of the two new subdomains, we compute the polynomial approximation (explained in Section III-C) and proceed recursively, if the approximation error is larger than the target error. Notice, that the encoding of this step takes two bits. In practise we observed that it is worth to select in the first stage *three* worst components $\mathcal{C}_i$. Then we chose such a component that minimizes the approximation error the most. It takes 4 bits to encode this version.

C. Approximation and encoding of the segmentation

Once we have computed the pruned segmentation of step III-A, we need to efficiently encode it. Recall that the second generation methods [21] did not fully succeed in solving this problem and therefore were not able to compete with the more conventional transform-based image compression algorithms.

First, we approximate the pruned segmentation by downsampling it. We fix a parameter $\epsilon_C \in \mathrm{N}$ as the maximum error of the approximation in pixel-distance. Choosing a small ϵ_C gives a very good approximation to the segmentation curves, but leads to a higher bit budget. We downsample the pruned segmentation at the rate ϵ_C and generate its lower resolution, as shown at Figure 4(a). Observe that from this downsampled segmentation, we can reconstruct the original 'jaggy' segmentation (see Figure 4(b)) with a maximal error of ϵ_C. After the downsampled segmentation is decoded, it is upsampled back to the original resolution of the image and smoothed using cubic spline least squares approximation (see Figure 4(c)).

The downsampled segmentation polylines are encoded using a high order chain coding algorithm [15],[19] where each contour is represented as a starting point and a sequence of travel directions. In our algorithm, we encode the location of the next point in the polyline, by encoding one of eight possible directions from the previous point (see [9] for a simpler lossy version). We use arithmetic coding combined with context modeling, where each context is determined by the previous four neighbors or equivalently, previous three travel directions. In some sense, this is equivalent to predicting the next point by extrapolating a cubic curve segment that interpolates the previous four points. Our algorithm achieves, on average, a bit-rate of 0.1-0.3 bit/contour point (compare with [2] where a bit-rate of 1.3 bit/contour point was obtained).

D. Creating the edge bands

The main goal of *bands* is to deal with the distortion introduced by the segmentation approximation (down-sampling). The concept of bands also appears in [29] and [18], where the authors justify the bands by claiming that the edges in real-life images are ”...most often ill-defined...”. Indeed, we fix this by allocating some width to the edge singularities computed in step III-A thereby creating edge bands. First, the band's width size in pixel distance, $\epsilon_B \in$ N, is computed from two previous parameters: ϵ_G, the Gaussian window width of step III-A and ϵ_C, the curve approximation error of step III-C. If these two parameters are small, then we allocate small width to the bands and visa-versa. An example of such bands can be seen in Figure 5(a).

We then mark each pixel whose distance is ϵ_B from a reconstructed segmentation pixel as a band pixel. In our algorithm, the values at band pixels are never encoded, since they are in the vicinity of an edge singularity, whose exact location is unknown to the decoder. Instead, the decoder reconstructs the values of band pixels by interpolating the decoded pixel values of the inner sub-domains, where we predict the function to be smoother.

E. Polynomial approximation and quantization in the subdomains of smoothness

In each subdomain Ω created at stage III-B we compute the low-order polynomial approximation $\mathcal{P}_\Omega$ of the target function using the least-squares technique. To ensure stability of the quantization of the polynomial's coefficients, we first compute a representation of $\mathcal{P}_\Omega$ in an orthonormal basis of $\Pi_{r-1} \bigcap L^2(\Omega)$. For example, in the case of bivariate linear polynomials, this simply amounts to transforming the standard polynomial basis $\{1,\ x_1,\ x_2\}$ using a Graham-Schmidt process to an orthonormal basis of $\Pi_1 \bigcap L^2(\Omega)$. The polynomial $\mathcal{P}_\Omega$ is subsequently quantized and encoded in this stable basis. Since at the time of decoding the decoder has full knowledge of the geometry of Ω, the Graham-Schmidt process is carried out exactly as it is proceeded during the encoding. At the decoder, the quantized coefficients of the polynomial are decoded and the polynomial is reconstructed. This quantized version of $\mathcal{P}_\Omega$ is the final polynomial representation.

F. Illustration of the algorithm

We have chosen Lena image to illustrate the steps of the GPP algorithm in Figures 3, 4, 5, and 6.

G. Experimental results

We applied our GPP algorithm to established test images at low-bit rates and compared the performance of GPP to the best wavelet image coding algorithm known to us: Taubman's Kakadu implementation [26] of the JPEG2000 standard [27]. It should be noted that not all JPEG2000 compression algorithms provide the tame coding performance, and therefore one should take care and reference the correct JPEG2000 implementation. We have observed that at the low-bit rates, where we

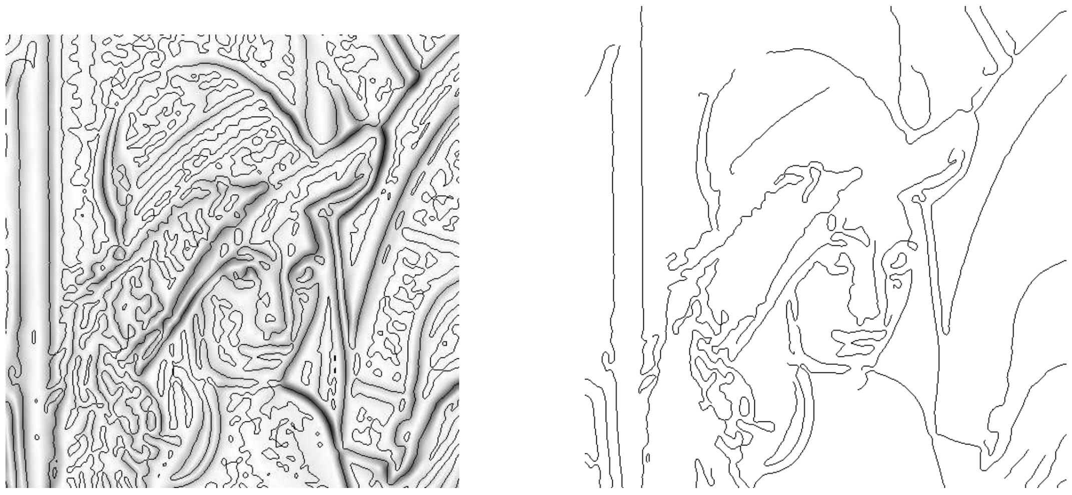

(a) Segmentation of ZeroCross of L o G (b) Pruned segmentation

Fig. 3. (a) Pruning of the initial segmentation of Lena, notice the sorting out of the jagged edges

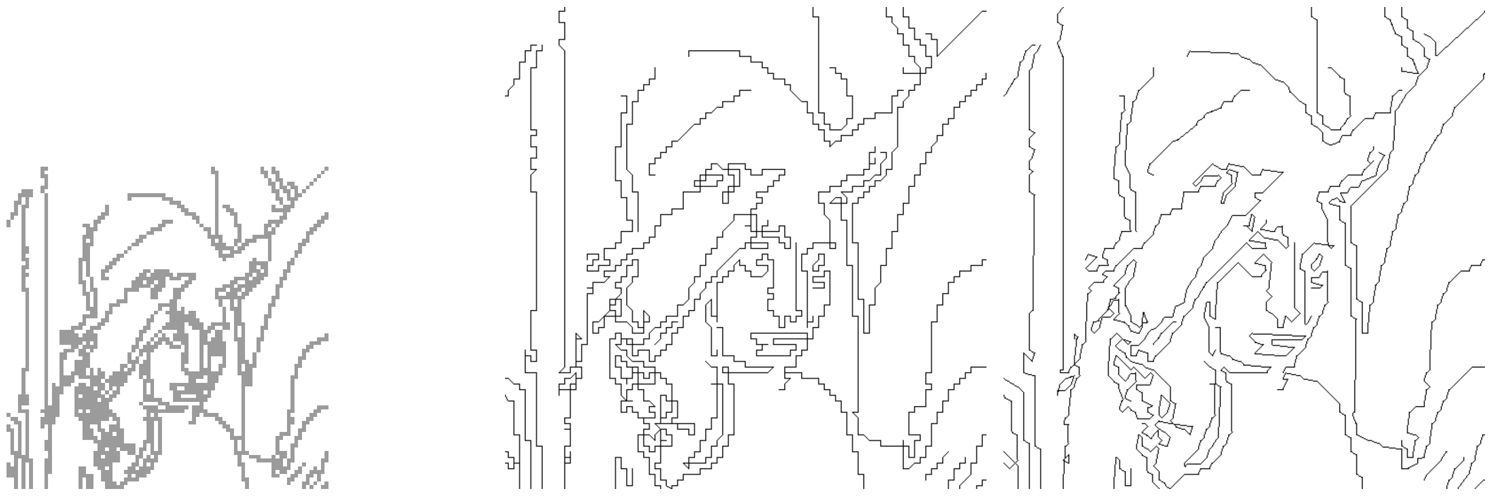
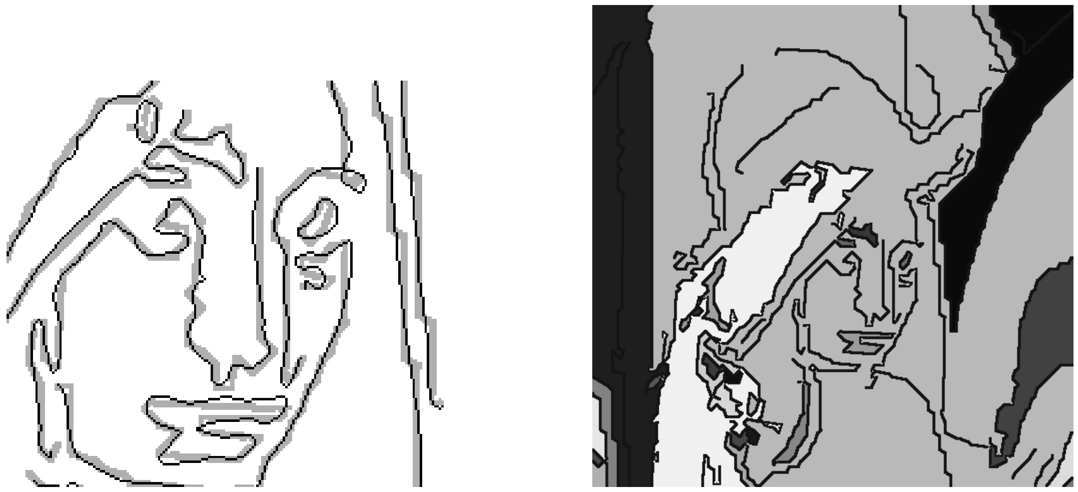

(a) Downscale before encoding (b) Upscale after decoding (c) Smoothing the decoded segmentation

Fig. 4. (a) 1-D approximation employs downscaling, before lossless encoding (b) shows the 'jaggy' reconstruction (upscaling) after decoding, (c) smoothing prediction is subsequently applied to produce smooth curves out of the 'jaggy' reconstruction

made our comparisons, the SPIHT algorithm [22] attains similar results to Kakadu algorithm.

The GPP algorithm efficiently encodes artificial geometric images (Figure 7). In addition, at very low bitrate, the GPP algorithm encodes images with relatively good visual quality, where the main features are preserved as can be seen in Figures 9 and 8. Roughly speaking, the performance of the GPP coding algorithm depends on the amount of 'geometric structure' the image has. For example, our experience is that the Cameraman image is more geometric than Lena and thus can be better encoded using the GPP algorithm.



(a) Bands (b) Initial domains

Fig. 5. (a) Adding bands as to the decoded lossy segmentation of (4(c)), (b) the initial domains of smoothness

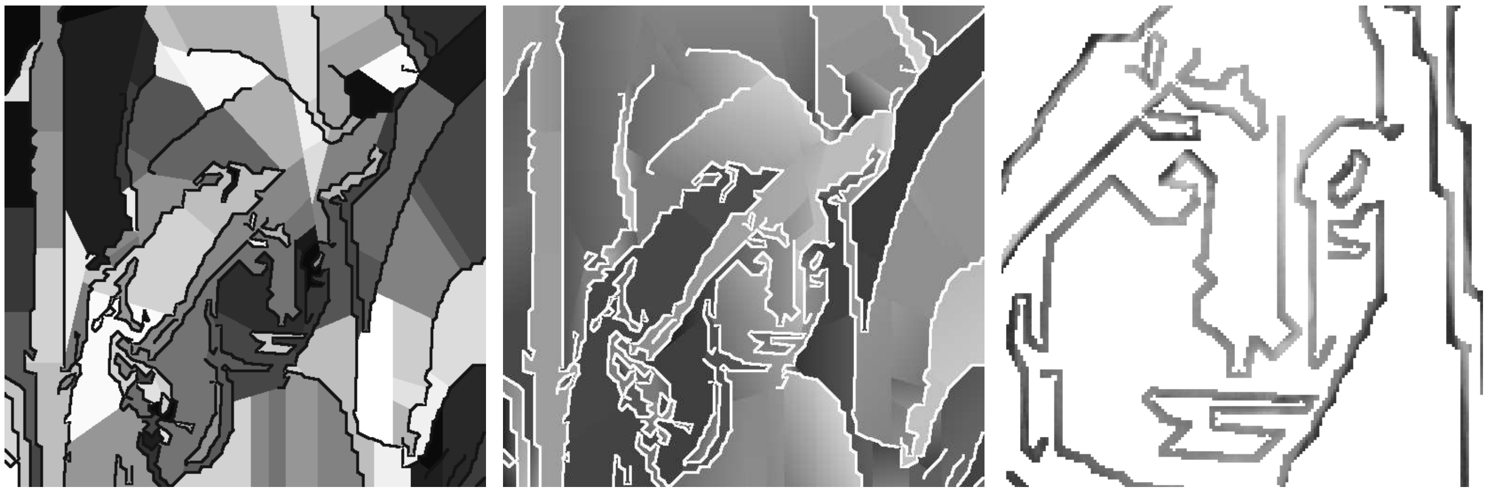


Fig. 6. (a) The final partition obtained with the convex-based dyadic partitioning, (b) its corresponding polynomial approximation, (c) the linear interpolation at the bands

References

[1] E. Candes and D. Donoho, "Curvelets and curvilinear integrals," *J. Approx. Theory*, vol. 113, pp. 59–90, 2001.

[2] S. Carlsson, "Sketch based coding of grey level images," *Signal Processing*, vol. 15, pp. 57–83, July 1988.

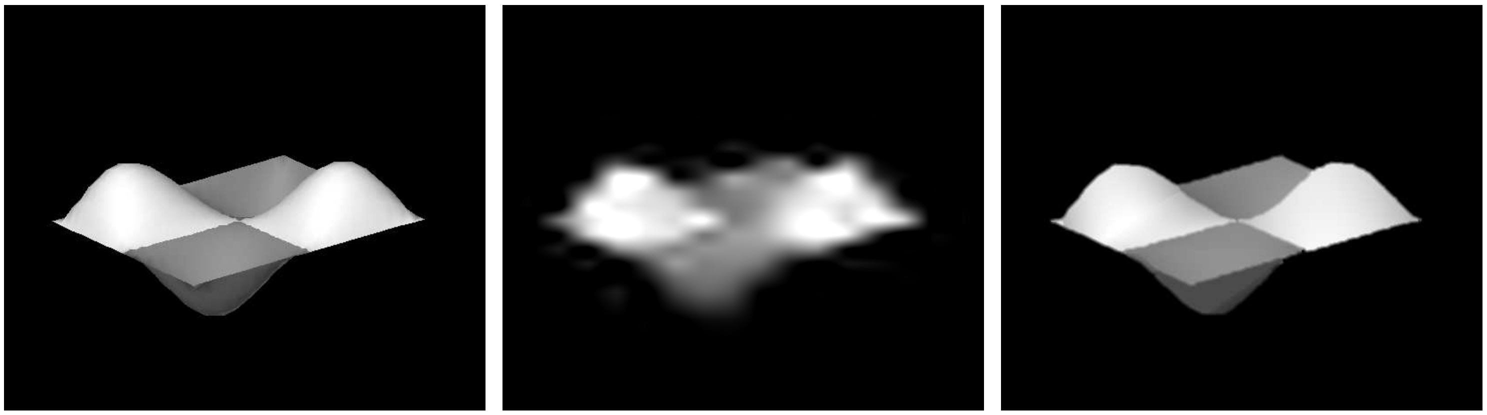
[3] R. L. Claypoole, G. M. Davis, W. Sweldens, and R. G. Baraniuk, "Nonlinear wavelet transforms for image coding via lifting," *IEEE Trans. Image Process.*, vol. 12, pp. 1449–1459, December 2003.

[4] A. Cohen and B. Matei, "Compact representations of images by edge adapted multiscale transforms," *IEEE ICIP Conference Thessaloniki*, September 2001.

[5] S. Dekel and D. Leviatan, "The bramble-hilbert lemma for convex domains," *SIAM Journal on Mathematical Analysis*, vol. 35, pp. 1203–1212, 2004.

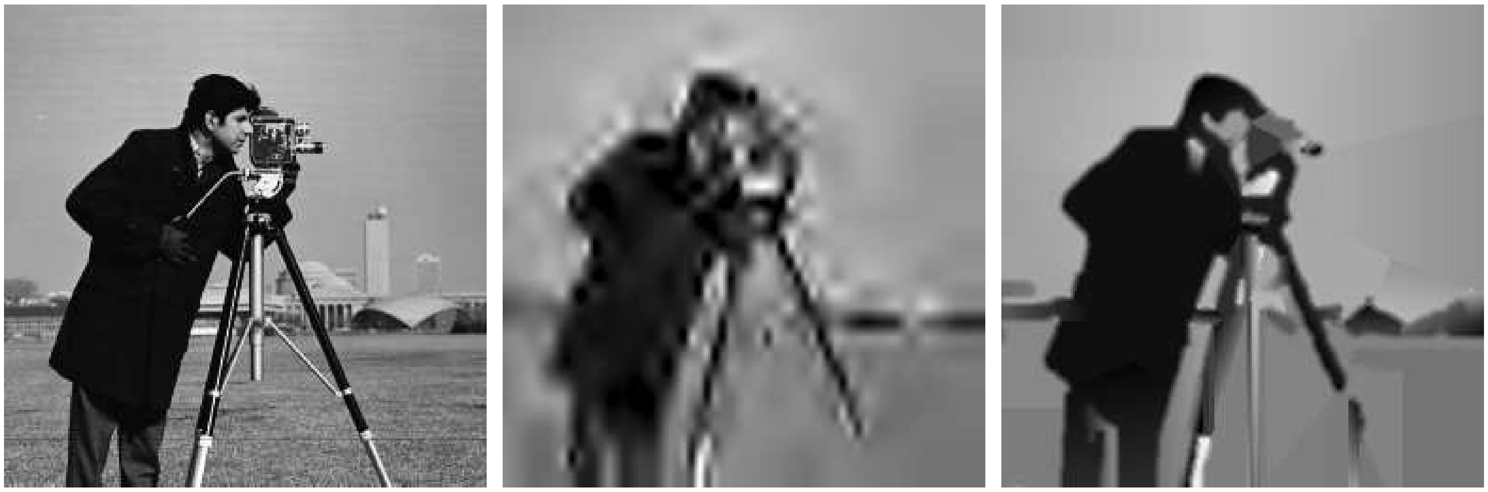
[6] ——, "Adaptive multivariate approximation using binary space partitions and geometric wavelets," *SIAM journal on numerical analysis*, vol. 43, pp. 707–732, 2005.

[7] S. Dekel, D. Leviatan, and M. Sharir, "On bivariate smoothness



(a) Original (b) Jpeg2000 coding at 0.01 bpp (c) GPP coding at 0.01 bpp

Fig. 7. The artificial Eggmean [13] with JPEG2000 Kakadu coding PSNR=24.4dB, versus GPP coding PSNR=29.7dB



(a) Cameraman image (b) Jpeg2000 coding at 0.45 bpp (c) GPP coding at 0.05 bpp

Fig. 8. The Cameraman JPEG2000 coding PSNR=20.1dB, versus GPP coding PSNR=21.5dB

spaces associated with nonlinear approximation," *Constructive Approximation*, vol. 20, pp. 625–646, 2004.

[8] L. Demaret, N. Dyn, and A. Iske, "Image compression by linear splines over adaptive triangulations." [Online]. Available: citeseer.ist.psu.edu/646599.html

[9] U. Y. Desai, M. M. Mizuki, I. Masaki, and B. K.P.Horn, "Edge and mean based image compression," *International Conference on Acoustics, Speech and Signal Process.-ICASSP*, 1996.

[10] R. DeVore and A. Lorentz, *Constructive approximation*. Springler-Verlag,New York,USA, 1993.

[11] R. DeVore and B. Lucier, *Wavelets*. Acta Numerica, 1992, vol. 1.

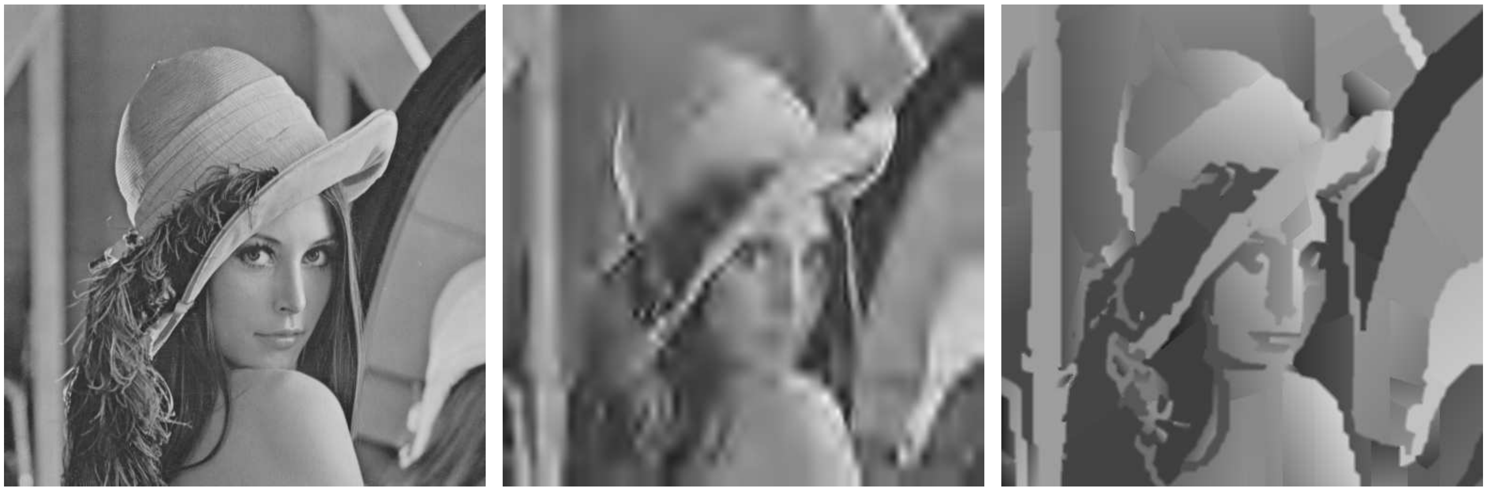
[12] M. N. Do and M. Vetterli, "The contourlet transform: an efficient directional multiresolution image representation," *IEEE Trans. Image Process.*, vol. 14, no. 12, pp. 2091–2106, December 2005.

[13] G. Elber. (1996) Irit solid modeler, technion. [Online]. Available: www.cs.technion.ac.il

[14] J. Froment and S. Mallat, *Wavelets: A Tutorial in Theory and Applications*, C. K. Chui, Ed. Academic Press, New York, 1992, second generation compact image coding with wavelets.

[15] A. K. Jain, *Fundamentals of digital image processing*. Prentice-Hall, 1989.

[16] B. Karaivanov and P. Petrushev, "Nonlinear piecewise polynomial approximation beyond besov spaces," *Appl. Comput. Harmon. Anal.*, vol. 15, pp. 177–223, 2003.



(a) Lena (b) Jpeg2000 coding at 0.02 bpp (c) GPP coding at 0.02 bpp

Fig. 9. The Lena JPEG2000 coding PSNR=25 dB, versus GPP coding PSNR=25 dB

[17] R. Kazinnik, "Image compression using geometric piecewise polynomials," Ph.D. dissertation, School of Mathematics, Tel Aviv University, Tel Aviv Israel, in preparation.

[18] E. LePennec and S. Mallat, "Sparse geometric image representation with bandelets," *IEEE Trans. Image Process.*, vol. 14(4), pp. 423–438, April 2005.

[19] J. Lim, *Two-dimensional signal and image processing*. Prentice Hall, 1990.

[20] H. Radha, M. Vetterli, and R. Leonardi, "Image compression using binary space partitioning trees," *IEEE Trans. Image Process.*, vol. 5, pp. 1610–1624, December 1996.

[21] M. M. Reid, R. J. Millar, and N. D. Black, "Second-generation image coding: an overview," *ACM Computing Surveys*, vol. 29, pp. 3–29, March 1997.

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

[23] J. M. Shapiro, "Embeded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Process.*, vol. 41, pp. 3445 – 3462, December 1993.

[24] R. Shukla, P. L. Dragotti, M. N. Do, and M. Vetterli, "Rate-distortion optimized tree structured compression algorithms for piecewise smooth images," *IEEE Trans. Image Process.*, vol. 14(3), pp. 343–359, March 2005.

[25] D. Taubman, "High performance scalable image compression with ebcot," *IEEE Trans. Image Process.*, vol. 9, pp. 1158–1170, July 2000.

[26] ——. (2005) Kakadu software toolkit for jpeg2000 developers. [Online]. Available: http://www.kakadusoftware.com

[27] D. Taubman and M. Marcellin, *Jpeg2000: Image Compression Fundamentals, Standards, and Practice*. Kluwer International Series in Engineering and Computer Science, 2001.

[28] M. Wakin, J. Romberg, H. Choi, and R. Baraniuk, "Geometric methods for wavelet-based image compression," *SPIE Conference proceedings X, M. Unser, A. Aldroubi, A. Laine, Editors*, vol. 5207, pp. 507–520, 2003.

[29] Y.Yomdin and Y. Elichai, "Normal forms representation: a technology for image compression," *Proc. SPIE - Intern. Soc. for Optical Eng.*, vol. 1903, pp. 204–214, February 1993.

Roman Kazinnik Roman Kazinnik studied Applied Math in Petersburg Polytechnic Institute, USSR, and after four years transferred to Technion, Haifa, Israel, where he graduated in 1998 with Master of Computer Science. He is now finishing Ph.D. at School of Mathematical Sciences of Tel Aviv University. He defines his current interests as Numerical methods of Approximation and its applications to Stochastic Dynamics.

Shai Dekel Shai Dekel received his Ph.D. degree in mathematics from the Tel-Aviv University, Israel, in 2000. Presently, he is a researcher at GE healthcare. His research interests include approximation theory, harmonic analysis and their applications in image processing.

Nira Dyn Nira Dyn is a professor of Applied Mathematics at Tel-Aviv University in Israel since 1984. Her main field of activity is Geometric Modeling and Approximation Theory. She is now serving in the editorial boards of the Journal of Approximation Theory and of the journal Computer Aided Geometric Design.

She wrote more than 130 papers, and participated actively in more than 80 conferences and workshops. Her education was in Applied Mathematics in Israeli institutions; B.Sc from the Technion, Haifa, in 1965, M.Sc from the Weizmann Institute, Rehovot, in 1967, Ph.D. from the Weizmann Institute, Rehovot in 1970.