Adaptive Image Approximation by Linear Splines over Locally Optimal Delaunay Triangulations

Laurent Demaret and Armin Iske

***Abstract*— Locally optimal Delaunay triangulations are constructed to improve previous image approximation schemes. Our construction relies on a local optimization procedure, termed exchange. The efficient implementation of the exchange algorithm is addressed and its complexity is discussed. The good performance of our improved image approximation is illustrated by numerical comparisons.**

***Index Terms*— Image approximation, linear splines, Delaunay triangulations, adaptive thinning algorithms, local optimization.**

I. Introduction

In our previous paper [2], a new method for image compression was proposed. The compression method combines a recent image approximation scheme with scattered data coding [3]. The image approximation in [2] works with linear splines over adaptive Delaunay triangulations, where the construction of the spline approximation space relies on *adaptive thinning*, a data-dependent recursive pixel removal scheme which selects a small subset of significant pixels from the given image. This set of significant pixels yields an anisotropic Delaunay triangulation, and so a suitable linear spline space for image approximation.

In this letter, we show how to improve the approximation quality of *any* linear spline space by the construction of *locally optimal* Delaunay triangulations. This is done by using a postprocessing local optimization algorithm, exchange, which helps to improve the image compression method of [2].

We first review basic ingredients of our image compression method [2] in Section II, before the exchange algorithm is introduced in Section III. Important computational aspects are then addressed in Section IV. Supporting numerical comparisons are finally presented in Section V.

II. Image Approximation by Linear Splines

A. 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.

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 $[X]$.

Manuscript received December 13, 2005.

L. Demaret is with the GSF, Institut für Biomathematik und Biometrie, D-85764 Neuherberg, Germany, laurent.demaret@gsf.de

A. Iske is with the Department of Mathematics, University of Hamburg, D-20146 Hamburg, Germany, iske@math.uni-hamburg.de

B. Linear Splines over Delaunay Triangulations

Let Π_1 denote the space of linear bivariate polynomials. For $Y \subset X$, let $\mathcal{D}(Y)$ denote the Delaunay triangulation [6] of Y, and $C([Y])$ the set of continuous functions on $[Y]$. We define the linear spline space $\mathcal{S}_Y$ by

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

Any element in $\mathcal{S}_Y$ is referred to as a *linear spline* over $\mathcal{D}(Y)$. For given luminances $\{I(y) : y \in Y\}$ there is a unique linear spline $L(Y, I) \in \mathcal{S}_Y$ with $L(Y, I)(y) = I(y)$ for all $y \in Y$. For a fixed subset $Y \subset X$ satisfying $[Y] = [X]$ we take $\mathcal{S}_Y$ as an approximation space for the image I, defined over the domain $[X]$.

C. Quality Measure for Image Approximation

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

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

where, for any $Y \subset X$ and with $|X|$ being the size of X,

$$\bar{\eta}(Y; X) = \eta(Y; X)/|X|$$

denotes the *mean square error* (MSE) and

$$\eta(Y; X) = \sum_{x \in X} |L(Y, I)(x) - I(x)|^2 \qquad (1)$$

is the square error of the approximation $L(Y, I)$ to image I.

The image approximation scheme in our previous paper [2] relies on the construction of a small subset $Y \subset X$ of significant pixels, whose distribution captures the geometry of the image, see also the related methods in [4], [8]. The selection of the significant pixels in Y aims at the reduction of the approximation error $\eta(Y; X)$. This is in [2] done by a greedy removal scheme, termed adaptive thinning, which recursively removes less significant pixels from X, using a suitable criterion for measuring the significance of a pixel $y \in Y$.

The exchange algorithm proposed in this letter reduces, for any given subset $Y \subset X$, the approximation error $\eta(Y; X)$. This is done by a postprocessing local optimization procedure which outputs a *locally optimal subset* $Y^* \subset X$ of size $|Y^*| = |Y|$ satisfying $\eta(Y^*; X) \leq \eta(Y; X)$.

But it is recommended to use exchange only in combination with a preprocessing adaptive thinning or any other suitable pixel reduction algorithm, which outputs a subset $Y \subset X$ with *small* approximation error $\eta(Y; X)$. In this case, Y is usually *close* to a locally optimal subset, so that the required

computational costs for exchange are small, and the composite pixel selection algorithm outputs a suitable locally optimal subset $Y^* \subset X$ with reduced approximation error $\eta(Y^*; X) \leq \eta(Y; X)$, see the numerical comparisons in Section V.

D. Pixel Significance Measures

For any pixel $y \in Y \subset X$, we measure the *significance* of y in Y w.r.t. X by $\eta(Y \setminus y; X)$. An equivalent significance measure to η is given by

$$e_\delta(y; Y) = \eta(Y \setminus y; Y) - \eta(Y; X). \quad (2)$$

Note that

$$e_\delta(y; Y) = \eta(Y \setminus y; X \cap \mathcal{C}(y; Y)) - \eta(Y; X \cap \mathcal{C}(y; Y)), \quad (3)$$

where $\mathcal{C}(y; Y)$ is the cell of y in the Delaunay triangulation $\mathcal{D}(Y)$ of Y. Therefore, e_δ is a *local* measure for the square error η in the cell $\mathcal{C}(y; Y)$, being incurred by the removal of y.

III. Locally Optimal Delaunay Triangulations

A. Local Optimization by Exchange

Definition 1: For any $Y \subset X$, let $Z = X \setminus Y$. A pixel pair $(y, z) \in Y \times Z$ satisfying $\eta((Y \cup z) \setminus y; X) < \eta(Y; X)$ is said to be `exchangeable`. A subset $Y \subset X$ is said to be `locally optimal` in X, iff there is no exchangeable pixel pair $(y, z) \in Y \times Z$, and in this case we also say that the Delaunay triangulation $\mathcal{D}(Y)$ of Y is `locally optimal`.

Note that by an exchange of any exchangeable pixel pair $(y, z) \in Y \times Z$, the approximation error $\eta(Y; X)$ is strictly reduced. This motivates the introduction of the following local optimization algorithm which computes a locally optimal subset in X from any input $Y \subset X$.

Algorithm 1: **(Exchange)**
INPUT: $Y \subset X$;
(1) Let $Z = X \setminus Y$;
(2) WHILE (Y not locally optimal in X**)**
  (2a) Locate an exchangeable pair $(y, z) \in Y \times Z$;
  (2b) Let $Y = (Y \setminus y) \cup z$ and $Z = (Z \setminus z) \cup y$;
OUTPUT: $Y \subset X$, locally optimal in X.

Note that the exchange algorithm terminates after finitely many steps. Indeed, this is because the input set X is finite, and each exchange in step **(2b)** strictly reduces the current (non-negative) approximation error $\eta(Y; X)$. By construction, the output set $Y \subset X$ is locally optimal in X.

B. Characterization of Exchangeable Pixel Pairs

Efficient implementation of the exchange algorithm requires a suitable characterization of exchangeable pixel pairs. To this end, note

$$\eta(Y \setminus y; X) = \eta(Y; X) + e_\delta(y; Y), \qquad \text{for } y \in Y, \quad (4)$$

from (2). By first letting $Y = Y \cup z$ in (4), for $z \in Z = X \setminus Y$, and then $y = z$, this provides two useful relations,

$$\begin{aligned} \eta((Y \cup z) \setminus y; X) &= \eta(Y \cup z; X) + e_\delta(y; Y \cup z), \\ \eta(Y; X) &= \eta(Y \cup z; X) + e_\delta(z; Y \cup z). \end{aligned}$$

Therefore, $(y, z) \in Y \times Z$ are exchangeable, iff

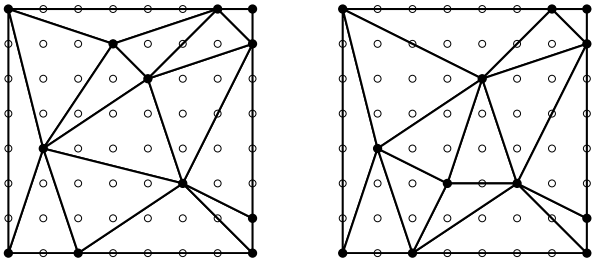
$$e_\delta(z; Y \cup z) > e_\delta(y; Y \cup z), \qquad \text{for } (y, z) \in Y \times Z. \quad (5)$$

Note that for any pixel pair $(y, z) \in Y \times Z$, which is *unconnected* in the Delaunay triangulation $\mathcal{D}(Y \cup z)$ of $Y \cup z$, i.e., $[y; z] \notin \mathcal{D}(Y \cup z)$, this condition can be rewritten as

$$e_\delta(z; Y \cup z) > e_\delta(y; Y), \qquad \text{for } [y; z] \notin \mathcal{D}(Y \cup z). \quad (6)$$

Indeed, if $[y; z]$ is not an edge in $\mathcal{D}(Y \cup z)$, then $\mathcal{C}(y; Y) = \mathcal{C}(y; Y \cup z)$, and so $e_\delta(y; Y) = e_\delta(y; Y \cup z)$ by (3).

For illustration, Figure 1 shows one exchange for an unconnected pixel pair $(y, z) \in Y \times Z$ satisfying $[y; z] \notin \mathcal{D}(Y \cup z)$.



(a) before exchange (b) after exchange

Fig. 1. Exchange of an unconnected pixel pair $(y, z) \in Y \times Z$, satisfying $[y; z] \notin \mathcal{D}(Y \cup z)$, where the image has size 8×8 and $y = (3, 6)$, $z = (3, 2)$. The two corresponding Delaunay triangulations, (a) $\mathcal{D}(Y)$ before exchange and (b) $\mathcal{D}((Y \cup z) \setminus y)$ after exchange, are also shown.

IV. Computational Aspects of Exchange

A. Efficient Implementation of Priority Queues

Efficient implementation the exchange algorithm requires, in combination with the two exchange criteria, (5) and (6), a suitable data structure for efficient maintenance of priority queues. To this end, we prefer to work with the data structure *heap*, a binary tree (see [1] for details), which was also used in our previous paper [2].

In that paper [2], one heap, say `heapY`, is used to store the pixels in $Y \subset X$ in the nodes of `heapY`, where for each node $y \in Y$ its key is given by the local error $e_\delta(y; Y)$ of y in Y. Moreover, the binary tree `heapY` is organized such that the key of a node is *smaller* than the keys of its two children. Hence, due to the heap condition, the root of `heapY` contains a pixel $y^* \in Y$, which minimizes the error $e_\delta(y; Y)$ among all pixels in Y.

In the terminology of [2], the pixel y^* is said to be *least significant*. The adaptive thinning algorithm in [2] recursively removes least significant pixels from X, and after each removal of a least significant pixel, y^*, the involved data structures, $\mathcal{D}(Y)$ and `heapY`, are updated. Recall that any update of `heapY` requires at most $\mathcal{O}(\log N)$ operations, where $N = |X|$. Therefore, the thinning algorithm in [2] has complexity $\mathcal{O}(N \log(N))$, see [2] for details.

(a) Original 256×256 image (b) Reconstruction by $\mathbf{AT}$ (c) Reconstruction by $\mathbf{E} \circ \mathbf{AT}$ (d) Locally optimal triangulation

Fig. 2. Test image `Lena`. Comparison between compression methods $\mathbf{AT}$ and $\mathbf{E} \circ \mathbf{AT}$, whose image reconstruction is shown in (b) for $\mathbf{AT}$ at 0.279 `bpp` with `PSNR` 28.77 `dB` and in (c) for $\mathbf{E} \circ \mathbf{AT}$ with `PSNR` 29.19 `dB`. The locally optimal Delaunay triangulation output by $\mathbf{E} \circ \mathbf{AT}$ is shown in (d).

B. Maintenance of Three Priority Queues

Due to the exchange criteria (5) and (6), the exchange algorithm can be implemented efficiently by using `heapY` and two other heaps, called `heapZ` and `heapE`.

The heap `heapZ` contains the pixels from $Z = X \setminus Y$ and is used for the maintenance of their local error reduction $e_\delta(z; Y \cup z)$, which would be incurred by the insertion of a pixel $z \in Z$ into Y. The heap `heapE` contains all pairs of pixel *neighbours* $(y, z) \in Y \times Z$ satisfying $[y; z] \in \mathcal{D}(Y \cup z)$, i.e., $[y; z]$ is an edge in $\mathcal{D}(Y \cup z)$. The key of any edge $[y; z]$ in `heapE` is given by the local error difference

$$\delta([y; z]) = e_\delta(z; Y \cup z) - e_\delta(y; Y \cup z), \tag{7}$$

which would be incurred by the exchange of pixels y and z.

C. Efficient Localization of Exchangeable Pixels

In order to efficiently locate an exchangeable pixel pair, as required in step **(2a)** of the exchange algorithm, the ordering in the three priority queues are organized, such that `HeapY` contains a pixel with *smallest* error $e_\delta(y; Y)$ at its head, `HeapZ` contains a pixel with *largest* error $e_\delta(z; Y \cup z)$ at its head, and `HeapE` contains an edge with *largest* error difference $\delta([y; z])$ at its head. Each of the three priority queues is updated after each pixel exchange.

To locate an exchangeable pixel pair, if any, the following two checks are performed in step **(2a)** of the exchange algorithm.

In the first check, we regard the key $\delta([y^*; z^*])$ of the root $[y^*; z^*]$ in `heapE`. If $\delta([y^*; z^*]) > 0$, then $(y^*; z^*)$ are exchangeable by criterion (5), in which case we will exchange the pixels y^* and z^* in step **(2b)** of Algorithm 1. Otherwise (i.e., if $\delta([y^*; z^*]) \leq 0$), there is no exchangeable pair of pixel neighbours contained in the edges of `heapE`, in which case we perform a second check.

In the second check, we first locate an unconnected pixel pair $(y^*, z^*) \in Y \times Z$, satisfying $[y^*; z^*] \notin \mathcal{D}(Y \cup z^*)$, which maximizes the difference

$$\delta(y; z) = e_\delta(z; Y \cup z) - e_\delta(y; Y), \qquad \text{for } [y; z] \notin \mathcal{D}(Y \cup z),$$

among all unconnected pixel pairs $(y, z) \in Y \times Z$. Such a pixel pair can be located in only at most $\mathcal{O}(\log N)$ operations by using the other two heaps, `heapY` and `heapZ`, provided that for any $(y, z) \in Y \times Z$ the number of triangles in the cell $\mathcal{C}(y; Y \cup z)$ is bounded above by a small number $k = \mathcal{O}(1)$. For computational details on this, we refer to our paper [5].

If $\delta(y^*, z^*) > 0$, then (y^*, z^*) are exchangeable. Otherwise (i.e., if $\delta(y^*, z^*) \leq 0$), there is no exchangeable pixel pair $(y, z) \in Y \times Z$ with $[y; z] \notin \mathcal{D}(Y \cup z)$.

D. Complexity for one Pixel Exchange

Note that by performing the above two checks we can *efficiently* locate an exchangeable pixel pair in $Y \times Z$, if any. Indeed, the first check, covering the class of neighbouring pixel pairs, requires only one operation. The second check, covering the class of unconnected pixel pairs, can be done in at most $\mathcal{O}(\log N)$ operations [5].

In particular, by the two checks we can decide in at most $\mathcal{O}(\log N)$ operations whether or not there is an exchangeable pixel pair contained in $Y \times Z$. Indeed, if $\delta([y^*; z^*]) \leq 0$ in the first check and $\delta(y^*; z^*) \leq 0$ in the second check, then there is no exchangeable pixel pair contained in $Y \times Z$, where in the *best* case each of the two checks requires only *one* comparison. In the worst case, the two checks may require at most $\mathcal{O}(\log N)$ operations, i.e., one for the first and at most $\mathcal{O}(\log N)$ for the second check.

Now note that the number of updates which are required in the three heaps (`heapY`, `heapZ` and `heapE`) after one pixel exchange is constant. Indeed, this is because for each heap the keys of their nodes correspond to *local* error measures, respectively, see the definitions of e_δ in (3) (for the keys in `heapY` and `heapZ`) and δ in (7) (for the keys in `heapE`).

Since each update in either heap (`heapY`, `heapZ`, or `heapE`) costs only $\mathcal{O}(\log N)$ steps [1], we can conclude that the complexity for *one* pixel exchange is only at most $\mathcal{O}(\log N)$.

V. Numerical Comparisons

We compare the performance of the image approximation method proposed in this letter with that of our previous paper [2]. We recall that the image approximation in [2] relies on adaptive thinning, a recursive pixel removal scheme, which computes a subset $Y \subset X$ of significant pixels, such that the resulting square error $\eta(Y; X)$ in (1) is small.

In this comparison, we take the significant pixels Y, obtained from adaptive thinning, as input for the exchange algorithm to compute a locally optimal subset $Y^* \subset X$ of equal size, $|Y| = |Y^*|$, satisfying $\eta(Y^*; X) \leq \eta(Y; X)$.

Each of these two image approximations is combined with our scattered data coding scheme [3] to obtain two complete image compression methods, here called **AT** and $\mathbf{E} \circ \mathbf{AT}$. We remark that the computational complexity for coding is very small, namely only at most $\mathcal{O}(m \log(N))$, where $m = |Y|$ and $N = |X|$, see the analysis in [3]. Moreover, we recall that the image compression scheme **AT** of our previous paper [2] is competitive to JPEG2000 [7].

For the purpose of comparison, we first consider using the popular test image `Lena`, of size 256×256, as shown in Figure 2(a). We let $|Y| = 1536$ for the number of significant pixels, yielding a compression rate of 0.279 `bpp` (bits per pixel). The reconstruction of `Lena` obtained from **AT** is shown in Figure 2(b), that one obtained from $\mathbf{E} \circ \mathbf{AT}$ can be seen in Figure 2(c). The locally optimal Delaunay triangulation $\mathcal{D}(Y^*)$, computed from $\mathcal{D}(Y)$, is shown in Figure 2(d).

The compression method **AT** yields the PSNR value 28.77 `dB`, whereas $\mathbf{E} \circ \mathbf{AT}$ provides the superior PSNR value 29.19 `dB`, after only $n = 203$ exchange steps. Note that the two PSNR values correspond to MSE $\bar{\eta}(Y; X) = 86.99$ and $\bar{\eta}(Y^*; X) = 78.97$. In conclusion, the exchange algorithm provides a reduction in MSE by about 10 %, which complies with our numerical results obtained in similar numerical comparisons, including the standard test cases `Lena`, `Fruits`, and `Peppers`, which were also used in [2]. Table I reflects our numerical results, where $|Y|$ is the size of $Y \subset X$ and n is the number of exchange steps.

TABLE I
COMPARISON BETWEEN **AT** AND $\mathbf{E} \circ \mathbf{AT}$.

Test Case	PSNR(**AT**)	PSNR($\mathbf{E} \circ \mathbf{AT}$)	$\lvert Y \rvert$	n
Lena	28.77 `dB`	29.19 `dB`	1536	203
Lena	30.04 `dB`	30.30 `dB`	2536	110
Fruits	31.14 `dB`	31.53 `dB`	2736	253
Peppers	31.38 `dB`	31.75 `dB`	2536	217

REFERENCES

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

[2] L. Demaret, N. Dyn, and A. Iske, "Image Compression by Linear Splines over Adaptive Triangulations", to appear in *Signal Processing*.

[3] 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, pp. 107-117, 2003.

[4] Y. Eldar, M. Lindenbaum, M. Porat, and Y.Y. Zeevi, "The Farthest Point Strategy for Progressive Image Sampling", *IEEE Trans. Image Process.*, vol. 6, no. 9, pp. 1305–1315, Sep 1997.

[5] A. Iske, "Progressive Scattered Data Filtering", *J. Comput. Appl. Math.*, vol. 158, no. 2, pp. 297–316, 2003.

[6] F.P. Preparata, M.I. Shamos, *Computational Geometry*, 2nd edition, Springer, New York, 1988.

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

[8] S.-J. Wang, L.-C. Kuo, H.-H. Jong, and Z.-H. Wu, "Representing Images Using Points on Image Surfaces", *IEEE Trans. Image Process.*, vol. 14, no. 8, pp. 1043–1056, Aug. 2005.

Laurent Demaret works in the Institute of Biomathematics and Biometry at the GSF research center in Munich, Germany, since Feb. 2004. He received a Ph.D. in Signal Processing from the University of Rennes I in 2002. He has been a postdoc at Munich University of Technology from Feb. 2002 to Jan. 2004. His research interests include image compression, scattered data simplification, triangular meshes, wedge approximations and partial differential equations with applications to biological models.

Armin Iske, professor in Applied Mathematics at the University of Hamburg, Germany, received a Ph.D. in Approximation Theory from Göttingen University (1994), and a Habilitation in Mathematics from Munich University of Technology (2002). Previous posts include a postdoc at SINTEF Applied Mathematics in Oslo, a research position at Hewlett-Packard Mechanical Design Division, and a Lecturer in Applied Mathematics at Leicester University, UK. His main research activities are in numerical analysis and scientific computing, especially meshfree methods for partial differential equations, multiresolution methods in scattered data modelling, radial basis functions and applications.