SURFACE BLENDING MODEL FOR IMAGE CODING

Joceli Mayer
LINSE: Circuitos e Processamento de Sinais
Departamento de Engenharia Elétrica
Universidade Federal de Santa Catarina
Campus Universitário, 88040-900 Florianópolis SC Brazil
E-mail: mayer@eel.ufsc.br

Abstract

This work proposes an efficient image representation based on triangular blending surfaces. The recursive triangular partitioning proposed represents an image as segments comprised of variable size right-angle triangles. Triangular partitioning is shown [11] to be more efficient than square partitioning. A novel and economic parametric blending model represents each triangular surface. The framework for designing blending surfaces for triangular regions is presented. This economic model allows coefficients (control points) to be shared among neighboring triangles. This coefficient sharing results in blockiness reduction as compared to block based techniques. The technique is specially appealing for encoding images presenting smooth transitions. Compression and visual quality results compare favorably against a popular wavelet codec using decomposition into seven bands. A greedy algorithm based on priority queues proposed in [6] is used to further reduce the entropy of the control point bitstream while preserves the quality. This algorithm provides a better performance in a rate-distortion sense than using uniform quantization of the control points.

Keywords: Image Coding and Blending Surfaces.

1. Introduction

Image models based on polynomial surfaces provides an interesting lossy representation. Regions represented by these surfaces are smooth and absent of noise, which usually result in good perceived quality. These models are specially appealing to efficiently represent smooth regions and provide an alternative to the traditional orthogonal basis representation as found by using KLT, DCT, LOT transforms and wavelets decomposition.

Second generation coding schemes based on fitting parametric polynomials to regions within an image are proposed in the literature [13] [12]. In order to represent the image into separated smooth regions, these techniques require boundary extraction and encoding of the resulting pixel values. For most images, a large portion of the bit-rate is allocated for this side information. Another approach is found in the block based encoder [14] using B-splines polynomials. This technique requires encoding of coefficients (from 9 to 25) to represent each fixed size block. Despite of the coefficients and/or boundary representation cost, the techniques described above are proven competitive to the JPEG-DCT algorithm.

We propose a new model for image coding using polynomial surfaces. The segmentation or region building procedure is replaced by the Recursive Triangular Partitioning (RTP) which is similar to quadtree partitioning. The proposed model uses variable size regions similar to the region-based models in [13] [12], while avoids the costs associated to boundary extraction. Our partitioning creates triangular shaped regions. The proposed blending model shares an important property of the parametric Bézier and B-splines modeling [14] [7] [8]: coefficients are shared by neighboring regions. This property provides an economic representation while reduces blocking artifacts generated by techniques that encode regions independently of the neighboring regions. This is the first work, besides the thesis in [11], that we discuss with details the triangular partitioning, the mathematical framework for designing blending surfaces and an example for a second degree polynomial basis. Results are presented to emphasize the potential of the blending model as encoding technique. A greedy optimization algorithm is applied [6, 5] to further improve the compression and quality.

2. Triangular Partitioning

Consider the following problem: assume an image I with $N \times N$ pixels segmented into $N_R = 2^k$ regions of same area $A \in \Re$ and $k \geq 2$ where k is an integer. We may use square block regions or right-angle triangular regions as shown in Fig. 1. Assume that both representations require the pixel amplitudes (coefficients) at the vertices (control points) of each block or triangular region. Therefore, each block needs 4 coefficients and each right-angle triangle needs 3 coefficients. An example for $k = 2$ and $k = 4$ is shown in Fig. 1. For a given k, we provide the expressions for the required number of coefficients for the square block and the triangular partitioning. By applying mathematical induction on the geometric problem as explained in [11], we find the closed form expressions.

i) For the square block partitioning:

$$V_\diamond(k) = (\sqrt{N_R} + 1)^2 = \\ = (\sqrt{2^{k/2} \cdot 2^{k/2}} + 1)^2 = (2^{k/2} + 1)^2 \quad (1)$$

for k even. It is not possible to segment a square image with $N \times N$ into square blocks for odd k.

ii) For triangular partitioning:

$$V_\triangleright(k) = V_\diamond(k-1) = (2^{(k-1)/2} + 1)^2 = (\frac{2^{k/2}}{\sqrt{2}} + 1)^2 \quad (2)$$

for k odd.

$$V_\triangleright(k) = 2V_\diamond(k-2) - 2\sqrt{V_\diamond(k-2)} + 1 = \\ = 2^{k-1} + 2^{k/2} + 1 \quad (3)$$

for k even. We can show that the ratio of the number of regions by the number of coefficients required is asymptotically twice higher for the triangular partitioning than for the square partitioning. The limit for the ratio of the number of regions over the number of coefficients, when $k \to \infty$, is given as:

For square block partitioning:

$$\lim_{k \to \infty} \frac{2^k}{V_\diamond} = \frac{2^k}{(2^{k/2} + 1)^2} = \frac{2^k}{2^k + 2^{k/2+1} + 1} = 1 \quad (4)$$

for k even.

For triangular partitioning:

$$\lim_{k \to \infty} \frac{2^k}{V_\triangleright} = \frac{2^k}{2^{k-1} + 2^{k/2+1/2} + 1} = 2 \quad (5)$$

for k odd.

$$\lim_{k \to \infty} \frac{2^k}{V_\triangleright} = \frac{2^k}{2^{k-1} + 2^{k/2} + 1} = 2 \quad (6)$$

for k even. These results can be stated as:

"*For a given square image I, a given number of regions $N_R = 2^k$ of same area A, the triangular partitioning requires about half of the coefficients required for the square block partitioning*".

For the specific partitioning case, where polygons have the same area using an image representation which requires the intensities at vertices of the polygons being encoded, the triangular partitioning is more efficient than square block partitioning. For a general case, where regions may have different areas, the superiority of triangular partitioning was clearly observed by simulations in [11]. This result suggests our investigation of an image representation using triangular partitioning. Our approach represents triangular regions using blending polynomials as explained in the next section.

3. Design of Blending Surfaces for Triangular Regions

Let us represent a right-angle triangle T by its three vertices (control points), C, L, R as illustrate in Fig. 2.

Allow us define the vectors $\vec{V} = \overline{CL} = \vec{L} - \vec{C}$ and $\vec{U} = \overline{CR} = \vec{R} - \vec{C}$ and the set $S_\triangle = \{P : P \subset R_\triangle\}$, where point $P \in \Re^2$ and region $R_\triangle$ is in the convex hull defined by the 2-D coordinates of the control points. For $\forall P \in S_\triangle$ $\exists$ parameterized coordinates (u, v) such that:

$$\vec{P} = \vec{C} + u \cdot \vec{U} + v \cdot \vec{V}, u, v \in [0, 1] \quad (7)$$

where $\vec{P}$ and $\vec{C}$ represent the coordinates of points P and C. The point C indicates the reference for the parametric space as illustrate in Fig. 2. In order to consider only points inside the triangle T, we need the additional constraint $(u + v) \leq 1$. Points at diagonal line segment $\overline{RL}$ have coordinates such that $(u+v) = 1$.

For a given image I, and a location defined by a point P, the image intensity at that point is given by $I(\vec{P})$. For a triangle T defining a region $R_\triangle$, we can approximate the image intensities over this region, $I_\triangle(u, v)$, by the parametric surface $\widetilde{I}_\triangle(u, v)$ using a parametric polynomial basis Φ:

$$I_\triangle(u, v) \cong \widetilde{I}_\triangle(u, v) = \sum_M \alpha_i \cdot \Phi_i(u, v) = \\ = \vec{\alpha}^T_\triangle \cdot \vec{\Phi} = \vec{\Phi}^T \cdot \vec{\alpha}_\triangle \quad (8)$$

where M is the number of coefficients, $\vec{\alpha}_\triangle$ is the coefficient vector, $\vec{\Phi}$ is the polynomial basis vector and $\vec{X}^T$ indicates the transpose of vector $\vec{X}$.

The proposed polynomial blending surface representation interpolates at the control points. The surface is completely defined by the intensities at control point locations. As a result, it avoids coefficient determination as in the traditional polynomial fitting. Moreover, the coefficient vector $\overrightarrow{\alpha}_{\triangle}$ is obtained by a fixed linear combination of the intensities at the control points:

$$\overrightarrow{\alpha_{\triangle}} = A \cdot \overrightarrow{G_{\triangle}} \tag{9}$$

where $\overrightarrow{G_{\triangle}}$ is the intensity control points vector for the region $R_{\triangle}$. A is a $M \times M$ matrix, which is *independent* of the region $R_{\triangle}$.

We find the matrix A by choosing surface interpolation at control points and using Eq. 8:

$$\widetilde{I_{\triangle}}(\overrightarrow{P_P}) = \overrightarrow{G_{\triangle}} = \overrightarrow{\Phi}^T(\overrightarrow{P_P}) \cdot \overrightarrow{\alpha_{\triangle}} \tag{10}$$

from Eq. 9 follows:

$$\overrightarrow{G_{\triangle}} = \overrightarrow{\Phi}^T(\overrightarrow{P_P}) \cdot A \cdot \overrightarrow{G_{\triangle}} \Rightarrow \overrightarrow{\Phi}^T(\overrightarrow{P_P}) \cdot A = I \tag{11}$$

where I is the identity matrix. For a 2nd degree surface we may choose the parametric control point vector as $\overrightarrow{P_P}^T = [\ \overrightarrow{C}\quad \overrightarrow{L}\quad \overrightarrow{R}\quad \overrightarrow{S}\quad \overrightarrow{M}\quad \overrightarrow{N}\ ]$. These points are illustrated in Fig. 2, which present the following parametric coordinates: $\overrightarrow{C} = (0,0)$, $\overrightarrow{L} = (0,1)$, $\overrightarrow{R} = (1,0)$, $\overrightarrow{S} = (0.5,0.5)$, $\overrightarrow{M} = (0,0.5)$ and $\overrightarrow{N} = (0.5,0)$.

Let us choose the following parametric polynomial basis:

$$\overrightarrow{\Phi}^T(u,v) = [\ 1 \quad u \quad v \quad u\cdot v \quad u^2 \quad v^2\ ] \tag{12}$$

which, when applied to the control point vector, $\overrightarrow{P_P}$, results:

$$\overrightarrow{\Phi}^T(\overrightarrow{P_P}) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1/2 & 1/2 & 1/4 & 1/4 & 1/4 \\ 1 & 0 & 1/2 & 0 & 0 & 1/4 \\ 1 & 1/2 & 0 & 0 & 1/4 & 0 \end{bmatrix} \tag{13}$$

From Eq. 11, we obtain the desired matrix A by inverting the matrix above:

$$A = \left(\overrightarrow{\Phi}^T(\overrightarrow{P_P})\right)^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -3 & 0 & -1 & 0 & 0 & 4 \\ -3 & -1 & 0 & 0 & 4 & 0 \\ 4 & 0 & 0 & 4 & -4 & -4 \\ 2 & 0 & 2 & 0 & 0 & -4 \\ 2 & 2 & 0 & 0 & -4 & 0 \end{bmatrix} \tag{14}$$

ALGORITHM **RTP**(Triangle T)
$\rightarrow$ **Blend**(T,BLEND_MODEL);
$\rightarrow$ IF (**P**SNR(T) $\geq$ MINIMUM_PSNR) THEN RETURN;
$\rightarrow$ IF (**Area**(T) $<$ MINIMUM_AREA) THEN RETURN;
$\rightarrow$ **Split_into**(T, T_L, T_R);
$\rightarrow$ **RTP**(T_L);
$\rightarrow$ **RTP**(T_R);

Table 1. Recursive Triangular Partitioning (RTP)

The resulting coefficient vector is given by Eq. 9, where:

$$\overrightarrow{G_{\triangle}} = \left[I_{\triangle}(\overrightarrow{C})\ I_{\triangle}(\overrightarrow{L})\ I_{\triangle}(\overrightarrow{R})\ I_{\triangle}(\overrightarrow{S})\ I_{\triangle}(\overrightarrow{M})\ I_{\triangle}(\overrightarrow{N})\right]^T \tag{15}$$

$I_{\triangle}(\overrightarrow{P})$ corresponds to the intensity at parameterized position of point P inside the region $R_{\triangle}$. Finally, the second degree parametric blending surface is given by:

$$\widetilde{I_{\triangle}}(u,v) = \overrightarrow{\Phi}^T(u,v) \cdot \overrightarrow{\alpha_{\triangle}}, u,v \in [0,1], (u+v) \leq 1 \tag{16}$$

Different blending surfaces can be derived from the framework described above depending on the polynomial basis chosen and how many and where the control points are located in the triangle. Further discussion on this topic is found in [11].

4. Recursive Triangular Partitioning

The proposed partitioning algorithm is described in the following. The algorithm starts by partitioning a square image I (which might be a region of the entire image) into four right-angle triangles, T_a, T_b, T_c, T_d as shown in Fig. 3. The recursive routine RTP(T) is applied to these triangles.

The RTP algorithm is presented in Table 1. The algorithm computes the surface representation for the triangle T and checks if the resulting PSNR(T) is satisfactory. When the triangle T is successfully represented by the model, the algorithm is done with this region $R_{\triangle}$, otherwise the triangle area is computed. If the area is greater than an allowed minimum, the algorithm splits the region $R_{\triangle}$ into two other regions $R_{\triangle L}$ and $R_{\triangle R}$. The algorithm is then called recursively for these new triangular regions. The triangular splitting is illustrate in Fig. 5. Each time a triangle is split, a bit set to 1 is sent to the decoder; when the triangle is successfully represented by the blending model, a bit set to 0 is sent. This splitting information is necessary to recover the partitioning information at decoder side. An illustration of partitioning is given in Fig. 3. In Fig. 4 we present the partitioning for the "Lenna" image.

When the algorithm reaches a triangle with area smaller than the minimum, it indicates an active region (with noise, textures and/or edges) where polynomials models are not efficient. In general, models based on polynomial surfaces are better for representing smooth regions. For those active triangles, it is computed the difference between the blending surface and the original image at same region. This prediction error is then quantized, entropy encoded and sent to the decoder.

5. Results with Wavelets Decomposition

Similar to the Spline Model proposed in [13], our approach represents the lowest frequency band (baseband) using a polynomial surface model. We decompose the original image using the FBI fingerprints wavelets [9] into 7 bands (2 levels of 2D filtering). This experimental codec uses the RTP algorithm with the 2nd order blending surface proposed to represent the baseband. The 6 other bands (detail bands) are uniformly quantized. The resulting bitstreams from the 6 bands, the control points, splitting information and quantized prediction error are entropy encoded using the arithmetic coder proposed by [3]. The encoder uses an efficient memory implementation with 3rd order context model presented in [10]. The results for the proposed codec are presented in the following table:

Images	Size (bytes)	Bit-Rate (bpp)	PSNR (dB)
Lenna	9601	0.293	32.15
Peppers	9125	0.278	30.81
Balls	4433	0.135	33.51

We computed the results for the wavelet codec proposed by [2] using only 7 bands:

Images	Size (bytes)	Bit-Rate (bpp)	PSNR (dB)
Lenna	10057	0.307	30.82
Peppers	9093	0.277	29.90
Balls	4464	0.136	28.34

The representation of the baseband by the proposed blending model resulted in a more efficient encoding than the codec in [2] when both codecs use only 7 bands. Zerotree quantization is not used neither in our codec nor in [2]. For further decomposition (>7 bands) our simple experimental codec, as it is (see the next section), does not provide a better tradeoff compression/quality. Nevertheless, it presents both an objective quality and a resulting perceived quality superior to the 7-bands codec in [2] for the same bit-rate. Fig. 6 and Fig. 7 show comparisons for the "Lenna" image. The additional computational cost associated to the proposed region based coding is very low since there is no optimization involved, boundary estimation or even coefficient determination as in [13] [12].

6. Greedy Optimization for the Control Point Bitstream

In this section we investigate the optimality of the proposed model. An optimal solution should jointly optimize both the partitioning and the control point bitstream quantization. This optimization would be extremely complex because the triangular regions cannot be considered as independent units neither for distortion nor bit-rate. Since control points are shared among neighbor regions: modification of a control point amplitude for a given triangle also affects the distortion of the neighboring triangles. The dependency problem is discussed in [1], where optimality is found using the Lagrangian optimization approach when the distortion and the bitrate contribution of each coding unit can be computed independently. When independence cannot be assumed, the optimal solution become exponentially complex on the number of coding units. In these cases, simplification assumptions and heuristics are used in order to achieve a practical solution.

In face of this dependency problem and associated complexity, we proposed a sub-optimal solution for quantization of the control points in [6, 4, 5]. Our low complexity iterative greedy algorithm provides improvements in both compression and quality when compared to the original RTP algorithm. The algorithm assumes that the image is partitioned by the RTP algorithm and the resulting control points are uniformly quantized into Nq levels.

7. Results

The proposed algorithm was applied to the "baseline" RTP algorithm (without wavelet decomposition). First, the RTP produces a control point bitstream uniform quantized with N_q levels. The greedy algorithm [6, 4, 5] is then applied to this uniform quantized bitstream using N_q bins. For the following results we used $N_q = 64$:

Images (512×512)	Non-optimized	Optimized
Lenna	18442 bytes 31.12 dB 0.56 bpp	15102 bytes 31.43 dB 0.46 bpp
Peppers	23937 bytes 30.14 dB 0.73 bpp	19261 bytes 30.26 dB 0.59 bpp
Balls	4192 bytes 32.05 dB 0.13 bpp	3723 bytes 32.35 dB 0.11 bpp

The results shown above indicate improvements on both quality and compression rate. The quantization takes about 15 minutes to improve the bitstream quantization for an image of 512 by 512 pixels, without wavelet decomposition. However, the decoding time is not affected at all; the decoding speed is the same as the non-optimized case. By using the greedy quantization and wavelet decomposition with 7 bands and 10 bands, we achieve the following results:

Images (512×512)	7 bands	10 bands
Lenna	9276 bytes 32.46 dB 0.28 bpp	5364 bytes 30.2 dB 0.16 bpp
Peppers	8772 bytes 31.16 dB 0.27 bpp	5130 bytes 29.4 dB 0.16 bpp
Balls	4229 bytes 34.19 dB 0.13 bpp	2863 bytes 33.6 dB 0.087 bpp

For the 7 bands decomposition case, the greedy quantization considerably improves both compression and quality as reported in Section 5. For the codec in [2] with 10 bands and no zerotree quantization, the compression and quality achieved are almost equivalent to our proposed codec. Moreover, the perceived quality of our optimized method is superior for all images, specially for images presenting many smooth regions, as in the "Balls" image.

The overhead due to the optimization quantization becomes proportionally smaller as the baseband gets smaller due to the $O(n \cdot log(n))$ algorithm complexity [4]. For the 7 bands decomposition the encoding time is about 1 minute, whereas for the 10 bands decomposition the encoding time is about 20 seconds. These simulations were performed in an AMD K6-2 300 MHz notebook under Linux OS (kernel 2.0).

The proposed blending model with the greedy optimization provides a good representation for the baseband. This representation provided reduced banding artifacts in baseband when compared to other techniques. This technique is specially appealing for images presenting smooth regions. The presented results can be further improved using more elaborated quantization schemes for the detail bands, as the zerotree quantization.

References

[1] Antonio Ortega and Kannan Ramchandran. Rate-Distortion Methods for Image and Video Compression. *Signal Processing Magazine*, 15:23–50, November 1998.

[2] Geoff Davis, John Danskin and Ray Heasman. Wavelet Image Compression Construction Kit. *http://www.cs.dartmouth.edu/˜gdavis/wavelet/wavelet.html*, 1997.

[3] Ian H. Witten, Radford Neal and John Clearly. Arithmetic Coding for Data Compression. *Communications of the ACM*, February 1997.

[4] Joceli Mayer. A Blending Model for Efficient Compression of Smooth Images. *Data Compression Conference, DCC'98*, March 1999.

[5] Joceli Mayer. Greedy Quantization of Control Points for 2D and 3D Data Using Blending Surfaces Representation. *33rd Asilomar Conference on Signals, Systems and Computers*, November 1999.

[6] Joceli Mayer. On the Quantization of Control Points for 2D and 3D Blending Surfaces. *XVIII Simpósio Brasileiro de Telecomunicações*, Setembro 2000.

[7] Joceli Mayer and Glen Langdon. Application of Bezier Functions to the Post-Enhancement of Decompressed Images. *31st Asilomar Conference on Signals, Systems and Computers*, November 1997.

[8] Joceli Mayer and Glen Langdon. Post-Processing Enhancement of Decompressed Images Using Variable Order Bezier Polynomials and Distance Transform. *Data Compression Conference, DCC'98*, March 1998.

[9] Jonathan N. Bradley, Christopher M. Brislawn and Tom Hopper. The FBI Wavelet/Scalar Quantization Standard for Gray-Scale Fingerprint Image Compression. *SPIE - Visual Information Processing II*, 1961:293–304, April 1993.

[10] Mark R. Nelson. Arithmetic Coding and Statistical Modeling. *Dr. Dobbs Magazine*, February 1991.

[11] J. Mayer. *Blending Models for Image Enhancement and Coding*. PhD thesis, University of California, Santa Cruz, CA, 1999.

[12] Paul Ausbeck Jr. Piecewise-Smooth Modeling of Digital Images. *Ph.D. Thesis, UC - Santa Cruz*, 1996.

[13] R. Baseri and J. W. Modestino. Region-Based Coding of Images using a Spline Model. *Proceedings of 1st International Conference on Image Processing - ICIP*, November 1994.

[14] Toshiaki Watanabe. Image Coding Making Use of B-Splines Surfaces. *IEEE Transactions on Circuits and Circuits for Video Technology*, pages 409–413, April 1997.

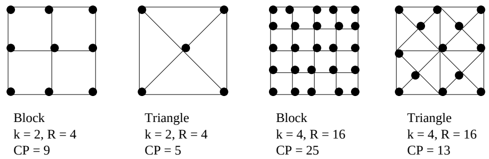

Figure 1. Triangular and block partitioning (k=2 and k=4).

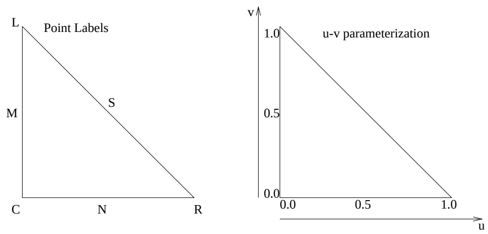

Figure 2. Parameterization (u, v) and control points of a right-angle triangle.

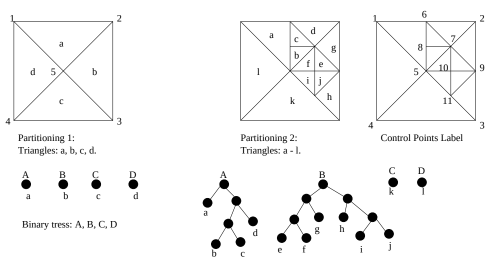

Figure 3. Partitioning and corresponding binary trees.

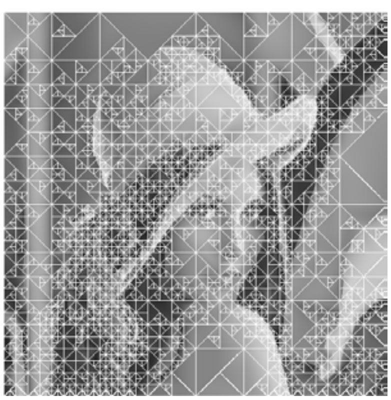

Figure 4. Triangular partitioning using RTP applied to "Lenna".

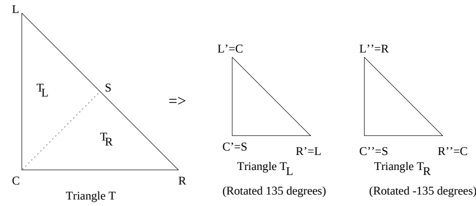

Figure 5. Splitting a triangle T into 2 triangles.

Figure 6. Result for [2] using 7 bands. PSNR = 30.82 dB, Bit-rate = 0.307 bpp.

Figure 7. Result for the proposed Blending Model applied to the lowest frequency band of the wavelet decomposition [6] into 7 bands. PSNR = 32.15 dB, Bit-rate = 0.293 bpp.