ON THE QUANTIZATION OF CONTROL POINTS FOR 2D AND 3D BLENDING SURFACES

Joceli Mayer
Linse: Circuitos e Processamento de Sinais
Departamento de Engenharia Elétrica
Universidade Federal de Santa Catarina
Campus Universitário, 88040-900 - Florianópolis SC - Brasil
Tel: (0xx48) 331-9504, Fax: (0xx48) 331-9091, j.mayer@ieee.org

ABSTRACT

Blending surfaces are extensively used for graphics modeling. Recently, we proposed a still image coding using these surfaces. These surfaces are completely defined by their control points. The usual approach is to partition an image or a 3D object into regions of same shape. Each region is represented by one blending surface. However, a surface shares control points with the neighboring regions. This results in a dependence problem which cannot be addressed by traditional techniques like Loyd-Max quantization or Laplacian optimization.

We describe an iterative greedy algorithm to address this problem and efficiently quantize these control points. It has a computational time complexity of $O(n \cdot log(n))$, for n control points. Examples of application of the algorithm are provided for still image coding and encoding of VRML files. We achieved an excellent tradeoff between bit-rate and distortion as illustrated by comparisons to uniform quantization.

Keywords: Image Coding, Data Compression, Blending Surfaces, Bit Allocation and Data Quantization.

1. INTRODUCTION TO BLENDING SURFACES

Images and 3D objects can be efficiently represented by blending surfaces. Examples of 2D representations by these parameterized surfaces are presented in [9, 7]. These surfaces are extensively used to represent 3D objects in many graphics applications [1, 6]. The popular file format for representing 3D objects, the VRML (Virtual Reality Modeling Language), also uses blending surfaces [3]. In the literature, many techniques split regions into separate patches and represent each patch by a surface constructed by Bézier polynomials [1, 6] .

Usually, the blending surface representation of images/objects is based on two steps: 1) the image/object is partitioned into small regions; 2) each region is represented by a polynomial blending surface. The partitioning process splits the image/object into regions of different areas presenting a same regular shape. In this way, the locations inside the region can be parameterized. Triangular shape is used in [9, 7, 15]. Rectangular shape is also popular in many partitioning algorithms, as for example, in quadtrees [14]. Since our interest is in the efficient image/object representation, we compare these two shapes (triangular and rectangular) and conclude in [8] that triangular shape partitioning provides a better tradeoff between compression ratio and distortion. An example of 2D partitioning is illustrated in Fig. 1. An example of 3D partitioning of a sphere is presented in Fig. 2.

The blending representation differs from polynomial fitting. The resulting blending surface for a given region is based on control points (amplitudes, coordinates), which are located at specific parameterized coordinates. The polynomial fitting provides a surface based on coefficients computed for each region in order to minimize some average error. This coefficient determination can generate unstable surfaces for high polynomial orders and it is region dependent. The blending approach does not compute coefficients, instead it uses amplitudes/coordinates from specific (parameterized) locations inside the region in order to generate the surface.

One important characteristic of the blending representation is that control points located at each region boundary can be shared by the neighboring regions.

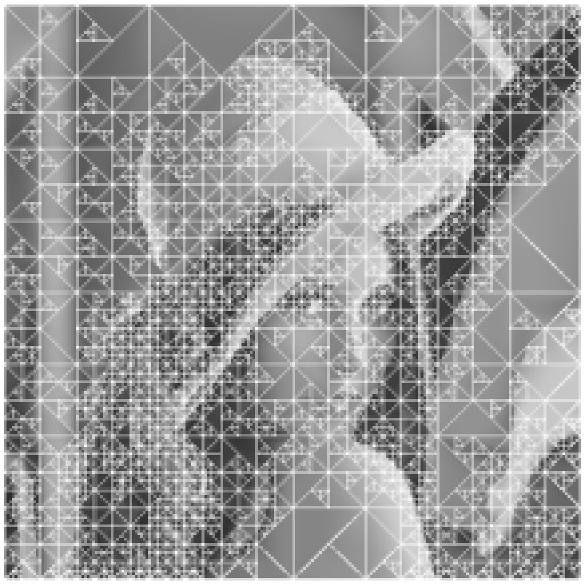

Fig. 1. 2D Partitioning illustration and blending representation for the "lenna" image.

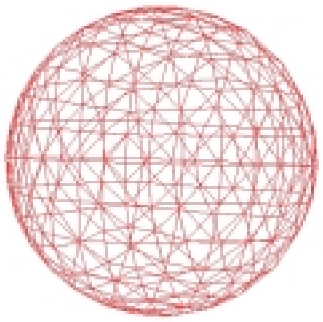

Fig. 2. 3D Partitioning illustration for a sphere.

This results in three desirable properties: 1) potentially better compression by sharing the coefficients [7], 2) reduced blockiness (distortion at the boundaries) since the resulting neighboring surfaces are forced to have closer amplitudes at the boundaries [10], 3) better and stable adjusting of the surface by changing one or more amplitudes of the control points.

In this work, we describe a general greedy approach and its application to the quantization of control points in a blending representation of images or 3D objects.

2. THE QUANTIZATION PROBLEM

Recalling that some control points are shared among neighboring regions [10], by changing a control point amplitude, we affect the surfaces of the regions that share this control point. This dependence complicates the quantization design. The same problem appears in still image coding using blending surfaces [7] and in the quantization of geometric coordinates for 3D objects described in VRML [3]. Traditional quantization techniques, like the Loyd-Max quantization [13, 2, 11], cannot be applied for this problem because the assumption of independence of distortion (the overall distortion as the sum of individual distortions) does not hold for the blending representation.

The dependence problem is discussed in [4], where optimality is found using the Lagrangian optimization approach when the distortion and the bit rate contribution of each coding unit can be computed independently. When independence cannot be assumed, the computational cost of the optimal solution becomes exponentially high on the number of coding units. In these cases, simplifying assumptions and heuristics are necessary in order to achieve a practical solution. In face of this dependence problem and associated complexity, we propose in the following a sub-optimal greedy approach for quantization of the control points.

3. BEING GREEDY

The greedy approach is used in many instances of the Computer Science literature. Sometimes this approach is able to produce the optimal solution [6]. One very famous example of optimal greedy approach is the Huffman coding [5]. However, many times a sub-optimal solution is acceptable when the computational cost of an optimal solution is extremely high. The idea of being greedy is to look for a solution which provides the best local improvement at each step of the algorithm. This approach may not produce the best overall result for a given number of steps.

In most cases, a greedy approach implies in an iterative algorithm. Our approach is to look for a greedy way to quantize the control points at each iteration. We modify only one control point per iteration. The problem is to define which control point to change (quantize) in order to obtain a good tradeoff between compression and distortion. In the following we formalize the problem at hand.

4. DISTORTION AND ENTROPY

4.1. Entropy of a m-th order Markov Source

Given a sequence of control points in the set $X = x_1, \cdots, x_n$ which defines the blending surfaces to represent the regions of an image or a 3D object. Without lost of generality, let us assume that each control point is represented by a fixed number of bits or equivalently, by uniform quantization using N_q levels. Therefore, this data source has at most N_q different symbols. Considering that this data source is going to be entropy encoded by either Huffman [5] or arithmetic coding [12], the overall bit-rate can be estimated by the entropy of this source [11]:

$$H(S) = \sum_{S^m} p(s_{j1}, s_{j2}, \cdots, s_{jm}) H(S|s_{j1}, s_{j2}, \cdots, s_{jm}) \tag{1}$$

where $p(s_{j1}, s_{j2}, \cdots, s_{jm})$ is the probability of the last m symbols be the ordered set $S_j = s_{j1}, s_{j2}, \cdots, s_{jm}$. This also can be interpreted as the probability of a m-th Markov source be in the state S_j. S^m indicates that the summation is applied to all possible states. The source entropy for a given state S_j, $H(S|s_{j1}, s_{j2}, \cdots, s_{jm})$, is given by:

$$-\sum_{i=1}^{Nq} p(s_i|s_{j1}, s_{j2}, \cdots, s_{jm}) \cdot log(p(s_i|s_{j1}, s_{j2}, \cdots, s_{jm})) \tag{2}$$

In practice, the entropy coders can achieve a bit rate very close (typically $> 95\%$) to the expression 1.

4.2. Distortion Associated with Control Points

Each control point x_i is represented by a symbol j or equivalently, it is represented by one of the N_q quantization levels j (also called bin j). Let us call this association as $x_{i,j}$. Let us define $D(i,j)$ as the distortion of a control point i, associated with a quantization level j. Since the control points are shared by some of the neighboring regions $s = 1, \cdots, NR$, we compute the distortion $D(i,j)$ as the average of the distortions DR_s of these sharing regions. DR_s is computed as the sum of the squared distance between points inside the quantized blending surface and points in the original surface for the region s. By changing a control point $x_{i,j}$ to another quantization level k, the new associated distortion becomes $D(i,k)$ and it also affects the associated distortions of the control points located in regions that share the control point x_i.

The formulation above is not restricted to the problem of blending surfaces, it can be applied to any case where exists dependence among coding units.

5. THE GREEDY WAY

The greedy approach determines at each iteration which control point should be moved to another quantization level such that the entropy reduces and the introduced distortion is the smallest among all other possibilities. This approach needs a set of comparisons for all possible distortions generated by changing the quantization levels of the control points. This greedy approach reduces the entropy and introduces the smallest distortion possible at each iteration. The overall distortion and bit-rate can be controlled by monitoring the introduced variations of bit-rate and distortion at each iteration.

This approach provides a finer bit-rate/distortion control as opposed to the uniform quantization approach. For a given data source, the uniform quantization can provide some points in the resulting rate-distortion (R-D) curve. The number of points depends on the number of different quantization levels used to generate this experimental curve. However, only a few quantization levels are practical: those which provide acceptable tradeoff between distortion and bit-rate. The proposed approach starts from one these points in the R-D curve and produces other non-existent points with lower entropy.

In the following we present two examples of application of the proposed greedy quantization approach. For simplicity, the following algorithms model the source data by a 0-th Markov model.

6. THE 0-th MARKOV MODEL

For this simplest model Eq. 1 becomes:

$$H = -\sum_{l} \left(\frac{f_l}{n} log \left(\frac{f_l}{n} \right) \right) \tag{3}$$

Given the quantized control points sequence $X = x_1, \cdots, x_n$, the algorithm searches for a new sequence $X' = x'_1, \cdots, x'_n$ presenting a lower bit-rate. Recalling that each control point x'_i is quantized to a bin j; every

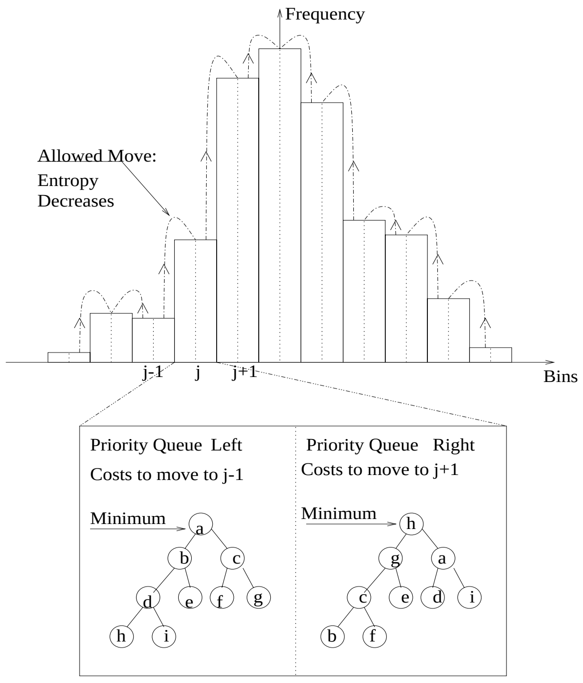

Fig. 3. Priority queues associated with a bin j in a histogram of control points.

bin j, $j > 1$, $j < N_q$, has two neighboring bins: the left bin $j-1$ and the right bin $j+1$, as illustrate by the histogram in Fig. 3. The distortion associated with the control point $x'_{i,j}$ is $D(i,j)$. This distortion takes into account those neighboring triangular surfaces sharing the point $x_{i,j}$. Moreover, two other costs are associated with each control point $x'_{i,j}$: the cost of moving it to the left bin, $D(i,j-1)$ and the cost of moving it to the right bin, $D(i,j+1)$. The greedy approach finds at each iteration the element $x'_{i,j}$ (associated with bin j) and moves it to a neighboring bin k, such that:

A) the frequencies associated with the bins j and k follow the inequality: $f_j \leq f_k$;

B) the element $x'_{i,j}$ when moved to a neighboring bin, either $k = j-1$, or $k = j+1$, has the minimum associated distortion $D(i,k)$ among all other possible movements of elements and respective distortions presenting the property A.

In this algorithm, at each iteration, the entropy reduces and the distortion is increased by the least possible increment. This property is verified by the simulations and it is demonstrated in the following.

7. ENTROPY REDUCTION

Hypothesis: "The 0-th order entropy of a control point bitstream reduces if an element $x_{i,j}$ associated with bin j, with frequency f_j, is moved to a bin k, presenting frequency $f_k \geq f_j$". The entropy before moving the element is:

$$H = -\sum_l \left(\frac{f_l}{n} log\left(\frac{f_l}{n}\right)\right) \qquad (4)$$

The variation of entropy after moving the element is:

$$\triangle H = -\frac{f_k+1}{n} log\left(\frac{f_k+1}{n}\right) + \frac{f_k}{n} log\left(\frac{f_k}{n}\right) - \\ -\frac{f_j-1}{n} log\left(\frac{f_j-1}{n}\right) + \frac{f_j}{n} log\left(\frac{f_j}{n}\right) \qquad (5)$$

To prove the original hypothesis we only need to show that $\triangle H < 0$. This inequality is arranged as:

$$f_k \cdot log\left(\frac{f_k+1}{f_k}\right) + log\left(f_k+1\right) + \\ + f_j \cdot log\left(\frac{f_j-1}{f_j}\right) - log(f_j-1) > 0 \qquad (6)$$

Note that frequencies are positive integers. It is easy to see from the last expression that the worst case, where this inequality could fail, is when $f_k = f_j$. If the inequality is valid for this worst case, it will also be valid for $f_k > f_j$. Therefore, we only need to show the following sufficient condition:

$$(f+1)^{f+1} \cdot (f-1)^{f-1} \left(\frac{1}{f}\right)^{2f} > 1 \qquad (7)$$

This inequality is obtained from the previous equation using $f = f_k = f_j$. The above inequality holds for $f \geq 1$, therefore the initial hypothesis is valid. This property was observed in the simulations: each iteration resulted in an entropy reduction. Since we are using entropy encoders, the entropy reduction at every step improves the compression, as observed in the simulations.

8. FAST SEARCH USING PRIORITY QUEUES

At each iteration, we need to find the element which has the two properties described before. It is necessary to compare all distortions and to determine the element presenting the minimum $D(i,k)$ where property A is valid. A straightforward implementation would require n comparisons for each iteration. We propose a more efficient algorithm by keeping the left costs and right costs of each element in two priority queues (binary heaps) [16] for each bin. Basically, a priority queue needs two operations: insertion and deletion of an element. The insertion algorithm used here is very similar

```
ALGORITHM Deletion(BinaryHeap A, Position $p$)
REM A [0, $\cdots$, heapsize-1] array holds the binary heap
REM X is in A[$p$] and L is in A[heapsize-1]
→ Exchange X ⟷ L;
→ heapsize - -;
→ IF Parent(L) < L
→→ IF (X < L)
→→→ Heapify(L);
→→→ done;
→ ELSE
→→ WHILE Parent(L) > L
→→→ Exchange Parent(L) ⟷ L;
```

Table 1. Deletion algorithm for priority queues. See [16] for Heapify() routine.

to the one presented in [16]. However, our proposed deletion algorithm can remove any element in a known position p. This new deletion algorithm is described in Table 1. Both operations have complexity of $O(log(n))$, where n is the number of elements in the queue. Assuming a number of iterations proportional to n, we can show that the overall complexity of our greedy algorithm is $O(n.log(n))$ [7]. This complexity increases considerably if priority queues, or similar scheme, are not used: $O(n^2)$.

9. RESULTS FOR STILL IMAGE CODING

The greedy approach was applied to the simplest form (without wavelet decomposition) of the still image coding algorithm in proposed [10]. First, the coding algorithm produces a control point bitstream uniform quantized into N_q levels. The greedy algorithm is then applied to this quantized bitstream. The resulting control point bitstream is entropy encoded by arithmetic coding with a 2nd order context model. For results in Table 2 we use $N_q = 64$. We illustrate in this table a great improvement by using the greedy approach: we reduce the bit rate considerably while marginally reducing the overall distortion.

10. GREEDY QUANTIZATION FOR 3D OBJECTS

We investigate the problem of quantizing coordinates in a 3D space. We assume that 3D objects are constructed by a mesh of M triangles and each triangle is completely defined by 3 points A, B and C. Each point is defined by its coordinates (x_i, y_i, z_i). As before, each point is shared by the neighboring triangles. Therefore, the distortion due to quantization of a

Images	Non-optimized [10]	Optimized (greedy)
Lenna	0.52 bpp, 31.23 dB	0.42 bpp, 31.43 dB
Peppers	0.67 bpp, 30.14 dB	0.56 bpp, 30.26 dB
Balls	0.12 bpp, 32.05 dB	0.10 bpp, 32.35 dB
Hotely	0.69 bpp, 30.14 dB	0.56 bpp, 30.26 dB
Barbara	0.73 bpp, 23.76 dB	0.64 bpp, 23.86 dB

Table 2. Results and comparisons for greedy algorithm, no wavelet decomposition.

given point is propagated to all neighboring triangular surfaces.

We modified the greedy algorithm to quantize the set of coordinates which represent a 3D object. The approach is to change the iterative process such that we quantize the coordinates of a point in a sequential fashion. We split the coordinates into 3 vectors, one for the "x" component, one for the "y" component and one for the "z" component. Each vector is independently uniform quantized and the elements are properly inserted in the corresponding priority queues. The iterative process works in one element each time; first we find the element in the "x" vector which has the two properties described at Section 6 and move it to the proper bin and associated priority queues. Next, we repeat the one element process to the "y" vector and then to the "z" vector. The next iteration starts by modifying an element in the "x" vector followed by modifying the "y" and "z" vectors.

The process stops when a certain number of iterations is achieved or a given bit-rate or distortion is obtained. It is important to notice that the 3 vectors are not independent since when a component (in the "x" vector for instance) is quantized to another bin, the introduced distortion affects the distortion associated with other components (in the "y" and "z" vectors for instance). This dependence is considered in the iterative process by updating the distortions in the affected priority queues.

In order to evaluate the described algorithm extension to the 3rd dimension, we use a sphere generated by a mesh of triangles. Two models are considered: model 3 with 512 triangles and model 4 with 2048 triangles. We generate the VRML representation for these models and it resulted in two files, one with 17463 bytes (model 3) and the other with 69411 bytes (model 4). The usual compression scheme for VRML is the Lempel-Ziv algorithm. The resulting compressed files for the model 3 and 4, using the GNU gzip, have sizes of 4088 bytes and 16358 bytes respectively.

We use the Euclidean distance as a measure of dis-

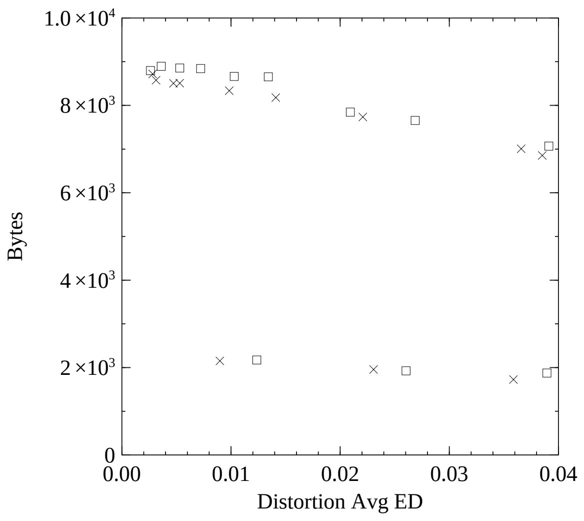

Fig. 4. Comparison between scheme 1 (only quantization, □) and scheme 2 (greedy, ×). Distortion computed as average Euclidean distance versus bit-rate computed as bytes.

tortion. The greedy quantized vectors are encoded by arithmetic coding with a 2nd order context model. We can achieve a compression ratio of about two over the traditional VRML [3] compression using Lempel-Ziv based coders (e.g. zip and gzip). For instance, for the model 3 the algorithm produces a file with 2152 bytes and no visible distortion. For the model 4, the algorithm generates a file of 8179 bytes also with no visible distortion. The quantization process takes only a couple of seconds in an AMD K6-3 450 MHz running on Linux operating system.

The Figure 4 shows a comparison between the uniform quantized only version and the greedy optimization algorithm. We can see that the greedy optimization can provide a higher compression for the same quality or a higher quality for the same compression. Moreover, the gain over the version using only quantization is small when compared to the bigger improvement achieved by the 2D case using our 2nd degree blending functions [10].

11. CONCLUSIONS

We describe an $O(n \cdot log(n))$ iterative algorithm for quantization of blending surfaces. The greedy approach provides an excellent performance when compared to uniform quantization. We illustrate this performance by given two examples for image and 3D object compression.

REFERENCES

[1] A. Watt and M. Watt. Advanced Animation and Rendering Techniques - Theory and Practice. *New York, ACM Press, Addison-Wesley*, 1995.

[2] Allen Gersho and Robert M. Gray. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, Boston, 1992.

[3] Andre Gueziec, Gabriel Taubin and Bill Horn. A Framework for Streaming Geometry in VRML. *IEEE Computer Graphics and Applications*, 19(2):68–78, April 1999.

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

[5] David A. Huffman. A Method for Contruction of Minimum Redundancy Codes. *Proceedings of IRE*, 40:1098–1101, 1952.

[6] James Foley et al. *Computer Graphics - Principles and Practice.* Addison Wesley, New York, 1996.

[7] Joceli Mayer. A Blending Model for Efficient Compression of Smooth Images. In *Data Compression Conference, Snowbird*, March 1999.

[8] Joceli Mayer. *Blending Models for Image Enhancement and Coding.* PhD thesis, University of Califonia Santa Cruz, 1999.

[9] Joceli Mayer. Greedy Quantization of Control Points for 2D and 3D data Using Blending Surfaces Representation. In *33rd Asilomar Conference on Signals, Systems, and Computers"*, October 1999.

[10] Joceli Mayer and Glen Langdon. Region Based Image Compression using Recursive Triangular Partitioning with a Blending Model. In *32nd Asilomar Conference on Signals, Systems and Computers*, November 1998.

[11] Majid Rabbani and Paul W. Jones. *Digital Image Compression Techniques.* SPIE Optical Engineering Press, Washington, 1991.

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

[13] Martin Vetterli and Jelena Kovacevic. *Wavelets and Subband Coding.* Signal Processing Series. Prentice Hall, Englewood Cliffs - New Jersey, 1995.

[14] Raj Kumar Dash. Image Processing Using Quadtrees. *Dr. Dobb's Journal*, July 1993.

[15] Riccardo Distasi, Michele Nappi and Sergio Vitulano. Image Compression by B-Tree Triangular Coding. *IEEE Transactions on Communications*, 45(9):1095–1100, September 1997.

[16] Thomas H. Cormen, Charles E. Leiserson and Ronald L. Rivest. *Introduction to Algorithms.* The MIT Press, Cambridge, 1990.