LEAST-SQUARES MESH MODEL FOR IMAGE COMPRESSION[1]

Iciar Álvarez-Cascos and Yongyi Yang

Dept. of Electrical and Computer Engineering
Illinois Institute of Technology
3301 S. Dearborn St., Chicago, IL 60616, USA

ABSTRACT

In this work we explore the use of a content-adaptive mesh model for image compression. We first model the image to be compressed by a quadtree mesh representation, in which the nodal values are determined using a least squares fit. The resulting mesh structure is coded using a 4-ary tree, and the mesh nodal values are coded using a hierarchical predictive coding scheme. Our experimental results demonstrate that the proposed approach can achieve good compression performance, and can significantly outperform JPEG both subjectively and objectively in low bit-rate applications.

1. INTRODUCTION

Mesh modeling provides an efficient and compact representation of an image and is an effective tool for tracking rigid and non-rigid motion in image sequences. As a result, mesh modeling has recently found many important applications in image processing, including image compression [1-6], motion tracking and compensation [7-11], image processing through geometric manipulation [12], and medical image analysis [13].

Mesh modeling of an image involves partitioning the image domain into a collection of non-overlapping (generally polygonal) patches called *mesh elements*. The image function is then determined over each element by interpolation based on the values at the vertices, called *mesh nodes*, of the element.

In our previous work we developed a fast and accurate approach for image representation using a content-adaptive mesh model (CAMM) [14], and applied this approach to image restoration [15] and tomographic image reconstruction [16,17]. In this work we propose a CAMM approach for image compression.

The CAMM is essentially a form of image representation based on non-uniform sampling. Specifically, in a CAMM smaller mesh elements (hence more samples) are placed in regions of an image containing high frequency features, while larger elements are placed in regions containing predominantly low frequency components.

[1] This research was supported by NIH/NHLBI grant HL65425.

Our approach is different from existing mesh based methods for image compression in the following two aspects: 1) for the mesh model we employ a least-squares fit rather than interpolation, which can significantly improve the representation accuracy by a given mesh structure; and 2) we propose a hierarchical predictive scheme for coding the mesh nodal values, which is demonstrated to be very efficient.

2. MESH MODEL GENERATION FOR IMAGE REPRESENTATION

2.1 Mesh representation

Let $f(\mathbf{x})$ denote an image function defined over a domain D. In a mesh model, the domain D is partitioned into M non-overlapping mesh elements, denoted by D_m, $m = 1,2,\cdots,M$, such that $D = \bigcup_{m=1}^{M} D_m$. The image function $f(\mathbf{x})$ is then approximated over each element as

$$\hat{f}(\mathbf{x}) = \sum_{k=1}^{K} f(\mathbf{x}_k)\varphi_k(\mathbf{x}) \quad \text{for } \mathbf{x} \in D, \qquad (1)$$

where $\mathbf{x}_k$ is the kth mesh node, $\varphi_k(\mathbf{x})$ is the interpolation basis function associated with $\mathbf{x}_k$, K is the total number of mesh nodes used. Note that the support of each basis function $\varphi_k(\mathbf{x})$ is spatially limited to only those elements attached to the node $\mathbf{x}_k$. For convenience we denote the mesh nodes using a single index k, with the understanding that any given mesh node is typically shared among several mesh elements.

The mesh representation in (1) is an image description based on nonuniform sampling, wherein the mesh nodes are the sample points. Therefore, it is desirable that the mesh elements (and thus the nodes) be placed strategically according to the local content of the image, with samples placed most densely in areas having the most image detail.

2.2 Quadtree mesh generation procedure

While several different methods have been studied in the literature for generation of a content-adaptive mesh for a given image, in this study we use a so-called quadtree mesh structure [1,4], which is an effective tool for adaptive mesh generation. As will be seen in the next section, the structure of a quadtree mesh can be coded

efficiently using only a small number of bits, thereby avoiding the transmission of individual nodal locations.

The quadtree method starts with a coarse rectangular mesh (initially the whole image as a single element), and successively refines it in a hierarchical manner. At each step, the mesh element with the largest interpolation error is split into four sub-elements. This process continues until the desired level of accuracy (or number of nodes) is achieved. In our implementation, the interpolation error for each element D_m is computed as

$$E_m = \int_{D_m} \left[f(\mathbf{x}) - \hat{f}(\mathbf{x}) \right]^2 d\mathbf{x} . \qquad (2)$$

For digital images, the integration in (2) corresponds to a summation over discrete pixels.

To demonstrate the idea, we show in Fig. 1(b) the quadtree mesh obtained from the first few iterations of this procedure for the image in Fig. 1(a).

2.3 Least-squares mesh representation

In a conventional mesh representation as in (1), interpolation from the nodal sample values $f(\mathbf{x}_k)$ is used. However, we previously demonstrated [14] that use of a least-squares (LS) fit rather than interpolation can significantly improve the representation accuracy of the image. Therefore, in this study we determine the nodal values in (1) using a LS fit. Specifically, let f_k denote the value of the mesh representation $\hat{f}(\mathbf{x})$ at $\mathbf{x}_k$, k=1,2,…,K. Then

$$\hat{f}(\mathbf{x}) = \sum_{k=1}^{K} f_k \varphi_k(\mathbf{x}) = \mathbf{\Phi}^T(\mathbf{x}) \mathbf{f}_{\text{node}} , \qquad (3)$$

where

$$\mathbf{\Phi}(\mathbf{x}) = (\varphi_1(\mathbf{x}), \varphi_2(\mathbf{x}), ..., \varphi_K(\mathbf{x}))^T , \qquad (4)$$

$$\mathbf{f}_{\text{node}} = (f_1, f_2, ..., f_K)^T . \qquad (5)$$

Then the LS fit for the nodal values, denoted by $\mathbf{f}^*_{\text{node}}$, is found by minimizing the following mean square error (MSE) of $\hat{f}(\mathbf{x})$:

$$E = \int_D \left[f(\mathbf{x}) - \mathbf{\Phi}^T(\mathbf{x}) \mathbf{f}_{\mathbf{node}} \right]^2 d\mathbf{x} . \qquad (6)$$

The LS mesh representation of $f(\mathbf{x})$ is then computed as

$$\hat{f}^*(\mathbf{x}) = \mathbf{\Phi}^T(\mathbf{x}) \mathbf{f}^*_{\text{node}} . \qquad (7)$$

In our implementation the steepest descent algorithm was used for minimization of the quadratic objective function in (6). The LS representation in (7) was applied in each step of the quadtree method, during which the element with the largest error, as calculated in (2), was further subdivided into four equally-sized rectangular elements. The process was stopped once a prescribed error level or bit budget was reached.

3. MESH REPRESENTATION ENCODING

Once the mesh representation of an image is obtained, it is then encoded using the following steps: 1) 4-ary tree coding of the mesh structure, 2) hierarchical predictive coding of the mesh nodal values, and 3) entropy coding of the prediction errors.

3.1 Coding the mesh structure

The mesh structure obtained from the quadtree procedure can be represented by a 4-ary tree, in which the nodes are used to represent the mesh elements. Specifically, starting from the root node (i.e., the whole image), we represent each refinement of a mesh element by branching from the corresponding node of the element with 4 new child nodes. The resulting tree is then encoded using a unique binary bit-stream. As an example, in Fig. 1(c) we show the tree structure corresponding to the mesh in Fig. 1(b). This tree is then coded as 00101111011111011110111101111, where each parent node is coded as '0', and each leaf node is coded as '1'. Note that the mesh structure can be easily reconstructed from its binary code. The image will then be determined using this mesh structure.

3.2 Coding the nodal values

We adopt a systematic predictive scheme for coding the nodal values, taking advantage of the hierarchical nature of the quadtree mesh structure. Starting with the root element (the whole image), each subsequent refinement of a mesh element will produce a total of five new mesh nodes. These five mesh nodes are first predicted using (7) from their immediate parent element (i.e., the element just refined). The differences of these five predicted values from their actual values are computed and then transmitted (upon further entropy coding). In this hierarchical procedure, it may happen that among these five newly introduced nodes some have already been transmitted (when they are shared by neighboring elements). In such a case, it is no longer necessary to transmit these nodes.

The nodal predication errors are transmitted in the order defined by the 4-array tree of the mesh structure. The four nodal values for the initial mesh (i.e. the whole image) are all predicted using the mean value (128 for 8-bit images).

3.3 Entropy coding

The nodal predictive errors are next quantized, and further compressed using entropy coding. In our experiments, 8-bit images were used; the nodal predictive errors were quantized to integers in the range of $[-256, 255]$, which were subsequently coded using a Huffman encoder. For demonstration purpose, a database of 22 images, all in size of 128×128, was used in this study to generate the histogram of the prediction errors, based on which the Huffman codes were designed.

We acknowledge that the database size used in our experiments might seem somewhat moderate; however, it is sufficient to demonstrate our basic compression algorithm in this preliminary study. We will investigate in the future with a larger database the benefit of adjusting the quantization step-size according to the bit rate.

3.4 Bit stream

The final compressed bit stream for an image consists of a concatenation of the following data fields: 1) a header, used to define the dimensions of the image; 2) quardtree code, used to define the mesh structure; and 3) Huffman codes for the nodal prediction errors.

4. EXPERIMENTAL RESULTS

In this section we present some results to demonstrate the performance of the proposed least-squares mesh (LSM) compression algorithm. Shown in Fig. 2(b) is the 128×128 "Afmsurf" image (shown in Fig. 2(a)) compressed by the proposed algorithm at a bit-rate of 0.132 bits/pixel (bpp); for comparison, we show in Fig. 2(c) the same image compressed by JPEG at 0.141 bpp. In addition, we show in Fig. 2(d) the LSM compressed image at 0.25 bpp, and in Fig. 2(e) JPEG compressed image at 0.275 bpp.

In Fig. 3 we show a similar set of compression results by LSM as compared to JPEG using a different test image. As can be seen, these results demonstrate that the LSM compressed images suffer much less distortion than their JPEG counterparts.

As an objective measure of compression distortion, we also computed the peak-signal to noise ratio (PSNR) for a compressed image:

$$\mathrm{PSNR} = 10 \log_{10} \frac{255^2 \times M \times N}{\sum_{i=1}^{M} \sum_{j=1}^{N} \left[f(i,j) - \hat{f}(i,j) \right]^2} \qquad (8)$$

where M,N correspond to the image size.

Finally, we show in Fig. 4 the rate-distortion (R-D) curves computed for the proposed LSM compression algorithm and JPEG, obtained for the two test images. The proposed LSM method significantly outperforms JPEG at low bit-rates.

5. CONCLUSIONS

In this paper we proposed an image compression approach based on least-squares mesh modeling. In this approach we first model the image to compressed by a content-adaptive mesh model, which is essentially a form of signal representation by non-uniform sampling. This mesh representation is then coded efficiently using hierarchical predictive coding. Our preliminary results demonstrate that the proposed approach can achieve good compression performance, significantly outperforming JPEG in low bit-rate applications.

6. REFERENCES

[1] E. Shusterman and M. Feder, "Image compression via improved quadtree decomposition algorithms," *IEEE Trans. Image Proc.*, vol. 3, no. 2, pp. 207-215, 1994.

[2] K. Aizawa and T. S. Huang, "Model-based image coding: advanced video coding techniques for very low bit-rate applications," *Proc. of IEEE*, vol. 83, no.2, pp. 259-271, 1995.

[3] F. Davoine, M. Antonini, J. Chassery, and M. Barlaud, "Fractal image compression based on Delaunay triangulation and vector quantization," *IEEE Trans. Image Proc.*, vol. 5, no. 2, pp. 338-346, 1996.

[4] J. Knipe and X. Li, "On the reconstruction of quadtree data," *IEEE Trans. Image Proc.,* vol. 7, no. 12, 1998.

[5] H. Benoit-Cattin, P. Joachimsmann, A. Planat, S. Valette, A. Baskurt, R. Prost, "Active mesh texture coding based on warping and DCT," *IEEE Inter. Conf. Image Proc*., Kobe, Japan, 1999.

[6] L. Demaret, G. Robert, N. Laurent, A. Buisson, "Scalable image coder mixing DCT and triangular meshes," *IEEE Inter. Conf. Image Proc*., vol. 3, pp. 849-852, Vancouver, Sept., 2000.

[7] Y. Wang and O. Lee, "Active mesh--a feature seeking and tracking image sequence representation scheme," *IEEE Trans. Imag. Proc*., vol. 3, no. 5, pp. 610-624, 1994.

[8] Y. Altunbasak and A. M. Tekalp, "Closed-form connectivity-preserving solutions for motion compensation using 2-D meshes," *IEEE Trans. Imag. Proc.*, vol. 6, no. 9, 1997.

[9] G. Marquant, S. Pateux, and C. Labit, "Mesh and "crack lines": application to object-based motion estimation and higher scalability," *IEEE Inter. Conf. Image Proc*., vol. 2, pp. 554-557, Vancouver, Sept., 2000.

[10] A. Nosratinia, "New kernels for fast mesh-based motion estimation," *IEEE Trans. Circuits and Systems for Video Tech*., vol. 11, no. 1, pp. 40--51, 2001.

[11] P. Hsu, K. J. R. Liu, and T. Chen, "A low bit-rate video codec based on two-dimensional mesh motion compensation with adaptive interpolation," *IEEE Trans. Circuits and Systems for Video Tech*., vol. 11, no. 1, pp. 111 -117, 2001.

[12] M. A. Garcia and B. X. Vintimilla, "Acceleration of filtering and enhancement operations through geometric processing of gray-level images," *IEEE Inter. Conf. Image Proc*., vol. 1, pp. 97-100, Vancouver, Sept., 2000.

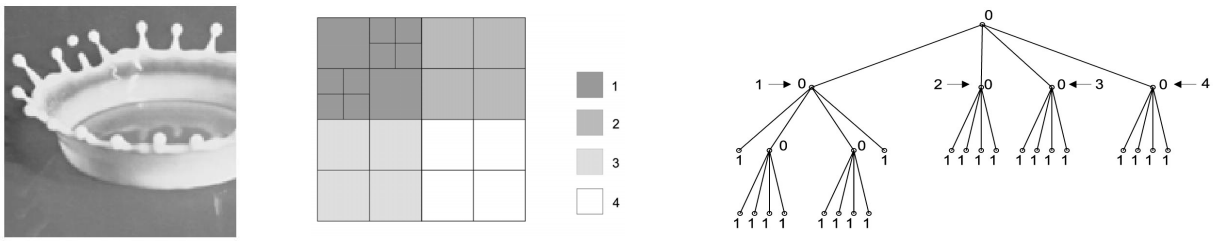
[13] A. Singh, D. Goldgof, and D. Terzopoulos, ed., *Deformable Models in Medical Image Analysis*, IEEE Computer Society Press, 1998.

[14] Y. Yang, J. Brankov, and M. Wernick, "A computationally efficient approach for accurate content-adaptive mesh generation," *IEEE Trans. Image Proc.,* vol. 12, no. 8, 2003.

[15] J. G. Brankov, Y. Yang, and N. P. Galatsanos, "Image restoration using content-adaptive mesh modeling," *IEEE Inter. Conf. Image Proc*., vol. 2, Barcelona, Spain, Sept. 14-17, 2003.

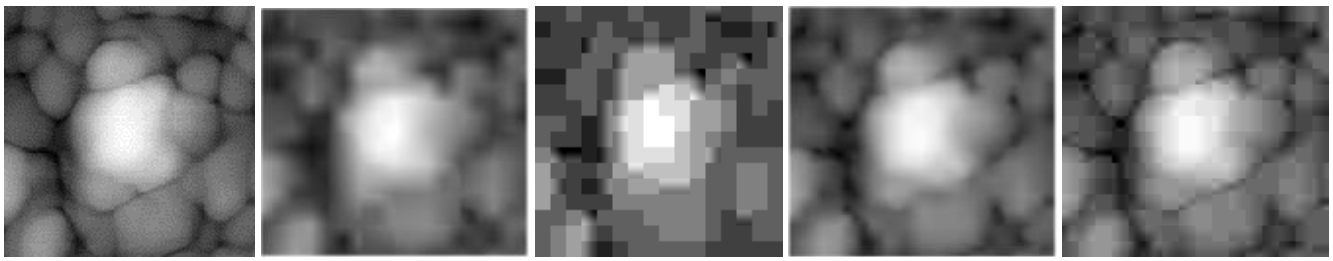
[16] J. G. Brankov, Y. Yang, and M. N. Wernick, "Tomographic image reconstruction using content-adaptive mesh modeling," *IEEE Inter. Conf. Image Proc*., vol. 1, pp. 690-693, Thessaloniki, Greece, Oct., 2001.

[17] Y. Yang, J. G. Brankov, and M. N. Wernick, "Content-adaptive mesh modeling for fully-3D tomographic image reconstruction," *IEEE Inter. Conf. on Image Proc., Rochester*, vol. 2, pp. 621-624, New York, Sept. 22-25, 2002.



(a) (b) (c)

Figure 1. (a) original image; (b) first few iterations of the quadtree mesh structure for the image in (a); (c) tree structure representing the mesh in (b).

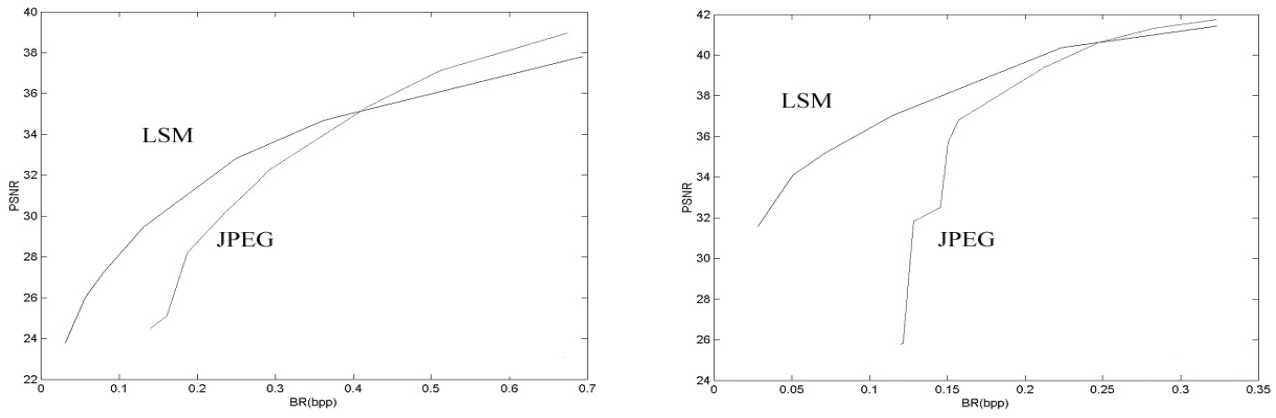


(a) (b) (c) (d) (e)

Figure 2. (a) original image; (b) mesh compression at 0.132 bpp, 29.47 dB; (c) JPEG compression at 0.141 bpp, 24.52 dB; (d) mesh compression at 0.25 bpp, 32.8 dB; and (e) JPEG compression at 0.275 bpp, 31.59 dB.

(a) (b) (c) (d) (e)

Figure 3. (a) original image; (b) mesh compression at 0.09 bpp, 29.47 dB; (c) JPEG compression at 0.12 bpp, 25.8 dB; (d) mesh compression at 0.18 bpp, 40.36 dB; and (e) JPEG compression at 0.21 bpp, 39.39 dB.



(a) (b)

Figure 4. R-D curves for JPEG and mesh compression for test images in Figs. 2(a) and 3(a), respectively.