Joint Embedded Coding of Data and Grid Using First-Generation Wavelet Transforms

James E. Fowler Yonghui Wang

Department of Electrical and Computer Engineering
Engineering Research Center
Mississippi State University, Starkville, Mississippi

Abstract

Many applications in a variety of scientific domains produce datasets that consist of a data field lying on a sampling grid that may not be uniformly spaced. However, progressive access for visualization, exploration, and communication of these datasets is a critical issue, and wavelet-based embedded coding an attractive solution given its prior success in realms such as image coding. As grid information may compose a significant portion, or even a majority, of that of the overall dataset, coding the grid as initial overhead is often impractical. Approaches for the joint embedded coding of data and grid are proposed using first-generation wavelet transforms so as not to require prior grid knowledge for transform inversion. In one proposed technique, two independently generated embedded codings, one for data and one for grid, are interleaved. As an alternative, an embedded vector-valued coder, in which data and grid are combined into a single vector-valued field, is considered. Experimental results are reported that favor the former approach over the latter.

Introduction

Wavelet-based embedded coding has recently become a preferred paradigm for lossy coding, particular so in the realm of still-image coding. One would like to capitalize upon the success of modern embedded image coders by directly applying their techniques in other applications. However, images are a relatively simple data format, being two-dimensional fields of pixel values lying on a sampling grid that is nearly always rectangular and uniformly spaced. Being implicit to the image, such a uniform rectangular sampling grid is known to both the encoder and decoder once described with a minimal amount of information (i.e., a width and a height), so that the grid itself need not be explicitly coded.

However, in a variety of applications, datasets possess both a data field and an *explicit* grid field which is often more complicated than the rectangular grids ubiquitous to image coding. For example, computational field simulations and empirically measured fields in oceanographic, aerodynamic, and electromagnetic applications, among others, produce datasets in which the grid is usually nonuniformly spaced. In these applications, the density of grid points is concentrated in regions of greatest relevance to the application to thereby focus computational resources on such regions. In addition to employing such nonuniform grids, these applications routinely produce datasets of unprecedented size (terabyte and petabyte range). The coupling of large size and nonuniform sampling in these applications presents several challenges to the deployment of lossy compression. The work presented

This work was funded in part by the National Science Foundation under the Large Data and Scientific Software Visualization Program, Grant No. ACI-9982344.

in *Proceedings of the IEEE Data Compression Conference*, J. A. Storer and M. Cohn, Eds., Snowbird, UT, April 2002, pp. 432-441.

here is an investigation of issues surrounding wavelet-based embedded coding for nontraditional datasets and a development of techniques to overcome the associated challenges.

The primary challenge is one of access. As a consequence of the extremely large data size involved in the applications of interest, access to the dataset for visualization, communication across a network, "browsing" or exploration, and other processes, is a critical issue. For these tasks, progressive transmission is greatly needed to circumvent practical limitations on resources such as network bandwidth and computer memory. Wavelet-based embedded coding is an attractive approach to providing such progressive access. The straightforward solution would be to transmit some high-fidelity, or lossless, coding of the grid field as initial "overhead" and follow with an embedded representation of the data field using techniques found in modern wavelet-based embedded coders. Unfortunately, this simple approach is usually impractical—for the applications considered here, it often is the case that the amount of information to be coded in the grid field is on the order of, or perhaps even larger than, that in the data field. For example, each node of a dataset consisting of a scalar data field lying in a three-dimensional space possesses one scalar data value and three scalar grid values, so one would expect that the storage burden for the grid field could be three times that of the data field. The straightforward solution provides an embedded coding of the data field, but not of the dataset as a whole. To meet this latter goal, the grid itself should be coded *jointly* with the data field in an embedded fashion.

In the remainder of the manuscript, we first consider the deployment of wavelet transforms on nonuniform grids and examine the issue of calculating distortions between datasets lying on imperfectly reconstructed sampling grids. We then propose two methods for jointly coding data and grid information. The first involves the production of an approximately embedded bitstream for the dataset as a whole by interleaving fixed-length packets extracted from two independently produced embedded bitstreams, one for data and one for grid. The second approach consists of combining the data and grid information together into a single vector-valued field and employing a vector-valued embedded coder based on successive-approximation vector quantization (SAVQ). We conclude with experimental results that compare the vector-coding method and several implementations of the interleaving approach and that indicate that the interleaving approach significantly outperforms its vector-based counterpart.

Before entering the heart of our discourse, a remark concerning topology is in order. In a one-dimensional space, all sampling grids are isomorphic to the "rectangular lattice," i.e., the uniformly spaced lattice. However, in higher dimensions, other topologies become possible, such as triangular meshes in two dimensions and tetrahedral meshes in three dimensions. Although certain applications do rely on complicated gridding topologies, a majority of the grids encountered in applications of interest here employ rectilinear grids (grids composed of rectangles), curvilinear grids (rectilinear grids that are "bent" or "warped" in space), or multiple blocks of such gridding structures. Additionally, even though there exists some limited work (e.g., [1]) exploring wavelets outside the traditional realm of rectangular lattices, it is currently unknown how to provide wavelet transforms on arbitrary topologies. As a consequence, we restrict our attention to rectilinear and curvilinear gridding structures isomorphic to rectangular lattices. In fact, in order to simplify discussion and experimental results, we investigate wavelet transforms and embedded coding for datasets lying in a one-dimensional space and thus necessarily employ a rectilinear grid.

in *Proceedings of the IEEE Data Compression Conference*, J. A. Storer and M. Cohn, Eds., Snowbird, UT, April 2002, pp. 432-441.

However, the applicability of the proposed techniques to higher-dimensional spaces, which are more relevant to the applications of interest here, holds in that the one-dimensional techniques can be applied in the usual "separable" fashion in higher dimensions as long as the topology remains recti- or curvilinear.

Wavelet Transforms on Nonuniform Grids

Suppose we have a scalar-valued function, $f(t)$, which we sample at points t_i so that we have a data field consisting of values $f_i = f(t_i)$ located at grid points t_i along the time axis. Assume, without loss of generality, that $t \in [0, 1]$, and that, for the moment, the t_i grid points may be arbitrarily (i.e., nonuniformly) spaced. Then, in a general sense, wavelet transforms can be considered to consist of two key components: 1) a downsampling/upsampling scheme that produces coarsely-spaced grids from finely-spaced grids and vice versa, and 2) a set of filter coefficients that control how coefficients at one scale are derived from those at the next finer or coarser scale.

First-Generation Wavelet Transforms

In familiar first-generation wavelet systems, both the sampling and filtering schemes take on their simplest form; that is, the sampling scheme is *regular*, and the filtering scheme is *invariant* across time and scale. Specifically, in a regular upsampling scheme, the grid points added to a coarse grid to produce the finer grid of the next scale are the midpoints of the intervals between neighboring points in the coarse grid. That is, a regular upsampling scheme implies that the t_i grid points are uniformly spaced. In terms of filtering, in first-generation systems, the filtering scheme is invariant across time and scale so that the coefficients associated with the grid points of the next-finer scale are derived from coefficients at the coarse scale *independently* of the location in time or scale of the grid point. In perhaps more familiar terms, the result of regular sampling and invariant filtering is that the scaling and wavelet functions for first-generation wavelet systems are translates and dilates of a single pair of functions.

As an example of a first-generation wavelet system, consider the familiar DWT as used extensively in compression. In these applications, the data is sampled using a regular sampling strategy; i.e., the sample points t_i are equidistant, while the upsampling process produces fine-scale grid points at the midpoints between two existing coarse-scale grid points. In this case, one does not need to actually code the sample points t_i since the decoder knows this information *a priori* due to the regularity of the sampling scheme.

Second-Generation Wavelet Transforms

The fact that downsampling/upsampling and filtering schemes do not necessarily have to be regular and time-scale invariant yields the possibility of second-generation wavelet systems [2] whose sampling strategies place grid points at locations other than the interval midpoints and whose filtering may change across both time and scale. In these second-generation systems, wavelet and scaling functions are no longer translates and dilates of mother wavelet and scaling functions but have shapes that depend on the location in time and scale of the corresponding grid point.

in *Proceedings of the IEEE Data Compression Conference*, J. A. Storer and M. Cohn, Eds., Snowbird, UT, April 2002, pp. 432-441.

A second-generation lifting wavelet transform was proposed for nonuniformly sampled data in [2], in which detail coefficients are calculated as the difference between a polynomial-based interpolation on scaling coefficients at "even" grid points and the scaling coefficient at the "odd" grid point under consideration. The transform adapts to nonuniform grid spacing since the interpolation is calculated at the nonuniformly spaced locations of samples being interpolated; the locations of sample points in the finest-scale grid determine the overall sampling scheme at all scales.

First-Generation Transforms on Nonuiform Grids

For nonuniformly sampled data, one would prefer to use second-generation wavelet systems since the ability of the interpolation used in lifting to adapt to the nonuniform sampling means more accurate prediction, less energy in detail coefficients, and therefore better potential for compression. However, the interpolation used by nonuniformly sampled lifting wavelets depends irrevocably on the sampling grid. Thus, to correctly invert the transform, the decoder needs to know, exactly and *a priori*, the nonuniform sampling arrangement in the time domain of the finest scale. As discussed previously, such grid pretransmission is impractical.

Realizing this impediment to second-generation systems, we instead propose employing first-generation strategies. One approach would be to resample the data field to a suitable uniform grid; however, resampling strategies are prone to serious artifacts [3], and our applications may be such that a representation of the original grid is necessary at the decoder side for certain processing and visualization tasks. Instead, we eliminate the regular-sampling restriction of first-generation wavelet systems while retaining invariant filtering. That is, we transform samples f_i using traditional first-generation wavelet filters (that are invariant in time and space) while ignoring that the f_i values come from nonuniformly space sample locations t_i. Of course, the cost associated with this approach is increased distortion over second-generation techniques since the encoder can no longer exploit the nonuniform sampling to minimize detail-coefficient energy. The advantage, though, is that the inverse transform on the data field no longer depends on the grid. The end result is that the data field and its grid may be compressed *simultaneously* in an embedded fashion since the decoder can invert the transform on the data field regardless of the accuracy of the current representation of the grid.

Distortion Measure for Nonuniformly Sampled Datasets

During joint coding of data and grid, an original dataset consisting of data-grid pairs (f_i, t_i), is represented as a set of reconstructed data-grid pairs $(\hat{f}_i, \hat{t}_i)$. We propose measuring the distortion between the original and reconstructed datasets using a mean square error (MSE) defined as $MSE = \frac{1}{N}\sum_{i=1}^{N}(f'_i - \hat{f}_i)^2$, where N is the number of samples in the dataset, and f'_i are values intended to represent the original data field as described below. This MSE definition is deceptively simple. The reconstructed data values $\hat{f}_i$ are output directly from the decoder. However, to obtain original data values f'_i, we have to consider two cases: 1) original continuous function $f(t)$ is known, and 2) $f(t)$ is not available (i.e., only discrete data f_i available). The latter case is the usual situation since empirical data is likely to be available only discretely while data arising in computational simulations is necessarily discrete.

in *Proceedings of the IEEE Data Compression Conference*, J. A. Storer and M. Cohn, Eds., Snowbird, UT, April 2002, pp. 432-441.

For the first case, we obtain original data values f_i' by sampling original function $f(t)$ at the reconstructed (decoded) grid locations, $\hat{t}_i$, which are output from the decoder. That is, $f_i' = f(\hat{t}_i)$. For the second case, because the original function is not available, we cannot get data values by directly sampling at the reconstructed grid points as in the preceding case. Instead, we use linear interpolation to approximate the function. Specifically, for a reconstructed grid point $\hat{t}_i$, we find a pair (t_j, t_k) such that $t_j < \hat{t}_i < t_k$ and $t_k - t_j$ is minimal, and derive f_i' as $f_i' = \frac{\hat{t}_i - t_j}{t_k - t_j}(f_k - f_j) + f_j$, with obvious modifications when $\hat{t}_i < t_1$ or $\hat{t}_i > t_N$. Both of these approaches the distortion calculation are imperfect but are reasonable assuming that the reconstructed grid values $\hat{t}_i$ are in the vicinity of the true grid locations t_i.

Interleaving of Independent Embedded Codings

In the this and the following section, we propose two methods for jointly coding data and grid information. The first technique, described in this section, involves the production of an approximately embedded bitstream for a dataset by the interleaving of two independently produced embedded bitstreams, one for data and one for grid. We first discuss the embedded coding of a single field and then present several packetization procedures that produce a final embedded representation of the dataset as a whole. We consider the second technique, a vector-valued embedded coder, in the subsequent section.

To produce an embedded bitstream of a single field, either data or grid, we first transform the field values using a first-generation lifting implementation of the 5-3 biorthogonal wavelet (i.e., linear lifting [2]) with symmetric extension at signal boundaries. To embeddedly code the resulting wavelet coefficients, a number of embedded coding algorithms could be used. For this work, we use a simple but effective successive-approximation coder [4] developed recently for the coding of ocean-temperature datasets. This coder, termed successive-approximation runlength (SARL) coding, combines successive-approximation embedded coding in the form of "bitplane" coding with efficient runlength coding similar to that of [5].

In order to produce an approximately embedded bitstream for a dataset as a whole, we produce SARL codings of the data field and grid field independently, and then "interleave" the resulting bitstreams. The interleaving of the two bitstreams takes place in the creation of packets and assembling these packets into a final bitstream.

Assume the final bitstream produced by the interleaving process is composed of M packets, where each packet has a fixed-length payload size of L bits and a fixed-length header size of H bits. The header is used to describe the partitioning of the current packet; the size of this header varies depending on the method of packetization. In the payload of packet m, the packetization procedure places L_g^m bits from the embedded bitstream for the grid field followed by L_d^m bits from the embedded bitstream for the data field, such that $L_g^m + L_d^m = L$ for all m. Thus, the interleaved bitstream consisting of M packets carries an embedded bitstream of length $L_g(M) = \sum_{m=1}^{M} L_g^m$ bits for the grid field and an embedded bitstream of length $L_d(M) = \sum_{m=1}^{M} L_d^m$ bits for the data field. Let the distortion obtained after M packets be $D(L_g(M), L_d(M))$. The main issue is then how one determines the partitioning of each packet between grid and data—or, equivalently, how one determines a

sequence of L_g^m values for a given dataset—so as to minimize the distortion of the reconstruction of the dataset for a given number of packets M. We overview several proposals for such optimization algorithms below.

Constant Partitioning—With constant partitioning, the ratio L_g^m/L_d^m is independent of m; that is, we use the same number of bits from the grid bitstream in each packet payload. Hereafter, we use term "k%-Grid" to indicate constant partitioning in which the grid-field coding occupies k% of each packet. For example, 70%-Grid consists of packets in which $L_g^m = 0.7 \cdot L$ for all m. Since the partitioning ratio is constant, the header overhead for constant partitioning is $H = 0$ bits for each packet.

Gradient-Descent Partitioning—In the gradient-descent method, we search for a locally optimal partitioning of the current packet using a gradient descent on distortion. That is, to determine the partitioning for packet n, the grid field is reconstructed from a prefix of length $L_g(n) = \sum_{m=1}^{n} L_g^m$ bits from the embedded grid-field bitstream, while the data field is reconstructed from a prefix of length $L_d(n) = \sum_{m=1}^{n} L_d^m$ bits from its corresponding bitstream. The distortion $D(L_g(n), L_d(n))$ is then calculated for the resulting reconstructed dataset, and the partitioning of the current packet, i.e., as represented by the L_g^n and L_d^n values, is optimized via a gradient descent. For packet $n > 1$, the gradient descent starts from the partitioning used in the previous packet, while the optimization of the first packet starts from an equal balance between data and grid.

Clearly, a major drawback to the gradient-descent approach to packetization is that a large number of reconstructions may be needed before the optimization converges, entailing large computational burden. Additionally, to permit the deinterleaving of the data and grid bitstreams for decoding, each packet carries a header overhead of $H = \lceil \log_2(L - 1) \rceil$ bits to describe the partitioning chosen for that packet, since L_g^m can take on values $1, 2, \ldots, L-1$.

Incremental Partitioning—In order to reduce the number of reconstructions from that of the gradient-descent approach, the last method we propose is that of incremental partitioning. In this approach, two reconstructions are performed for packet n, one using the L_g^n bits for the grid alone, and one using both the L_g^n grid bits and the L_d^n data bits. The distortion of the first representation is $D(L_g(n), L_d(n-1))$ while that of the second is $D(L_g(n), L_d(n))$. Define $\Delta D_g = D(L_g(n-1), L_d(n-1)) - D(L_g(n), L_d(n-1))$ as the improvement in distortion given by the first representation, and $\Delta D_d = D(L_g(n), L_d(n-1)) - D(L_g(n), L_d(n))$ as the additional improvement to be had by decoding the data bits too. If $\Delta D_g > \Delta D_d$, then the next packet uses $L_g^{n+1} = L_g^n + \Delta L$, while if $\Delta D_g < \Delta D_d$, the next packet has $L_g^{n+1} = L_g^n - \Delta L$. That is, we adjust the partitioning between data and grid by a small amount, $\pm\Delta L$, from the partitioning used in the previous packet, based on whether we expect data or grid to result in quicker refinement of the representation. In addition to reduced computation burden relative to gradient descent, the incremental-partitioning method has reduced header burden as well—a header of $H = 1$ bit suffices to distinguish between the two candidate partitions for the current packet.

Vector-Based Embedded Coding

An alternative to the interleaving of independent embedded codings as described above would be to assemble data and grid together and subject the resulting vector-valued data

in *Proceedings of the IEEE Data Compression Conference*, J. A. Storer and M. Cohn, Eds., Snowbird, UT, April 2002, pp. 432-441.

to vector-based embedded coding. That is, we represent the dataset as a single vector-valued field, $\bar{f}_i = [f_i, t_i]^T$, and, in creating a single embedded bitstream from the vector-valued data, we jointly code both data and grid in an embedded fashion. To implement this technique, we need a vector version of the SARL coder described previously. Such a vector-valued SARL (VSARL) coder must employ successive approximation of vectors in lieu of scalar-based bitplane coding.

Successive approximation for vector data has existed for quite some time under the guise of tree-structured vector quantization (TSVQ). A variant of TSVQ, multistage VQ (MSVQ), in particular, has been proposed frequently for SAVQ. Notably, a vector version of a zerotree coder was constructed from a hybrid of MSVQ and gain-shape VQ [6]. Our implementation of a SARL algorithm for vectors uses this hybrid MSVQ coder.

The SAVQ approach of [6] uses a scalar threshold that successively diminishes as in embedded scalar coding; however, the notion of bitplanes does not exist for vectors. Instead, the scalar threshold is compared to the *magnitude* of a vector to determine the significance or insignificance of that vector. Once a vector is determined to be significant, rather than coding coefficient-sign information as in scalar coding, an orientation angle from a predetermined codebook of possible angles is chosen and coded. Subsequent SAVQ passes refine the residual vector between the original vector and its current reconstruction with the same approach, coding magnitude significance as well as orientation angles.

In our experiments, vector-based embedded coding of the dataset employs the 5-3 biorthogonal wavelet transform on each vector component individually, and then follows with a VSARL coder, a vector version of the SARL algorithm. In VSARL, we replace the bitplane coding of SARL with the SAVQ procedure of [6]. Specifically, the runlength coding of significance is done the same as in SARL; however, the punctuation symbols between successive runlengths are symbols associated with orientation angles rather than coefficient signs. Additionally, a distinct refinement pass is not used; instead, residual vectors are refined using the same runlength-angle coding process. For the experimental results later, we use $\{0^\circ, 60^\circ, 120^\circ, 180^\circ, 240^\circ, 300^\circ\}$ as the codebook of orientation angles.

Experimental Results

In the experimental results presented below, we use two datasets intended to be representative of the two different paradigms through which nonuniformly sampled data may arise. The first dataset is derived from a piecewise-continuous scalar function $f(t)$, shown in Fig. 1(a). The dataset itself consists of 1024 data values that have been sampled, in a nonuniform fashion, according to the sampling-grid density depicted in Fig. 1(b), which places greater numbers of grid points in regions in which the function is changing rapidly. Since the original function $f(t)$ is available, distortion calculations on this dataset use $f'_i = f(\hat{t}_i)$ in the MSE calculation.

The second dataset (Fig. 1(c)) we use below is a set of discrete data values resulting from a computational simulation. The data field represents an air-velocity magnitude in a simulation of air flow passing over an iced airfoil of an aircraft at low flight speed. Like the artificial dataset above, the grid field for this simulation dataset is significantly nonuniformly spaced (Fig. 1(d)). As this dataset is inherently discrete, no continuous function is known, so the MSE distortion calculation uses interpolation estimates of the original data field at reconstructed grid locations as discussed previously.

in *Proceedings of the IEEE Data Compression Conference*, J. A. Storer and M. Cohn, Eds., Snowbird, UT, April 2002, pp. 432-441.

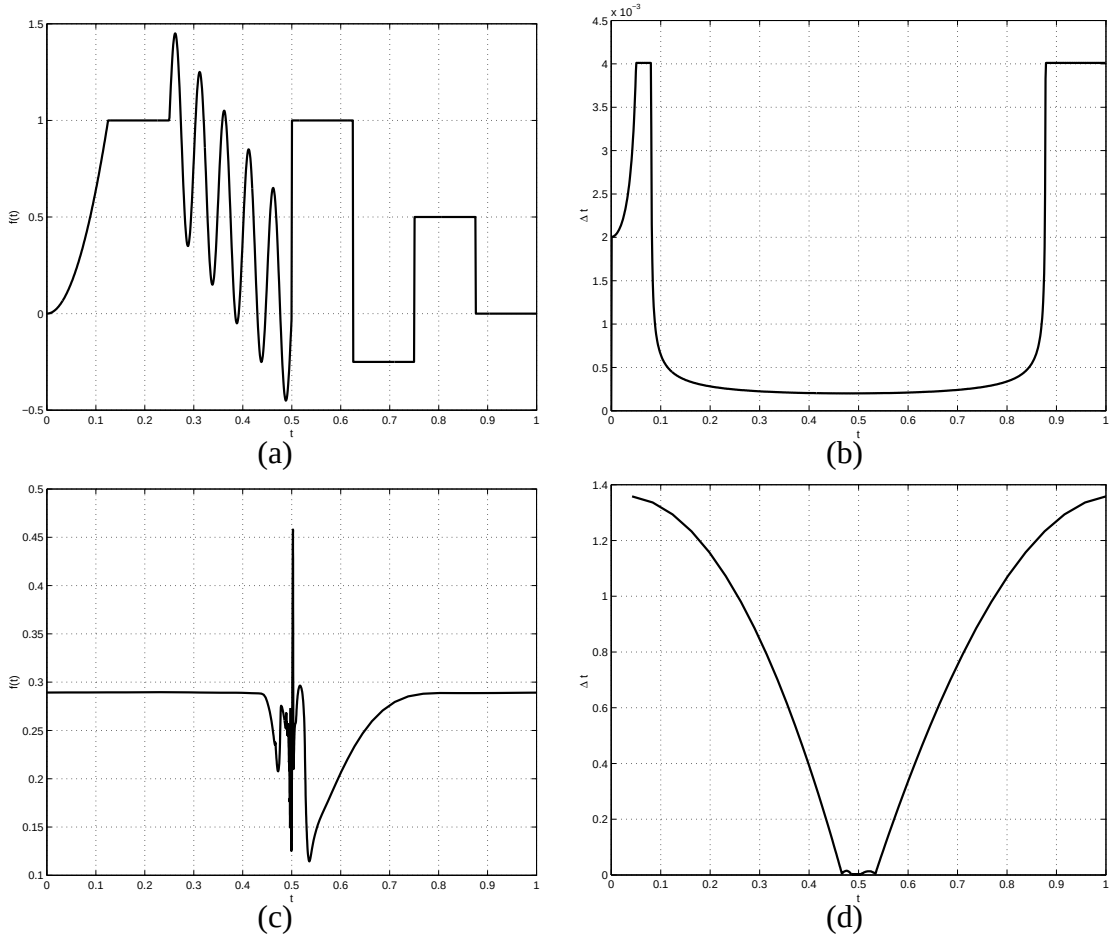


Figure 1: (a) An artificial piecewise-continuous function $f(t)$, and (b), its grid density represented as separation between adjacent grid points. (c) Iced airfoil data, a discrete dataset arising from a computational simulation and, (d) its grid density.

In all experiments, instead of directly coding grid values, we code the difference between the actual grid and a uniform grid of the same number of points. Reconstruction thus initiates with the uniform grid and approaches the true nonuniform grid as the rate increases.

We first investigate the performance of the interleaving of independent embedded codings of data and grid. Fig. 2 shows a comparison of the distortion-rate performance of the various packetization techniques for the two datasets under consideration. For each of the packetization schemes, we use a packet size of $L = 16$ bits with headers of $H = 0$, $H = 1$, and $H = 4$ bits, respectively, for the constant, incremental, and gradient-descent packetization approaches. For incremental partitioning, we use $\Delta L = 1$ bit, and all rate figures reported include packet-header overhead. The constant-partitioning curves shown are drawn from a larger body of results and represent the best performance obtained for the two datasets. We see that, for lower bit rates, the packetization techniques produce roughly equivalent results. However, for the higher-fidelity reconstructions of the dataset occurring at larger rates, a broader range in performance is observed. Note that, although the constant partitioning method produces better high-rate performance in each case, the better

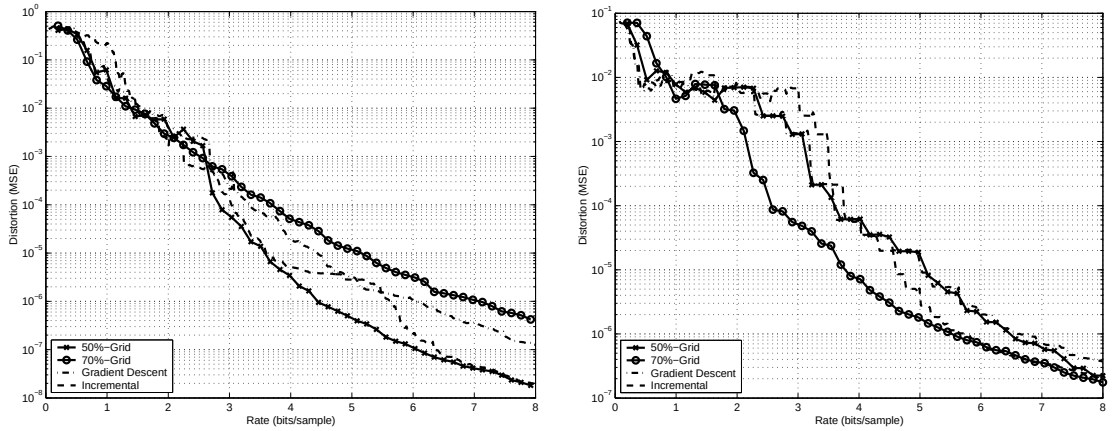


Figure 2: Distortion-rate performance of interleaving of independent embedded codings of data and grid using various packetization techniques. (a) Artificial piecewise-continuous function, (b) iced airfoil dataset.

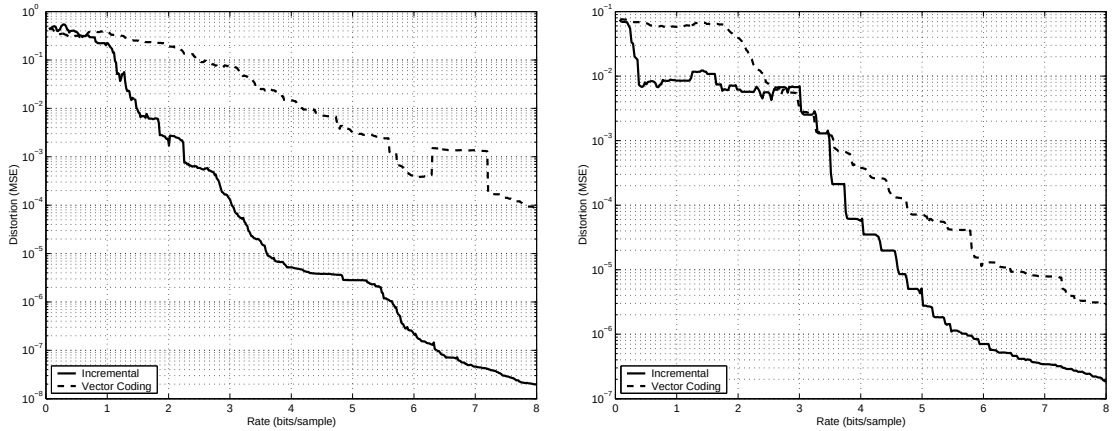
constant mixture ratio between data and grid is different for the two datasets—50% for the artifi cial signal of and 70% for the iced airfoil. This discrepancy suggests that datasets may have their own "optimal" data-grid mixture, and that constant partitioning would work well if this data-grid ratio could be predicted *a priori*. Since we are unable to make such a prediction, we opt for packetization procedures that attempt to "learn" the optimal ratio as coding progresses. The results indicate that the incremental-partitioning procedure is better capable than the gradient-descent approach at this task. Since it is also signifi cantly less computational expensive, incremental partitioning is thus the clear winner in performance.

Since Fig. 2 indicates that incremental partitioning is the preferred method for the interleaving of independent embedded codings of data and grid, we compare this approach to the VSARL vector-coding technique. We see from Fig. 3 that the incremental-partitioning approach easily outperforms the vector-based coder. We hypothesize that the reason for such a large discrepancy is that the SAVQ process in VSARL attempts to successively reduce distortion between the reconstructed and original vectors as measured in the two-dimensional Euclidean space in which the vectors reside, rather than distortion as measured on the reconstructed dataset. On the other hand, the packetization procedures employ a minimization that takes into account the reconstruction of the dataset, even though the individual embedded codings driving the packetization do not. The end result is that packetization of independent codings achieves a better reconstruction for the dataset.

Conclusions

In this paper, we have investigated issues surrounding the joint embedded coding of data and its underlying nonuniform sampling grid. We have explored the ramifi cations of nonuniform sampling on wavelet transforms, proposing fi rst-generation techniques over second-generation transforms in order to avoid the pretransmission of often-sizable grid information required by the latter for correct transform inversion. Additionally, we found that joint coding of data and grid through a vector-based method yields inferior results as compared to those observed for interleaving of independent embedded codings. We attribute

in *Proceedings of the IEEE Data Compression Conference*, J. A. Storer and M. Cohn, Eds., Snowbird, UT, April 2002, pp. 432-441.



(a) (b)

Figure 3: Distortion-rate performance of interleaving of independent embedded codings of data and grid using the incremental packetization technique versus the vector-coding approach using VSARL. (a) Artificial piecewise-continuous function, (b) Iced airfoil dataset.

this deficiency in the vector-based approach to a mismatch between vector distortion and distortion as measured on the reconstructed dataset.

Throughout this paper, we have attempted to propose techniques of a practical nature. That said, there are, perhaps, several impediments to the practicality of the methods explored here—notably, for example, the numerous reconstructions required for packetization when the packet size is small may remove interleaving of independent embedded codings from consideration in some applications. Despite this shortcoming of the present work, we believe that joint embedded coders for data and grid, such as those we have explored here and others inspired by this work, will eventually play a key role in next-generation systems for visualization, exploration, and communication of the large datasets arising in numerous scientific domains.

References

[1] P. Schröder and W. Sweldens, "Spherical Wavelets: Efficiently Representing Functions on the Sphere," in *Computer Graphics (Proceedings of SIGGRAPH 95)*, 1995, pp. 161–172.

[2] W. Sweldens and P. Schröder, "Building Your Own Wavelets at Home," in *Wavelets in Computer Graphics*, pp. 15–87. ACM SIGGRAPH Course notes, 1996.

[3] I. Daubechies, I. Guskov, P. Schröder, and W. Sweldens, "Wavelets on Irregular Point Sets," *The Royal Society of London. Philosophical Transactions. Series A. Mathematical, Physical and Engineering Sciences*, vol. 357, no. 1760, pp. 2397–2413, September 1999.

[4] J. E. Fowler and D. N. Fox, "Embedded Wavelet-Based Coding of Three-Dimensional Oceanographic Images With Land Masses," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 39, no. 2, pp. 284–290, February 2001.

[5] M.-J. Tsai, J. D. Villasenor, and F. Chen, "Stack-Run Image Coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 5, pp. 519–521, October 1996.

[6] M. Crazier, E. A. B. da Silva, and E. G. Ramos, "Convergent Algorithms for Successive Approximation Vector Quantization with Applications to Wavelet Image Compression," *IEE Proceedings: Vision, Image and Signal Processing*, vol. 146, no. 3, pp. 159–164, June 1999.