Compact Description and Modeling of Multidimensional Information

Krasimir Kolarov and William Lynch
Interval Research Corporation, 1801-C Page Mill Road, Palo Alto, CA 94304

Abstract

We present a technique to compress data defined as functions in high dimensional spaces. The objects in the spaces are represented by manifolds. Traditionally, data compression methods have been applied to functions defined on simple manifolds such as the real line (e.g., audio), a rectangle (e.g., images), or a three–dimensional open–ended box (e.g., video). However, many conventional data compression technologies, unmodified, are not suitable for compression of data defined on more complex geometries such as spheres, general polytopes, etc. Accordingly, this paper seeks to provide a transform compression technique for addressing 2–manifold domains using second generation wavelet transforms and zerotree coding.

1. Introduction

Many of the current topics of computational interest involve the representation of and the computation on continuous spaces. Obvious examples are images, video, and audio. In these cases data is distributed over and attached to a continuous rather than a discrete space. A photograph is well modeled as a reflectance over a rectangle.

The spaces to which data (the functions of interest) are attached are not always exactly the familiar Euclidean 1, 2, or 3 dimensional spaces. For example, the photograph is not defined on the plane (Euclidean 2–space) but rather on a rectangular subset of it. And its boundary, while locally Euclidean, is a closed curve. Such examples lead one naturally to the notion of a manifold — a space which is locally Euclidean but which may have a boundary (yet another manifold) and which may close back on itself (e.g., the sphere). In many applications, the primary space of interest is fairly simple, but its boundary is less simple and the locus of interesting features within it is less simple still.

Another interesting notion is that of subdividing a manifold. By subdividing a manifold into a larger number of smaller manifolds (with perhaps a complex boundary — another manifold — between them) a function attached to the manifold can be described subdivision–by–subdivision with the function on each particular subdivision being much easier to describe. In addition, we can describe a manifold as a collection of Euclidean balls (called cells), each with a spherical boundary, and with the collection joined together by sharing patches of their spherical boundaries. Such a description is conventionally called a regular CW complex and is an abstract description of the geometry of a manifold.

We can analyze the description of the functions defined on manifolds by considering the related notion of a multi–resolution description of a function. By going to several levels of sub–division, we can use each level as an approximation to the function being described. If we are careful, we can avoid repeating any information about the coarser levels, so that we get an increasingly detailed description of the function.

One way of doing this is by locally representing the function as a piecewise polynomial. A technique for doing this is the subject of "second generation wavelets" [7]. Another technique that has been used with great success is "multi–pole," where functions are represented by the ratios of polynomials.

Our computer representations are necessarily finitely generated, so we are always talking about representations that approximate the real world. Of course, we need to measure and control the degree of approximation appropriate to our specific application. A good approximation schemes will not only have appropriate and measurable fidelity, but will also be economical of both computational and memory resources. Such approximations are often called compression schemes, because they greatly reduce storage requirements, and often also enable a great reduction in the computation required.

In this paper we present a technique to compress functions defined on high dimensional manifolds. Our approach combines discrete wavelet transforms with zerotree compression, building on ideas from three previous developments: the lifting scheme, spherical wavelets, and embedded zerotree coding methods. §2 of the paper briefly describes the second generation wavelet transforms and the lifting scheme, as formulated in [8]. §3 reviews our implementation of the zerotree method for general 2–manifolds described in detail in [3] and [5]. §4 reports the results of applying our method to data defined on the surface of 3D objects. The function we used is a topographic elevation data computed on the surface of the Earth (a 2–sphere) from a satellite. The geometries considered include a sphere and the surface of a 3D model of a cat.

2. Function Representation with Wavelets

2.1. Wavelets on manifolds

The transform compression of a function involves three steps: a) transformation of the function, b) quantization, and c) entropy encoding. During the first step the function is subjected to a reversible linear transformation in order to concentrate most of its entropy (i.e., information) into a low dimensional subspace, thus simplifying its description. A wide variety of transformation techniques are currently in use, of which we will use wavelets.

Wavelets supply a basis for the functions we are representing. They decorrelate the data because in some way they resemble the data we want to represent. They are local in space and frequency. Typically they have compact support (localization in space), are smooth (decay towards high frequencies), and have vanishing moments (decay towards low frequencies). Finally, the wavelet representation of a data set can be found quickly. The fast decorrelation power of wavelets is the key to compression applications.

We can not use easily traditional wavelets for the transformation of data defined on complex manifolds. They are built using translation and dilation of a "mother wavelet" — a process that only makes sense in a Euclidean space. Instead, we will use the "second generation wavelets" introduced in [8] for building wavelets on a sphere. The basic philosophy behind second generation wavelets is to build wavelets with all desirable properties (basis, localization, fast transform) adapted to much more general settings than the real line, e.g., wavelets on manifolds. Such adaptations will depend on and vary with the local and global topological and geometric properties of the manifold.The main difference with the classical wavelets is that the filter coefficients of second generation wavelets are not the same throughout, but can change locally to reflect the changing (non–translation invariant) nature of the manifold.

2.2. The lifting scheme

In order to construct second generation wavelets we need to use a technique different than the Fourier transform which uses translation and dilation as algebraic operations. Sweldens [8] introduced one such technique called "the lifting scheme." A canonical case of lifting consists of three stages, which are referred to as: split, predict, and update. Given an abstract data set λ_0, the first stage of the method splits the data into two smaller subsets λ_{-1} and γ_{-1}. γ_{-1} is referred to as the wavelet subset. The simplest way to do the split is by dividing the "odd" and the "even" indexed points of the data set into two disjoint sets. In a second stage, the subset λ_{-1} is used to predict the γ_{-1} subset based on the correlation present in the original data. γ_{-1} is then replaced with the difference between itself and its predicted value. The wavelet subset encodes how much the data deviates from the model on which the prediction operator P was built. Finally the data subset λ_{-1} is updated so that some global properties of the original data set is maintained in the smaller versions λ_{-j} (e.g. same overall brightness of an image).

At this moment the original data can be replaced with the smaller set λ_{-1} and the wavelet set γ_{-1}. This scheme can now be iterated. After n steps the original data have been replaced with the wavelet representation $(\lambda_{-n}, \gamma_{-n}, ..., \gamma_{-1})$. Given that the wavelet sets encode the difference with some predicted value based on a correlation model, this is likely to give a more compact representation. An important property of lifting is that once the forward transform is built, one can immediately derive the inverse.

2.3. High dimensional wavelets

Using the lifting scheme we can construct wavelets on, e.g., the surface of any 3D object. In this paper we will use a 3D model of a cat as an illustration. The base complex was obtained via 3D scanning of a plastic model with a Cyberware scanner at the Graphics Lab in the Computer Science Department of Stanford University. The resulting manifold is a 3D triangular mesh.

Figure 1 illustrates the base manifold for the cat and figure 2 is the result (geometrical) of the first subdivision step. We can use different subdivision methods. For 2D manifolds, midpoint subdivision (common in graphics) results in the most regular results and was used throughout this paper. Other subdivision schemes are possible.

Each vertex of the model has a data value (function value) attached to it. The "even" vertices used by the lifting scheme are simply the ones from the coarser (i.e., pre–subdivision) level while the "odds"

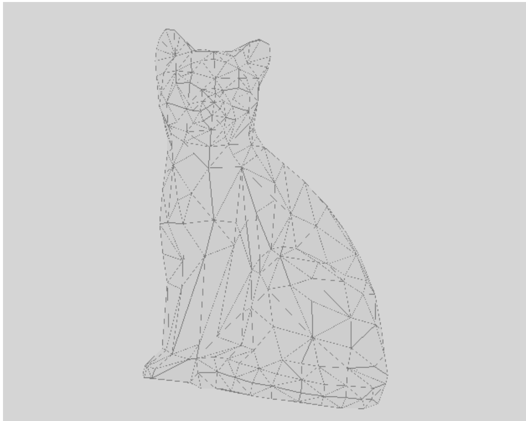

Figure 1: Cat — base model mesh

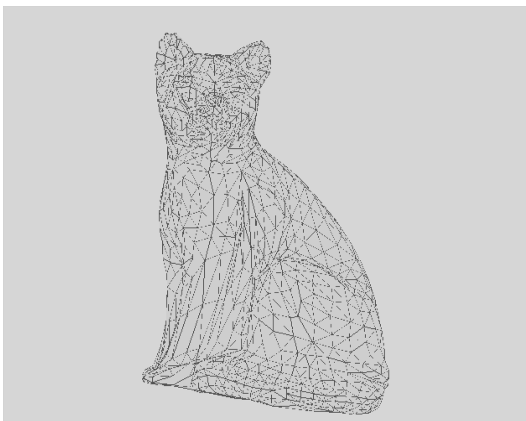

Figure 2: Cat — one subdivision level mesh

are the remaining ones.

There are a number of possibilities for the prediction operator. The simplest one is to take the mean of the two endpoint vertices for each "odd" midpoint generated. If we want however to account for, e.g. the sharp edges around the cat ears, we need to use a larger stencil. Here we use the "butterfly" scheme [2], where the odd values are predicted by a weighted average of 8 neighboring even values (depicted in Figure 3) as follows:

$$m = 1/2(v_1+v_2)+1/8(f_1+f_2)-1/16(e_1+e_2+e_3+e_4) \quad (1)$$

The actual wavelets and scaling functions can now be constructed by iterating the prediction operator. This process defines the scaling function on a dense subset of the surface. This function is then extended to the whole surface by continuity. The wavelets are simple linear combinations of the scaling functions. The update operator is a result of the condition of preservation of first order vanishing moments. The process is continued until appropriate coverage of the data set is achieved.

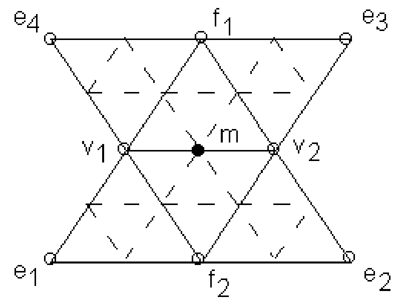

Figure 3: Odd vertex (m) surrounded by even neighbors $(v_1, v_2, f_1, f_2, e_1, e_2, e_3, e_4)$

3. Compression of the Wavelet Coefficients

3.1. Quantization of the coefficients

Once we generate the wavelet representation of the function data, we would like to take advantage of its decorellation properties and build a compressed description. As a first step, the coefficients of the transformed function are quantized, i.e., scaled and truncated appropriately. As it is shown in [1], the contrast sensitivity curve for the human visual system is best approximated by using the L_1 norm and its associated scaling. We have experimented with using different scaling models the result of which can be found in [4]. During decompression the wavelets are descaled by the inverse of the scaling factor used during compression.

Quantization is performed, after scaling, by thresholding. The minimum quantization is determined by the encoder and a maximum quantization by the decoder. The coefficients are coded out bit plane by bit plane for descending n. This results in a progressive, embedded approximation in which the more important bits appear earlier in the data stream.

3.2. Entropy encoding — construction of the G–tree

The remaining coefficients after the quantization step are encoded using a suitable entropy encoding technique. Not all possible values of the coefficients are equally likely nor are all sequences of coefficients equally likely. Entropy encoding exploits statistics about the data to use shorter codes for the more frequent situations and longer codes for the less frequent situations. In particular, the transformation and quantization steps produce many zero values so that a major task of an entropy compression technique is to code the geometric location of these zeros. Most of the bits produced will be leading zeros, zeros which precede the most significant bit of the coefficient. Coefficients whose bit is significant (not a leading zero) are so much in the minority that the problem of designating which coefficients are significant far outweighs the problem of the value of those coefficients. The locations of the significant coefficients are heavily correlated by location and scale and are compactly encoded by a quadtree–like technique known as zerotrees [6].

Zerotrees are based on the observation that insignificant coefficients are typically clustered spatially and strongly correlated from between subdivision levels at the same spatial location. This tree structure has the property that one coefficient is below another if the former represents a subdivision refinement of the latter. The zerotree hypothesis suggests that if a coefficient is insignificant then all coefficients below it are also insignificant. If the tree is constructed in a reasonable way the zerotree hypothesis will be true with very high probability. Thus, trees for which the hypothesis is true can be coded with very short codewords while long codewords can be used in the rare cases where the hypothesis is false.

The zerotree algorithm was introduced for effective and fast embedded (progressive) compression of images. In our context that algorithm processes the wavelet coefficients generated from the transform analysis part based on significance with respect to given threshold. The cells (triangles, edges and vertices) of the tessellation of the 2–manifold must be arranged into a tree, which we call a G–tree. Possible constructions for G–trees are described in detail in [5]. The original data and the coefficients of its wavelet transform are attached to a subset of the nodes of the G–tree.

The coded file begins with a preamble that details the base complex, the subdivision method, the number of subdivision levels, and the scaling of the coefficients. The preamble information is sufficient to reconstruct the G–tree and the decoder initializes by reconstructing the tree from the preamble.

Using a modified zerotree scheme, the G–tree is processed threshold by descending threshold, encoding bits indicative of significant G–tree nodes and the corresponding coefficient bits. The decoding algorithm inputs bits according to the modified zerotree scheme into the G–tree structure, refining the wavelet coefficients. The canonical ordering of the bits is similarly generated algorithmically by both the encoder and the decoder. De–scaling and inverse second generation wavelets complete the synthesis of the original function.

4. Simulation Results and Applications

The simulations below were coded in C++ on a UNIX platform (SGI Indigo 2 Impact 10000). The user can interactively select the type of wavelets to be used, the base complex , the number of levels of subdivision, the function to be compressed, and the desired compression.

For the Cat example we compressed the topographic function that is the elevation (with respect to sea level) of the Earth. This function is initially approximated by the ETOPO10 data set which samples the Earth every 10 arc minutes (1.5 million points approximately 17 km. apart). The resulting data set is mapped on the surface of the cat by a radial projection. The center of the sphere of the projection is located at the mass center of the cat and the data value for each point on the surface of the cat is calculated by intersecting the ray from the center to the point with the sphere wrapped around. The topographic elevation at each vertex was then determined by interpolation of the ETOPO10 data set. The result is color coded based on that elevation.

When new points are generated via subdivision their function values are calculated with the same procedure. Their geometrical location is computed using the butterfly scheme over the spatial (x, y, z) coordinates of the coarser level vertices. The elevation data is wavelet transformed using butterfly lifting and compressed at various ratios.

The base complex of the cat has 366 vertices, 728 triangles and 1092 edges. After 5 levels of subdivision (the maximum allowed by the hardware) we generate 372,738 vertices (wavelet coefficients) that cover about 1/4 of the available data points in ETOPO10 (we also have 745,472 triangular faces and 1,118,208 edges).

For the Earth example we use the same function mapped on the surface of a sphere approximating the Earth. The base manifold is an icosahedron (12 vertices, 20 triangular faces and 30 edges) that is subdivided using midpoint subdivision. Geometrically the newly generated points are projected up on a sphere using geodetic projection. We use the butterfly scheme as a prediction operator. The icosahedron is subdivided 8 times which results in 655,362 vertices (covering about half of the ETOPO10 data points).

Figure 4 summarizes the results for the peak signal–to–noise ratio (PSNR) for the two examples above for several different compression ratios.Each row in the table corresponds to the number of bitplanes read during the decompression. Every coefficient is represented with 10 bits. In all cases the compression subroutine writes out, bitplane by bitplane, the significant bits associated with all 10 bits for all the coefficients. The first row in the table corresponds to the case when during decompression all 10 bitplanes of significant coefficients are read in. At the next row we read only 9 bitplanes (the most significant ones) for all coefficients, etc. Even when we

	CAT (5 levels, 372738 coeff.)			Earth (8 levels, 655362 coeff.)		
bitplanes	PSNR	b/vertex	Compress	PSNR	b/vertex	Compress
10	57.94	2.49	3.2:1	41.98	0.47	17:1
9	50.93	1.53	5.2:1	37.68	0.17	48:1
8	45.15	0.88	9:1	34.08	0.067	120:1
7	40.66	0.5	16:1	31.1	0.028	291:1
6	36.98	0.24	33:1	28.43	0.012	686:1
5	33.85	0.12	67:1	25.89	0.005	1623:1
4	31.23	0.07	114:1	23.66	0.0022	3659:1
3	28.82	0.045	178:1	21.81	0.0013	6402:1
2	26.87	0.036	222:1	20.12	0.0008	10006:1
1	25.37	0.032	250:1	18.76	0.0007	11599:1

Figure 4: PSNR data for different compression ratios.

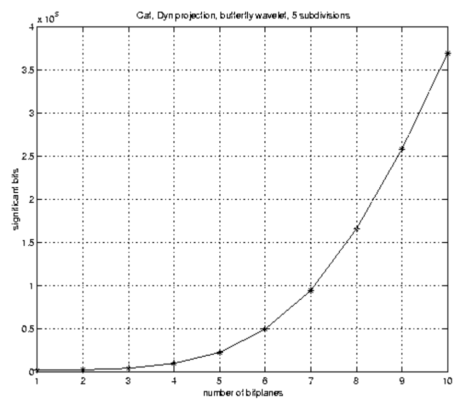

Figure 5: Significance Bits Allocation.

read in all the bitplanes encoded, there is still a 3:1 compression due to the zerotree structure.

The number of significant bits increase exponentially with the number of bitplanes retained in the representation of the coefficients. For the Cat example, the most significant bitplane (bitplane 10) has only 1871 bits that are significant for all 372,738 coefficients. In contrast, the least significant bitplane (bitplane 1) has 369,270 significant bits. The curve in Figure 5 illustrates the relationship between number of significant bits and number of bitplanes used for coefficient representation. That curve shows that if we select the number of bitplanes for decompression such that we guarantee adequate PSNR, we can achieve a significant amount of compression. In fact with bitplane reduction we can achieve close to 100:1 compression with the PSNR being in the virtually lossless range.

Visually Figures 6 and 7 illustrate the quality of the compression. The results look exactly as what we would expect for good compression given the resolution that we are achieving.

Applications of our method lie in the interactive multiresolution viewing, processing and editing; efficient storage and rapid transmission of complex

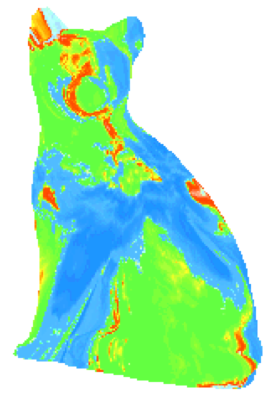

Figure 6: Cat manifold with ETOPO10 data mapped.

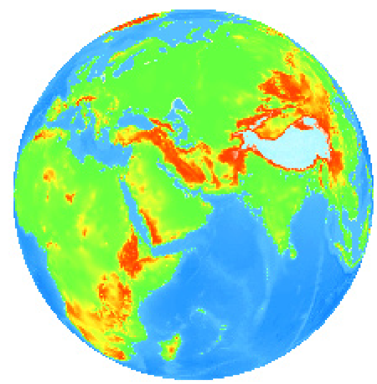

Figure 7: Earth manifold with ETOPO10 data mapped.

data sets. Typical data sets include earth topography, satellite images, and complex surface parametrizations.

References

[1] DeVore, R. A., Jawerth, B., and Lucier, B. J. Image compression through wavelet transform coding. IEEE Trans. Inform. Theory, vol. 38, pp. 719-746, March 1992.

[2] Dyn, N., Levin, D., and Gregory, J. A butterfly subdivision scheme for surface interpolation with tension control. Transactions on Graphics, 9(2):160–169, 1990.

[3] Kolarov, K., and Lynch, W. Compression of functions defined on surfaces of 3D objects. Data Compression Conference Proceedings, IEEE Computer Society Press, 1997.

[4] Kolarov, K., and Lynch, W. Optimization of the SW algorithm for high-dimensional compression. Compression and Complexity of SEQUENCES 1997 , IEEE Computer Society Press, 1997.

[5] Kolarov, K., and Lynch, W. Wavelet compression for 3D and higher-dimensional objects.

Proceedings of SPIE Conference on Applications of Digital Image Processing, Volume 3164, San Diego, July 1997.

[6] Said, A., and Pearlman, W. A new fast and efficient image codec based on set partitioning in hierarchical trees. IEEE Transactions on Circuits and Systems for Video Technology.

[7] Schröder, P., and Sweldens, W. Spherical wavelets: Efficiently representing functions on the sphere. *Computer Graphics Proceedings, (SIGGRAPH 95)* (1995), 161–172.

[8] Sweldens, W. The lifting scheme: A custom-design construction of biorthogonal wavelets. Tech. Rep. 1994:7, Industrial Mathematics Initiative, Department of Mathematics, University of South Carolina, 1994.