Coding of Ocean-Temperature Volumes Using Binary Set Splitting with $k$-d Trees

Justin T. Rucker and James E. Fowler
Department of Electrical and Computer Engineering
GeoResources Institute, Mississippi State ERC
Mississippi State, MS 39762

Abstract—An embedded wavelet-based coder for the shape-adaptive coding of ocean-temperature data is described. The proposed coder, 3D binary set splitting with $k$-d trees (3D-BISK), is based upon the popular bitplane-coding paradigm and is specifically designed for shape-adaptive coding. Other similar coding methods use octree-based set partitioning; however, 3D-BISK employs a simpler set decomposition based on $k$-d trees which makes it more flexible when considering shape-adaptive coding. The performance of 3D-BISK is compared to prominent shape-adaptive coders and superior performance is demonstrated for a variety of ocean-temperature datasets.

I. INTRODUCTION

While the compression of imagery with arbitrary shape has become an important issue in image and video coding, such as in the recent MPEG-4 standard, certain geoscience applications can also benefit from such shape-adaptive coding. For example, the US Naval Oceanographic Office (NAVO-CENO) generates three-dimensional oceanographic temperature datasets for rectangular regions of sea and land at standard ocean depths. Data that refers to land or points beyond the bathymetry are considered to have no valid data. Thus, the compression of such ocean-temperature data requires shape-adaptive coding.

Modern 3D embedded wavelet-based coders are designed upon three major precepts: a 3D wavelet transform, significance-map encoding, and successive-approximation quantization (i.e., bitplane coding). Key to this process is the mapping of the significance state of each coefficient (i.e., whether or not the coefficient is greater than or less than the current threshold) into a binary-valued significance map with the threshold decreasing for each successive pass through the dataset. Such coders can easily be made shape-adaptive and applied to the ocean-temperature compression problem by employing a 3D shape-adaptive wavelet transform which transforms only the ocean regions and considers land regions to be permanently insignificant. The major difference between wavelet-based compression schemes lies in the method for coding the significance map; consequently, the key to shape-adaptive coding is to modify this significance-map encoding to accommodate the presence of land regions wherein no valid data lies.

In this paper, we describe 3D binary set splitting with $k$-d trees (3D-BISK), which is a 3D extension of the 2D-BISK coder proposed in [1]. BISK is itself a variant of the well-known, state-of-the-art SPECK algorithm [2, 3]; its shape-adaptive version, object-based SPECK (OB-SPECK) [4]; and its 3D extension, 3D-SPECK [5]. Experimental evidence has shown that 3D-SPECK demonstrates performance roughly equivalent to that of the prominent JPEG-2000 standard in tasks such as the compression of hyperspectral image cubes. JPEG-2000, on the other hand, does not support shape-adaptive coding. The main contribution of this paper is the development of the 3D-BISK algorithm which replaces the octree set-partitioning operation of 3D-SPECK with $k$-d trees [6], a simpler set decomposition particularly well-suited to shape-adaptive coding due to its greater flexibility at capturing arbitrarily shaped regions. Additionally, 3D-BISK aggressively discards land regions from consideration by shrinking the decomposed sets to the bounding box of their ocean regions.

In the following sections, we describe the 3D-BISK algorithm in detail and compare it to 3D-OB-SPECK and the shape-adaptive version of a popular zerotree coder, 3D-SPIHT [7]. Empirical results are presented which show that 3D-BISK performs significantly better than both 3D-SPIHT and 3D-OB-SPECK in the coding of ocean-temperature volumes.

II. SHAPE ADAPTIVE CODING

In this section, we summarize prior approaches to 3D shape-adaptive coding. Essentially, 3D shape-adaptive coders are direct extensions to 3D of algorithms developed for 2D imagery with arbitrary shape. Regardless of the dimensionality of the data, the straightforward approach to shape-adaptive coding involves applying a transform to only the valid ocean data and treating the remaining land regions as permanently “insignificant.” The significance and refinement passes can then process these land regions in the same way as other insignificant coefficients. While most shape-adaptive coders are based on this idea, a number of approaches employ modifications to the significance-map encoding to increase performance.

A. Shape-Adaptive 3D-SPIHT

Significance-map encoding for 3D-SPIHT involves the coding of the insignificance of entire tree-structured sets across multiple scales of a wavelet transform. The shape-adaptive version of 3D-SPIHT follows the straightforward approach described above by aggregating large land regions and insignificant ocean regions into zerotree symbols. Further refinement to the algorithm can be made by discarding subtrees consisting...
entirely of land regions (which are permanently insignificant) from further consideration [8]. The result is typically a modest increase in performance related to the sizes of land regions in the dataset.

B. 3D-OB-SPECK

The recently developed SPECK algorithm eliminates the cross-scale aggregation of coefficients that occurs in SPIHT and other wavelet-based algorithms and instead applies partitioning to sets of contiguous coefficients within each individual subband. In 3D-OB-SPECK, the significance state of an entire set is tested and coded, then, if the set was known to contain at least one significant coefficient, the set is split into eight subsets (i.e., octree partitioning), and the process is repeated for each subset. 3D-OB-SPECK is similar to shape-adaptive 3D-SPIHT in that land is considered to be permanently insignificant, and, when a set contains only land coefficients, it is removed from further consideration.

III. 3D BINARY SET-SPLITTING WITH k-d TREES

In this section, we describe our 3D-BISK algorithm and the k-d tree set-partitioning structure upon which it is based. 3D-BISK is a variant of 3D-OB-SPECK that is well-suited to shape-adaptive coding. The octree-partitioning of 3D-OB-SPECK is replaced by binary set splitting of k-d trees which allows for a more flexible coding of arbitrarily shaped regions. An additional key difference between 3D-OB-SPECK and 3D-SPIHT is the aggressive shrinking of the sets to the bounding box of the ocean coefficients contained in the set, which is responsible for a large part of the performance gain.

A. Set Partitioning with k-d Trees

Octrees and k-d trees [6] are two well-known methods for partitioning sets of data. Octrees divide a set into eight equally sized subblocks, whereas in k-d trees, a set is divided along a single dimension into two arbitrarily sized subsets. We have investigated several strategies for determining the location and dimension of the set-splitting operation and have determined that splitting so as to maintain sets as close to cubes as possible generally yields the best performance for shape-adaptive compression. Consequently, we divide a set into approximately equal halves, and we choose the split to occur along the dimension in which the set is the longest. It is straightforward to see that k-d trees can achieve a partitioning of a set identical to that resulting from an octree decomposition, although usually more levels of decomposition are needed. However, we demonstrate below that the k-d trees decomposition is advantageous for shape-adaptive coding of the significance map.

B. The 3D-BISK Algorithm

Following a 3D wavelet transform, the 3D-BISK algorithm begins by placing the subbands of the transformed coefficients into a list of insignificant sets (LIS); subsequently, each subband is “shrunk” to the bounding volume of its ocean coefficients. As in 3D-OB-SPECK, each LIS is indexed, and a given set \( S \) resides in the LIS with index \( N(S) \), i.e., in \( \text{LIS}_{N(S)} \). The LIS index, \( N(S) \), is the total number of decompositions, or splits, that has produced the set. The algorithm continues in the usual bitplane-coding fashion with sorting and refinement passes. The algorithm is as follows:

```plaintext
procedure BISK(\( \mathcal{X} \))
    Initialization(\( \mathcal{X} \))
    \( n \leftarrow \) max bitplane
    while (true)
        SortingPass()
        RefinementPass()
        \( n \leftarrow n - 1 \)
    
    procedure Initialization(\( \mathcal{X} \))
        for each subband \( S \) in \( \mathcal{X} \)
            \( N(S) \leftarrow \) total number of decompositions in all dimensions
            ShrinkSet(S)
            append \( S \) to LIS\(_{N(S)}\)
            LSP \( \leftarrow \) \( \emptyset \)

    procedure SortingPass()
        \( l \) = number of LIS lists
        while \( l > 0 \)
            for each \( S \in \text{LIS}_{N(S)} \)
                ProcessSet(S)
            \( l \leftarrow l - 1 \)

    procedure RefinementPass()
        for each \( S \in \text{LSP} \)
            output \( n^{th} \) bitplane value of coefficient magnitude

    procedure ProcessSet(S)
        if \( S = \emptyset \)
            remove \( S \) from LIS\(_{N(S)}\)
        else
            output \( \Gamma_n(S) \)
            if \( \Gamma_n(S) = 1 \)
                remove \( S \) from LIS\(_{N(S)}\)
            if \( |S| = 1 \)
                output sign of \( S \)
                append \( S \) to LSP
            else
                CodeSet(S)

    procedure CodeSet(S)
        \( \{S_1, S_2\} = \text{PartitionSet}(S) \)
```

if $S_1 \neq \emptyset$
  append $S_1$ to LIS$_{N(S_1)}$  
  ProcessSet($S_1$)
  append $S_2$ to LIS$_{N(S_2)}$  
  ProcessSet($S_2$)

procedure PartitionSet($S$)
  if $z(S) \geq y(S) \geq x(S)$
    split $S$ depth-wise into $S_1$ and $S_2$:
    $S_1: \lfloor z(S)/2 \rfloor \times y(S) \times x(S)$
    $S_2: (z(S) - \lfloor z(S)/2 \rfloor) \times y(S) \times x(S)$
  else
    if $x(S) \geq y(S) > z(S)$
      split $S$ horizontally into $S_1$ and $S_2$:
      $S_1: z(S) \times y(S) \times \lfloor x(S)/2 \rfloor$
      $S_2: z(S) \times y(S) \times (x(S) - \lfloor x(S)/2 \rfloor)$
    else
      if $y(S) > x(S) > z(S)$
        split $S$ vertically into $S_1$ and $S_2$:
        $S_1: z(S) \times \lfloor y(S)/2 \rfloor \times x(S)$
        $S_2: z(S) \times (y(S) - \lfloor y(S)/2 \rfloor) \times x(S)$

$N(S_1) = N(S) + 1$
$N(S_2) = N(S) + 1$
ShrinkSet($S_1$)
ShrinkSet($S_2$)

Above, $\Gamma_n(S)$ is the significance state of set $S$, and $z(S)$, $y(S)$, and $x(S)$ are the number of ocean depths, rows, and columns, respectively, of set $S$.

The use of k-d trees in 3D-BISK is advantageous for the adaptive arithmetic coder used to code the significance information. Specifically, when set $S$ is split into $S_1$ and $S_2$, and $S_1$ is known to be insignificant (or empty), the significance state of $S_2$ is guaranteed to be significant. In this case, the significance state $\Gamma_n(S_2)$ is not coded into the bitstream. In the other case, the coding of $\Gamma_n(S_2)$ is conditioned with the knowledge that $S_1$ is significant.$^1$ By contrast, the first seven subsets of an octree decomposition must be known to be insignificant for 3D-OB-SPECK to benefit from the same strategy. The contexts used for set-significance coding within the BISK algorithm are as follows:

\[
c(S_1) \leftarrow \text{CONTEXT}_S1
\]
\[
\text{if } S_1 = \emptyset \text{ or } \Gamma_n(S_1) = 0
\]
\[
c(S_2) \leftarrow \text{CONTEXT}_\text{NOCODE}
\]
else
\[
c(S_2) \leftarrow \text{CONTEXT}_S2
\]

Above, $c(S_i)$ denotes the context that will be used to code $\Gamma_n(S_i)$.

IV. EXPERIMENTAL RESULTS

The performance of our 3D-BISK algorithm is measured using data from the study in [9, 10]. While only ocean-temperature data was considered in this work, the 3D-BISK implementation is general and can accommodate any type of 3D data that requires shape-adaptive coding. All algorithms are tested using a three-level shape-adaptive wavelet-packet transform using the popular 9/7 biorthogonal filter. Specifically, we implement the 3D wavelet transform in separable fashion, employing 1D transforms separately in the horizontal, vertical, and depth directions. For all algorithms, we use the wavelet-packet transform employed in [7], in which each depth plane is decomposed using a separable 2D transform and followed with a 1D decomposition in the depth direction. We use the QcPack [11] implementations of 3D-BISK, 3D-OB-SPECK, and 3D-SPIHT, each of which supports shape-adaptive coding as described in this paper.

The rate-distortion results shown in Figs. 1-4 indicate that 3D-BISK exhibits a substantial performance gain over the other two algorithms over a range of compression ratios. The results in Table I show that the 3D-BISK consistently outperforms both 3D-OB-SPECK and shape-adaptive 3D-SPIHT for a variety of ocean-temperature datasets.

V. CONCLUSIONS

In this paper, we described 3D-BISK, an embedded, wavelet-based, 3D shape-adaptive coder for the compression of ocean-temperature data. We compared 3D-BISK to 3D-SPIHT and 3D-OB-SPECK, two prominent 3D wavelet-based coding techniques, and experimental results demonstrated a substantial performance gain for a variety of ocean-temperature datasets. The performance gain is attributed to aggressive discarding of land-only sets and the simpler, more flexible partitioning of the sets that results from the binary k-d tree set-decomposition structure.

Interestingly, k-d trees often require more decompositions to represent a dataset than octrees. This suggests that binary set splitting allows for a simpler, more efficient entropy coding of the significance map. To confirm this hypothesis, we coded full datasets with no land so that ShrinkSet() had no effect on the coding process. In the case of coding these datasets, 3D-BISK and 3D-OB-SPECK exhibited virtually identical performance despite that fact that 3D-BISK incurred roughly seven times as many set-decomposition operations.

References

Fig. 1. Rate-distortion performance for guama. Rate is in bits per voxel (bpv) and distortion is measured as signal-to-noise ratio (SNR) in dB.

Fig. 2. Rate-distortion performance for ylsoj.

Fig. 3. Rate-distortion performance for okina.

Fig. 4. Rate-distortion performance for hawai.


<table>
<thead>
<tr>
<th>Dataset</th>
<th>3D-BISK</th>
<th>3D-OB-SPECK</th>
<th>3D-SPIHT</th>
</tr>
</thead>
<tbody>
<tr>
<td>adrtc</td>
<td>37.4</td>
<td>36.9</td>
<td>33.1</td>
</tr>
<tr>
<td>bisca</td>
<td>49.5</td>
<td>49.2</td>
<td>47.8</td>
</tr>
<tr>
<td>ginse</td>
<td>45.9</td>
<td>45.6</td>
<td>44.2</td>
</tr>
<tr>
<td>guama</td>
<td>65.2</td>
<td>64.0</td>
<td>61.3</td>
</tr>
<tr>
<td>hawai</td>
<td>63.6</td>
<td>63.2</td>
<td>61.1</td>
</tr>
<tr>
<td>med</td>
<td>45.0</td>
<td>44.4</td>
<td>42.0</td>
</tr>
<tr>
<td>nwlan</td>
<td>55.5</td>
<td>55.2</td>
<td>53.8</td>
</tr>
<tr>
<td>okina</td>
<td>46.0</td>
<td>45.3</td>
<td>41.7</td>
</tr>
<tr>
<td>socal</td>
<td>37.3</td>
<td>36.9</td>
<td>31.2</td>
</tr>
<tr>
<td>ylsoj</td>
<td>52.2</td>
<td>51.6</td>
<td>49.6</td>
</tr>
</tbody>
</table>