FPGA-Based Hyperspectral Data Compression Using Spectral Unmixing and the Pixel Purity Index Algorithm

David Valencia and Antonio Plaza

Department of Computer Science, University of Extremadura
Avda. de la Universidad s/n, E-10071 Caceres, Spain
{davaleco, aplaza}@unex.es

Abstract. Hyperspectral data compression is expected to play a crucial role in remote sensing applications. Most available approaches have largely overlooked the impact of mixed pixels and subpixel targets, which can be accurately modeled and uncovered by resorting to the wealth of spectral information provided by hyperspectral image data. In this paper, we develop an FPGA-based data compression technique based on the concept of spectral unmixing. It has been implemented on a Xilinx Virtex-II FPGA formed by several millions of gates, and with high computational power and compact size, which make this reconfigurable device very appealing for onboard, real-time data processing.

1 Introduction

Our focus in this work is to design a hyperspectral data compression technique able to reduce significantly the large volume of information contained in hyperspectral data while, at the same time, being able to retain information that is crucial to deal with mixed pixels and subpixel targets. A mixed pixel is a mixture of two or more different substances present in the same pixel[1]. A subpixel target is a mixed pixel with size smaller than the available pixel size (spatial resolution). So, it is embedded in a single pixel and its existence can only be verified by using the wealth of spectral information provided by hyperspectral sensors. In this case, spectral information can greatly help to effectively characterize the substances within the mixed pixel via spectral unmixing techniques. However, a major drawback of spectral-based data compression methods for hyperspectral imaging is their computational complexity[2]. The possibility of real-time, onboard data compression is a highly desirable feature to overcome the problem of transmitting a sheer volume of high-dimensional data to Earth control stations via downlink connections. In this work, we develop and FPGA-based compression algorithm based on spectral unmixing concepts. Section 2 develops the lossy compression algorithm. Section 3 maps the algorithm in hardware using systolic array design. Section 4 evaluates the algorithm in terms of both mixed-pixel and subpixel characterization accuracy, using a real image data set collected by the NASA Jet Propulsion Laboratory's Airborne Visible Infra-Red Imaging Spectrometer (AVIRIS). Performance data in a Xilinx Virtex-II FPGA are also given. Section 5 concludes with some remarks.

V.N. Alexandrov et al. (Eds.): ICCS 2006, Part I, LNCS 3991, pp. 888 – 891, 2006.

2 Hyperspectral Data Compression Algorithm

The first step of the algorithm consists of extracting endmembers from the input data. A well-known approach to accomplish this goal is the PPI algorithm[3], which proceeds by generating a large number of N-dimensional (N-D) random unit vectors called "skewers" through the dataset. Every data point is projected onto each skewer, and the data points that correspond to extrema in the direction of a skewer are identified and placed on a list. The number of times a given pixel is placed on the list is tallied, and the t pixels with the highest tallies are considered the final endmembers.

Using the set of t endmembers above, an inversion model is required to estimate the fractional abundances of each of the endmembers at the mixed pixels. Here, we use a commonly adopted technique as the second step of our compression algorithm, i.e., the linear spectral unmixing (LSU)[1] technique. Suppose that there are t endmembers $\{\mathbf{e}_i\}_{i=1}^{t}$ in a hyperspectral image scene, and let $\mathbf{f}$ be a mixed pixel vector. LSU assumes that the spectral signature of $\mathbf{f}$ can be represented by a linear mixture of $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_t$ with appropriate abundance fractions specified by an abundance vector $\mathbf{a} = \{a_1, a_2, \ldots, a_t\}$. Then, we can model the spectral signature of an image pixel $\mathbf{f}$ by a linear regression form $\mathbf{f} = \mathbf{e}_1 \cdot a_1 + \mathbf{e}_2 \cdot a_2 + \cdots + \mathbf{e}_t \cdot a_t$. Two constraints are usually imposed on this model to produce adequate solutions[1], i.e., abundance sum-to-one constraint: $\sum_{i=1}^{t} a_i = 1$, and abundance non-negativity constraint: $a_i \geq 0$ for $1 \leq i \leq t$.

Taking advantage of PPI and LSU, the idea of the proposed data compression algorithm is to represent a hyperspectral image cube by a set of fractional abundance images[4]. More precisely, for each N-D pixel vector $\mathbf{f}$, its associated LSU-derived abundance vector $\mathbf{a}$ of t dimensions is used as a fingerprint of $\mathbf{f}$ with regards to t endmembers obtained by the PPI algorithm. The implementation of the PPI/LSU algorithm can be summarized as follows:

1. Use the PPI to generate a set of t endmembers $\{\mathbf{e}_i\}_{i=1}^{t}$.
2. Use the LSU algorithm to estimate the corresponding endmember abundance fractions $\mathbf{a} = \{a_1, a_2, \ldots, a_t\}$ and approximate each pixel vector as $\mathbf{f} = \mathbf{e}_1 \cdot a_1 + \mathbf{e}_2 \cdot a_2 + \cdots + \mathbf{e}_t \cdot a_t$. Note that this is a reconstruction of $\mathbf{f}$.
3. Construct t fractional abundance images, one for each PPI-derived endmember.
4. Apply lossless predictive coding to reduce spatial redundancy within each of the t fractional abundance images, using Huffman coding to encode predictive errors.

3 FPGA-Based Hardware Implementation

Fig. 1 shows our proposed systolic array design for FPGA implementation of the PPI algorithm. The nodes labeled as "dot" in Fig. 1 perform the individual products for the skewer projections, while the nodes labeled as "max" and "min" respectively compute the maxima and minima projections after the dot product calculations have been completed (asterisks in Fig. 1 represent delays). In Fig. 1, $s_j^{(i)}$ denotes the j-th value of the i-th skewer vector, with $i \in \{1,\ldots,k\}$, and $j \in \{1,\ldots,b\}$, where b is the

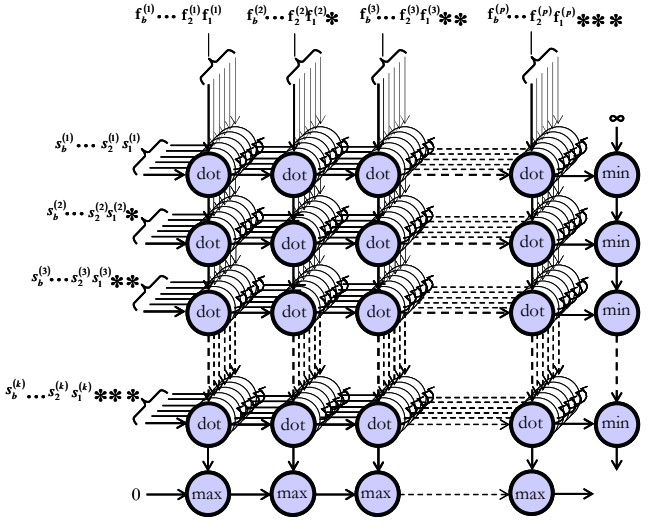

Fig. 1. Systolic array design for the proposed FPGA implementation of the PPI algorithm

number of bands. Similarly, $f_j^{(m)}$ denotes the reflectance value of the j-th band of the m-th pixel, with $m \in \{1,..., p\}$ and p is the total number of pixels in the scene.

We are still working towards the implementation of the LSU algorithm in hardware, so timing results in this work will be related with the optimization introduced by PPI-based FPGA design. The synthesis was performed using Handel-C, a hardware design and prototyping language. The implementation was compiled by using the DK3.1 software package. We also used other tools such as Xilinx ISE 6.1i to carry out automatic place and route (PAR), and to adapt the implementation to the Virtex-II FPGA used in experiments.

4 Experimental Results

The algorithm was applied to a real hyperspectral scene collected by an AVIRIS flight over the Cuprite mining district in Nevada, which consists of 614x512 pixels and 224 bands (available online from http://aviris.jpl.nasa.gov). Table 1 reports the spectral angle-based similarity scores[1] among U.S. Geological Survey reference signatures (see http://speclab.cr.usgs.gov) and the PPI-extracted endmembers from the resulting images after data compression (the lowest the scores, the highest the similarity). As the compression ratio was increased, the quality of extracted endmembers was decreased. We also included results by two standard methods: the wavelet-based JPEG2000 multicomponent[5] and the set partitioning in hierarchical trees (SPIHT)[6]. Results in Table 1 show that such 3-D techniques, which enjoy success in classical image compression, did not find equal success in hyperspectral image compression.

Table 1. Spectral similarity scores among ground-truth spectra and the endmembers extracted and from several reconstructed versions of the original image after compression

Mineral	Original image	PPI/LSU			JPEG2000			SPIHT		
		20:1	40:1	80:1	20:1	40:1	80:1	20:1	40:1	80:1
Alunite	0.063	0.069	0.078	0.085	0.112	0.123	0.133	0.106	0.119	0.129
Buddingt.	0.042	0.053	0.061	0.068	0.105	0.131	0.142	0.102	0.125	0.127
Calcite	0.055	0.057	0.063	0.074	0.102	0.128	0.139	0.097	0.122	0.134
Kaolinite	0.054	0.059	0.062	0.071	0.114	0.140	0.151	0.110	0.134	0.146
Muscovite	0.067	0.074	0.082	0.089	0.123	0.145	0.167	0.116	0.139	0.152

Finally, Table 2 shows the resource utilization by our systolic array-based implementation of PPI/LSU on the Xilinx Virtex-II XC2V6000-6 FPGA, which compressed the full AVIRIS scene in only 39 seconds approaching (near) real time performance.

Table 2. Summary of resource utilization for the FPGA-based implementation

Number of gates	Number of slices	Percentage of total	Maximum operation frequency (MHz)
526944	12418	36%	18.032

5 Conclusions

On-board compression of hyperspectral imagery has been a long-awaited goal by the remote sensing community. This paper investigated the importance of mixed pixels in hyperspectral data compression and further proposed an innovative, application-oriented data compression technique which is based on the pixel purity index (PPI) algorithm and linear spectral unmixing (LSU). Experimental results demonstrate that the algorithm provides very high compression ratios with no apparent spectral signature degradation. A systolic array-based FPGA implementation of the algorithm on a Xilinx Virtex-II XC2V6000-6 FPGA was also provided.

References

1. Chang, C.-I: Hyperspectral Imaging: Detection and Classification. Kluwer, New York (2003)
2. Motta, G., Storer, J. (eds.): Hyperspectral Data Compression. Springer-Verlag, Berlin Heidelberg New York (2005)
3. Chang, C.-I, Plaza, A.: A Fast Iterative Implementation of the Pixel Purity Index Algorithm. IEEE Geoscience and Remote Sensing Letters, 3 (2006) 63–67
4. Du, J., Chang, C.-I: Linear Mixture Analysis-Based Compression for Hyperspectral Image Analysis. IEEE Trans. Geoscience and Remote Sensing, 42 (2004) 875–891
5. Taubman, D.S., Marcellin, M.W.: JPEG2000: Image Compression Fundamentals, Standard and Practice. Kluwer, Boston (2002)
6. Said, A., Pearlman, W.A.: A New, Fast, and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees. IEEE Trans. Circuits and Systems, 6 (1996) 243–350