Presented at the NASA Jet Propulsion Laboratory AVIRIS Airborne Earth Science Workshop
Pasadena, California, March 31-April 2, 2004.

PARALLEL IMPLEMENTATION OF ALGORITHMS FOR ENDMEMBER EXTRACTION FROM AVIRIS HYPERSPECTRAL IMAGERY

Antonio Plaza[1], David Valencia, Pablo Martínez and Javier Plaza

1. Introduction

Hyperspectral imaging systems, used in conjunction with appropriate detection and recognition algorithms, have demonstrated to be very useful tools in many different remote sensing applications [1]. These instruments are capable of collecting hundreds of images, corresponding to different wavelength channels, for the same area on the surface of the Earth. A chief hyperspectral sensor is the NASA's Jet Propulsion Laboratory Airborne Visible-Infrared Imaging Spectrometer (AVIRIS) system, which currently covers the wavelength region from 0.4 to 2.5 mm using 224 spectral channels, at a nominal spectral resolution of 10 nm [2]. On other hand, the Hyperion hyperspectral imager aboard Earth Observing-1 (EO-1) spacecraft has been NASA's first hyperspectral imager to become operational on-orbit. It routinely collects images hundreds of kilometers long with 220 spectral bands from 0.4 to 2.5 mm. With such spectral detail, the ability to detect and identify target materials is greatly enhanced with respect to other techniques available, such as multispectral imaging, which typically collects only tens of images. In the near future, the proliferation of hyperspectral sensors installed aboard satellite platforms will produce a nearly continual stream of multidimensional data, and this expected high data volume would demand fast and efficient means for storage, transmission and analysis.

A diverse array of analysis techniques has been applied to hyperspectral image analysis during the last decade [1]. They are inherently either full pixel techniques or mixed pixel techniques, where each pixel vector in a hyperspectral image records the spectral information. The underlying assumption governing full pixel techniques is that each pixel vector measures the response of one predominantly underlying material at each site in a scene. However, a number of the pixel vectors in the scene will likely measure the spectral response of a mixture of underlying materials. Mixed pixel techniques have overcome some of the weaknesses of full pixel approaches by using linear and/or nonlinear mixture modeling and signal processing techniques. Spectral mixture analysis usually involves two steps: to find spectrally unique signatures of pure ground components (usually referred to as endmembers) and to express individual pixels in terms of linear/nonlinear combinations of endmembers.

Most available techniques for endmember selection and spectral unmixing focus on analyzing the data without incorporating information on the spatially adjacent data; i.e. the data is treated not as an image but as an unordered listing of spectral measurements where the spatial coordinates can be shuffled arbitrarily without affecting analysis [3]. However, one of the distinguishing properties of hyperspectral data, as collected by available spectrometers, is the multivariate information coupled with a two-dimensional (2-D) pictorial representation amenable to image interpretation. Subsequently, there is a need to incorporate the image representation of the data in the analysis5. By taking into account the complementary nature of spatial and spectral information in simultaneous fashion, it is possible to alleviate the problems related to each of them taken separately. While integrated spatial/spectral developments in hyperspectral technology hold great promise for improving endmember extraction, they create new processing challenges [4]. In particular, the price paid for the wealth spatial and spectral information available from hyperspectral sensors is the enormous amounts of data that they generate. As a result, analysis techniques that make combined use of spatial and spectral information are often computationally tedious, and require lengthy durations to calculate desired quantities. However, several applications exist where having the desired information calculated in real-time or near real-time is

[1]Neural Networks & Signal Processing Group (GRNPS), Computer Science Dept. University of Extremadura, Spain
Contact e-mail: jplaza@unex.es, Phone: +34 927 257254, Fax: +34 927 257203

desired, e.g. military applications or those pursuing detection and tracking of environmental disasters such as forest fires, oil spills, and other types of chemical contamination.

Parallel processing has been reported as an efficient technique to tackle large remotely sensed data sets, and to get reasonable response times in complex analysis scenarios [5-7]. In parallel processing, high performance can be achieved by two complementary means: Increased efficiency of the algorithms used in the codes and increased performance of the computers on which the codes are run. Parallel computer facilities offer the possibility of performance in hundreds of gigaflops, and memory capacity sufficient for advanced standoff detection in large, high-dimensional image data. Parallel architectures have been used in the past to improve computational performance of remotely sensed image analysis techniques. Thanks to the geographic local organization of the pixels in an image as a 2-D mesh, and to the regularity of most low-level computations, mesh-based parallel architectures are quite popular for image analysis applications [8]. However, the situation is more complex when dealing with hyperspectral image data, where spatial and spectral information can be equally employed to conduct the analysis. In order to take advantage of parallel computers, designing and implementing well-optimized hyperspectral analysis approaches can significantly improve their computational performance, and reduce the total research time to complete these studies.

In this paper, we discuss parallel implementations of a novel algorithm for endmember extraction in hyperspectral imagery, which makes combined use of spatial and spectral information. The paper is organized as follows. In section 2, we describe the fundamentals of the proposed method, which is based on mathematical morphology concepts. Section 3 provides an overview of its parallel implementation. The performance of the proposed parallel approach is demonstrated in section 4 using real hyperspectral data sets, collected by the AVIRIS imaging spectrometer. Specific factors affecting the performance of algorithm implementations in parallel computers are examined in this section, including impact of interprocessor communication and coordination, number of processors, and speedup ratios. Finally, section 5 concludes with some remarks and future research lines.

2. Morphological endmember extraction

In this section, we give a description of the proposed algorithm for endmember extraction in hyperspectral imagery. In order to render this algorithm computationally it will need to be embedded in a parallel implementation, which is given in the following section. Before describing the algorithm, we emphasize that it considers hyperspectral image analysis from a broader perspective than the standard methods currently available. Instead of focusing exclusively on the spectral information contained in the data, it focuses on analyzing both spatial and spectral responses in a simultaneous manner. This is achieved by extending spatial-based mathematical morphology [9-10] operations to the spectral domain. Section 2.1 provides an overview of the proposed framework to extend morphological operations to multidimensional images. Section 2.2 describes the general algorithm, along with its parameters and requirements.

2.1. Extension of morphological operations to hyperspectral images

Mathematical morphology is a classic nonlinear image processing technique that was originally defined for binary images [9]. Based on set theory, binary morphology was established by introducing fundamental operators applied to two sets. One set is processed by another set of carefully selected shape and size, known as structuring element (SE), which is translated over the image. The SE acts as a probe for extracting or suppressing specific structures of the image objects by checking, for each position of the SE, whether it fits or not within the image objects. In the erosion (dilation) operation, the minimum (maximum) value of the image pixels within the SE is selected as the resulting value of the morphological operation, and used to create a new image which is known as eroded (dilated) image. Morphological operations have been extended to gray-tone (mono-channel) images by viewing these data as an imaginary topographic relief; the brighter the gray tone, the higher the corresponding elevation [10]. The extension of the concepts of grayscale morphology to hyperspectral images is not straightforward. When such techniques are applied independently to each image channel (marginal morphology), there is a possibility for loss or corruption of information of the image due to the likely fact that new spectral constituents, i.e. not available in the original image, may be created as a result of processing image channels separately. An alternative way to approach this problem is to treat the data at each pixel as a vector. In order to define the basic morphological operations in N-D space, a concept for a maximum (or minimum) is necessary, and thus it is important to define an appropriate ordering of vectors in the selected vector space. Our innovative approach to the above problem has been the definition of a vector ordering scheme based on the spectral singularity of hyperspectral pixel vectors. First, a lattice structure is imposed onto N-D space by the definition of a cumulative metric based on the spectral angle distance (SAD). Second, basic morphological operations such as erosion and dilation are defined by extension.

2.2. Proposed algorithm

The automated morphological endmember extraction (AMEE) algorithm [11] is the only available endmember extraction algorithm that makes simultaneous use of spatial and spectral information via multi-channel morphological processing. The input to the AMEE method is the full image data cube, with no previous dimensionality reduction. Let $\boldsymbol{h}$ denote the input hyperspectral data cube and $\boldsymbol{h}(\mathrm{x},\mathrm{y})$ denote the pixel vector at spatial location (x,y). Similarly, let K be a kernel defined in the spatial domain of the image (the x - y plane). This kernel, called SE in mathematical morphology terminology, is translated over the image. The SE acts as a probe for extracting or suppressing specific structures of the image objects, according to the size and shape of the SE. Having the above definitions in mind, the AMEE method is based on the application of multi-channel erosion and dilation operations to the data. The above operations are respectively defined as follows [12-13].

$$(\boldsymbol{h}\otimes \mathrm{K})(\mathrm{x},\mathrm{y}) = \mathrm{arg\_\ Min}_{(\mathrm{s},\mathrm{t})\in \mathrm{K}} \left\{ \sum_{\mathrm{s}} \sum_{\mathrm{t}} \mathrm{dist}(\boldsymbol{h}(\mathrm{x},\mathrm{y}), \boldsymbol{h}(\mathrm{x}+\mathrm{s},\mathrm{y}+\mathrm{t})) \right\}, \tag{1}$$

$$(\boldsymbol{h}\oplus \mathrm{K})(\mathrm{x},\mathrm{y}) = \mathrm{arg\_\ Max}_{(\mathrm{s},\mathrm{t})\in \mathrm{K}} \left\{ \sum_{\mathrm{s}} \sum_{\mathrm{t}} \mathrm{dist}(\boldsymbol{h}(\mathrm{x},\mathrm{y}), \boldsymbol{h}(\mathrm{x}-\mathrm{s},\mathrm{y}-\mathrm{t})) \right\}, \tag{2}$$

where dist is the spectral angle mapper (SAM). Multi-channel erosion (respectively, dilation) selects the pixel vector which minimizes (respectively, maximizes) a cumulative distance-based cost function, based on the sum of the SAM distance scores between each pixel in the spatial neighborhood defined by K and all the other pixels in the neighborhood. As a result, multi-channel erosion extracts the pixel vector that is more similar to its neighbors as opposed to multi-channel dilation, which extracts the most spectrally distinct pixel in the neighborhood (endmember candidate). It should be noted that, according to the definition of morphological erosion and dilation, the above operations are sensitive to the size and shape of the SE used in the computation. In our application, a morphological eccentricity index (MEI) is defined for each endmember candidate by calculating the SAM distance between the pixel provided by the dilation operation and the pixel provided by the erosion. This operation is repeated for all the pixels in the scene, using SE's with a range of different sizes, until a final MEI image is generated. For illustrative purposes, we provide next a step-by-step algorithmic description of AMEE that will be used as a reference in the development of a parallel implementation of the algorithm.

AMEE algorithm

Inputs: N-D image $\boldsymbol{h}$, Structuring element B, Number of iterations $\mathrm{I_{MAX}}$, Number of endmembers p.

Outputs: Set of endmembers $\{\boldsymbol{e}_j\}_{j=1}^{p}$; Set of fractional abundances $\{\alpha_i(x,y)\}_{i=1}^{p}$ for each pixel $\boldsymbol{h}(x,y)$.

1. Set $i=1$ and initialize a morphological eccentricity index score $\mathrm{MEI}(x,y)=0$ for each pixel $\boldsymbol{h}(x,y)$.
2. Move B through all the pixels of $\boldsymbol{h}$, defining a local spatial search area around each $\boldsymbol{h}(x,y)$ and calculate the maximum pixel $(\boldsymbol{h}\oplus B)(x,y)$ and the minimum pixel $(\boldsymbol{h}\Theta B)(x,y)$ at each B-neighborhood. Update the resulting MEI score at each pixel selected as a local maximum, $\boldsymbol{h}(x',y')=(\boldsymbol{h}\oplus B)(x,y)$, using the following expression:

$$\mathrm{MEI}(x',y')=\mathrm{MEI}(x',y')+\mathrm{SAM}[(\boldsymbol{h}\oplus B)(x,y),(\boldsymbol{h}\Theta B)(x,y)] \tag{3}$$

3. Set $i=i+1$. If $i=\mathrm{I_{max}}$ then go to step 4. Otherwise, set $\boldsymbol{h}=(\boldsymbol{h}\oplus B)$ and go to step 2.
4. Select the set of p pixels $\{\boldsymbol{e}_j\}_{j=1}^{p}$ in $\boldsymbol{h}$ with higher score in the resulting MEI image. These pixels form the final endmember set.
5. Estimate the fractional abundance $\alpha_i(x,y)$ of each $\{\boldsymbol{e}_i\}_{i=1}^{p}$ at $\boldsymbol{h}(x,y)$ by using the linear mixture model $\boldsymbol{h}(x,y)=\sum_{i=1}^{p}\alpha_i(x,y)\cdot\boldsymbol{e}_i$, subject to $\sum_{i=1}^{p}\alpha_i(x,y)=1$ (sum-to-one) and $\alpha_i(x,y)\geq 0$ (non-negativity) constraints. The fully constrained least-squares (FCLSU) algorithm [14] is used to perform the abundance estimation, which results in a set of fractional abundances $\{\alpha_i(x,y)\}_{i=1}^{p}$ for each $\boldsymbol{h}(x,y)$.

It should be noted that the pixel at spatial coordinates (x',y') in equation (3) might be the maximum pixel at the neighborhood of two or more pixels. When this happens, the value of is uniquely defined as the sum of the SAM scores between $\boldsymbol{h}(x',y')$ and the minimum pixel in the neighborhood of each of the pixels above. As a result, if $\boldsymbol{h}(x',y')$ is selected several times as a maximum pixel, then its associated MEI score will be increased according to the number of times it was selected.

3. Parallel implementation

In this section, we provide an overview of a parallel version that improves the computational efficiency of the code described in section 2, so that the time in conducting scientific studies involving hyperspectral data analysis can be reduced. The proposed parallel implementation is based on domain decomposition techniques and the Message Passing Interface (MPI) library. Our major goal was designing an efficient parallel version of the algorithm running on multiple processors executing with significant speedup. To achieve this objective, we focused on the data structures of the code to discover all possible data dependencies. In order to achieve load balance and to exploit parallelism as much as possible, a general N-dimensional domain decomposition-based parallel partitioner (NDPP) was developed. The spatial domain of the hyperspectral data was chosen as the basis for the decomposition, so that the message passing was minimized and most of the code was executed in parallel on multiple processors. There are several reasons that justify the above decision. First, the application of spatial-domain partitioning is a natural approach for low level image processing, as many operations require the same function to be applied to a small set of elements around each data element present in the image data structure. A second reason has to do with the cost of inter-processor communication. In spectral-domain parallelism, the SE-based calculations made for each hyperspectral pixel need to originate from several PEs, and thus require intensive inter-processor communication. The overhead introduced by inter-processor communication will increase linearly with the increase in the number of PE's, and this will easily involve load unbalance problems. As a result, a final major reason for the use of a spatial domain-based parallel model is that load balancing (i.e., evenly distributing all work among the available processing units) is generally much more difficult when a spectral-domain parallel model is adopted. It is particularly so in the case of a system that involves highly varying inter-processor communication operations, where it is difficult to ensure that each processor has exactly the same amount of work to do. Hereinafter, each partition produced by our NDPP will be referred to as parallelizable spatial/spectral pattern (PSSP). Next, we show how such patterns are obtained, and their further use in the implementation of the parallel algorithm.

3.1. Parallelizable spatial/spectral patterns

The concept of parallelizable spatial/spectral pattern (PSSP) is defined as the maximum amount of work that –when applied to partial image data– can be performed without communication [8]. As stated above, the so-called NDPP module divides the hyperspectral image into a collection of PSSPs and maps each pattern to a node, so that all internal data accesses refer to data local to the node that executes the operation. Then, each node performs automated analysis on its associated data set. Analysis of each pattern involves of a series of independent tasks, where a task specifies what hyperspectral pixels in the N-D input image must be read in order to update (write) the MEI value of a single data pixel in a 2-D destination image called MEI image. Each task is tied to a range of spatial locations, which constitute a subset of the positions inside the spatial domain of a partition of the source N-D scene. These positions are given by the shape and size of the structuring element. For illustrative purposes, Fig. 1 shows an example where MEI scores are calculated for two hyperspectral image pixels (P1 and P2), using a square-shaped 3x3 SE and two processing nodes. As shown in Fig. 1, each pixel value in the output 2-D image depends on the hyperspectral pixels in the neighborhood of the pixel at the same position in the input image, which is given by the related kernel structure. In the end, the master node gathers back the results that are produced from each node.

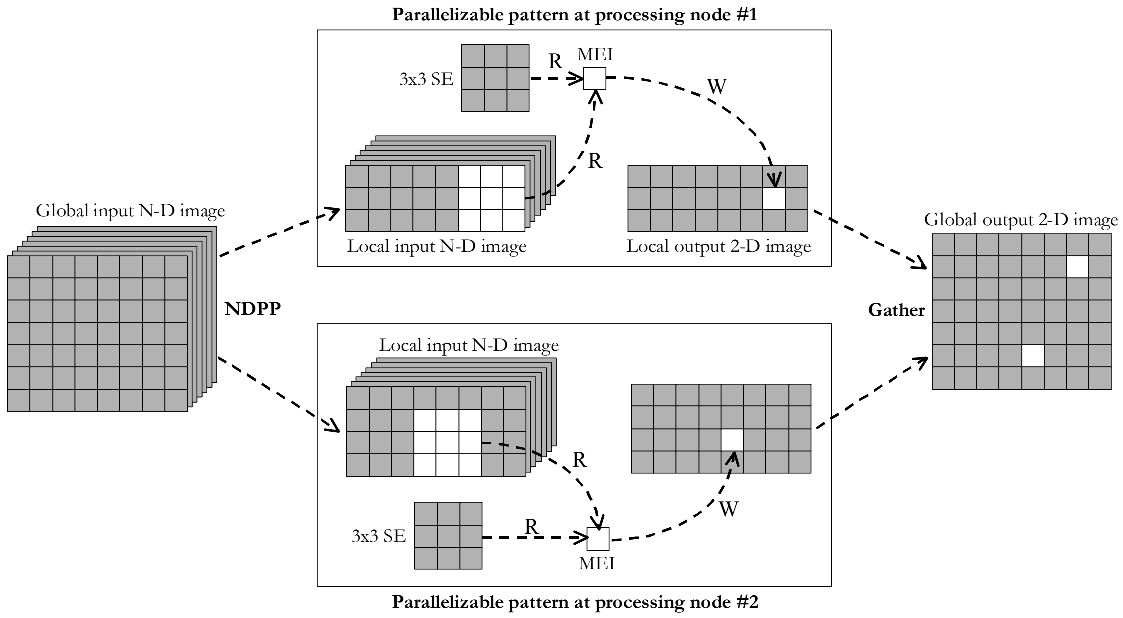


Figure 1. Partitioning of hyperspectral image using the NDPP, and assignment of each parallelizable pattern to processing units for local MEI index computation.

An important issue in kernel-based image processing operations is that accesses to pixels outside the input image's domain are possible. In sequential implementations of such kernel-based operations, it is common practice to redirect such accesses according to a predefined border handling strategy. A better approach for parallel implementation, however, is to separate the border handling from the actual kernel-based operation. This makes implementations more robust and, in general, also faster. Two specific border-handling strategies are implemented in our case as follows:

1. When accesses to pixels outside the image domain are necessary, only those pixels inside the image are considered for the MEI calculation.
2. Additional communication is required between processing nodes when the kernel computation is split amongst several processing nodes [see Fig. 2(a)]. In such cases, we allow an overlapping scatter of the global input image to avoid additional communication [see Fig. 2(b)]. In our implementation, the NDPP module automatically determines the size in pixels of the overlapping section.

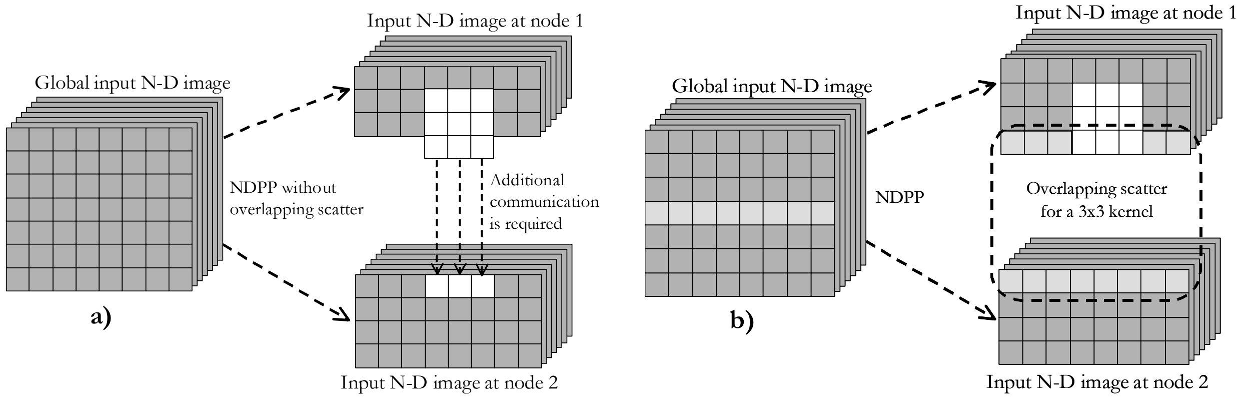

Global input N-D image
Input N-D image at node 1
NDPP without overlapping scatter
Additional communication is required
Input N-D image at node 2
a)
Global input N-D image
Input N-D image at node 1
NDPP
Overlapping scatter for a 3x3 kernel
Input N-D image at node 2
b)

Figure 2. (a) SE-based computation split among several processors. (b) Overlapping scatter avoids communication.

The generalized description of parallelizable patterns given above, along with border-handling strategies, is important as it states the requirements for parallel implementation of a large set of processing operations. In addition, for each spatial/spectral pattern implemented on the basis of the generalized description, a parallelization strategy directly follows. As such, code reusability is maximized, and flexibility is enhanced. At the same time, communication overhead is minimized, while – for the given parallelization granularity – the available parallelism is fully exploited.

3.2. Proposed parallel algorithm

The parallel implementation of the AMEE algorithm (AMEEPAR) entirely relies on the spatial-domain decomposition scheme implemented by a master node, which also acts as a root node in charge of all I/O operations [15]. The partitioner has been implemented so that it automatically determines the optimum size for the PSSPs to be distributed between the PEs. This is done by taking into account both the spatial and spectral resolution of the input hyperspectral image $\boldsymbol{h}$. By distributing data evenly among the processors, load balance is achieved. Also, the utilization of the concept of PSSP allows us to greatly minimize inter-processor communication overhead. The proposed parallel algorithm has been implemented in the `C++` programming language using calls to message passing interface (`MPI`), an application programmer interface which provides a substantial set of libraries for writing, debugging and performance testing of distributed programs. We briefly describe next the sequence of operations executed by the master node in the proposed `MPI`-based parallel implementation.

AMEEPAR algorithm

Inputs: N-D image $\boldsymbol{h}$, Structuring element B, Number of iterations I_{MAX}, Number of endmembers p.

Outputs: Set of endmembers $\{\boldsymbol{e}_j\}_{j=1}^{p}$, Set of fractional abundances $\{\alpha_i(x,y)\}_{i=1}^{p}$ for each pixel $\boldsymbol{h}(x,y)$.

1. Initialize `MPI` using `MPI_Init()` and generate necessary system information, including the total number of available PEs (denoted by K), each processor's ID number, timing and other data using `MPI_Comm_Size()` and `MPI_Comm_Rank()`.
2. Obtain a set of K partitions $\{PSSP_j\}_{j=1}^{K}$ of the original image $\boldsymbol{h}$, so that $\boldsymbol{h} = \bigcup_{j=1}^{K} PSSP_j$. Apply all necessary border-handling strategies in the construction of the partition set $\{PSSP_j\}_{j=1}^{K}$.
3. Using the `MPI_Send()` function, send each $PSSP_j$ to a different PE along with parameters B and I_{MAX}. With the above information, steps 1-3 of the sequential AMEE algorithm will be executed locally at each processor for the corresponding $PSSP_j$, giving an image MEI_j as a result.
4. Using the `MPI_Receive()` function, collect all the individual images $\{MEI_j\}_{j=1}^{K-1}$ produced by each PE, and merge them together to form a final image $MEI = \bigcup_{j=1}^{K} MEI_j$.
5. Execute step 4 of the sequential AMEE algorithm, thus obtaining a set of p endmembers $\{\boldsymbol{e}_j\}_{j=1}^{p}$.
6. Use the `MPI_BCast()` function to distribute the set $\{\boldsymbol{e}_j\}_{j=1}^{p}$ to all available PEs, where step 5 of the sequential AMEE algorithm will be locally executed for each $PSSP_j$ giving a set of fractional abundances $\{\alpha_i^{(j)}(x_j,y_j)\}_{i=1}^{p}$ as a result, where (x_j,y_j) refer to the spatial coordinates of pixels in the $PSSP_j$.
7. Using the `MPI_Receive()` function, collect all the individual sets of fractional abundances $\{\alpha_i^{(j)}(x_j,y_j)\}_{i=1}^{p}$ calculated for every $PSSP_j$, and form a final set of fractional abundances designated by $\{\alpha_i(x,y)\}_{i=1}^{p} = \bigcup_{j=1}^{K}\{\alpha_i^{(j)}(x_j,y_j)\}_{i=1}^{p}$ for each pixel $\boldsymbol{h}(x,y)$ of the original image.

As a final note, we emphasize that the AMEEPAR algorithm is an excellent application for parallel computation, mainly because there is no dependence between the calculations made at each $PSSP_j$ and only minimal communication is required for the entire calculation. The `MPI` code summarized above is portable to any type of distributed memory system (provided that the memory available to each PE is large enough to store the respective PSSP). Performance data for the parallel algorithm are given in section 4.

4. Experimental results

Various code performance tests were carried out for both sequential and parallel codes. The parallel algorithm described in section 3 has been implemented on the SGI Origin 2000 Silicon Graphics supercomputer at CEPBA (European Center for Parallelism of Barcelona) and the Thunderhead Beowulf cluster at NASA's Goddard Space Flight Center (see Fig. 3). The former is a distributed memory, message-passing parallel machine composed of 64 MIPS R10000 processors (each one with 4 MB of cache) and 12 Gb of main memory, interconnected via 1.2 Gbps communication network. The latter is a commodity cluster computer composed of 256 dual 2.4 Ghz Intel Xeon nodes, each with 1 Gb of memory and 80 Gb of main memory, interconnected via 2 Ghz optical fibre Myrinet.

(a) SGI Origin 2000 (b) Thunderhead

Figure 3. (a) SGI Origin 2000 multicomputer at European Center for Parallelism of Barcelona; (b) Thunderhead Beowulf cluster at NASA's Goddard Space Flight Center.

To empirically investigate the scaling properties of the algorithm described in this paper, it was applied to two standard AVIRIS hyperspectral data sets, described in Table 1 (see also Fig. 4), where AVCUP97 refers to a scene collected over Cuprite mining district in Nevada and AVJRBP97 refers to a scene collected over Jasper Ridge biological preserve in California. We have selected the above datasets because they are widely available online (from http://aviris.jpl.nasa.gov), and have served as benchmark data for many scientific studies, including performance evaluation of hyperspectral analysis algorithms. In addition, detailed ground-truth information is available for the scenes above, which can be used to validate performance of endmember extraction techniques.

Image	Location	Year	Pixels	Bands	Size	Spectral range	Pixel size	Applications
AVCUP97	Cuprite,NV	1997	614x512	224	137.5 Mb	0.4 – 2.5 μm	20 meters	Geology
AVJRBP97	Jasper Ridge,CA	1997	614x512	224	137.5 Mb	0.4 – 2.5 μm	20 meters	Vegetation

Table 1. Summary of hyperspectral images used in experimental results.

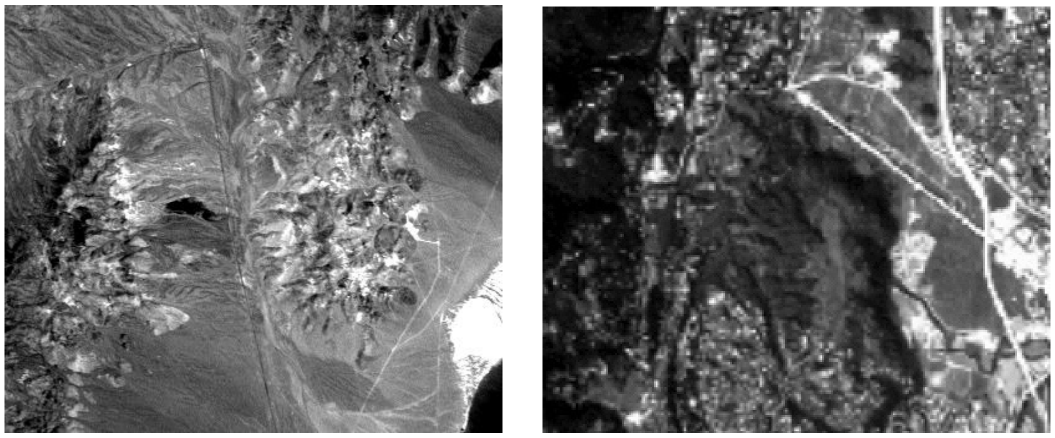

(a) (b)

Figure 4. Representative spectral bands from AVIRIS hyperspectral scenes: (a) AVCUP95. (b) AVJREF97.

To empirically investigate the scaling properties of the parallel algorithm, its performance was tested by timing the program over a variety of input hyperspectral scenes and number of processors. The measured speedups were simply computed by dividing the parallel times by the single processor times. If we approximate the real time required to complete a task on K parallel processors, $T(K)$, as

$$T(K) = A + \frac{B}{n}, \tag{4}$$

where A is the serial (non-parallelizable) portion of the computation, and B is the parallel portion, then the parameters A, and B can be determined by linear regression of measured CPU times versus the inverse number of CPUs, $1/K$. In our implementation, B refers to spatial/spectral competitive endmember selection through morphological operations, and A refers to automated identification of endmembers. We can define the speedup for K CPUs, c_K, as

$$c_K = \frac{T(1)}{T(K)} = \frac{A+B}{A+(B/K)}. \tag{5}$$

The relationship above, usually expressed as an inequality to account for parallelization overhead, is generally known as Amdahl's Law [16]. It is obvious from this expression that the speedup of a parallel algorithm does not continue to increase with increasing the number of processors. Since only the parallel component scales while the time required to complete the serial component remains constant, there is a theoretical limit for the maximum parallel speedup, denoted as

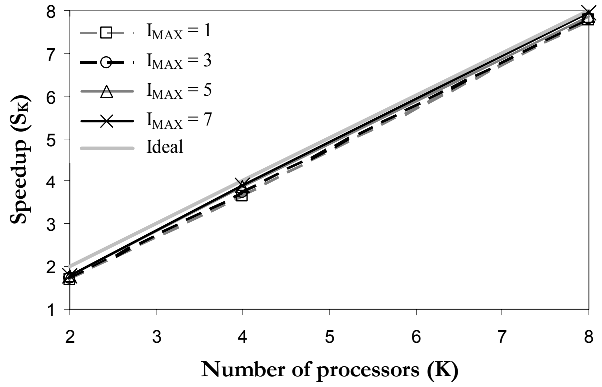
$$c_\infty = \lim_{K\to\infty} c_K = \frac{A+B}{A} = 1 + \frac{B}{A}. \tag{6}$$

Taking into account the limit imposed by Amdahl 's law, a series of application-specific experiments, focused on analyzing the scalability and classification accuracy of the proposed parallel algorithm, are shown in Table 2 and Fig. 5. Firstly, an experiment-based cross examination on AMEEPAR's endmember extraction accuracy is presented in Table 2, which tabulates the scores obtained after comparing endmembers extracted from AVCUP97 and AVJRBP97 to ground-truth signatures available from USGS and University of California, respectively (see Fig. 6). In both cases, SAM is used as a spectral similarity metric to compute the fitness between extracted endmembers and ground-truth signatures (the closer the value of SAD to 0, the highest the spectral similarity between the endmembers and the signatures). For comparative purposes, results obtained by other well-known endmember extraction algorithms such as Pixel Purity Index (PPI) [17], N-FINDR [18] and Iterative Error Analysis (IEA) [19] are also displayed. Results in Table 2 reveal that the proposed method is able to extract image endmembers which are almost identical, spectrally, to reference signatures, in particular when $I_{MAX} \geq 3$.

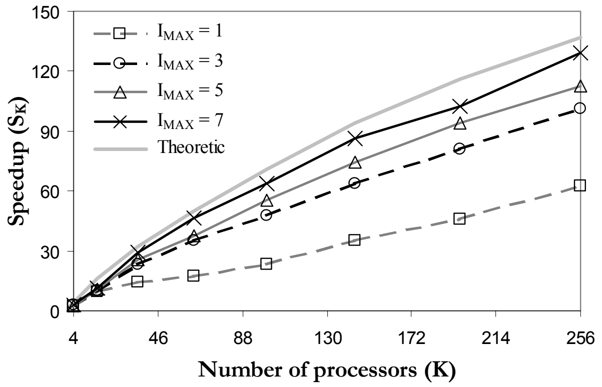
In order to investigate computational performance on different parallel platforms, Fig. 5 compares measured, ideal and theoretic speedups achieved by the AMEEPAR algorithm, using AVCUP97 and AVJRBP as benchmark data on the SGI Origin 2000 and Thunderhead computers. From Fig. 5, it can be observed that a very good scalability of the parallel code is achieved on the SGI Origin 2000 [see Figs. 5(a) and 5(c)]. On other hand, results on the Beowulf cluster show speedups which are very close to the theoretic limit values provided by Amdahl's law [see Figs. 5(b) and 5(d)]. It should be noted that processing times obtained in the multicomputer prevent near real-time exploitation of the scene, mainly due to the limited number of PEs available at the time of the experiments. However, the improvement of the parallel implementation with regard to the serial implementation is very significant as shown by the almost ideal measured speedups. On other hand, despite the extremely large size of both AVCUP97 and AVJRBP97 scenes (see Table 1), the computational time to obtain high-quality endmember signatures can be as low as 39 seconds in the Thunderhead cluster for $K \geq 128$ processors. This fact reveals that our parallel algorithm is able to accomplish fast and accurate endmember extraction from full-size AVIRIS scenes, provided that the number of PEs is sufficiently large to accommodate the extremely high computational requirements involved in those calculations.

	AVCUP97					AVJRBP97				
	Alunite	Buddingtonite	Calcite	Kaolinite	Muscovite	Soil	Forest	Grass	Chaparral	Shade
PPI	0.084	0.065	0.075	0.087	0.066	0.027	0.022	0.021	0.019	0.017
N-FINDR	0.072	0.062	0.054	0.081	0.069	0.028	0.025	0.022	0.020	0.019
IEA	0.046	0.051	0.047	0.071	0.056	0.016	0.013	0.015	0.008	0.009
AMEEPAR $I_{MAX} = 1$	0.101	0.098	0.105	0.123	0.106	0.020	0.018	0.020	0.015	0.015
AMEEPAR $I_{MAX} = 3$	0.097	0.094	0.098	0.103	0.095	0.017	0.015	0.018	0.012	0.012
AMEEPAR $I_{MAX} = 5$	0.077	0.073	0.043	0.081	0.063	0.015	0.012	0.013	0.010	0.011
AMEEPAR $I_{MAX} = 7$	0.036	0.018	0.034	0.073	0.053	0.012	0.011	0.011	0.007	0.010

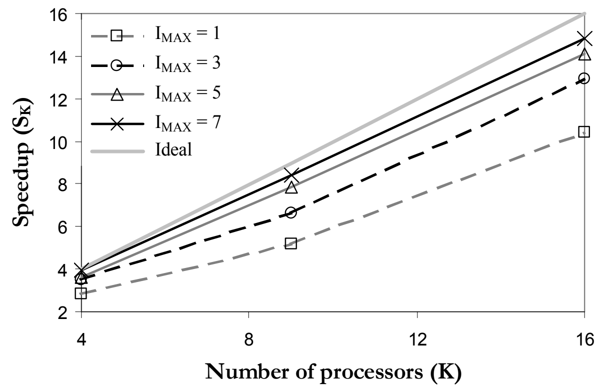
Table 2. SAD-based spectral similarity scores between endmember signatures extracted by PPI, N-FINDR, IEA and AMEEPAR and ground-truth spectral signatures.



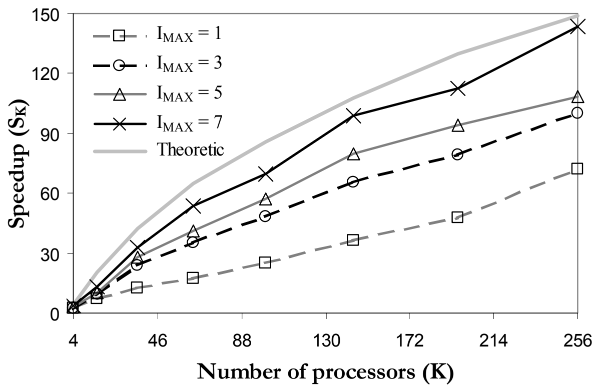
(a) Processing AVCUP97 in SGI Origin 2000



(b) Processing AVCUP97 in Thunderhead

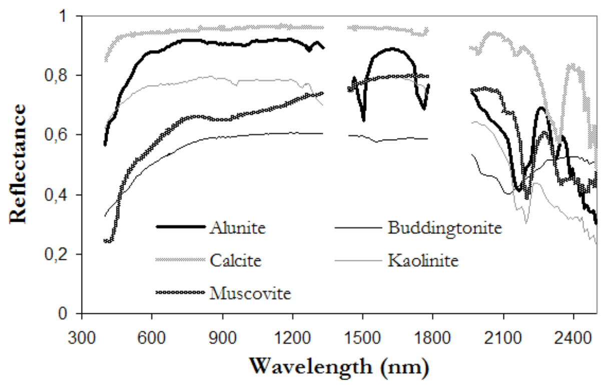


(c) Processing AVJRBP97 in SGI Origin 2000

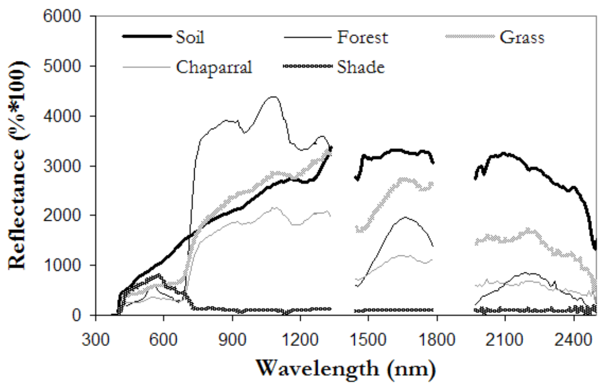


(d) Processing AVJRBP97 in Thunderhead

Figure 5. Parallel performance of AMEEPAR algorithm.



(a) AVJRBP97



(b) AVCUP97

Figure 6. Ground-truth spectral signatures used in experimental evaluation of endmember extraction algorithms.

5. Conclusions and future research

The aim of this paper has been the parallel implementation on high performance computers of an innovative technique for endmember extraction and hyperspectral image analysis. We demonstrate that spatial/spectral algorithms, based on classic spatial-based morphological operations, can be efficiently implemented on massively parallel computers. We have also shown that the framework of mathematical morphology is very suitable to the designing of efficient hyperspectral analysis algorithms. In the proposed parallel approach, code reusability is enhanced by the application of so-called parallelizable patterns. Essentially, such patterns define the maximum amount of work that can be executed by a single processing unit without having to communicate to obtain data values that reside elsewhere. Experimental results in this paper suggest that our parallel algorithm provides adequate results in both the quality of the solutions and the time to obtain them, in particular, when it is implemented on a commodity Beowulf cluster. With the above issues in mind, the present investigation indicates the feasibility of on-board processing of remotely sensed data, using parallel computing techniques, to interpret and classify hyperspectral data more accurately and efficiently than is currently possible. As future work, we plan to implement the parallel algorithm in other high performance parallel computing architectures, such as the Medusa Beowulf cluster at NASA/GSFC or the Bull NovaScale 5160 multicomputer at CEPBA. We are also working toward a field programmable gate array (FPGA)-based implementation that may allow us to accomplish the goal of near real-time processing of hyperspectral image data, with potential applications in hyperspectral image compression.

Acknowledgement

The authors would like to acknowledge Dr. Robert O. Green and the AVIRIS team at NASA's Jet Propulsion Laboratory for their advice and support to this work. They also thank Profs. Mateo Valero and Francisco Tirado for their suggestions and support, and Drs. James C. Tilton, J. Anthony Gualtieri and Glenn Gardner (system administrator) for their help with experiments in the Thunderhead cluster. Funding from Junta de Extremadura (local government) under project number 2PR03A026 is gratefully acknowledged. Antonio Plaza would also like to thank for support received from the Spanish Ministry of Education and Science (PR2003-0360 Fellowship).

References

[1] C.-I Chang, *Hyperspectral imaging: Techniques for spectral detection and classification*. Kluwer Academic Publishers, 2003.

[2] R.O. Green et al., "Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (AVIRIS)," *Remote Sensing of Environment*, vol. 65, pp. 227–248, 1998.

[3] A. Plaza, P. Martinez, R.M. Perez, J. Plaza, "A quantitative and comparative analysis of endmember extraction algorithms from hyperspectral data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, no. 3, pp. 650-663, 2004.

[4] J. A. Gualtieri and J. C. Tilton, "Hierarchical segmentation of hyperspectral data," *XI NASA/Jet Propulsion Laboratory Airborne Earth Science Workshop*, Pasadena, CA, USA, 2002.

[5] R. Brightwell, L. A. Fisk, D. S. Greenberg, T. Hudson, M. Levenhagen, A. B. Maccabe and R. Riesen, "Massively parallel computing using commodity components," *Parallel Computing*, vol. 26, pp. 243-266. 2000.

[6] J. Dorband, J. Palencia and U. Ranawake, "Commodity computing clusters at Goddard Space Flight Center," *Journal of Space Communication*, vol. 1, no. 3, 2003.

[7] P. Wang, K. Y. Liu, T. Cwik, R.O. Green, "MODTRAN on supercomputers and parallel computers," *Parallel Computing*, vol. 28, pp. 53–64, 2002.

[8] F. J. Seinstra, D. Koelma and J. M. Geusebroek, "A software architecture for user transparent parallel image processing," *Parallel Computing*, vol. 28, pp. 967-993, 2002.

[9] J. Serra, *Image Analysis and Mathematical Morphology*. Academic: New York, 1982.

[10] P. Soille, *Morphological Image Analysis: Principles and Applications*, *2nd ed.* Springer: Berlin, 2003.

[11] A. Plaza, P. Martinez, R.M. Perez, J. Plaza, "Spatial/spectral endmember extraction by multidimensional morphological operations," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 40, no. 9, pp. 2025-2041, Sept. 2002.

[12] A. Plaza, P. Martinez, R.M. Perez, J. Plaza, "A new approach to mixed pixel classification of hyperspectral imagery based on extended morphological profiles," *Pattern Recognition*, vol. 37, pp. 1097-1116, 2004.

[13] A. Plaza, P. Martinez, J.A. Gualtieri, R.M. Perez, "Spatial/spectral identification of endmembers from AVIRIS data using mathematical morphology," *Proc. X NASA/Jet Propulsion Laboratory Airborne Earth Science Workshop*, Pasadena, CA, USA, 2001.

[14] D. Heinz and C.-I Chang, "Fully constrained least squares linear mixture analysis for material quantification in hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 39, pp. 529–545, 2000.

[15] A. Plaza, "Extended morphological profiles for hyperspectral image analysis on parallel computers," *Neural Information Processing Systems Conference*, Vancouver and Whistler, British Columbia, Canada, 2003.

[16] J. L. Hennessy, D. A. Patterson, *Computer architecture: A quantitative approach, 3rd ed*. Morgan Kaufmann Publishers, 2002.

[17] J.W. Boardman, F.A. Kruse and R.O. Green, "Mapping target signatures via partial unmixing of AVIRIS data," *Summaries of JPL Airborne Earth Science Workshop*, Pasadena, CA, 1995.

[18] M.E. Winter, "N-FINDR: an algorithm for fast autonomous spectral endmember determination in hyperspectral data," *Image Spectrometry V*, Proc. SPIE 3753, pp. 266-277, 1999.

[19] R.A. Neville, K. Staenz, T. Szeredi, J. Lefebvre and P. Hauff, "Automatic endmember extraction from hyperspectral data for mineral exploration," *4th International Airborne Remote Sensing Conf. and Exhibition/21st Canadian Symposium on remote Sensing*, Ottawa, Ontario, Canada, pp. 21-24, June 1999.