Parallel implementation of algorithms for standoff detection in hyperspectral imagery

Antonio Plaza*, Pablo Martínez, Javier Plaza, Rosa Pérez
Neural Networks and Signal Processing Group (GRNPS), Computer Science Department,
University of Extremadura, Avda. de la Universidad S/N, 10071 Cáceres, SPAIN.

ABSTRACT

Hyperspectral imaging systems, used in conjunction with appropriate detection and recognition algorithms, have demonstrated to be very appropriate tools for standoff detection in many different environments. Compared to other techniques available such as multispectral imaging, which typically collects only tens of images, hyperspectral instruments are capable of collecting hundreds of images, corresponding to different wavelength channels, for the same area on the surface of the Earth. While developments in hyperspectral technology hold great promise for advanced standoff detection, they create new processing challenges. 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. However, several applications exist where having the desired information calculated in real-time or near real-time is desired, e.g. those focused on the detection and tracking of forest fires, oil spills and other types of chemical contamination. In this paper, we discuss parallel implementations of a novel morphological algorithm for standoff detection in hyperspectral imagery. Several illustrative processing examples are provided, thus allowing an evaluation of the speedup factors provided by implementing the above algorithm, characterized by its combined use of spatial and spectral information, on massively parallel computer facilities.

Keywords: Standoff detection, Hyperspectral imagery, Parallel computing, Morphological analysis.

1. INTRODUCTION

Standoff detection involves passive and active methods for sensing of chemical and biological material when the sensor is physically separated from the site of interest (target area). Remote sensing and supporting technologies, such as GPS and GIS, are increasingly being used in environmental activities such as monitoring of natural resources, as well as detection and tracking of natural disasters such as forest fires, oil spills, and other types of chemical contamination. Hyperspectral imaging systems, used in conjunction with appropriate detection and recognition algorithms, have demonstrated to be very useful tools for standoff detection[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[2], 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. 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[3].

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

* E-mail: aplaza@unex.es; Phone: +34 927257195; Fax: +34 927257203.

techniques have overcome some of the weaknesses of full pixel approaches by using linear and/or nonlinear mixture modeling and signal processing techniques[4]. 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. 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 analysis[5]. 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 advanced standoff detection, they create new processing challenges[6]. 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 desired, e.g. those pursuing detection and tracking of environmental disasters such as forest fires, oil spills, and other types of chemical contamination. In the above cases, timely classification is highly desirable in order to design an effective environmental protection and response plan, which could help to reduce the environmental consequences of the event, as well as to protect human life.

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[7,8]. 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[9]. 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[6]. 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 standoff detection 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 STANDOFF DETECTION IN HYPERSPECTRAL IMAGERY

In this section, we give a description of the proposed algorithm for standoff detection 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[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 mathematical morphology operations to hyperspectral images

Mathematical morphology[10] is a classic nonlinear image processing technique that was originally defined for binary images. 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[11]. 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 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. 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[11], 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[12].

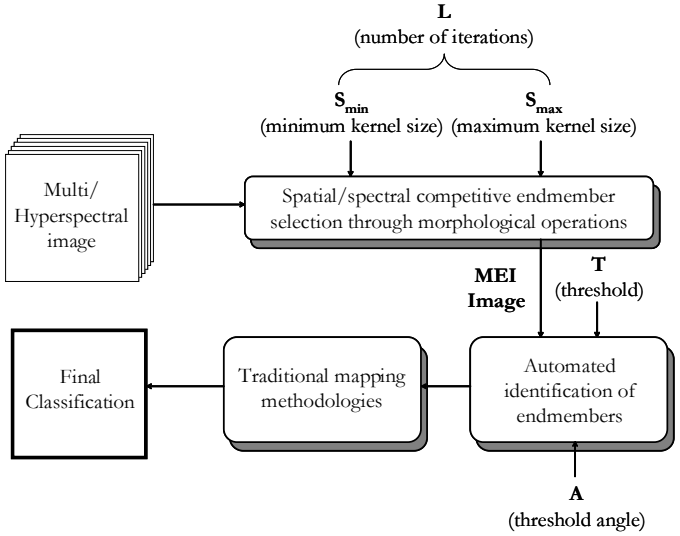

Figure 1. Schematic representation of the proposed standoff detection algorithm.

2.2. Standoff detection algorithm

A general block diagram of the proposed algorithm is given in Fig. 1. As shown in this figure, the input to the method is the full image data cube, with no previous dimensionality reduction. First, a minimum kernel $K = S_{min}$ is considered. This element is moved through all the pixels of the image, defining a spatial context around each hyperspectral pixel $\boldsymbol{h}(x, y)$. The spectrally purest ($\boldsymbol{p}$) and the spectrally most highly mixed ($\boldsymbol{m}$) spectral signatures are respectively obtained at the neighborhood of $\boldsymbol{h}(x, y)$ defined by K using the following cumulative distance-based extended morphological operations[12].

$$\boldsymbol{p} = \arg\_\,\mathrm{Max}_{(s,t)\in K}\left\{\sum_s\sum_t \mathrm{dist}\left(\boldsymbol{h}(x, y), \boldsymbol{h}(x-s, y-t)\right)\right\}, \qquad \forall (s, t)\in K, \tag{1}$$

$$\boldsymbol{m} = \arg\_\,\mathrm{Min}_{(s,t)\in K}\left\{\sum_s\sum_t \mathrm{dist}\left(\boldsymbol{h}(x, y), \boldsymbol{h}(x+s, y+t)\right)\right\}, \qquad \forall (s, t)\in K, \tag{2}$$

where dist is the SAD distance. A morphological eccentricity index (MEI) is then obtained by calculating the SAD between the two signatures above. This operation is repeated for all the pixels in the scene, using kernels of progressively increased size, and the resulting scores are used to evaluate each pixel in both spatial and spectral terms. The algorithm performs as many iterations as needed until $K = S_{max}$. The associated MEI value of selected pixels at subsequent iterations is updated by means of newly obtained values, as a larger spatial context is considered, until a final MEI image is generated. This approach ensures a complete spatial/spectral description of the image and provides an efficient tool to integrate both types of information simultaneously[13]. Automated pure pixel (endmember) selection is performed from the MEI image by using a threshold value. A final selection of representative spectral signatures for each material present in the scene is refined by using a region-growing procedure that incorporates neighboring pixels that are sufficiently similar (according to a certain threshold angle parameter) to the regions obtained after thresholding[14]. Mean spectra are obtained for the resulting regions after the region-growing process above, and a final set of endmember signatures is obtained. Traditional methodologies such as the spectral angle mapper (SAM) or linear spectral unmixing (LSU) can be used for the purpose of mapping the obtained endmembers over the original image, and obtaining a final classification result in terms of full pixel assignment to classes, or estimation of sub-pixel abundances in mixed pixels.

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[15] (`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. Each partition produced by NDPP is called a parallelizable spatial/spectral pattern[16]. Next, we show how such patterns are obtained, and their further use in the implementation of the parallel algorithm. The section concludes with an overview of the sequence of operations performed by the proposed parallel algorithm.

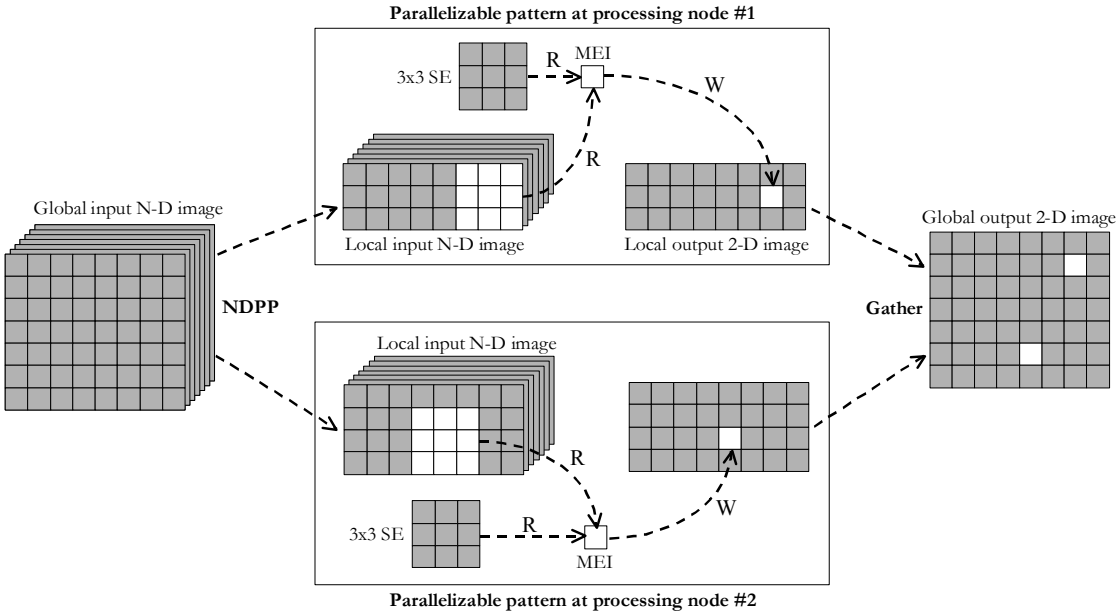

Figure 2. Partitioning of hyperspectral image using the NDPP, and assignment of parallelizable patterns to processing units for local MEI index computation. Each processing node calculates the MEI score for a series of pixels. These partial values are combined to form a global output MEI image.

3.1. Parallelizable spatial/spectral patterns

The concept of parallelizable spatial/spectral pattern is defined as the maximum amount of work that –when applied to partial image data– can be performed without communication[16]. As stated above, the so-called NDPP module divides the hyperspectral image into a collection of parallelizable patterns 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. 2 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. 2, 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.

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[16]. 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. 3(a)]. In such cases, we allow an overlapping scatter of the global input image to avoid additional communication [see Fig. 3(b)]. In our implementation, the NDPP module automatically determines the size in pixels of the overlapping section.

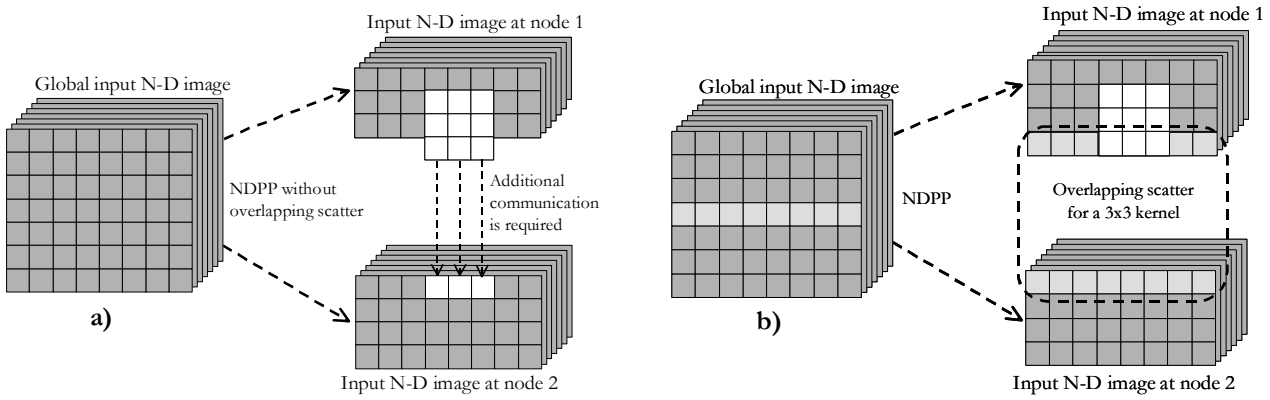

Figure 3. a) Kernel computation split among several processors. b) Overlapping scatter avoids further communication.

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.

3.2. Summary of operations

As shown in the previous section, the parallel implementation of the proposed morphological algorithm is based on an efficient decomposition scheme, provided by a so-called NDPP module that also acts as master node in charge of all I/O operations. The partitioner has been implemented so that it automatically determines an optimum size for parallelizable patterns to be distributed among processing nodes. This is done by taking into account both spatial and spectral resolutions. By distributing data evenly among the processors, load balance is achieved. In the proposed `MPI`-based

implementation, the NDPP distributes data to the other processors through the function `MPI_Bcast()`, i.e. a single-source broadcast to all processors. The master processor then sends each slave processor its portion of the input hyperspectral image through a pair of functions, `MPI_Send()` and `MPI_Recv()`. After each processor has finished working on its portion of the data, it must send its results back to the master processor for output to the resulting MEI image file. This is accomplished again through the send/receive function pairs. In a nutshell, executing the following sequence of operations on a parallel platform carries out the entire parallel computation:

1. Read the original hyperspectral image, the structuring element, the number of available processors and other input data.
2. Use the `MPI` library to generate necessary system information, including the total number of processors, each processor's ID number, timing and other data.
3. Partition the original hyperspectral image using the NDPP, obtaining a collection of parallelizable spatial/spectral patterns.
4. For each parallelizable pattern, execute the sequential algorithm in a different processor, obtaining a collection of local 2-D MEI images.
5. Collect the final results from all processors and form a final MEI image with the same spatial dimensions (i.e. number of pixels) as the original input data.
6. Perform automated endmember extraction from resulting MEI image by using a combination of automated thresholding techniques and spatial/spectral region growing[17]. It must be taken into account that these operations have not yet been parallelized. Also, the mapping/abundance estimation stage of the algorithm (see Fig. 1) has not been considered for the parallel implementation, which is reduced in this work to the competitive spatial/spectral endmember selection.

As a final note, we emphasize that the original algorithm is an excellent application for parallel computing, because there is no dependence between the calculations made at each spatial/spectral pattern, and only minimal communication is required for the entire calculation. Performance data for the parallel implementation described above are given in the next section.

4. RESULTS AND DISCUSSION

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). The system is a distributed memory, message-passing parallel machine of the MIMD class. It is 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 that has the topology of a three-dimensional hyper-crossbar. The theoretical peak performance of the system is 32 Gflop/s. The operating system is Irix 6.5, with single-kernel architecture. Our algorithm was coded entirely in C++, and compiled using version 7.3.1.2 of the MIPSpro compiler suite. The communication library used was the message-passing interface `MPI`.

To empirically investigate the scaling properties of the algorithm described in this paper, it was applied to four standard AVIRIS hyperspectral data sets, described in Table 1 (see also Fig. 4). We have selected the above datasets because they are widely available online[18], and have served as benchmark data for many scientific studies, including performance evaluation of hyperspectral analysis algorithms. In addition, there is ground-truth information available for the scenes that can be used to validate performance of analysis techniques.

Identifier	Area	Year	Pixels	Bands	Spectral range	Units	Application area
AVCUP95	Cuprite, NV	1995	320x200	50	2 – 2.5 μm	Reflectance	Geology
AVIP92	Indian Pines, IN	1992	145x145	220	0.4 – 2.5 μm	Radiance	Agriculture
AVJRAD97	Jasper Ridge, CA	1997	614x512	224	0.4 – 2.5 μm	Radiance	Vegetation and soil studies
AVJREF97	Jasper Ridge, CA	1997	614x512	224	0.4 – 2.5 μm	Reflectance	Vegetation and soil studies

Table 1. Database of AVIRIS hyperspectral scenes used to illustrate performance of the proposed parallel algorithm.

a) b) c)

Figure 4. Representative spectral bands from AVIRIS hyperspectral scenes. a) AVCUP95. b) AVIP92. c) AVJREF97.

The proposed morphological method was run on the above images serially and with two- through 4-, 8-, and 16-fold parallelism. Due to the operating system configuration on the supercomputer used in this study, parallel computation on more than 16 CPUs was not available to us at the time experiments were carried out. The performance of the parallel implementation was tested by timing the program over a variety of inputs and numbers of processors (see Table 2). Two different structuring element sizes, i.e. 3x3 pixels and 15x15 pixels were considered in each algorithm run, bearing in mind that 15x15 pixel-sized kernels generally produce more accurate results in the analysis. 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 n parallel processors, $T(n)$, as

$$T(n) = A + \frac{B}{n}, \tag{3}$$

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, $1/n$. In our implementation, B refers to spatial/spectral competitive endmember selection through morphological operations, and A refers to automated identification of endmembers (see Fig. 1). We can define the speedup for n CPUs, c_n, as

$$c_n = \frac{T(1)}{T(n)} = \frac{A + B}{A + (B/n)}. \tag{4}$$

The relationship above, usually expressed as an inequality to account for parallelization overhead, is generally known as Amdahl's Law[19]. 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

$$c_\infty = \lim_{n\to\infty} c_n = \frac{A + B}{A} = 1 + \frac{B}{A}. \tag{5}$$

From the results shown in Table 2, it can be noticed that the proposed parallel algorithm achieves significant speedup when compared to the serial implementation, in particular when 16-fold parallelism is applied. It can be derived from Table 2 that the measured speedup of the program drops for 8, 4 and 2 processing nodes. The measured speedup is higher for large structuring elements, a fact that reveals that morphological operations are suitable to the designing of efficient implementations of algorithms for hyperspectral image analysis. The algorithm achieves similar speedup factors for radiance and reflectance data. Further results are still required in order to determine if adding more processors may be counterproductive due to the large impact of communication. As an important final note, we should also state that the region growing process, which has not yet been parallelized, has a strong impact on the final results[17]. As reported in previous work, this step is important in order to reduce algorithm sensitivity to noise and outliers in the data. For example, the region-growing stage takes approximately 86 seconds when the algorithm is run serially on the AVIP92 scene. On other hand, this stage takes only 38 and 35 seconds in the AVJRAD97 and AVJREF97 scenes, respectively. This result reveals that the region-growing is scene-dependent. As a result, although the parallelization of the spatial/spectral competitive procedure task leads to a significant improvement in the computational performance of the algorithm, additional work is still required in order to parallelize the automated selection of endmembers performed by the algorithm, which includes the computationally expensive and scene-dependent region-growing procedure.

Original Image	Structuring element size	Serial proc. time T(1) in seconds	Number of Processors n	Parallel proc. time T(n) in seconds	Measured speedup c_n
AVCUP95	3x3	109	2	88	1.23
			4	57	1.89
			8	39	2.79
			16	28	3.89
	15x15	257	2	194	1.32
			4	112	2.28
			8	62	4.11
			16	48	5.35
AVIP92	3x3	148	2	124	1.19
			4	88	1.67
			8	55	2.65
			16	41	3.60
	15x15	349	2	258	1.35
			4	146	2.38
			8	83	4.19
			16	65	5.36
AVJRAD97	3x3	403	2	341	1.18
			4	245	1.64
			8	182	2.21
			16	142	2.83
	15x15	985	2	781	1.26
			4	437	2.25
			8	282	3.49
			16	212	4.64
AVJREF97	3x3	397	2	328	1.21
			4	235	1.69
			8	176	2.25
			16	136	2.91
	15x15	972	2	741	1.31
			4	428	2.27
			8	275	3.53
			16	203	4.78

Table 2. Measured parallel speedup for different images/structuring elements, and different numbers of processors.

CONCLUSIONS AND FUTURE LINES

This paper was dedicated to the interest of parallel asynchronous computations in hyperspectral image analysis. We showed that extended spatial/spectral algorithms, based on classic spatial-based morphological operations, can be efficiently implemented on massively parallel computers. We have also confirmed that the framework of mathematical morphology is very suitable to the designing of efficient hyperspectral analysis algorithms. In the proposed scheme, 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. A limitation of this approach is that not all of the steps of the proposed algorithm have been yet optimized for parallel processing. A research topic deserving future research is the parallelization of the spatial/spectral region-growing step, which refines and greatly improves the task of extracting endmembers for spectral mixture analysis of hyperspectral data. 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. Finally, we will continue implementing example programs to investigate the implication of parallelization of typical applications, especially in the area of near real-time hyperspectral image processing.

ACKNOWLEDGEMENT

We gratefully thank Dr. James C. Tilton (Applied Information Sciences Branch, NASA/Goddard Space Flight Center) for providing us with an MPI-based parallel implementation of his Hierarchical Segmentation (HSEG) program, which served as a guide for the development of parallel implementations described in this work. We also thank Prof. Mateo Valero (Polytechnic University of Catalonia), for granting us account-based access to the SGI Origin 2000 Silicon Graphics supercomputer at CEPBA (European Center for Parallelism of Barcelona), which was used in computational performance tests. Support of the present research by Prof. Francisco Tirado (Free University of Madrid) is also gratefully acknowledged.

REFERENCES

1. C.-I Chang, *Hyperspectral imaging: 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. D. Landgrebe, "Hyperspectral image data analysis," *IEEE Signal Processing Magazine*, vol. 19, pp. 17–28, 2002.
4. J.B. Adams, M.O. Smith, and A.R. Gillispie, "Imaging spectroscopy: interpretations based on spectral mixture analysis," in *Remote Geochemical Analysis: Elemental and Mineralogical Composition*, C. M. Pieters and P. A. Englert, Eds. Cambridge, U.K.: Univ. of Cambridge, 1993, pp. 145-166.
5. V. Madhok and D. Landgrebe, *Spectral-spatial analysis of remote sensing data: An image model and a procedural design*, Ph.D. dissertation, School of Electrical Engineering and Computer Science, Purdue University, Lafayette, IN, 1998.
6. J. Le Moigne and J.C. Tilton, "Refining image segmentation by integration of edge and region data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 33, pp. 605–615, 1995.
7. J.A. Gualtieri and J. C. Tilton, "Hierarchical segmentation of hyperspectral data," in *Summaries of the 11th NASA/Jet Propulsion Laboratory Airborne Earth Science Workshop*, Pasadena, CA, 2002.
8. J.C. Tilton, "A recursive PVM implementation of an image segmentation algorithm with performance results comparing the HIVE and the Cray T3E," In *Proceedings of the 7th Symposium on the Frontiers of Massively Parallel Computation*, Annapolis, MD, 1999.
9. P. Wang, K.Y. Liu, T. Cwik, R.O. Green, "MODTRAN on supercomputers and parallel computers," *Parallel Computing*, vol. 28, pp. 53–64, 2002.
10. P. Soille, *Morphological image analysis: Principles and applications*, Springer-Verlag, 2003.
11. P. Soille and M. Pesaresi, "Advances in mathematical morphology applied to geoscience and remote sensing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 40, pp. 2042–2055, 2002.
12. A. Plaza, P. Martínez, R.M. Pérez, J. Plaza, "Spatial/spectral endmember extraction by multidimensional morphological operations," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 40, pp. 2025–2041, 2002.
13. A. Plaza, P. Martínez, J.A. Gualtieri, R.M. Pérez, "Spatial/spectral identification of endmembers from AVIRIS data using mathematical morphology," In *Summaries of the 10th NASA/Jet Propulsion Laboratory Airborne Earch Science Workshop*, Pasadena, CA, 2001.
14. A. Plaza, P. Martínez, J.A. Gualtieri, R.M. Pérez, "Automated identification of endmembers from hyperspectral images using mathematical morphology," In *Proceedings of the 8th SPIE International Symposium on Remote Sensing*, Toulouse, France, 2001.
15. MPICH: A portable implementation of MPI. Available online: http://www-unix.mcs.anl.gov/mpi/mpich/.
16. F.J. Seinstra, D. Koelma, J.M. Geusebroek, "A software architecture for transparent parallel image processing," *Parallel Computing*, vol. 28, pp. 967-923, 2002.
17. J. C. Tilton, *Method for recursive hierarchical segmentation by region growing and spectral clustering with a natural convergence criterion*, Disclosure of Invention and New Technology: NASA Case No. GSC 14,328-1, February 28, 2000.
18. California Institute of Technology. AVIRIS (Airborne Visible/Infrared Imaging Spectrometer) homepage. Available online: http://aviris.jpl.nasa.gov.
19. J.L. Hennessy and D.A. Patterson, *Computer architecture: A quantitative approach, Third edition*. Morgan Kaufmann Publishers, San Mateo, CA, 2002.