Parallel Implementation of RX Anomaly Detection on Multi-Core Processors: Impact of Data Partitioning Strategies

José M. Molero[a], Ester M. Garzón[a], Inmaculada García[b] and Antonio Plaza[c]

[a]Department of Computer Architecture and Electronics, University of Almería,Ctra Sacramento s/n. 04120 Almería, SPAIN;
[b]Department of Computer Architecture, University of Málaga, Escuela de Ingenierías, Campus de Teatinos, 29071 Málaga, SPAIN;
[c]Hyperspectral Computing Laboratory, University of Extremadura, Avda. de la Universidad s/n, E-10071 Cáceres, SPAIN

ABSTRACT

Anomaly detection is an important task for remotely sensed hyperspectral data exploitation. One of the most widely used and successful algorithms for anomaly detection in hyperspectral images is the Reed-Xiaoli (RX) algorithm. Despite its wide acceptance and high computational complexity when applied to real hyperspectral scenes, few documented parallel implementations of this algorithm exist, in particular for multi-core processors. The advantage of multi-core platforms over other specialized parallel architectures is that they are a low-power, inexpensive, widely available and well-known technology. A critical issue in the parallel implementation of RX is the sample covariance matrix calculation, which can be approached in global or local fashion. This aspect is crucial for the RX implementation since the consideration of a local or global strategy for the computation of the sample covariance matrix is expected to affect both the scalability of the parallel solution and the anomaly detection results. In this paper, we develop new parallel implementations of the RX in multi-core processors and specifically investigate the impact of different data partitioning strategies when parallelizing its computations. For this purpose, we consider both global and local data partitioning strategies in the spatial domain of the scene, and further analyze their scalability in different multi-core platforms. The numerical effectiveness of the considered solutions is evaluated using receiver operating characteristics (ROC) curves, analyzing their capacity to detect thermal hot spots (anomalies) in hyperspectral data collected by the NASA's Airborne Visible Infra-Red Imaging Spectrometer system over the World Trade Center in New York, five days after the terrorist attacks of September 11th, 2001.

Keywords: Hyperspectral imaging, anomaly detection, RX algorithm, multi-core processors

1. INTRODUCTION

Hyperspectral imaging[1] is concerned with the measurement, analysis, and interpretation of spectra acquired from a given scene (or specific object) at a short, medium or long distance by an airborne or satellite sensor.[2] Hyperspectral imaging instruments such as the NASA Jet Propulsion Laboratory's Airborne Visible Infra-Red Imaging Spectrometer (AVIRIS)[3] are now able to record the visible and near-infrared spectrum (wavelength region from 0.4 to 2.5 micrometers) of the reflected light of an area of 2 to 12 kilometers wide and several kilometers long using 224 spectral bands. The resulting "image cube" (see Fig. 1) is a stack of images in which each pixel (vector) has an associated spectral signature or *fingerprint* that uniquely characterizes the underlying objects.[4] The resulting data volume typically comprises several GBs per flight.[5]

Anomaly detection is an important task for hyperspectral data exploitation. An anomaly detector enables one to detect spectral signatures which are spectrally distinct from their surroundings with no *a priori* knowledge.

Send correspondence to José M. Molero[a]:
E-mail: jmp384@ual.es; Telephone: +34 950 214396 ; URL: http://www.hpca.ual.es/~jmolero

High-Performance Computing in Remote Sensing, edited by Bormin Huang, Antonio J. Plaza,
Proc. of SPIE Vol. 8183, 81830I · © 2011 SPIE · CCC code: 0277-786X/11/$18 · doi: 10.1117/12.897388

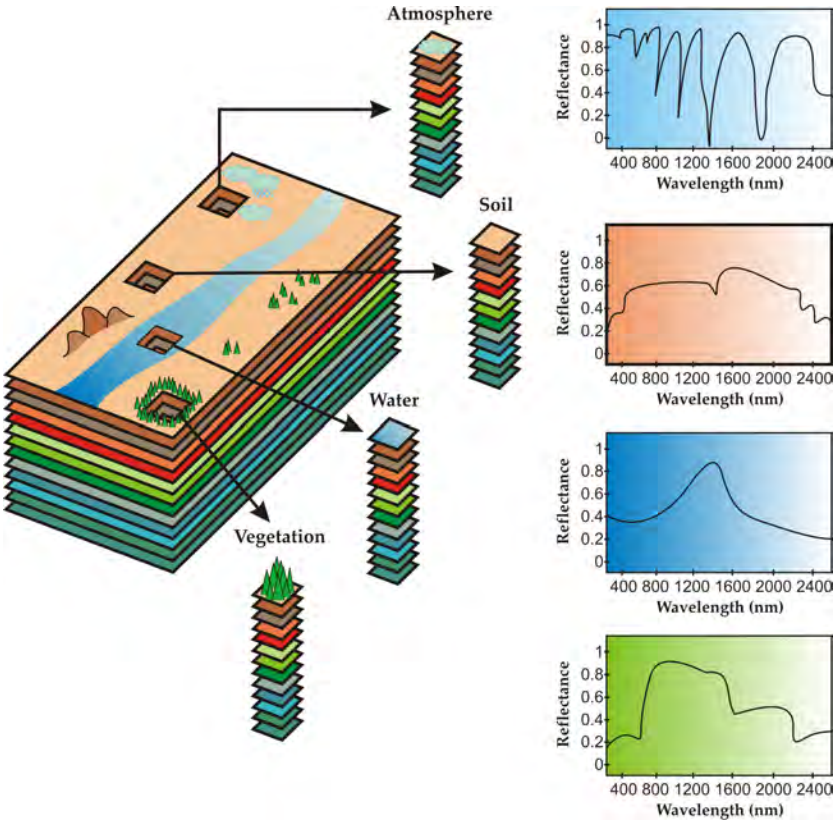

Figure 1. Concept of hyperspectral imaging.

In general, such anomalous signatures are relatively small compared to the image background, and only occur in the image with low probabilities. A well-known approach for anomaly detection was developed by Reed and Yu, and is referred to as the RX algorithm, which has shown success in anomaly detection for multispectral and hyperspectral images.[4] The RX uses the pixel currently being processed as the matched signal. Since the RX uses the sample covariance matrix to take into account the sample spectral correlation, it performs the same task as the Mahalanobis distance, which has been widely used in hyperspectral imaging applications.[6] A variation of the algorithm consists in applying the same concept in local neighborhoods centered around each image pixel, this is known as the kernel version of the RX algorithm.[7]

Despite its wide acceptance and high computational complexity when applied to real hyperspectral scenes, few approaches have been developed for parallel implementation of this algorithm due to the complexity of calculating the sample covariance matrix (and its inverse) in parallel.[8,9] This is in contrast with the significant amount of previous work focused on efficient implementation of hyperspectral imaging algorithms on high performance computing architectures.[10–14] A standard data partitioning framework may provide improved results for parallel implementation of the RX algorithm if the sample covariance matrices are calculated independently for relatively small subimages rather than computing the sample covariance matrix for the entire hyperspectral image, which from a parallel computing point of view would require additional interprocessor communications that may reduce parallel performance. This aspect is crucial for the RX implementation since the consideration of a local or global strategy for the computation of the sample covariance matrix is expected to significantly affect the scalability of the parallel solution. However, there is a trade-off between improving the performance of the parallel implementation and the quality of the final solution in terms of anomaly detection accuracy.

The evaluation of a hybrid parallel implementation of RX based on message passing interface (`MPI`)* and

*http://www.mcs.anl.gov/research/projects/mpi/

POSIX Threads (`Pthreads`)[†] on heterogeneous clusters has been previously carried out.[15] This heterogeneous version of RX associates each sub-image to each `MPI` process and, moreover, each `MPI` process is computed by a set of threads. In this way the two levels of parallelism of clusters of multi-cores can be exploited by this hybrid version.[15] In this paper, our interest is focused on multicore architectures. Bearing in mind that the hybrid parallel RX can be executed on such platforms, our aim is to optimize the exploitation of this kind of architectures by the hybrid parallel RX. Moreover, we investigate the impact of data partitioning on anomaly detection accuracy by means of receiver operating characteristics (ROC) curves,[7] analyzing their capacity to detect thermal hot spots (anomalies) in one example of hyperspectral data which comprises a data set collected by the NASA's Airborne Visible Infra-Red Imaging Spectrometer (AVIRIS) system over the World Trade Center (WTC) in New York, five days after the terrorist attacks.

The remainder of the paper is structured as follows. Section 2 briefly describes the classic RX algorithm. Section 3 describes the parallel implementation of RX algorithm evaluated in this work and the different data partitions. Section 4 describes the experimental results. First, the hyperspectral data set considered in experiments is described. Then, the algorithms are analyzed in terms of their anomaly detection accuracy using ROC curves for different data partitions in order to estimate the impact of the data partitioning in the anomaly detection accuracy. Finally, parallel performance and scalability issues are explored using a state-of-the-art multicore architecture. Finally, Section 5 concludes with some remarks and hints at plausible future research.

2. RX ALGORITHM

The RX algorithm has been widely used in signal and image processing.[16] The filter implemented by this algorithm is referred to as RX filter and defined by the following expression:

$$\delta^{RX}(\mathbf{x}) = (\mathbf{x} - \mu)^T \mathbf{K}^{-1} (\mathbf{x} - \mu), \tag{1}$$

where $\mathbf{x} = \left[x^{(0)}, x^{(1)}, \cdots, x^{(n)}\right]$ is a sample, n-dimensional hyperspectral pixel (vector), μ is the sample mean of the hyperspectral image and $\mathbf{K}$ is the sample data covariance matrix. As we can see, the form of δ^{RX} is actually the well-known Mahalanobis distance.[6] It is important to note that the images generated by the RX algorithm are generally gray scale images. In this case, the anomalies can be categorized in terms of the value returned by RX, so that the pixel with higher value of $\delta^{RX}(\mathbf{x})$ can be considered the first anomaly, and so on.

From a computational point of view, the RX algorithm can be computed by means of the following four steps:

1. Calculate the sample data covariance matrix $\mathbf{K}$ $[O(rows \cdot columns \cdot bands^2)]$
2. Calculate $\mathbf{K}^{-1}$ by the Gauss Method $[O(bands^3)]$
3. Calculate the Mahalanobis distance as δ^{RX} $[O(rows \cdot columns \cdot bands^2)]$
4. Locate the maximum values of δ^{RX} or anomalies $[O(rows \cdot columns)]$

The computational complexity of each step is described between brackets, where *rows* and *columns* are the spatial dimensions of the input hyperspectral image and *bands* is the spectral dimension. This is a relevant information for subsequent analysis of parallel RX algorithm.

3. HYBRID PARALLEL RX

Previous studies in the field of parallel hyperspectral image processing based on spatial domain partitioning and `MPI` parallel interface have reached a high performance on clusters of mono-processors.[17] For these implementations, a pixel vector is always entirely assigned to a single processor, and slabs of spatially adjacent pixel vectors are distributed among the processing nodes of the parallel system. The inter-processor communication is reduced, resulting from the fact that a single pixel vector is never partitioned and communications are not needed at the pixel level. Modern clusters based on heterogeneous multi-core nodes can significantly reduce the run-time

[†]https://computing.llnl.gov/tutorials/pthreads/

of hyperspectral image analysis algorithms in general and the RX anomaly detection algorithm in particular. In order to exploit the modern clusters of multi-core nodes, a hybrid parallel implementation of the RX algorithm has been developed.[15] It combines the distributed and shared memory parallel programming mode, using the parallel interfaces `MPI` and `PThreads`. So, every multi-core node of the cluster can execute several `MPI` processes which expands a set of threads.

In this context, the idea is that several nodes with different number of cores (and possibly connected using an external network) cooperate together (in a balanced way) to solve the problem. The hybrid parallel RX for anomaly detection is a variation of this idea.[8,9] It uses a combination of global/local approaches for the calculation of the covariance matrix where each `MPI` process computes the covariance matrix of its local partition based on a multi-threaded computing model. Two levels of parallelism are considered in the hybrid parallel RX. The first level is based on spatial decomposition of the image cube, using the `MPI` parallel interface, so every `MPI` process computes the RX algorithm on its sub-image without communications, however the maximum number of `MPI` processes is bounded because the accuracy of the RX algorithm decreases as the size of the sub-image decreases. The second level lets us exploit the shared-memory multi-processor architecture. So each `MPI` process expands a set of threads by means of the `PThreads` parallel interface. Taking into account the data dependencies of the RX algorithm and the computational load of the operations, specific loops have been selected to be parallelized in order to improve the performance.

Let N be the number of `MPI` processes and T the number of `PThreads` that every `MPI` process executes. Thus, the computational complexity of the hybrid parallel algorithm at each step is:

1. Compute the covariance matrix, each `MPI` process computes the matrix K for its local sub-image exploiting the multi-thread parallelism, then the parallel complexity for this step is $O(\frac{rows \cdot columns \cdot bands^2}{N \cdot T})$.
2. Compute K^{-1} by the Gauss Method, using the Automatically Tuned Linear Algebra Software (ATLAS) library[‡] that exploits the multi-thread parallelism. So, every `MPI` process computes its inverse matrix without communication between processes. The parallel complexity is $O(\frac{bands^3}{T})$.
3. Each `MPI` process applies the RX filter based on the Mahalanobis distance (δ^{RX}) on its sub-image by means of a multi-threaded approach. Then, the parallel complexity is $O(\frac{rows \cdot columns \cdot bands^2}{N \cdot T})$.
4. In order to locate the targets, a master processor selects the pixels with higher associated value of $\delta^{(RX)}$, and these are used to define a final set of targets or anomalies. So, in this step there are some communications and synchronization points among processes. Its complexity is about $O(\frac{rows \cdot columns}{N})$.

However, the capability and accuracy of the hybrid parallel RX algorithm to detect anomalies depends on the number of sub-images in which the full image is decomposed. The value of this parameter is chosen by the user by means of the number of `MPI` processes, N_{mpi}, since it is considered as a input parameter whose upper limit is the number of processors of the architecture where the algorithm is computed.

Evaluation results have shown that the hybrid parallel RX based on a local strategy: provides similar accuracy results to those reported for the (global) one adopted in previous work, and can reduce its run-times exploiting the modern clusters of heterogeneous multi-core nodes.[15] Moreover, the hybrid parallel RX is portable to multi-core systems and it can also exploit this kind of platforms, which represent a low-power, inexpensive, widely available and well-known technology. In this context the MPI communications at low level are translated into shared memory accesses and synchronism points. The following section is intended to evaluate the hybrid parallel RX on multicore platforms.

4. EXPERIMENTAL RESULTS

4.1 Hyperspectral data used in experiments

The image scene used for experiments in this work was collected by the AVIRIS instrument, which was flown by NASA's Jet Propulsion Laboratory over the World Trade Center (WTC) area in New York City on September

[‡]http://math-atlas.sourceforge.net/

Figure 2. False color composition of an AVIRIS hyperspectral image collected by NASA's Jet Propulsion Laboratory over lower Manhattan on Sept. 16, 2001 (left). Location of thermal hot spots in the fires observed in World Trade Center area, available online: http://pubs.usgs.gov/of/2001/ofr-01-0429/hotspot.key.tgif.gif (right).

16, 2001, just five days after the terrorist attacks that collapsed the two main towers and other buildings in the WTC complex. The full data set selected for experiments consists of 614×512 pixels, 224 spectral bands and a total size of (approximately) 140 MB. The spatial resolution is 1.7 meters per pixel. The leftmost part of Fig. 2 shows a false color composite of the data set selected for experiments using the 1682, 1107 and 655 nm channels, displayed as red, green and blue, respectively. Vegetated areas appear green in the leftmost part of Fig. 2, while burned areas appear dark gray. Smoke coming from the WTC area (in the red rectangle) and going down to south Manhattan appears bright blue due to high spectral reflectance in the 655 nm channel. Extensive reference information, collected by U.S. Geological Survey (USGS), is available for the WTC scene[§]. In this work, we use a U.S. Geological Survey thermal map[¶] which shows the locations of the thermal hot spots (which can be seen as anomalies) at the WTC area, displayed as bright red, orange and yellow spots at the rightmost part of Fig. 2. The Figure is centered at the region where the towers collapsed, and the temperatures of the targets range from 700F to 1300F.

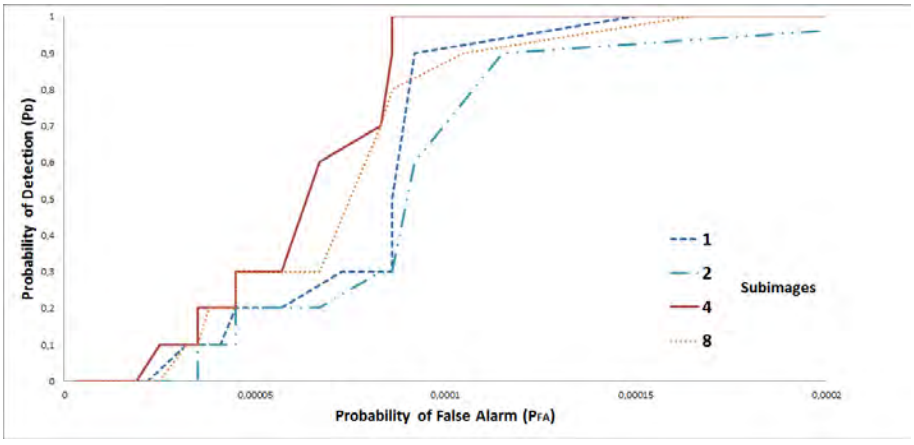

Figure 3. ROC curves obtained using 1, 2, 4 and 8 subimages.

[§]http://speclab.cr.usgs.gov/wtc
[¶]http://pubs.usgs.gov/of/2001/ofr-01-0429/hotspot.key.tgif.gif

4.2 Analysis of anomaly detection accuracy

In order to analyze the accuracy in anomaly detection by our proposed implementations, we use the thermal map displayed in the rightmost part of Fig. 2 as a ground-truth map. ROC curves[18] are used to measure such accuracy by measuring the probability of detection (P_D) rate versus the false alarm probability (P_{FA}) for several equidistant threshold values applied to the outcome of the RX algorithm δ^{RX}. This allows us to evaluate the performance of the RX filter independently of specific threshold values. Fig. 3 shows the ROC curves using 1, 2, 4 and 8 `MPI` processes (that is, applying the RX algorithm with 1, 2, 4 and 8 subimages) for processing the AVIRIS WTC image, where the false positive rate (Fig. 3) show ranges from 0 to 0.0002 (this means that we focus on the area of the ROC with very low false alarm probability). In ROC analysis, the area under the curve provides a reasonable estimate of anomaly detection accuracy. In Fig. 3, we can observe that the results obtained by the 4 and 8 subimages are slightly better than the result obtained by the global version using a single subimage or MPI process. This is mainly because the anomalies are located in a small area, thus, the process assigned for that partition can easily find the anomalies in the corresponding local partition. These observations are very important for the parallelization of the algorithm, which can greatly benefit from such local processing as it reduces inter-processor communications. These aspects will be explored in the following subsection devoted to analyzing parallel performance of the proposed implementations.

4.3 Analysis of parallel performance

The parallel computing platform used in experiments is the Dell PowerEdge R810, composed of 1 octo-core 1.87 GHz Intel Xeon L7555 (8 cores), with 16 Gb of main memory. The operating system used at the time of experiments was Debian, and `OpenMPI`[‖] was used as the interface for parallel programming. Although the selected parallel platform is based on a shared memory architecture, according to our experience `MPI` (in conjunction with `PThreads` can be effectively used to exploit this kind architecture and to adapt to the characteristic of our problem. Table 1 shows the execution time and the speedup for the pure multithreaded version, while Table 2 shows the execution time and the speedup for the pure `MPI` version. In the cases with `MPI` processes, a local approach is used for calculating the covariance matrix, while the pure multithreaded version uses a global approach. Nevertheless, we observed that the local strategy exhibited results which are almost identical to those obtained by the global strategy.

Table 1. Execution time (seconds) of the most relevant stages and speedup for our parallel implementation of the RX algorithm using `PThreads`.

Number of threads	Covariance Matrix	RX Filter	Total	Speedup
1	27,6	22,9	49,7	
2	14,1	11,1	25,2	1,9
4	7,9	5,6	13,6	3,6
8	6,6	2,5	9,1	5,4

Table 2. Execution time (seconds) of the most relevant stages and speedup for our parallel implementation of the RX algorithm using `MPI`.

Number of MPI processes	Covariance Matrix	RX Filter	Total	Speedup
1	27,6	22,1	49,7	
2	13,9	10,9	24,9	1,9
4	8,9	4,9	13,9	3,5
8	5,2	2,8	8,1	6,2

The results displayed on Tables 1 and 2 show that the two considered implementations (`PThreads` and `MPI`) achieve similar performance using 1, 2, 4 and 8 cores of the multicore system. However, it is worth noting that the `MPI` version offers slightly better performance with 8 processes, due to the use of a local strategy in the calculation of the covariance matrix. This implies that each process works with a portion of the image matrix. This strategy

[‖] http://www.open-mpi.org/

results in a less expensive computation, since the memory requirements to store hyperspectral images are large. We draw attention on the key fact that, in the developed parallel RX (which follows a local strategy), there are no communications among the processors. Such communications can severely penalize parallel performance when a global strategy is used, particularly as the number of MPI processes increase.

Table 3. Execution time (seconds) of the most relevant stages and speedup for the hybrid parallel implementation of the RX algorithm using both MPI and PTthreads.

MPI processes	PThreads	Cores	Covariance Matrix	RX Filter	Total	Speedup
1	**8**	**8**	6,6	2,5	9,1	5,4
2	**4**	**8**	4,1	2,4	6,5	7,6
4	**2**	**8**	4,8	2,4	7,3	6,8
8	**1**	**8**	5,2	2,8	8,1	6,2

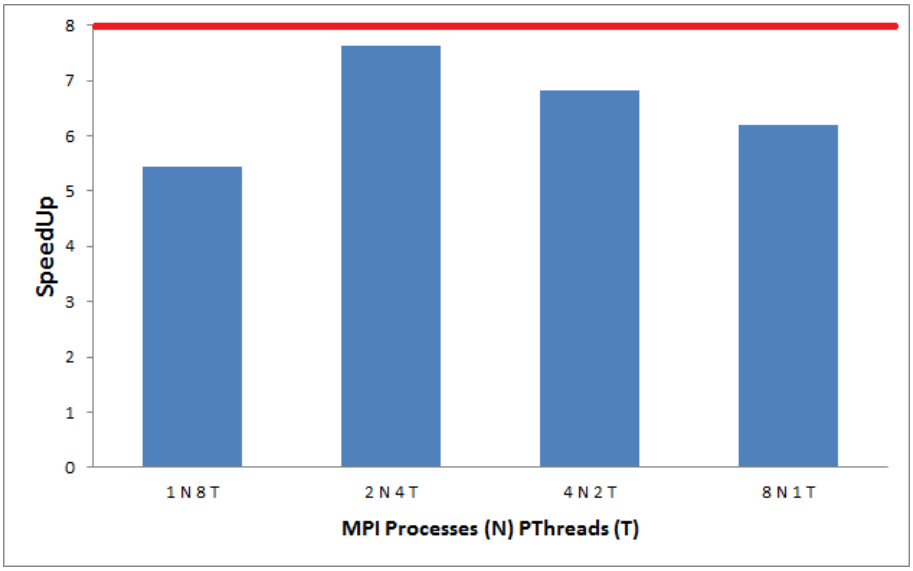

Figure 4. Speedup for the hybrid version (i.e., combining MPI and PThreads) over the cases in which only MPI or PThreads are used. The red line marks the ideal speedup for 8 cores.

For illustrative purposes, Table 3 shows the execution time and the measured speedup for the proposed (hybrid) parallel RX version using all available cores in our considered parallel platform. Here, we use different combinations of the number of MPI processes and PThreads. As shown by Fig. 4, the best results in terms of scalability are obtained by the hybrid version (i.e., combining MPI and PThreads) over the cases in which only MPI or PThreads are used. This confirms our introspection that the hybrid implementation can take advantage of two levels of parallelism to optimize the computing resources available in our considered platform. It also reveals the possibility to effectively used MPI in a shared memory architecture, which offers a new computing paradigm shown in this work to perform effectively in the context of anomaly detection applications in remotely sensed hyperspectral images.

5. CONCLUSIONS AND FUTURE RESEARCH LINES

In this paper, we have analyzed the anomaly detection accuracy and scalability of a new parallel implementation of the RX algorithm for hyperspectral image analysis. The proposed parallel algorithm uses a local approach for the calculation of the covariance matrix in parallel. It is shown to provide good anomaly detection results (which are not affected by the adopted partitioning strategy) as well as competitive advantages (in terms of scalability) with regards to the commonly used (global) strategies adopted for calculating the covariance matrix in parallel when implementing this algorithm. The parallel algorithm has been validated in the context of a real hyperspectral imaging application, focused on detecting the thermal hot spots (anomalies) of the fires in

the World Trade Center area in New York City, just few days after the terrorist attacks of September 11th, 2001. Our experimental results indicate that the proposed (local) strategy can even offer slight improvements in terms of anomaly detection accuracy with regards to those reported for the (global) strategy adopted in previous work. As the proposed (local) strategy reduces significantly the amount of inter-processor communications, the parallel version also scales significantly better than the global strategy. Although the results reported in this work are very encouraging, further experiments should be conducted in order to increase the scalability of the proposed parallel algorithms to a higher number of processors by resolving memory issues and optimizing the parallel design of such algorithms. Experiments with additional scenes under different target/anomaly detection scenarios are also highly desirable.

ACKNOWLEDGEMENT

This work has been funded by grants from the Spanish Ministry of Science and Innovation TIN2008-01117 and Junta de Andalucia (P08-TIC-3518, P10-TIC-6002), in part financed by the European Regional Development Fund (ERDF). The work has also been supported by the European Community's Marie Curie Research Training Networks Programme under reference MRTN-CT-2006-035927 (HYPER-I-NET). Funding from the Spanish Ministry of Science and Innovation (HYPERCOMP/EODIX project, reference AYA2008-05965-C04-02) and from Junta de Extremadura (PRI09A110 and GR10035 projects) are also very gratefully acknowledged.

REFERENCES

[1] Goetz, A. F. H., Vane, G., Solomon, J. E., and Rock, B. N., "Imaging spectrometry for Earth remote sensing," *Science* **228**, 1147–1153 (1985).

[2] Plaza, A., Benediktsson, J. A., Boardman, J., Brazile, J., Bruzzone, L., Camps-Valls, G., Chanussot, J., Fauvel, M., Gamba, P., Gualtieri, J., Marconcini, M., Tilton, J. C., and Trianni, G., "Recent advances in techniques for hyperspectral image processing," *Remote Sensing of Environment* **113**, 110–122 (2009).

[3] Green, R. O., Eastwood, M. L., Sarture, C. M., Chrien, T. G., Aronsson, M., Chippendale, B. J., Faust, J. A., Pavri, B. E., Chovit, C. J., Solis, M., et al., "Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (AVIRIS)," *Remote Sensing of Environment* **65**(3), 227–248 (1998).

[4] Chang, C.-I., [*Hyperspectral Imaging: Techniques for Spectral Detection and Classification.*], Norwell, MA: Kluwer (2003).

[5] Plaza, A. and Chang, C.-I., [*High performance computing in remote sensing*], CRC Press, Boca Raton (2007).

[6] Richards, J. A. and Jia, X., [*Remote Sensing Digital Image Analysis: An Introduction*], Springer (2006).

[7] Kwon, H. and Nasrabadi, N. M., "Hyperspectral anomaly detection using kernel rx-algorithm.," in [*International Conference on Image Processing (ICIP)*], (2004).

[8] Paz, A., Plaza, A., and Blazquez, S., "Parallel implementation of target and anomaly detection algorithms for hyperspectral imagery," *Proc. IEEE Geosci. Remote Sens. Symp.* **2**, 589–592 (2008).

[9] Paz, A., Plaza, A., and Plaza, J., "Comparative analysis of different implementations of a parallel algorithm for automatic target detection and classification of hyperspectral images," *Proc. SPIE* **7455**, 1–12 (2009).

[10] Plaza, A. and Chang, C.-I., [*High Performance Computing in Remote Sensing*], Taylor & Francis: Boca Raton, FL (2007).

[11] Plaza, A. and Chang, C.-I., "Special issue on high performance computing for hyperspectral imaging," *International Journal of High Performance Computing Applications* **22**(4), 363–365 (2008).

[12] Plaza, A., "Special issue on architectures and techniques for real-time processing of remotely sensed images," *Journal of Real-Time Image Processing* **4**(3), 191–193 (2009).

[13] Plaza, A., Du, Q., Y.-L-Chang, and King, R. L., "High performance computing for hyperspectral remote sensing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **4**(3) (2011).

[14] Plaza, A., Plaza, J., Paz, A., and Sanchez, S., "Parallel hyperspectral image and signal processing," *IEEE Signal Processing Magazine* **28**(3), 119–126 (2011).

[15] Molero, J., Paz, A., Garzón, E. M., Martínez, J., Plaza, A., and García, I., "Fast anomaly detection in hyperspectral images with RX method on heterogeneous clusters," *The Journal of Supercomputing* , 1–9 (2011). DOI:10.1007/s11227-011-0598-0.

[16] Reed, I. and X.Yu, "Adaptive multiple-band cfar detection of an optical pattern with unknown spectral distrihution.," *IEEE Trans. Acoustics, Speech and Signal Processing* **38**, 1760–1770 (1990).
[17] Plaza, A., Valencia, D., Plaza, J., and Martinez, P., "Commodity cluster-based parallel processing of hyperspectral imagery," *Journal of Parallel and Distributed Computing* **66**, 345–358 (2006).
[18] Manolakis, D., Marden, D., and Shaw, G. A., "Hyperspectral image processing for automatic target detection applications," *MIT Lincoln Laboratory Journal* **14**, 79–116 (2003).