A New Morphological Anomaly Detection Algorithm for Hyperspectral Images and its GPU Implementation

Abel Paz[a] and Antonio Plaza[a]

[a]Hyperspectral Computing Laboratory
Department of Technology of Computers and Communications
University of Extremadura, Avda. de la Universidad s/n
10071 Cáceres, Spain

ABSTRACT

Anomaly detection is considered a very important task for hyperspectral data exploitation. It is now routinely applied in many application domains, including defence and intelligence, public safety, precision agriculture, geology, or forestry. Many of these applications require timely responses for swift decisions which depend upon high computing performance of algorithm analysis. However, with the recent explosion in the amount and dimensionality of hyperspectral imagery, this problem calls for the incorporation of parallel computing techniques. In the past, clusters of computers have offered an attractive solution for fast anomaly detection in hyperspectral data sets already transmitted to Earth. However, these systems are expensive and difficult to adapt to on-board data processing scenarios, in which low-weight and low-power integrated components are essential to reduce mission payload and obtain analysis results in (near) real-time, i.e., at the same time as the data is collected by the sensor. An exciting new development in the field of commodity computing is the emergence of commodity graphics processing units (GPUs), which can now bridge the gap towards on-board processing of remotely sensed hyperspectral data. In this paper, we develop a new morphological algorithm for anomaly detection in hyperspectral images along with an efficient GPU implementation of the algorithm. The algorithm is implemented on latest-generation GPU architectures, and evaluated with regards to other anomaly detection algorithms using hyperspectral data collected by NASA's Airborne Visible Infra-Red Imaging Spectrometer (AVIRIS) over the World Trade Center (WTC) in New York, five days after the terrorist attacks that collapsed the two main towers in the WTC complex. The proposed GPU implementation achieves real-time performance in the considered case study.

Keywords: Hyperspectral imaging, morphological anomaly detection, parallel computing, graphics processing units (GPUs).

1. INTRODUCTION

The special properties of remotely sensed hyperspectral data have significantly expanded the domain of many analysis techniques, including (supervised and unsupervised) classification, spectral unmixing, compression, target and anomaly detection.[1–5] Specifically, the automatic detection of anomalies is highly relevant in many application domains,[6–8] including defense and security applications.[9,10] During the last few years, several algorithms have been developed for the aforementioned purposes, including the well-known RX algorithm developed by Reed and Xiaoli.[11] It is based on the application of a so-called RXD filter, given by the well-known Mahalanobis distance. Many other anomaly detection algorithms have also been proposed in the recent literature, using different concepts such as background modeling and characterization.[8,12]

Depending on the complexity and dimensionality of the input hyperspectral scene,[13] the anomaly detection task may be computationally expensive, a fact that limits the possibility of utilizing this technique in time-critical applications.[14] To address this issue, recent work has proposed to take advantage of the emergence of specialized hardware architectures such as commodity graphic processing units (GPUs),[15] which can now bridge the gap towards on-board processing of remotely sensed hyperspectral data in different applications.[10,16–24] The speed

Send correspondence to Antonio J. Plaza:
E-mail: aplaza@unex.es; Telephone: +34 927 257000 (Ext. 51662); URL: http://www.umbc.edu/rssipl/people/aplaza

Satellite Data Compression, Communications, and Processing VII, edited by Bormin Huang, Antonio J. Plaza, Carole Thiebaut, Proc. of SPIE Vol. 8157, 815708 · © 2011 SPIE · CCC code: 0277-786X/11/$18 · doi: 10.1117/12.892282

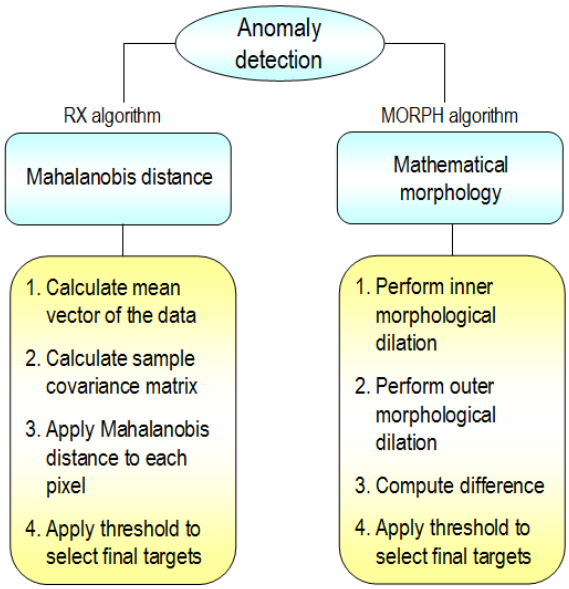


Figure 1. Summary of the RX and MORPH anomaly detection algorithms considered in this work.

of graphics hardware doubles approximately every six months, which is much faster than the improving rate of the CPUs (even those made up by multiple cores). The ever-growing computational requirements introduced by hyperspectral imaging applications can fully benefit from this type of specialized hardware and take advantage of the compact size and relatively low cost of these units, which make them appealing for onboard data processing at lower costs than those introduced by other hardware devices.[25]

In this paper we develop MORPH, a new morphological algorithm for anomaly detection in hyperspectral images which includes both spatial and spectral information. It adopts a morphological framework[26,27] that has been extended to hyperspectral imagery,[28–30] with the advantage that it considers jointly the two sources of information (spatial and spectral) when searching for anomalies in the hyperspectral data cube. An advantage of the algorithm is that it naturally appeals for parallel implementation in GPU devices.

The remainder of the paper is organized as follows. Section 2 describes the new MORPH algorithm as compared to the most widely used anomaly detection algorithm: the RX. Section 3 describes the parallel implementation of MORPH in GPUs. Section 4 evaluates the target detection accuracy and parallel performance of MORPH with regards to a previously developed parallel version of RX, using hyperspectral data collected by NASA's Airborne Visible Infra-Red Imaging Spectrometer (AVIRIS)[31] over the World Trade Center (WTC) in New York, five days after the terrorist attacks that collapsed the two main towers in the WTC complex. Finally, section 5 concludes with some remarks and hints at plausible future research.

2. ANOMALY DETECTION ALGORITHMS: MORPH VERSUS RX

Two anomaly detection algorithms are considered in this work: the well-known RX,[11] and the newly developed MORPH. The algorithms are summarized in Fig. 1. In the following, we describe each algorithm in step-by-step fashion.

2.1 RX algorithm

The RX algorithm has been widely used in signal and image processing.[11] This algorithm finds the pixel vectors which are spectrally distinct in a hyperspectral image $\mathbf{F}$ by applying the following steps:

1. Calculate the mean vector $\mathbf{m}$ of $\mathbf{F}$ using the following expression:

$$\mathbf{m} = \frac{1}{N \times M} \sum_{x=1}^{N} \sum_{y=1}^{M} [\mathbf{F}(x, y)], \tag{1}$$

where $\mathbf{F}(x,y)$ is the pixel vector at spatial coordinates (x,y), N is the total number of lines, and M is the total number of samples of the hyperspectral image $\mathbf{F}$.

2. Next, the algorithm computes the sample covariance matrix $\mathbf{K}_{n\times n}$, where n is the number of bands of the hyperspectral image $\mathbf{F}$, and then applies the following expression (which corresponds to the Mahalanobis distance):
$$\mathrm{RXD}(\mathbf{F}) = [\mathbf{F}(x,y) - \mathbf{m}]^T \mathbf{K}_{n\times n}^{-1} [\mathbf{F}(x,y) - \mathbf{m}]. \quad (2)$$

3. Resulting from the procedure above, each pixel receives a score that can be interpreted as the probability to represent an anomaly. In order to extract the final set of anomalies, a threshold value should be applied.

2.2 MORPH algorithm

The MORPH algorithm uses the concept of extended morphological operations.[28] These operations rely on a distance-based technique which utilizes a cumulative distance between one particular pixel vector $\mathbf{F}(x,y)$, and all the pixel vectors in the spatial neighborhood given by a structuring element denoted by S as follows:[29]

$$C(\mathbf{F}(x,y)) = \sum_{(s,t)\in S} \mathrm{SAD}(\mathbf{F}(x,y), \mathbf{F}(s,t)), \quad (3)$$

where SAD is the spectral angle distance. The SAD between two pixel vectors $\mathbf{F}(x,y)$ and $\mathbf{F}(s,t)$ is given by the following expression:

$$\mathrm{SAD}(\mathbf{F}(x,y), \mathbf{F}(s,t)) = cos^{-1}\left(\frac{\mathbf{F}(x,y)\cdot\mathbf{F}(s,t)}{\|\mathbf{F}(x,y)\|\cdot\|\mathbf{F}(s,t)\|}\right). \quad (4)$$

As a result, $C(\mathbf{F}(x,y))$ is given by the sum of SAD scores between $\mathbf{F}(x,y)$ and every other pixel vector in the neighborhood defined by the structuring element S. At this point, we need to be able to define a maximum and a minimum given an arbitrary set of vectors $\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k\}$, where k is the number of vectors in the set. This can be done by computing $C(\mathbf{V}) = \{C(\mathbf{v}_1), C(\mathbf{v}_2), \cdots, C(\mathbf{v}_k)\}$ and selecting $\mathbf{v}_i$ such that $C(\mathbf{v}_i)$ is the minimum of $C(\mathbf{V})$, with $1 \leq i \leq k$. In similar fashion, we can select $\mathbf{v}_j$ such that $C(\mathbf{v}_j)$ is the maximum of $C(\mathbf{V})$, with $1 \leq j \leq k$. Based on the definitions above, the extended erosion $\mathbf{F} \ominus S$ consists of selecting the pixel vector in the neighborhood given by S that produces the minimum value of C as follows:[28]

$$(\mathbf{F} \ominus S)(x,y) = argmin_{(s,t)\in S}\{C(\mathbf{F}(x+s, y+t)\}. \quad (5)$$

On the other hand, the extended dilation $\mathbf{F} \oplus S$ selects the pixel in the neighborhood given by S that produces the maximum value for C as follows:[28]

$$(\mathbf{F} \oplus S)(x,y) = argmax_{(s,t)\in S}\{C(\mathbf{F}(x-s, y-t)\}. \quad (6)$$

Based on the definitions above, the idea of the MORPH algorithm is to define two structuring elements, S_{internal} and S_{external}, around each pixel vector $\mathbf{F}(x,y)$. With this in mind, the algorithm performs a double morphological dilation operation for every pixel in the hyperspectral image, using first the internal window: $\mathbf{F} \oplus S_{\mathrm{internal}}(x,y)$, and then the external window: $\mathbf{F} \oplus S_{\mathrm{external}}(x,y)$. Once all the image pixels have been processed, the resulting values from the two dilation operations are compared with the original pixel $\mathbf{F}(x,y)$ using the SAD, accumulating the resulting values. The main goal of using a double dilation operation is to use the external window to emphasize even more the anomalies detected by the internal window. Since morphological dilation calculates a measure of eccentricity of the pixel under test with regards to the neighboring pixels as defined by the structuring element, this double dilation operation is expected to detect those pixels which are spectrally distinct with regards to their neighbors. As in the case of the RX algorithm, the outcome of this process is a score associated to each pixel that can be interpreted as the probability to represent an anomaly. In order to extract the final set of anomalies, a threshold value should be applied.

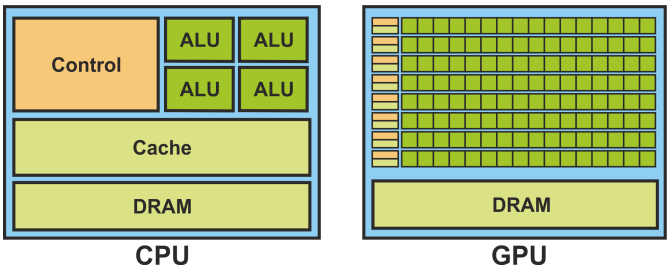


Figure 2. Comparison of CPU versus GPU architecture.

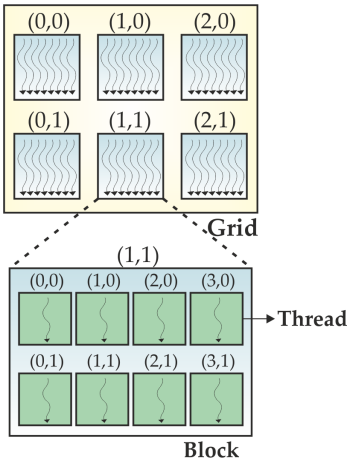


Figure 3. Processing in the GPU: grids made up of blocks with computing threads.

3. GPU IMPLEMENTATION OF THE MORPH ALGORITHM

In this section we develop a parallel version of the MORPH algorithm developed in the previous section. It should be noted that a GPU implementation for the RX algorithm is already available in the literature.[17] GPUs can be abstracted by assuming a much larger availability of processing cores than in standard CPU processing, with smaller processing capability of the cores and small control units associated to each core (see Fig. 2). Hence, the GPU is appropriate for algorithms that need to execute many repetitive tasks with fine grain parallelism and few coordination between tasks. In the GPU, algorithms are constructed by chaining so-called *kernels*, which define the minimum units of computations performed in the cores. Thereby, data-level parallelism is exposed to hardware, and kernels can be concurrently applied without any sort of synchronization. The kernels can perform a kind of batch processing arranged in the form of a grid of blocks, as displayed in Fig. 3, where each block is composed by a group of threads which share data efficiently through the shared local memory and synchronize their execution for coordinating accesses to memory. There is a maximum number of threads that a block can contain but the number of threads that can be concurrently executed is much larger (several blocks executed by the same kernel can be managed concurrently, at the expense of reducing the cooperation between threads since the threads in different blocks of the same grid cannot synchronize with the other threads). Finally, Fig. 4 shows the architecture of the GPU, which can be seen as a set of multiprocesors. Each multiprocessor is characterized by a single instruction multiple data (SIMD) architecture, i.e., in each clock cycle each processor of the multiprocessor executes the same instruction but operating on multiple data streams. Each processor has access to a local shared memory and also to local cache memories in the multiprocessor, while the multiprocessors have access to the global GPU (device) memory.

In order to implement MORPH in the GPU, we first define a processing grid given by as many blocks as pixel vectors are in the original hyperspectral image $\mathbf{F}$, and in which each block performs all the operations associated to the inner processing window S_{internal} centered around each pixel vector $\mathbf{F}(x, y)$ of the image. These operations are based on the calculation of the SAD distance between each pixel and the neighboring pixels within the window (these calculations are performed by the threads associated to each block, that will be as many as

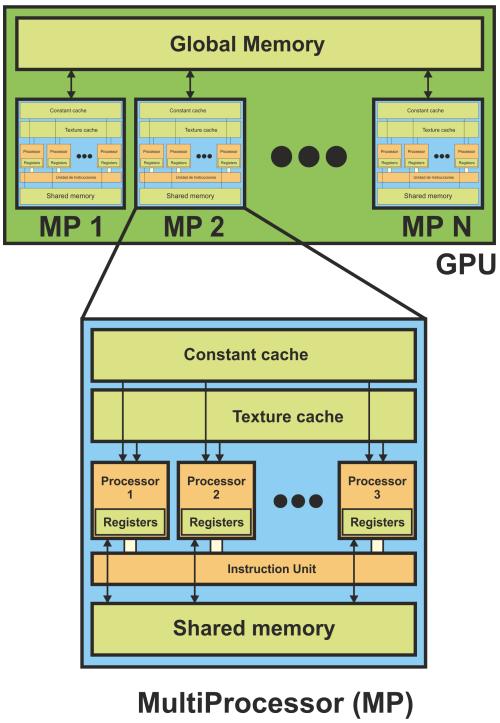


Figure 4. Hardware architecture of the GPU.

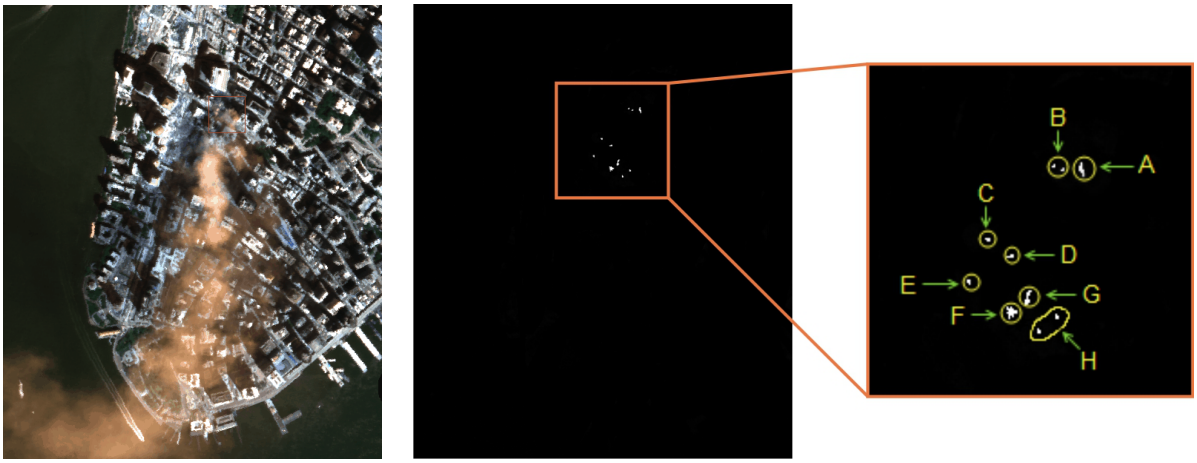
the number of pixels in the window). Once this process is finished, we perform the operations associated to the outer processing window S_{external} using a similar processing framework, and ultimately the SAD distance between the output of both operations is also calculated in the GPU. As it can be seen from the description above, the MORPH algorithm maps well in the GPU architecture as it is given by repetitive basic operations that can be applied with little synchronization. In our implementation we optimized the allocation of data to the local memories in order to guarantee the best possible performance.

4. EXPERIMENTAL RESULTS

The image scene used for experiments 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 16, 2001, just five days after the terrorist attacks that collapsed the two main towers and other buildings in the WTC complex. The data set consists of 512×614 pixels, 224 spectral bands, and a total size of (approximately) 140 MB. The spatial resolution is 1.7 meters per pixel. Fig. 5(a) shows a false color composite of the data set selected for experiments. Fig. 5(b) shows the locations of the thermal hot spots at the WTC area, which can be seen as anomalies in the scene. This information, available from U.S. Geological Survey (USGS) *, will be used in this work as ground-truth to validate the anomaly detection accuracy of the considered algorithms.

The GPU implementations of MORPH (developed in this work) and RX[17] were tested on a NVidia GeForce GTX 275 GPU, which features 240 processor cores operating at 1.550 GHz, 80 texture processing units, a 448-bit memory interface, and a 1792MB GDDR3 framebuffer at a 2520 MHz. It is based on the GT200 architecture. The GPU is connected to a CPU Intel Q9450 with 4 cores, which uses a motherboard ASUS Striker II NSE (with NVidia 790i chipset) and 4GB of RAM memory at 1333 MHz. Both the serial and GPU implementations of RX and MORPH obtained exactly the same results. Fig. 6 shows the receiver operating characteristic (ROC) curves used for evaluating the anomaly detection accuracy of RX and MORPH. These are graphical plots of the sensitivity, or true positive rate, versus the false positive rate obtained by the two methods in the detection of the ground-truth anomalies in Fig. 5 as the detection threshold involved in both algorithms is varied. The

*http://speclab.cr.usgs.gov/wtc



(a) (b)

Figure 5. (a) False color composition of the AVIRIS WTC scene. (b) Location of thermal hot spots.

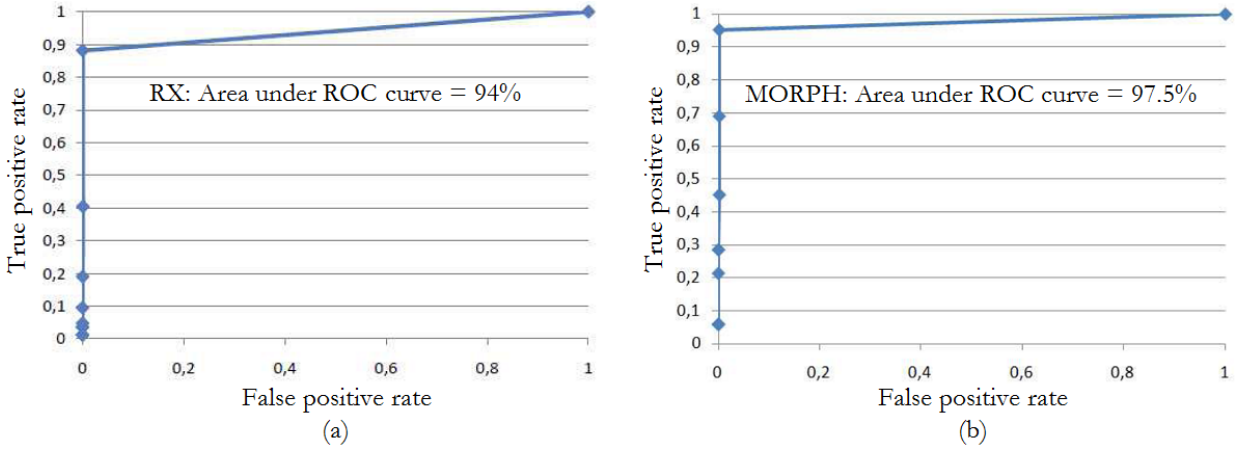


Figure 6. Receiver operating characteristics (ROC) curves for RX (a) and MORPH (b).

area under the ROC curve can be used as a metric of detection accuracy, and in our experiments it was around 94% for the RX algorithm and 97.5% for the MORPH algorithm, indicating better detection performance in the latter case. It should be noted that, for the MORPH algorithm, we optimized the input parameters by testing several configurations of the inner and outer processing windows, obtaining the best performance for an internal window S_{internal} of 3×3 pixels and an external window S_{external} of 5×5 pixels. The RX does not have any input parameters.

Regarding the computational performance of both implementations, Table 1 shows the processing times measured for the CPU and GPU versions of RX and MORPH, as well as the speedup achieved by each GPU version over its respective CPU counterpart. It should be noted that the CPU versions used all available cores. As shown by Table 1, the proposed GPU version of MORPH can still be optimized with regards to the GPU implementation of RX as the speedup achieved by the RX implementation in the GPU is higher, but in turn the GPU version of MORPH is able to provide a very low response time when processing the full AVIRIS WTC image. In fact, the processing time achieved by the GPU implementation of MORPH for this image (4293 milliseconds) is strictly in real-time as the cross-track line scan time in AVIRIS, a push-broom instrument,[31] is quite fast (8.3 milliseconds to collect 512 full pixel vectors). This introduces the need to process the considered AVIRIS WTC scene (512 lines and 614 samples) in less than 5096 milliseconds to fully achieve real-time performance. As a result, the proposed GPU implementation of MORPH opens innovative perspectives regarding the possibility to perform accurate and real-time analysis of hyperspectral data in the context of anomaly detection applications.

Algorithm	RX	MORPH
CPU	1995.152	78.016
GPU	24.512	4.293
Speedup	81.402	17.171

Table 1. Processing time (seconds) of the CPU and GPU implementations of RX and MORPH and speedups achieved by the GPU implementations over the CPU counterparts.

5. CONCLUSIONS AND FUTURE RESEARCH LINES

In this paper, we have developed a new morphological algorithm for hyperspectral images and its efficient implementation in GPU architectures. The algorithm includes both spatial and spectral information as opposed to other available techniques which only exploit spectral information. It allows adjusting the size of the targets to be detected by configuring an internal and an external processing window adopted for the morphological processing. The algorithm has been shown to map efficiently into GPU architectures, obtaining processing results in real-time for hyperspectral scenes collected by the AVIRIS sensor. Although the experimental results are quite promising, the parallel implementation in the GPU can still be further optimized in light of the obtained speedup factors, which in any case do not prevent the algorithm to perform in real-time. Our future work will be directed towards optimizing the GPU implementation of the algorithm. Further comparisons with other anomaly detection algorithms in different analysis scenarios will also be pursued.

ACKNOWLEDGEMENT

This work has 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 Junta de Extremadura (PRI09A110 and GR10035 projects) are also gratefully acknowledged.

REFERENCES

1. R. A. Schowengerdt, *Remote Sensing: Models and Methods for Image Processing, 2nd ed.*, Academic Press: New York, 1997.
2. D. A. Landgrebe, *Signal Theory Methods in Multispectral Remote Sensing*, John Wiley & Sons: New York, 2003.
3. J. A. Richards and X. Jia, *Remote Sensing Digital Image Analysis: An Introduction*, Springer, 2006.
4. C.-I. Chang, *Recent Advances in Hyperspectral Signal and Image Processing*, John Wiley & Sons: New York, 2007.
5. C.-I. Chang, *Hyperspectral Data Exploitation: Theory and Applications*, John Wiley & Sons: New York, 2007.
6. C.-I. Chang and H. Ren, "An experiment-based quantitative and comparative analysis of hyperspectral target detection and image classification algorithms for hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.* **2**, p. 1044.
7. H. Ren and C.-I. Chang, "Automatic spectral target recognition in hyperspectral imagery," *IEEE Trans. Aerosp. Electron. Syst.* **39**, pp. 1232–1249, 2003.
8. D. Manolakis, D. Marden, and G. A. Shaw, "Hyperspectral image processing for automatic target detection applications," *MIT Lincoln Laboratory Journal* **14**, pp. 79–116, 2003.
9. A. Paz, A. Plaza, and S. Blazquez, "Parallel implementation of target and anomaly detection algorithms for hyperspectral imagery," *Proc. IEEE Geosci. Remote Sens. Symp.* **2**, pp. 589–592, 2008.
10. Y. Tarabalka, T. V. Haavardsholm, I. Kasen, and T. Skauli, "Real-time anomaly detection in hyperspectral images using multivariate normal mixture models and gpu processing," *Journal of Real-Time Image Processing* **4**, pp. 1–14, 2009.
11. I. Reed and X.Yu, "Adaptive multiple-band cfar detection of an optical pattern with unknown spectral distrihution.," *IEEE Trans. Acoustics, Speech and Signal Processing* **38**, pp. 1760–1770, 1990.

12. N. Acito, M. Diani, and G. Corsini, "A new algorithm for robust estimation of the signal subspace in hyperspectral images in the presence of rare signal components," *IEEE Trans. Geosci. Remote Sens.* **47**(11), pp. 3844–3856, 2009.
13. A. Plaza and C.-I. Chang, *High performance computing in remote sensing*, CRC Press, Boca Raton, 2007.
14. A. Plaza, Q. Du, Y.-L-Chang, and R. L. King, "High performance computing for hyperspectral remote sensing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **4**(3), 2011.
15. C. A. Lee, S. D. Gasster, A. Plaza, C.-I. Chang, and B. Huang, "Recent developments in high performance computing for remote sensing: A review," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **4**(3), 2011.
16. A. Paz, A. Plaza, and J. Plaza, "Comparative analysis of different implementations of a parallel algorithm for automatic target detection and classification of hyperspectral images," *Proc. SPIE* **7455**, pp. 1–12, 2009.
17. A. Paz and A. Plaza, "Clusters versus GPUs for parallel automatic target detection in remotely sensed hyperspectral images," *EURASIP Journal of Advances in Signal Processing* **915639**, pp. 1–18, 2010.
18. C.-C. Chang, Y.-L. Chang, M.-Y. Huang, and B. Huang, "Accelerating regular LDPC code decoders on GPUs," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **4**(3), 2011.
19. J. Mielikainen, B. Huang, and A. Huang, "GPU-accelerated multi-profile radiative transfer model for the infrared atmospheric sounding interferometer," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **4**(3), 2011.
20. J. A. Goodman, D. Kaeli, and D. Schaa, "Accelerating an imaging spectroscopy algorithm for submerged marine environments using graphics processing units," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **4**(3), 2011.
21. E. Christophe, J. Michel, and J. Inglada, "Remote sensing processing: From multicore to GPU," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **4**(3), 2011.
22. S.-C. Wei and B. Huang, "GPU acceleration of predictive partitioned vector quantization for ultraspectral sounder data compression," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **4**(3), 2011.
23. H. Yang, Q. Du, and G. Chen, "Unsupervised hyperspectral band selection using graphics processing units," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **4**(3), 2011.
24. S. Sanchez, A. Paz, G. Martin, and A. Plaza, "Parallel unmixing of remotely sensed hyperspectral images on commodity graphics processing units," *Concurrency and Computation: Practice and Experience* **23**(12), 2011.
25. A. Plaza, J. Plaza, A. Paz, and S. Sanchez, "Parallel hyperspectral image and signal processing," *IEEE Signal Processing Magazine* **28**(3), pp. 119–126, 2011.
26. P. Soille, *Morphological image analysis: principles and applications*, Springer-Verlag, Berlin, 2003.
27. J. A. Benediktsson, J. A. Palmason, and J. R. Sveinsson, "Classification of hyperspectral data from urban areas based on extended morphological profiles," *IEEE Trans. Geosci. Remote Sens.* **42**, pp. 480–491, 2005.
28. A. Plaza, P. Martinez, R. Perez, and J. Plaza, "Spatial/spectral endmember extraction by multidimensional morphological operations," *IEEE Transactions on Geoscience and Remote Sensing* **40**(9), pp. 2025–2041, 2002.
29. A. Plaza, P. Martinez, J. Plaza, and R. Perez, "Dimensionality reduction and classification of hyperspectral image data using sequences of extended morphological transformations," *IEEE Trans. Geosci. Remote Sens.* **43**(3), pp. 466–479, 2005.
30. A. Plaza, J. A. Benediktsson, J. Boardman, J. Brazile, L. Bruzzone, G. Camps-Valls, J. Chanussot, M. Fauvel, P. Gamba, J. Gualtieri, M. Marconcini, J. C. Tilton, and G. Trianni, "Recent advances in techniques for hyperspectral image processing," *Remote Sensing of Environment* **113**, pp. 110–122, 2009.
31. R. O. Green, M. L. Eastwood, C. M. Sarture, T. G. Chrien, M. Aronsson, B. J. Chippendale, J. A. Faust, B. E. Pavri, C. J. Chovit, M. Solis, *et al.*, "Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (AVIRIS)," *Remote Sensing of Environment* **65**(3), pp. 227–248, 1998.