Hyperspectral Unmixing on GPUs and Multi-Core Processors: A Comparison

Sergio Bernabé, *Student Member, IEEE*, Sergio Sánchez, Antonio Plaza, *Senior Member, IEEE*, Sebastián López, *Member, IEEE*, Jón Atli Benediktsson, *Fellow, IEEE*, and Roberto Sarmiento

***Abstract*—One of the main problems in the analysis of remotely sensed hyperspectral data cubes is the presence of mixed pixels, which arise when the spatial resolution of the sensor is not able to separate spectrally distinct materials. Due to this reason, spectral unmixing has become one of the most important tasks for hyperspectral data exploitation. However, unmixing algorithms can be computationally very expensive, a fact that compromises their use in applications under real-time constraints. For this purpose, in this paper we develop two efficient implementations of a full hyperspectral unmixing chain on two different kinds of high performance computing architectures: graphics processing units (GPUs) and multi-core processors. The proposed full unmixing chain is composed for three stages: (i) estimation of the number of pure spectral signatures or *endmembers*, (ii) automatic identification of the estimated endmembers, and (iii) estimation of the fractional abundance of each endmember in each pixel of the scene. The two computing platforms used in this work are inter-compared in the context of hyperspectral unmixing applications. The GPU implementation of the proposed methodology has been implemented using the compute devide unified architecture (CUDA) and the `cuBLAS` library, and tested on two different GPU architectures: NVidia™ GeForce GTX 580 and NVidia™ Tesla C1060. It provides real-time unmixing performance in two different analysis scenarios using hyperspectral data collected by NASA's Airborne Visible Infra-Red Imaging Spectrometer (AVIRIS) over the Cuprite mining district in Nevada and the World Trade Center complex in New York City. The multi-core implementation, developed using the applications program interface (API) `OpenMP` and the Intel Math Kernel Library (MKL) used for matrix multiplications, achieved near real-time performance in the same scenarios. A comparison of both architectures in terms of performance, cost and mission payload considerations is given based on the results obtained in the two considered data analysis scenarios.**

***Index Terms*—Hyperspectral imaging, spectral unmixing, high performance computing, GPUs, multi-core platforms.**

Manuscript received September 30, 2012; revised January 04, 2013; accepted February 27, 2013. Date of publication April 18, 2013; date of current version June 17, 2013.

S. Bernabé is with the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura, E-10003 Cáceres, Spain, and also with the Faculty of Electrical and Computer Engineering, University of Iceland, 101 Reykjavik, Iceland (e-mail: sergiobernabe@unex.es, sbg30@hi.is).

S. Sánchez and A. Plaza are with the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura, E-10003 Cáceres, Spain (e-mail: sersanmar@unex.es, aplaza@unex.es).

J. A. Benediktsson is with the Faculty of Electrical and Computer Engineering, University of Iceland, 101 Reykjavik, Iceland (e-mail: benedikt@hi.is).

S. López and R. Sarmiento are with the Institute for Applied Microelectronics, University of Las Palmas de Gran Canaria, 35017 Tafira Baja, Spain (e-mail: seblopez@iuma.ulpgc.es, roberto@iuma.ulpgc.es).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSTARS.2013.2254470

I. Introduction

HYPERSPECTRAL imaging instruments are capable of collecting hundreds of images, corresponding to different wavelength channels, for the same area on the surface of the Earth [1]. For instance, NASA is continuously gathering imagery data with instruments such as the Jet Propulsion Laboratory's AVIRIS, which is able to record the visible and near-infrared spectrum (wavelength region from 0.4 to 2.5 micrometers) of reflected light in an area 2 to 12 kilometers wide and several kilometers long, using 224 spectral bands [2]. One of the main problems in the analysis of hyperspectral data cubes is the presence of mixed pixels [3], [4], which arise when the spatial resolution of the sensor is not fine enough to separate spectrally distinct materials. In this case, several spectrally pure signatures (endmembers) are combined into the same (mixed) pixel. Spectral unmixing [5], [6] involves the separation of a pixel spectrum into its endmember spectra, and the estimation of the abundance value for each endmember [7], [8]. A popular approach for this purpose in the literature has been linear spectral unmixing, which assumes that the endmembers substances are sitting side-by-side within the field of view of the imaging instrument [see Fig. 1(a)]. On the other hand, the nonlinear mixture model [9]–[11] assumes nonlinear interactions between endmember sustances [see Fig. 1(b)]. In practice, the linear model is more flexible and can be easily adapted to different analysis scenarios. Let $\mathbf{y}$ be a pixel vector given by a collection of values at different wavelengths. In the context of linear spectral unmixing, such vector can be modeled as:

$$\mathbf{y} \approx \mathbf{M}\alpha + \mathbf{n} = \sum_{i=1}^{p} \mathbf{e}_i \alpha_i + \mathbf{n}, \tag{1}$$

where $\mathbf{M} = \{\mathbf{e}_i\}_{i=1}^{p}$ is a matrix containing p endmember signatures, $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_p]$ is a p-dimensional vector containing the abundance fractions for each of the p endmembers in $\mathbf{M}$, and $\mathbf{n}$ is a noise term. The spectral unmixing chain considered in this work comprises three steps (see Fig. 2): 1) estimation of the number of pure spectral signatures (*endmembers*), p, in the hyperspectral scene; 2) identifying a collection of $\mathbf{M} = \{\mathbf{e}_i\}_{i=1}^{p}$ endmembers, and 3) estimating the abundances, in which the fractional coverage of each endmember is estimated for each pixel. The estimation error can be computed by reconstructing the original image (using the extracted endmembers and the derived abundances) and comparing the reconstructed image with the original one.

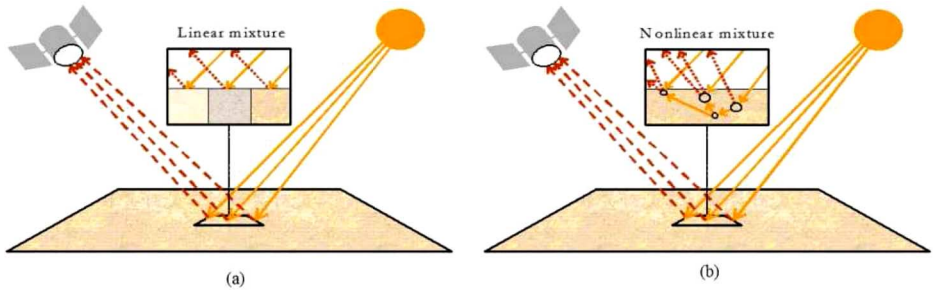


Fig. 1. Linear (a) versus nonlinear (b) mixture models in remotely sensed hyperspectral imaging. (a) Single scattering, (b) Multiple scattering.

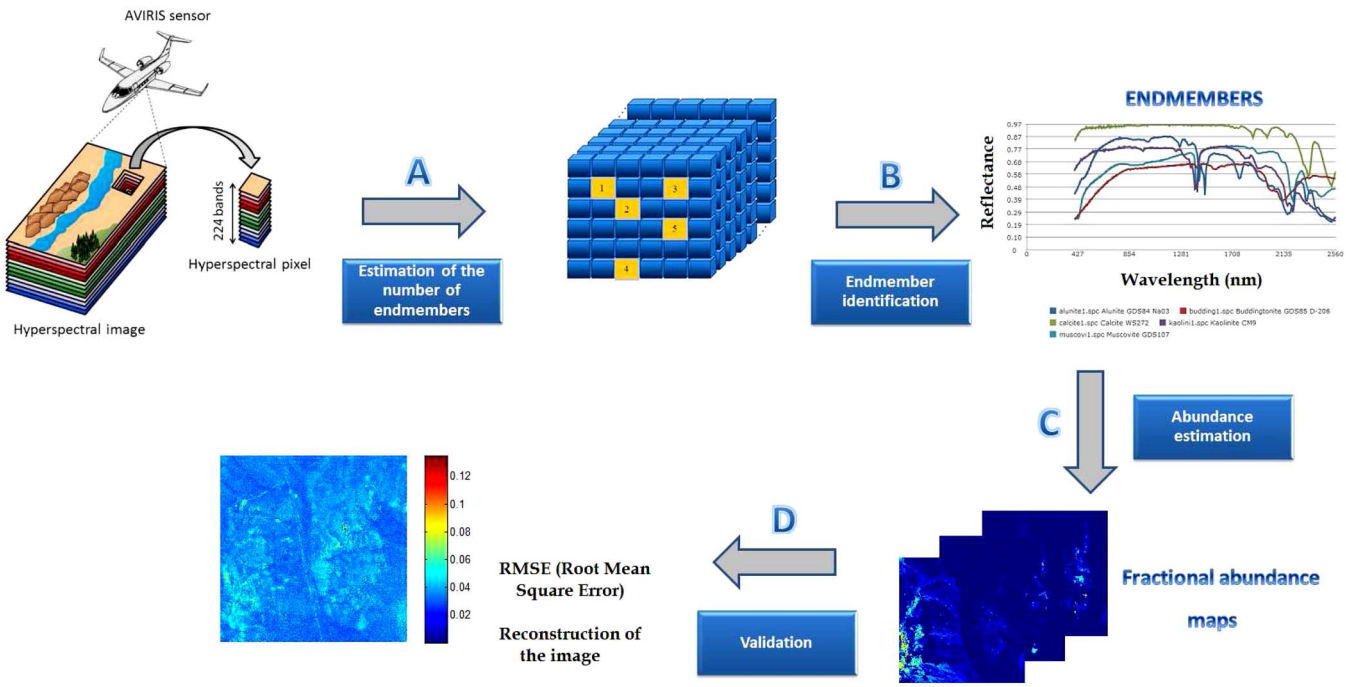


Fig. 2. Block diagram illustrating the full hyperspectral unmixing chain considered in this work.

In recent years, several techniques have been proposed to solve the aforementioned problem under the linear mixture model assumption (see [12]–[17], among several others), but all of them are quite expensive in computational terms. Although these techniques map nicely to high performance computing systems such as commodity clusters [18], these systems are difficult to apdapt to on-board processing requirements introduced by applications with real-time constraints such as wild land fire tracking, biological threat detection, monitoring of oil spills and other types of chemical contamination [19], [20]. In those cases, low-weight integrated components such as field programmable gate arrays (FPGAs) [21], [22], which offer a good compromise in terms of mission payload, have revealed as a feasible option, but one that generally requires a significant effort from the design and programmability point of view. Possible alternatives are multi-core processors [23] and commodity graphics processing units (GPUs) [24], which offer highly relevant computational power at low cost, this offering the opportunity to bridge the gap towards real-time analysis of remotely sensed hyperspectral data [25]–[27].

In this paper, we develop two computationally efficient implementations of a full hyperspectral unmixing chain on GPUs and multi-core processors. Although previous work has discussed the implementation of unmixing algorithms on GPUs [28], the implementation of a full hyperspectral unmixing chain on multi-core processors has not been discussed in previous contributions, to the best of our knowledge. The two considered platforms are inter-compared in the context of hyperspectral unmixing applications. In our comparisons, we have selected both domestic (e.g., multi-core processor i7 and NVidia GeForce GTX 580 GPU) and professional platforms (e.g., multi-core Xeon processor and NVidia Tesla C1060 GPU). Our study reveals that GPUs and multi-core processors can provide real-time unmixing performance. The implementations on GPUs have been carried out using NVidia CUDA and the `cuBLAS` library[1], an implementation of `BLAS` (basic linear algebra subprograms) on top of NVidia™ CUDA, while the multi-core implementations have been developed using

[1]http://developer.nvidia.com/cuBLAS

the API OpenMP[2] and Intel's Math Kernel Library (MKL)[3]. The remainder of the paper is organized as follows. Section II describes the different modules that conform to the proposed unmixing chain. Sections III and IV describe the GPU and multi-core implementations of these modules, respectively. Section V presents an experimental evaluation of the proposed implementations in terms of both unmixing accuracy and parallel performance, using two different hyperspectral scenes with reference data and collected by AVIRIS. Finally, Section VI concludes the paper with some remarks and hints at plausible future research lines.

II. Unmixing Chain Algorithms

A. Virtual Dimensionality (VD) Algorithm for Estimation of the Number of Endmembers

Let us denote by $\mathbf{Y} \equiv [y_1, y_2, \ldots, y_N]$ a hyperspectral image with N pixel vectors, each with L spectral bands. The VD first calculates the eigenvalues of the covariance matrix $\mathbf{K}_{L\times L} = 1/\mathrm{N}(\mathbf{Y} - \bar{\mathbf{Y}})^T(\mathbf{Y} - \bar{\mathbf{Y}})$ and the correlation matrix $\mathbf{R}_{L\times L} = \mathbf{K}_{L\times L} + \overline{\mathbf{Y}\mathbf{Y}}^T$, respectively referred to as covariance-eigenvalues and correlation-eigenvalues, for each of the spectral bands in the original hyperspectral image $\mathbf{Y}$. If a distinct spectral signature makes a contribution to the eigenvalue-represented signal energy in one spectral band, then its associated correlation eigenvalue will be greater than its corresponding covariance-eigenvalue in this particular band. Otherwise, the correlation eigenvalue would be very close to the covariance-eigenvalue, in which case only noise would be present in this particular band. By applying this concept, a Neyman-Pearson detector [29] is introduced to formulate the issue of whether a distinct signature is present or not in each of the spectral bands of $\mathbf{Y}$ as a binary hypothesis testing problem, where a so-called Neyman-Pearson detector is generated to serve as a decision maker based on a prescribed P_F (i.e., false alarm probability). In light of this interpretation, the issue of determining an appropriate estimation $\hat{p}$ for the number of endmembers is further simplified and reduced to a specific value of P_F that is preset by the Neyman-Pearson detector.

[2]http://openmp.org

[3]http://software.intel.com/en-us/intel-mkl

B. Orthogonal Subspace Projection With Gram-Schmidt Orthogonalization (OSP-GS) for Endmember Extraction

The orthogonal subspace projection (OSP) algorithm [12] was originally developed to find spectrally distinct signatures using orthogonal projections. For this work, we have used an optimization of this algorithm (see [30], [31]) which allows calculating the OSP without requiring the computation of the inverse of the matrix that contains the endmembers already identified in the image. This operation, which is difficult to implement in parallel, is accomplished using the Gram-Schmidt method for orthogonalization. This process selects a finite set of linearly independent vectors $\mathbf{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_p\}$ in the inner product space $\mathbf{R}^L$ in which the original hyperspectral image is defined, and generates an orthogonal set of vectors $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_p\}$ which spans the same *p-dimensional* subspace of $\mathbf{R}^L (p \leq L)$ as $\mathbf{A}$. In particular, $\mathbf{B}$ is obtained as follows: where the projection operator is defined in (3), in which $\langle \mathbf{a}, \mathbf{b} \rangle$ denotes the inner product of vectors $\mathbf{a}$ and $\mathbf{b}$.

$$\mathrm{proj}_{\mathbf{b}}(\mathbf{a}) = \frac{\langle \mathbf{a}, \mathbf{b} \rangle}{\langle \mathbf{b}, \mathbf{b} \rangle}\mathbf{b}. \tag{3}$$

The sequence $\mathbf{b}_1, \ldots, \mathbf{b}_p$ in (2) represents the set of orthogonal vectors generated by the Gram-Schmidt method, and thus, the normalized vectors $\mathbf{e}_1, \ldots, \mathbf{e}_p$ in (2) form an orthonormal set. As far as $\mathbf{B}$ spans the same p-dimensional subspace of $\mathbf{R}^L$ as $\mathbf{A}$, an additional vector $\mathbf{b}_{p+1}$ computed by following the procedure stated at (2) is also orthogonal to all the vectors included in $\mathbf{A}$ and $\mathbf{B}$. This algebraic assertion constitutes the cornerstone of the OSP method with Gram-Schmidt orthogonalization, referred to hereinafter as OSP-GS algorithm.

C. Unconstrained Least-Squares (UCLS) Algorithm for Abundance Estimation

Once the set of endmembers $\mathbf{M} = \{\mathbf{e}_i\}_{i=1}^p$ has been identified, their correspondent abundance fractions $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_p]$ in a specific, L-dimensional pixel vector $\mathbf{y}$ of the scene can be simply estimated (in least squares sense) by the following unconstrained expression:

$$\alpha = (\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T\mathbf{y}. \tag{4}$$

Two additional constraints can be imposed into the model described in (4), these are the abundance non-negativity constraint (ANC), i.e., $\alpha_i \geq 0$, and the abundance sum-to-one constraint (ASC), i.e., $\sum_{i=1}^p \alpha_i = 1$. However, in this work we focus on the unconstrained estimation only as it is much faster and

$$\begin{aligned}
&\mathbf{b}_1 = \mathbf{a}_1, && \mathbf{e}_1 = \tfrac{\mathbf{b}_1}{\|\mathbf{b}_1\|}\\
&\mathbf{b}_2 = \mathbf{a}_2 - \mathrm{proj}_{\mathbf{b}_1}(\mathbf{a}_2), && \mathbf{e}_2 = \tfrac{\mathbf{b}_2}{\|\mathbf{b}_2\|}\\
&\mathbf{b}_3 = \mathbf{a}_3 - \mathrm{proj}_{\mathbf{b}_1}(\mathbf{a}_3) - \mathrm{proj}_{\mathbf{b}_2}(\mathbf{a}_3), && \mathbf{e}_3 = \tfrac{\mathbf{b}_3}{\|\mathbf{b}_3\|}\\
&\mathbf{b}_4 = \mathbf{a}_4 - \mathrm{proj}_{\mathbf{b}_1}(\mathbf{a}_4) - \mathrm{proj}_{\mathbf{b}_2}(\mathbf{a}_4) - \mathrm{proj}_{\mathbf{b}_3}(\mathbf{a}_4), && \mathbf{e}_4 = \tfrac{\mathbf{b}_4}{\|\mathbf{b}_4\|}\\
&\vdots && \vdots\\
&\mathbf{b}_p = \mathbf{a}_p - \textstyle\sum_{j=1}^{p-1} \mathrm{proj}_{\mathbf{b}_j}(\mathbf{a}_p), && \mathbf{e}_p = \tfrac{\mathbf{b}_p}{\|\mathbf{b}_p\|},
\end{aligned} \tag{2}$$

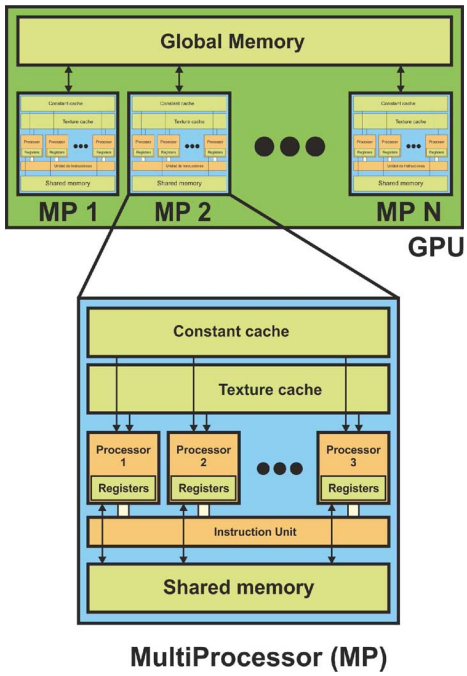


Fig. 3. Schematic overview of a GPU architecture, which can been seen as a set of multiprocessors (MPs).

it has been shown in practice to provide satisfactory results if the model endmembers are properly selected.

III. GPU Implementation

GPUs can be abstracted in terms of a stream model, under which all data sets are represented as streams (i.e., ordered data sets). Fig. 3 shows the architecture of a GPU, which can be seen as a set of multiprocessors (MPs). Each multiprocessor is characterized by a single instruction multiple data (SIMD) architecture, i.e., in each clock cycle each processor 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. Algorithms are constructed by chaining so-called *kernels* which operate on entire streams and which are executed by a multiprocessor, taking one or more streams as inputs and producing one or more streams as outputs. 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 (see Fig. 4), 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. As a result, there are different levels of memory in the GPU for the thread, block and grid concepts (see Fig. 5). There is also 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). With

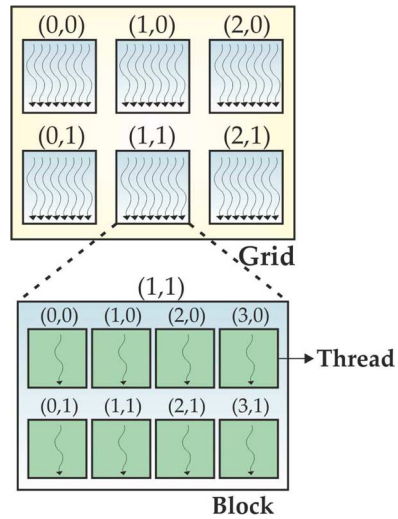


Fig. 4. Batch processing in the GPU: grids of blocks of threads, where each block is composed by a group of threads.

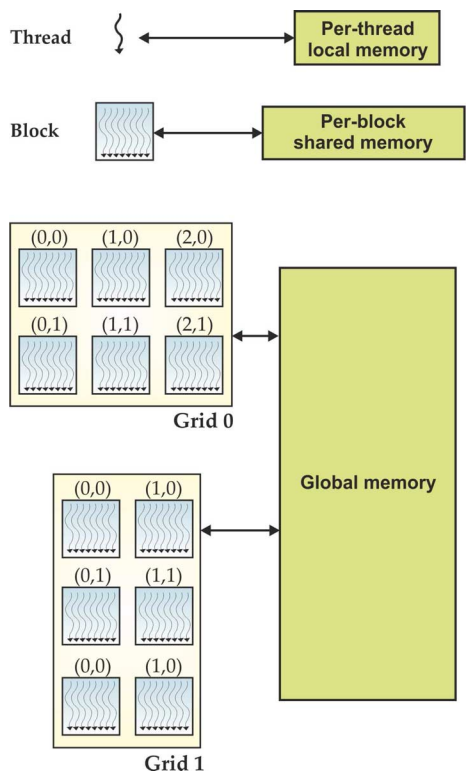


Fig. 5. Different levels of memory in the GPU for the thread, block and grid concepts.

the above ideas in mind, our GPU implementation of the hyperspectral unmixing chain comprises three stages: 1) GPU implementation of VD; 2) GPU implementation of OSP-GS; and 3) GPU implementation of UCLS.

A. GPU Implementation of VD

Once we load the full hyperspectral image $\mathbf{Y}$ pixel by pixel from disk to the main memory of the GPU, the first step is to calculate the covariance matrix $\mathbf{K}_{L\times L}$. For this purpose, we need to calculate the mean value $\bar{\mathbf{Y}}$ of each band of the image and substract this mean value to all the pixels in the same band.

To perform this calculation in the GPU, we use a kernel called *mean_pixel* configured with as many blocks as the number of bands L in the hyperspectral image. In each block, all available threads perform a reduction process using shared memory and coalesced memory accesses to add the values of all the pixels in the same band. Once this process is completed, another thread divides the computed value by the number of pixels in the original image, N, and the mean value is obtained. The resulting mean values of each band $\bar{Y}$ are stored in a structure as they will be needed for the calculation of the covariance matrix $\mathbf{K}_{L\times L}$ in the GPU by means of a matrix multiplication operation $(\mathbf{Y}-\bar{\mathbf{Y}})^T(\mathbf{Y}-\bar{\mathbf{Y}})$. This operation is performed using the `cuBLAS` library. Specifically, we use the `cublasSgemm` function of `cuBLAS`. The next step is to calculate the correlation matrix $\mathbf{R}_{L\times L}$ in the GPU. To achieve this, we use a kernel *correlation* which launches as many threads as elements in $\mathbf{R}_{L\times L}$, where each thread computes an element of the resulting matrix as follows: $\mathbf{R}_{ij} = \mathbf{K}_{ij} + \bar{\mathbf{Y}}_i\,\bar{\mathbf{Y}}_j$. Finally, we have observed that the remaining steps in the VD calculation (i.e., extraction of correlation-eigenvalues, covariance-eigenvalues and Neyman-Pearson test for estimation of the number of endmembers) can be computed very fast in the CPU.

B. GPU Implementation of OSP-GS

On the other hand, our implementation of OSP-GS for GPUs can be summarized by the following steps:

1) The first step is related with the proper arrangement of the hyperspectral data in the local GPU memory. In order to optimize accesses, we store the pixel vectors of the hyperspectral image $\mathbf{Y}$ by columns. Our arrangement is intended to access consecutive wavelength values in parallel by the processing kernels (coalesced accesses to memory). This technique is used to maximize global memory bandwith and minimize the number of bus transactions. Then, a structure is created in which the number of blocks equals the number of pixel vectors in the hyperspectral image divided by the number of threads per block, where the maximum number of supported threads depends on the considered GPU architecture. A kernel called `get_pixel_max_bright` is now used to calculate the brightest pixel $\mathbf{e}_1$ in $\mathbf{Y}$. This kernel computes (in parallel) the dot product between each pixel vector and its own transposed version, retaining the pixel that results in the maximum projection value.
2) Once the brightest pixel in $\mathbf{Y}$ has been identified, the pixel is allocated as the first column in matrix $\mathbf{M}$. The algorithm now calculates the orthogonal vectors through the Gram-Schmidt method as detailed in (2). This operation is performed in the CPU beacause this method operates on a small data structure and the results can be obtained very quickly. A new kernel is created, in which the number of blocks equals the number of pixel vectors in the hyperspectral image divided by the number of supported threads depends on the considered GPU architecture. This kernel, called `pixelProjection`, is now applied to project the orthogonal vector onto each pixel in the image. An important optimization applied at this point involves the effective use of the shared memories. We use these memories to store the most orthogonal vectors obtained at each iteration of OSP-GS (this is because these vectors will be accessed every time that the projection onto each pixel of the image is performed). The maximum of all projected pixels, is calculated using a separate reduction kernel `reductionProjection` which also uses the shared memory to store each of the projections and obtains the new endmember $\mathbf{e}_2$.
3) The algorithm now extends the endmember matrix as $\mathbf{M} = [\mathbf{e}_1, \mathbf{e}_2]$ and repeats from step 2) until the desired number of endmembers (specified by the input parameter p) has been extracted. The output of the algorithm is a set of endmembers $\mathbf{M} = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_p]$.

C. GPU Implementation of UCLS

Finally, our GPU implementation of UCLS can be summarized by the following steps:

1) The first step is to calculate the operation $(\mathbf{M}^T\mathbf{M})$, where $\mathbf{M} = \{\mathbf{e}_i\}_{i=1}^{p}$ is formed by the p endmembers extracted by the OSP-GS. This is performed by a kernel called `MtxM`. The inverse of this operation is calculated in the CPU mainly due to two reasons: i) its computation is relatively fast, and ii) the inverse operation remains the same throughout the whole execution of the code. A new kernel called `Mxinv` is now used to multiply $\mathbf{M}$ by its inverse.
2) The result calculated in the previous step is now multiplied by each pixel $\mathbf{y}$ in the hyperspectral image, thus obtaining a set of abundance vectors $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_p]$, each containing the fractional abundances of the p endmembers in $\mathbf{M}$. This is accomplished in the GPU by means of a specific kernel, called `get_abundances`, which produces p abundance maps.

IV. Multi-Core Implementation

A multi-core processor is a single CPU with two or more independent processors (called *cores*), which are the units that read and execute program instructions. The multiple cores can run multiple instructions at the same time, increasing the overall speed for programs amenable to parallel computing. In our implementation of the considered unmixing chain, we used `OpenMP` which is an API used to explicity address multithreaded, shared-memory parallelism. In `OpenMP` the users specify the regions in the code that are suitable for parallel implementation. The user also specifies necessary synchronization operations, such as locks or barriers, to ensure correct execution of the parallel region. At runtime the threads are executed in different processors but sharing the same memory and address space. In the following, we briefly summarize the main techniques used in the multi-core implementation of the considered unmixing chain:

- In our unmixing chain, we have several matrix multiplications. Mainly in the VD algorithm, we have a multiplication of high dimensionality in the *covariance* operation (the most expensive one of the algorithm). Our idea is based in [32], where we include the matrix multiplication routine using two levels of parallelim (`OpenMP+BLAS`). The first level of parallelism is set using the API `OpenMP` and the second level is achieved by invoking the *dgemm*

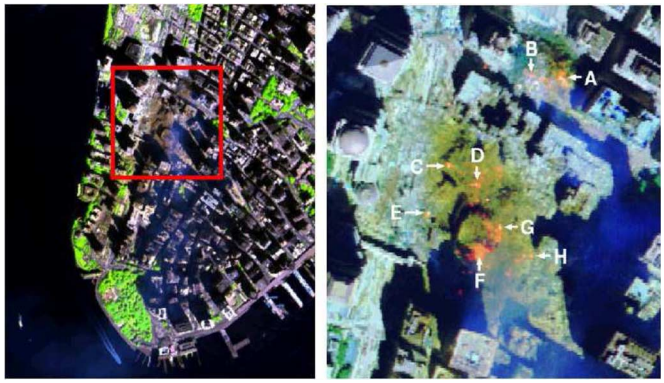


Fig. 6. False color composition of an AVIRIS hyperspectral image collected by NASAs Jet Propulsion Laboratory over lower Manhattan on Sept. 16, 2001 (left). Location of thermal hot spots in the fires observed in World Trade Center area, available online: http://pubs.usgs.gov/of/2001/ofr-01-0429/hotspot.key.tgif.gif (right).

routine of a multithreading implementation of `BLAS` (the MKL library of optimized math routines has been used in the experiments).

- One of the main techniques adopted in the `OpenMP` implementation of the considered unmixing chain is the use of locking routines. For instance, these routines are used in the OSP-GS algorithm to calculate the brighest pixel in the scene and the maximum projection operation.
- Another strategy adopted in the `OpenMP` implementation of the considered unmixing chain is the use of `parallel for` directives, which indicate the compiler that the structured block of code should be executed in parallel on multiple threads. Each thread will execute the same instruction stream, however not necessarily the same set of instructions. These directives are used with the locking routines and to implement the different steps for the UCLS algorithm in Section III.

V. Experimental Results

A. Hyperspectral Image Data

The first data set used in our experiments was collected by the AVIRIS sensor 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 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. 6 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. 6, 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 online for the WTC scene[4]. The rightmost part of Fig. 6 shows a USGS thermal map depicting the target locations of the thermal hot spots at the WTC area. The map is centered at the region where the towers collapsed, and the temperatures of the targets range from 700 F to 1300 F. These targets will be used as reference information for validation purposes in our comparison.

A second hyperspectral image scene has been considered for experiments. It is the well-known AVIRIS Cuprite scene [see Fig. 7(a)], collected by the Airborne Visible Infra-Red Imaging Spectrometer (AVIRIS) [2] in the summer of 1997 and available online in reflectance units after atmospheric correction[5]. The portion used in experiments corresponds to a 350×350-pixels subset of the sector labeled as f970619t01p02_r02_sc03.a.rfl in the online data, which comprises 188 spectral bands in the range from 400 to 2500 nm and a total size of around 50 MB. Water absorption bands as well as bands with low signal-to-noise ratio (SNR) were removed prior to the analysis. The site is well understood mineralogically, and has several exposed minerals of interest, including *alunite*, *buddingtonite*, *calcite*, *kaolinite*, and *muscovite*. Reference ground signatures of the above minerals, displayed in Fig. 7(b) and available in the USGS library,[6] will be used in this work for evaluation purposes.

B. Analysis of Algorithm Precision

The number of endmembers to be extracted from the AVIRIS Cuprite image was estimated as $p = 19$ after calculating the virtual dimensionality (VD) [33], which is the first stage in our proposed unmixing chain. Table I shows the spectral angles (in degrees) between the most similar endmembers extracted by the OSP-GS and the reference USGS spectral signatures available for this scene. The range of values for the spectral angle is $[0°, 90°]$. As shown by Table I, the endmembers extracted by the OSP-GS algorithm are very similar, spectrally, to the USGS reference signatures, despite the potential variations (due to posible interferers still remaining after the atmospheric correction process) between the ground signatures and the airbone data. For illustrative purposes, the fractional abundance maps obtained for the same representative minerals in the Cuprite mining district are displayed in Fig. 8(a)–(e). Since no reference information is available regarding the true abundance fractions of minerals in the AVIRIS Cuprite data, no quantitative experiments were conducted although the obtained mineral maps exhibit similar correlation with regards to previously published maps[7].

In any case, the results of spectral unmixing can also be evaluated in terms of the quality of the reconstruction of the original data set using the extracted endmembers, the estimated fractional abundances, and the linear mixture model. These results have been discussed in a previous work [34]. In this case, the metric employed to evaluate the goodness of the reconstruction is the root mean square error (RMSE) obtained after comparing the original scene with the reconstructed one. This metric is based on the assumption that a set of high-quality endmembers (and their corresponding estimated abundance fractions) may allow reconstruction of the original scene with higher precision

[4]http://speclab.cr.usgs.gov/wtc

[5]http://aviris.jpl.nasa.gov

[6]http://speclab.cr.usgs.gov

[7]http://speclab.cr.usgs.gov/cuprite.html

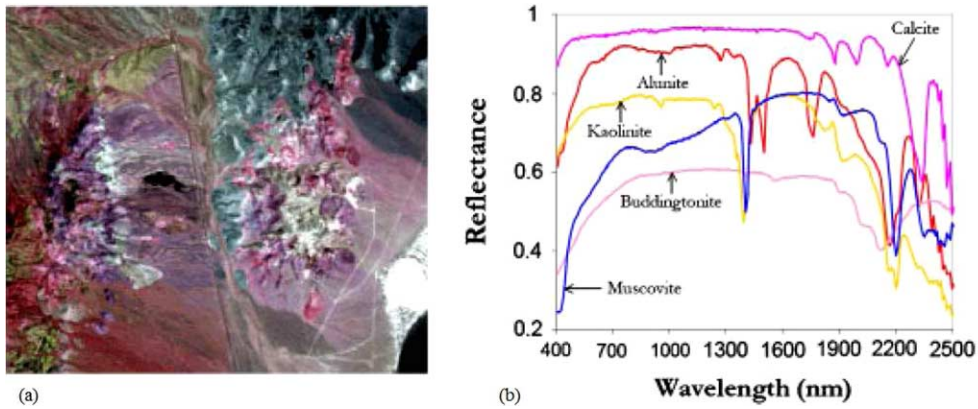


Fig. 7. (a) False color composition of the AVIRIS hyperspectral over the Cuprite mining district in Nevada and (b) U.S. Geological Survey mineral spectral signatures used for validation purposes.

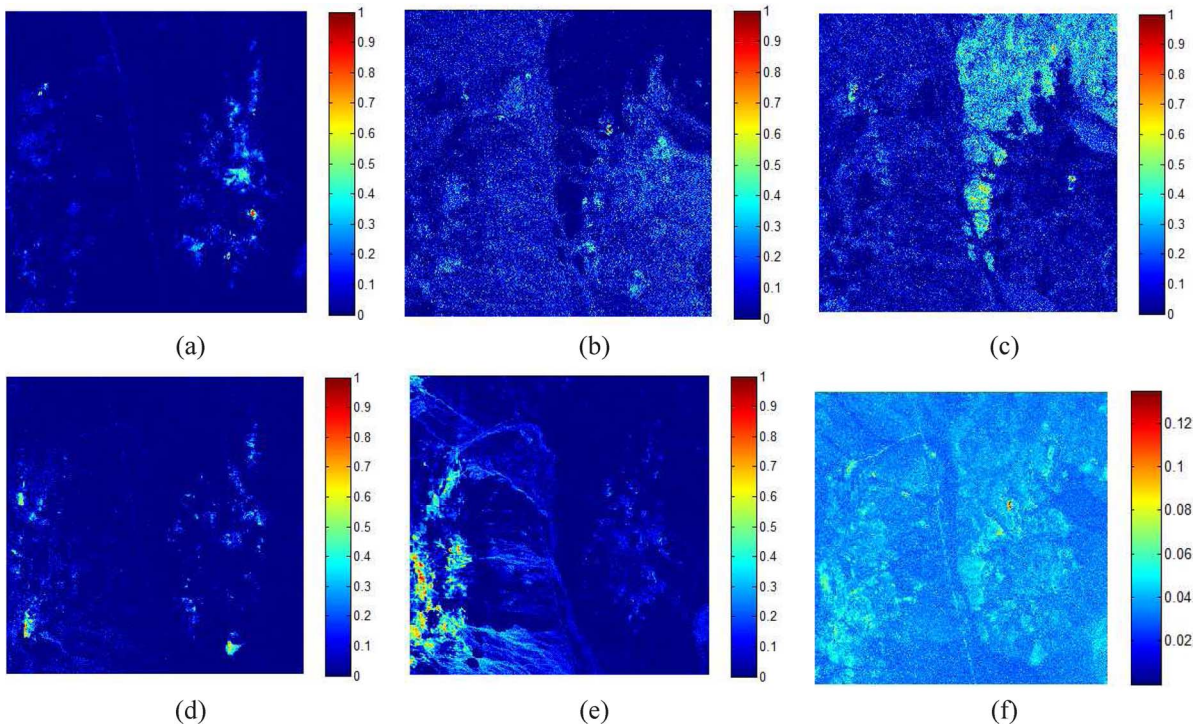


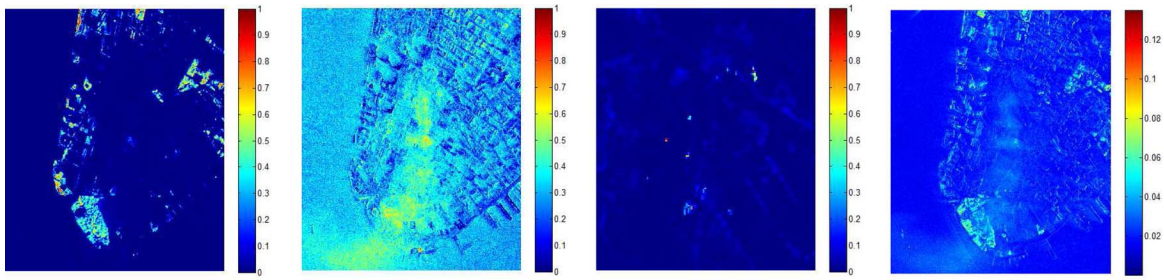
Fig. 8. Abundance maps extracted from the Cuprite scene for different minerals: (a) Alunite. (b) Budinggtonite. (c) Calcite. (d) Kaolinite. (e) Muscovite. (f) Per-pixel RMSE obtained in the reconstruction process of the AVIRIS Cuprite scene using $p = 19$ endmembers (the overall RMSE in this case was 0.0361).

TABLE I
SPECTRAL ANGLE VALUES (IN DEGREES) BETWEEN THE TARGET PIXELS EXTRACTED BY THE OSP-GS ALGORITHM AND THE REFERENCE USGS MINERAL SIGNATURES FOR THE AVIRIS CUPRITE SCENE

Alunite	Buddingtonite	Calcite	Kaolinite	Muscovite	Average
5.48°	4.08°	5.87°	11.14°	5.68°	6.45°

compared to a set of low-quality endmembers. In this case, the original scene is used as a reference to measure the fidelity of the reconstructed version on a per-pixel basis. For illustrative purposes, Fig. 8(f) graphically represents the per-pixel RMSE obtained in the reconstruction process of the AVIRIS Cuprite scene. The RMSE map in Fig. 8(f) generally reveals a good spatial distribution of the error, although some anomalous endmembers appear to be missing. In any event, the per-pixel RMSE values are quite low, indicating a good overall compromise in the reconstruction of the scene.

A similar experiment was also conducted for the AVIRIS WTC scene. Table II shows the spectral angles (in degrees) between the most similar endmember pixels detected by OSP-GS and the pixel vectors at the known target positions in the scene, labeled from 'A' to 'H' in the rightmost part of Fig. 6. The number of endmembers to be detected was $p = 31$ after calculating the virtual dimensionality (VD) of the hyperspectral data. As shown by Table II, the OSP-GS extracted endmembers which were very similar, spectrally, to be known reference pixels in Fig. 6 (this method was able to perfectly detect the



(a) (b) (c) (d)

Fig. 9. Abundance maps extracted from the WTC scene for different targets: (a) Vegetation. (b) Smoke. (c) Fire. (d) Per-pixel RMSE obtained in the reconstruction process of the AVIRIS WTC scene using $p = 31$ endmembers (the overall RMSE in this case was 0.0216).

TABLE II
SPECTRAL ANGLE VALUES (IN DEGREES) BETWEEN THE TARGET PIXELS EXTRACTED BY OSP-GS ALGORITHM AND THE KNOWN GROUND TARGETS IN THE AVIRIS WORLD TRADE CENTER SCENE

A	B	C	D	E	F	G	H
0.00°	27.16°	0.00°	15.62°	27.81°	3.98°	2.72°	24.26°

pixels labeled as 'A' and 'C', and had more difficulties in detecting very small targets). On the other hand, we could observe that the fractional abundance maps of vegetation, smoke and fire [see Fig. 9(a)–(c)] revealed features that cannot be easily appreciated in the false color composition displayed in Fig. 6. For illustrative purposes, the RMSE reconstruction map for this scene is also displayed in Fig. 9(d).

C. Analysis of Parallel Performance

The proposed full hyperspectral unmixing chain has been tested on four different platforms (two GPUs and two multi-core processors):

- The first GPU (denoted hereinafter as GPU1) is the NVidia™ Tesla C1060, which features 240 processor cores operating at 1.296 GHz, with single precision floating point performance of 933 Gflops, double precision floating point performance of 78 Gflops, total dedicated memory of 4 GB, 800 MHz memory (with 512-bit GDDR3 interface) and memory bandwidth of 102 GB/s[8].
- The second GPU (denoted hereinafter as GPU2) used is the NVidia™ GeForce GTX 580, which features 512 processor cores operating at 1.544 GHz, with single precision floating point performance of 1,354 Gflops, double precision floating point performance of 198 Gflops, total dedicated memory of 1,536 MB, 2,004 MHz memory (with 384-bit GDDR5 interface) and memory bandwidth of 192.4 GB/s[9].
- The two aforementioned GPUs are connected to a multi-core Intel i7 920 CPU (denoted hereinafter as MC1) at 2.67 GHz with 4 physical cores and 6 GB of DDR3 RAM memory, which uses a motherboard Asus P6T7 WS SuperComputer.
- Finally, another multi-core system (denoted hereinafter as MC2) is also used in our experiments. The system is made up of two Quad Core Intel Xeon at 2.53 GHz with 12 physical cores and 24 GB of DDR3 RAM memory, which uses a motherboard Supermicro X8DTT-H and it is mounted on a bullx R424-E2.

For illustrative purposes, Table III provides an indication of processor performance (including memory bandwidth and floating point performance) for both multi-core systems (MC1 and MC2) used in our experiments. These measures have been obtained using `Geekbench`[10] (version 2.4.0), a widely used benchmarking tool which provides a comprehensive set of benchmarks engineered to quickly and accurately measure processor and memory performance. Specifically, floating point performance has been evaluated using two `Geekbench` benchmarks based on the following calculations: vector dot product (single-core scalar, multi-core scalar, single-core vector and multi-core vector), and matrix LU decomposition (single-core scalar and multi-core scalar). On the other hand, memory bandwidth has been evaluated using two `Geekbench` benchmarks based on the following calculations: stream copy (single-core scalar and single-core vector) and stream add (single-core scalar and single-core vector).

Before describing our parallel performance results, it is important to emphasize that our GPU and multi-core versions provide exactly the same results as the serial versions of the implemented algorithms, using the `gcc` (gnu compiler default) with optimation flags `-O3` (for the single-core version) and `-fopenmp` (flag used for the multi-core version) to exploit data locality and avoid redundant computations. Hence, the only difference between the serial and parallel algorithms is the time they need to complete their calculations. The serial algorithms were executed in one of the available cores, while the multi-core versions were executed in all the available cores. For each experiment, ten runs were performed and the mean values were reported (these times were always very similar, with differences on the order of a few milliseconds only).

Tables IV and V summarize the timing results and speedups measured after processing two hyperspectral images on the considered GPU and multi-core platforms. It should be noted

[8]http://www.nvidia.com/object/product_tesla_c1060_us.html

[9]http://www.nvidia.com/object/product-geforce-gtx-580-us.html

[10]http://www.primatelabs.com/geekbench/

TABLE III
PROCESSOR PERFORMANCE (INCLUDING MEMORY BANDWITH AND FLOATING POINT PERFORMANCE) FOR THE TWO MULTI-CORE SYSTEMS (MC1 AND MC2) USED IN OUR EXPERIMENTS, MEASURED USING DIFFERENT GEEKBENCH BENCHMARKS

Benchmark	Description	MC1	MC2
Vector dot product (measures floating point performance in Gflops)	single-core scalar	1.53	1.39
	multi-core scalar	7.24	16.4
	single-core vector	4.83	4.34
	multi-core vector	20.7	51.5
Matrix LU decomposition (measures floating point performance in Gflops)	single-core scalar	2.04	1.82
	multi-core scalar	4.41	11.3
Stream copy (measures memory bandwidth in GB/sec)	single-core scalar	5.96	5.42
	single-core vector	8.59	6.80
Stream add (measures memory bandwidth in GB/sec)	single-core scalar	7.85	6.49
	single-core vector	8.45	7.11

TABLE IV
PROCESSING TIMES (IN SECONDS) AND SPEEDUPS ACHIEVED FOR THE PARALLEL UNMIXING CHAIN IN TWO DIFFERENT PLATFORMS: MULTI-CORE AND GPU, TESTED WITH THE AVIRIS CUPRITE SCENE

	Initialization	VD	OSP-GS	UCLS	Writing of final results	Total
Serial time	0.121	5.541	1.331	1.051	0.009	8.053
Parallel time GPU1	0.269	0.246	0.049	0.067	0.009	0.640
Parallel time GPU2	0.281	0.241	0.024	0.034	0.011	0.590
Parallel time MC1	0.126	0.924	0.516	0.277	0.010	1.853
Parallel time MC2	0.098	1.066	1.055	0.197	0.053	2.468
Speedup (GPU1)	–	22.48	27.26	15.74	–	12.58
Speedup (GPU2)	–	23.00	55.28	30.62	–	13.64
Speedup (MC1)	–	6.00	2.58	3.79	–	4.35
Speedup (MC2)	–	5.20	1.26	5.35	–	3.26

TABLE V
PROCESSING TIMES (IN SECONDS) AND SPEEDUPS ACHIEVED FOR THE PARALLEL UNMIXING CHAIN IN TWO DIFFERENT PLATFORMS: MULTI-CORE AND GPU, TESTED WITH THE AVIRIS WTC SCENE

	Initialization	VD	OSP-GS	UCLS	Writing of final results	Total
Serial time	0.364	20.149	9.979	10.314	0.036	40.842
Parallel time GPU1	0.522	0.711	0.202	0.280	0.039	1.755
Parallel time GPU2	0.535	0.499	0.109	0.133	0.037	1.313
Parallel time MC1	0.370	3.258	3.549	2.759	0.037	9.973
Parallel time MC2	0.281	3.782	4.612	1.620	0.206	10.501
Speedup (GPU1)	–	28.34	49.28	36.79	–	23.28
Speedup (GPU2)	–	40.37	91.49	77.66	–	31.12
Speedup (MC1)	–	6.18	2.81	3.74	–	4.10
Speedup (MC2)	–	5.33	2.16	6.37	–	3.89

that the cross-track line scan time in AVIRIS, a push-broom instrument [2], is quite fast (8.3 milliseconds to collect 512 full pixel vectors). This introduces the need to process the considered AVIRIS Cuprite scene (350×350 pixels and 188 spectral bands) in less than 1.985 seconds to fully achieve real-time performance. Similarly, the AVIRIS WTC scene needs to be processed in less than 5.096 seconds in order to achieve real-time performance. As shown by Table IV, the AVIRIS Cuprite scene could be processed in real-time using GPU1, GPU2 and MC1. On the other hand, Table V reveals that the AVIRIS WTC scene could only be processed in real-time in the GPU platforms.

For illustrative purposes, Fig. 10 shows the percentage of the total GPU execution time consumed by memory transfers and by each CUDA kernel (obtained after profiling the full implementation of the spectral unmixing chain) during the processing of the AVIRIS World Trade Center scene in the two considered GPUs. As shown by Fig. 10, the implementation on the GeForce

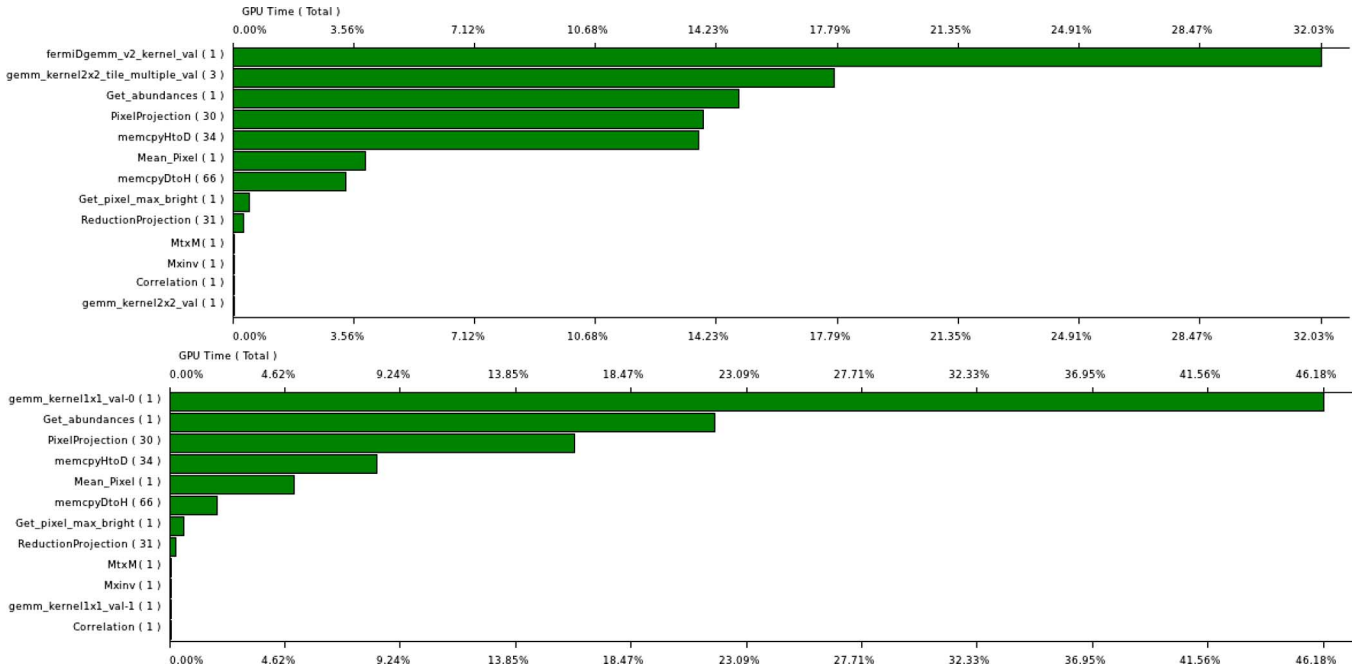


Fig. 10. Summary plot describing the percentage of the total GPU time consumed by memory transfer operations and by the different kernels used by the NVidia GeForce GTX 580 GPU (top) and the Tesla C1060 GPU (bottom) in the unmixing of the AVIRIS WTC image.

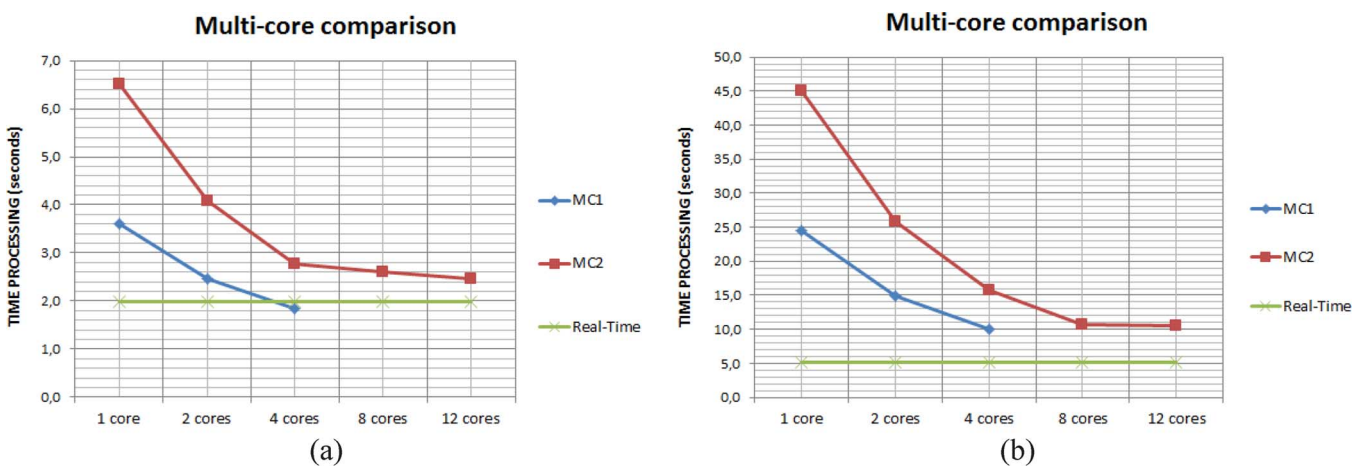


Fig. 11. Comparison of time processing between the two considered multi-core processors varying the number of cores using: (a) AVIRIS Cuprite scene. (b) AVIRIS WTC scene.

GTX 580 GPU uses approximately 85% of the execution time for executing the kernels and 15% of the time for memory transfers. On the other hand, the implementation on the Tesla C1060 GPU uses about 90% of the total GPU time for executing the kernels and only 10% of the time for memory transfers. This indicates that memory transfers are not a bottleneck for the proposed GPU implementations.

On the other hand, Fig. 11 compares the processing times measured in the two considered multi-core processors when the number of physical cores available was varied. With the increase of the number of cores, the processing time in MC1 significantly decreases and leads to real-time processing results, as illustrated in Fig. 11(a). However, real-time processing performance could never be achieved in MC2. This is due to the scalability problems observed when increasing the number of cores. Specifically, it can be seen in Figs. 11(a) and (b) that there was no significant difference between using 4 or 12 cores in this platform. This suggests that our multi-core implementation can be optimized to increase its scalability to a high number of cores.

Although the speedups obtained in all cases for the multi-core platforms are not very high (particularly for MC2), we believe that this kind of systems may provide a good alternative to GPUs for onboard processing of remote sensing data. This is particularly due to the fact that the power consumption of GPUs is quite high, an observation that may compromise mission payload and energy requirements. In this regard, multi-core platforms are evolving very quickly and it is expected that systems with hundreds of cores will be soon available, thus offering the possibility to replace many-core systems such as GPUs as the default platforms for high performance computing in many applications. At present, GPUs offer such possibility of massively parallel processing and we have illus-

trated in this work that their computational power can be readily exploited for providing real-time performance in an important exploitation-based application for hyperspectral data such as spectral unmixing.

VI. Conclusions and Future Research Lines

In this work, we have developed computationally efficient implementations of a full hyperspectral unmixing on multi-core processors and GPU platforms. Both platforms can boost the computational performance of the considered unmixing chain, using relatively inexpensive hardware. The performance of the proposed implementations has been evaluated (in terms of the quality of the solutions provided and its parallel performance) in the context of two analysis scenarios, using data sets collected by the AVIRIS instrument. The experimental results reported in this paper indicate that remotely sensed hyperspectral imaging can greatly benefit from the development of efficient implementations of unmixing algorithms in specialized hardware devices for better exploitation of high-dimensional data sets. In this case, real-time performance could be obtained using any of the considered GPU devices and one of the considered multi-core environments. GPUs compared with multi-core processors present more advantages because these devices offer few on-board restrictions in terms of cost and size, and these are important parameters when defining mission payload in remote sensing missions. In this regard, our contribution bridges the gap towards real-time unmixing of remotely sensed hyperspectral images in GPUs and multi-core processors.

Although the results reported in this paper are very encouraging, GPUs and multi-core processors are still rarely exploited in real missions due to power consumption and radiation tolerance issues, despite improvements in these directions are expected in upcoming years. Currently we are also experimenting with FPGAs and other spectral unmixing algorithms, i.e., a fully constrained linear spectral unmixing algorithm, in order to be able to adapt the proposed algorithms to hardware devices that can be mounted onboard hyperspectral imaging instruments after space qualification by international agencies. We are also investigating the use of OpenCL[11] as a computing standard for multi-core architectures that allows writing code for both GPUs and CPUs, and which has recently also seen a growing interest for FPGAs.

Acknowledgment

This work has been supported by the European Community Marie Curie Research Training Networks Programme under contract MRTN-CT-2006-035927, Hyperspectral Imaging Network (HYPER-I-NET). Funding from the Icelandic Research Fund and the Spanish Ministry of Science and Innovation (CEOS-SPAIN project, reference AYA2011-29334-C02-02) and DREAMS project (Reference TEC2011-28666-C04-04) are also gratefully acknowledged. Also, this work was partially supported by the computing facilities of Extremadura Research for Advanced Technologies (CETA-CIEMAT), funded by the European Regional Development Fund (ERDF). The CETA-CIEMAT belongs to the Spanish Ministry of Science and Innovation. Last but not least, we would like to take this opportunity to gratefully thank the Editor and the anonymous reviewers for their outstanding comments and suggestions, which have been really helpful in order to enhance the technical content and presentation of our manuscript.

[11]http://www.khronos.org/opencl

References

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

[2] R. O. Green *et al.*, "Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (AVIRIS)," *Remote Sens. Environ.*, vol. 65, no. 3, pp. 227–248, 1998.

[3] A. Plaza, Q. Du, J. Bioucas-Dias, X. Jia, and F. Kruse, "Foreword to the special issue on spectral unmixing of remotely sensed data," *IEEE Trans. Geosci. Remote Sens.*, vol. 19, no. 11, pp. 4103–4110, 2011.

[4] 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 Sens. Environ.*, vol. 113, pp. 110–122, 2009.

[5] J. B. Adams, M. O. Smith, and P. E. Johnson, "Spectral mixture modeling: A new analysis of rock and soil types at the Viking Lander 1 site," *J. Geophys. Res.*, vol. 91, pp. 8098–8112, 1986.

[6] N. Keshava and J. F. Mustard, "Spectral unmixing," *IEEE Signal Process. Mag.*, vol. 19, no. 1, pp. 44–57, 2002.

[7] C.-I. Chang and D. Heinz, "Constrained subpixel target detection for remotely sensed imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 38, pp. 1144–1159, 2000.

[8] D. Heinz and C.-I. Chang, "Fully constrained least squares linear mixture analysis for material quantification in hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, pp. 529–545, 2000.

[9] C. C. Borel and S. A. W. Gerslt, "Nonlinear spectral mixing models for vegetative and soil surfaces," *Remote Sens. Environ.*, vol. 47, pp. 403–416, 1994.

[10] W. Liu and E. Y. Wu, "Comparison of non-linear mixture models," *Remote Sens. Environ.*, vol. 18, pp. 1976–2003, 2004.

[11] K. J. Guilfoyle, M. L. Althouse, and C.-I. Chang, "A quantitative and comparative analysis of linear and nonlinear spectral mixture models using radial basis function neural networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, pp. 2314–2318, 2001.

[12] J. C. Harsanyi and C.-I. Chang, "Hyperspectral image classification and dimensionality reduction: An orthogonal subspace projection," *IEEE Trans. Geosci. Remote Sens.*, vol. 32, no. 4, pp. 779–785.

[13] J. Boardman, F. A. Kruse, and R. O. Green, "Mapping target signatures via partial unmixing of AVIRIS data," in *Summaries of the JPL Airborne Earth Science Workshop*, 1995, pp. 23–26, JPL Publication.

[14] J. H. Bowles, P. J. Palmadesso, J. A. Antoniades, M. M. Baumback, and L. J. Rickard, "Use of filter vectors in hyperspectral data analysis," in *Proc. SPIE Infrared Spaceborne Remote Sensing III*, 1995, vol. 2553, pp. 148–157.

[15] R. A. Neville, K. Staenz, T. Szeredi, J. Lefebvre, and P. Hauff, "Automatic endmember extraction from hyperspectral data for mineral exploration," in *Proc. 21st Can. Symp. Remote Sensing*, 1999, pp. 21–24.

[16] A. Ifarraguerri and C.-I. Chang, "Multispectral and hyperspectral image analysis with convex cones," *IEEE Trans. Geosci. Remote Sens.*, vol. 37, no. 2, pp. 756–770, 1999.

[17] M. Winter, "N-FINDR: An algorithm for fast autonomous spectral endmember determination in hyperspectral data," in *Proc. SPIE*, 1999, vol. 3753, pp. 266–270.

[18] A. Plaza, D. Valencia, J. Plaza, and P. Martinez, "Commodity cluster-based parallel processing of hyperspectral imagery," *J. Parallel Distrib. Comput.*, vol. 66, no. 3, pp. 345–358, 2006.

[19] A. Plaza and C.-I. Chang, *High Performance Computing in Remote Sensing*. Boca Raton, FL, USA: CRC Press, 2007.

[20] 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 J. Sel. Topics Appl. Earth Observ. Remote Sens.* , vol. 4, no. 3, pp. 508–527, 2011.

[21] A. Plaza and C.-I. Chang, "Clusters versus FPGA for parallel processing of hyperspectral imagery," *Int. J. High Performance Computing Applications*, vol. 22, no. 4, pp. 366–385, 2008.

[22] C. Gonzalez, D. Mozos, J. Resano, and A. Plaza, "FPGA implementation of the N-FINDR algorithm for remotely sensed hyperspectral image analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 2, pp. 374–388, 2012.
[23] A. Remon, S. Sanchez, A. Paz, E. S. Quintana-Orti, and A. Plaza, "Real-time endmember extraction on multi-core processors," *IEEE Geosci. Remote Sens. Lett.*, vol. 8, no. 5, pp. 924–928.
[24] J. Setoain, M. Prieto, C. Tenllado, and F. Tirado, "GPU for parallel on-board hyperspectral image processing," *Int. J. High Performance Computing Applications*, vol. 22, no. 4, pp. 424–437, 2008.
[25] 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," *J. Real-Time Image Process.*, vol. 4, pp. 1–14, 2009.
[26] H. Yang, Q. Du, and G. Chen, "Unsupervised hyperspectral band selection using graphics processing units," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.* , vol. 4, no. 3, pp. 660–668, 2011.
[27] C.-C. Chang, Y.-L. Chang, M.-Y. Huang, and B. Huang, "Accelerating regular LDPC code decoders on GPUs," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.* , vol. 4, no. 3, 2011.
[28] A. Plaza, J. Plaza, A. Paz, and S. Sánchez, "Parallel hyperspectral image and signal processing," *IEEE Signal Process. Mag.*, vol. 28, no. 3, pp. 119–126, 2011.
[29] Q. Du and C.-I. Chang, "Estimation of number of spectrally distinct signal sources in hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 3, pp. 608–619, 2004.
[30] S. Lopez, P. Horstrand, G. M. Callico, J. F. Lopez, and R. Sarmiento, "A low-computational-complexity algorithm for hyperspectral endmember extraction: Modified vertex component analysis," *IEEE Geosci. Remote Sens. Lett.*, vol. 9, no. 3, pp. 502–506, 2012.
[31] S. Bernabe, S. Lopez, A. Plaza, and R. Sarmiento, "GPU implementation of an automatic target detection and classification algorithm for hyperspectral image analysis," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 2, pp. 221–225, 2013.
[32] J. Camara, J. Cuenca, D. Gimenez, and A. Vidal, "Empirical autotuning of two-level parallel linear algebra routines on large CC-NUMA systems," in *Proc. 10th IEEE Int. Symp. Parallel and Distributed Processing With Applications*, 2012, pp. 843–844.
[33] C.-I. Chang, *Hyperspectral Imaging: Techniques for Spectral Detection and Classification*. New York, NY, USA: Kluwer Academic, 2003.
[34] 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*, vol. 23, no. 13, pp. 1538–1557, 2011.

Sergio Bernabé (S'12) received the Computer Engineer degree from the University of Extremadura, Caceres, Spain, and the M.Sc. degree from the same university. He is currently pursuing a joint Ph.D. between the University of Iceland and the University of Extremadura, Spain.

He has been a visiting researcher with the Institute for Applied Microelectronics, University of Las Palmas de Gran Canaria, Spain. His research interests comprise hyper/multi-spectral image analysis and efficient implementations of large-scale cientific problems on commodity Beowulf clusters and graphical processing units (GPUS).

Mr. Bernabe has served as a reviewer for the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING and for the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS.

Sergio Sánchez received the M.Sc. degree in 2010 and the Ph.D. degree in 2013.

He is currently a Research Associate with the Hyperspectral Computing Laboratory, Department of Technology of Computers and Communications, University of Extremadura, Spain. His main research interests comprise hyperspectral image analysis and efficient implementations of large-scale scientific problems on commodity graphical processing units (GPUs).

Antonio Plaza (SM'07) received the M.S. and Ph.D. degrees in computer engineering from the University of Extremadura, Extremadura, Spain, in 1999 and 2002, respectively.

He has been a Visiting Researcher with the Remote Sensing Signal and Image Processing Laboratory (RSSIPL), University of Maryland Baltimore County, Baltimore; with the Applied Information Sciences Branch, NASA Goddard Space Flight Center (GSFC), Greenbelt, MD; with the Airborne Visible Infrared Imaging Spectrometer Data Facility, NASA Jet Propulsion Laboratory (JPL), Pasadena, CA; with the Telecommunications and Remote Sensing Laboratory, University of Pavia, Pavia, Italy; and with the GIPSA-lab, Grenoble Images Parole Signal Automatique, Grenoble, France. He is currently an Associate Professor (with accreditation for Full Professor) with the Department of Technology of Computers and Communications, University of Extremadura, where he is the Head of the Hyperspectral Computing Laboratory (HyperComp). He was the Coordinator of the Hyperspectral Imaging Network, a European project designed to build an interdisciplinary research community focused on hyperspectral imaging activities. He is the author or coauthor of more than 370 publications on remotely sensed hyperspectral imaging, including around 100 journal citation report papers (51 since January 2011), around 20 book chapters, and over 230 conference proceeding papers. His research interests include remotely sensed hyperspectral imaging, pattern recognition, signal and image processing, and efficient implementation of large-scale scientific problems on parallel and distributed computer architectures.

Dr. Plaza has guest edited seven special issues in scientific journals on the topic of remotely sensed hyperspectral imaging. He has been a Chair for the IEEE Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (Whispers 2011). He has also been serving as a Chair for the SPIE Conference on Satellite Data Compression, Communications, and Processing, since 2009, and for the SPIE Europe Conference on High-Performance Computing in Remote Sensing, since 2011. He has been a recipient of the recognition of Best Reviewers of the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS in 2009 and a recipient of the recognition of Best Reviewers of the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING in 2010, a journal for which he has served as an Associate Editor since 2007 and for which he has reviewed more than 260 manuscripts. In 2011–2012 he served as the Director of Education Activities of the IEEE Geoscience and Remote Sensing Society (GRSS), and is currently serving as President of the Spanish Chapter of IEEE GRSS. He has been appointed Editor of the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, with a three-year term starting in January 2013.

Sebastián López (M'08) was born in Las Palmas de Gran Canaria, Spain, in 1978. He received the Electronic Engineer degree from the University of La Laguna, Santa Cruz de Tenerife, Spain, in 2001, obtaining regional and national awards for his CV during his degree, and the Ph.D. degree from the University of Las Palmas de Gran Canaria, Las Palmas de Gran Canaria, Spain, in 2006.

Currently, he is an Associate Professor at the University of Las Palmas de Gran Canaria, developing his research activities at the Integrated Systems Design Division of the Institute for Applied Microelectronics (IUMA). He has published more than 50 papers in international journals and conferences. His research interests include real-time hyperspectral imaging systems, reconfigurable architectures, video coding standards, and hyperspectral image compression systems.

Prof. Lopez is a member of the IEEE Geoscience and Remote Sensing Society and the IEEE Consumer Electronics Society as well as a member of the Publications Review Committee of the IEEE TRANSACTIONS ON CONSUMER ELECTRONICS. Additionally, he currently serves as an active reviewer of the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING, the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, the *Journal of Real Time Image Processing, Microprocessors and Microsystems: Embedded Hardware Design* (MICPRO), and the IET *Electronics Letters*.

Jón Atli Benediktsson (F'04) received the Cand.Sci. degree in electrical engineering from the University of Iceland, Reykjavik, Iceland, in 1984, and the M.S.E.E. and Ph.D. degrees in electrical engineering from Purdue University, West Lafayette, IN, USA, in 1987 and 1990, respectively.

He is currently Pro Rector for Academic Affairs and Professor of Electrical and Computer Engineering at the University of Iceland. His research interests are in remote sensing, biomedical analysis of signals, pattern recognition, image processing, and signal processing, and he has published extensively in those fields.

Prof. Benediktsson was the 2011–2012 President of the IEEE Geoscience and and Remote Sensing Society (GRSS) and has been on the GRSS AdCom since 2000. He was Editor-in-Chief of the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING (TGRS) from 2003 to 2008 and has served as Associate Editor of TGRS since 1999 and the IEEE *Geoscience and Remote Sensing Letters* since 2003. He was the Chairman of the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING (JSTARS) 2007–2010. Prof. Benediktsson is a co-founder of the biomedical start up company Oxymap. He received the Stevan J. Kristof Award from Purdue University in 1991 as outstanding graduate student in remote sensing. In 1997, Dr. Benediktsson was the recipient of the Icelandic Research Council's Outstanding Young Researcher Award, in 2000, he was granted the IEEE Third Millennium Medal, in 2004, he was a co-recipient of the University of Iceland's Technology Innovation Award, in 2006 he received the yearly research award from the Engineering Research Institute of the University of Iceland, and in 2007, he received the Outstanding Service Award from the IEEE Geoscience and Remote Sensing Society. He is co-recipient of the 2012 IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING Paper Award. He is a Fellow of IEEE and Fellow of SPIE, and a member of Societas Scinetiarum Islandica and Tau Beta Pi.

Roberto Sarmiento is a Full Professor with the Telecommunication Engineering School at University of Las Palmas de Gran Canaria (ULPGC), Spain, in the area of electronic engineering. He contributed to set this school up; he was the Dean of the Faculty from 1994 to 1995 and Vice-Chancellor for Academic Affairs and Staff at the ULPGC from 1998 to 2003. In 1993, he was a Visiting Professor at the University of Adelaide, South Australia, and later at the Edith Cowan University, also in Australia. He is a founder of the Research Institute for Applied Microelectronics (IUMA) and Director of the Integrated Systems Design Division of this Institute. Since 1990 he has published over 30 journal papers and book chapters and more than 100 conference papers and has participated in more than 35 projects and research programs funded by public and private organizations, in 15 of which he has been leader researcher.