2013 15th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing

A New Digital Repository for Remotely Sensed Hyperspectral Imagery on GPUs

Jorge Sevilla
Hyperspectral Computing Laboratory
Department of Technology of Computers
and Communications, University of Extremadura
Cáceres, Spain.
Email: jorgesece@unex.es

Antonio Plaza
Hyperspectral Computing Laboratory
Department of Technology of Computers
and Communications, University of Extremadura
Cáceres, Spain.
Email: aplaza@unex.es

***Abstract*—Hyperspectral imaging is a new technique in remote sensing in which an imaging spectrometer collects hundred of images (at different wavelength channels) for the same area on the surface of Earth. Over the last years, hyperspectral image data sets have been collected from a great amount of locations over the world using a variety of instruments for Earth observation. Only a small amount of them are available for public use and they are spread among different storage locations and exhibit significant heterogeneity regarding the storage format. Therefore, the development of a standardized hyperspectral data repository is a highly desired goal in the remote sensing community. In this paper, we describe the development of a shared digital repository for remotely sensed hyperspectral data, which allows uploading new hyperspectral data sets along with meta-data, ground-truth and analysis results (spectral information). Such repository is presented as a web service for providing the management of images through a web interface, and it is available online from http://www.hypercomp.es/repository. Most importantly, the developed system includes a spectral unmixing-based content based image retrieval (CBIR) functionality which allows searching for images from the database using spectrally pure components or endmembers in the scene. A full spectral unmixing chain is implemented for spectral information extraction, which allows filtering images using the similarity of the spectral signature and abundance of a given ground-truth. In order to accelerate the process of obtaining the spectral information for new entries in the system, we resort to an efficient implementations of spectral unmixing algorithms of graphics processing units (GPUs).**

Keywords*—*Hyperspectral imaging, repository, content-based image retrieval (CBIR), spectral unmixing, high performance computing, GPUs, distributed resources.

I. Introduction

Content-based image retrieval (CBIR) intends to retrieve, from real data stored in a database, information that is relevant to a query [1]. This is particularly important in large data repositories, such as those available in remotely sensed hyperspectral imaging [2]. For instance, the NASA Jet Propulsion Laboratory's Airborne Visible Infra-Red Imaging Spectrometer (AVIRIS) [3] is able to record the visible and near infrared spectrum of the reflected light of an area several kilometers long (depending on the duration of the flight) using hundreds of spectral bands. The resulting 'image cube' is a stack of images (see Fig. 1), in which each pixel (vector) has an associated spectral signature or 'fingerprint' that uniquely characterizes the underlying objects. The resulting data often comprises several Gigabytes per flight.

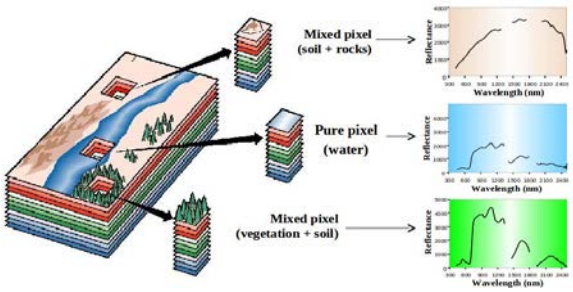


Fig. 1. The concept of hyperspectral imaging.

One of the main problems involved in hyperspectral data exploitation is spectral unmixing [4], as many of the pixels collected by imaging spectrometers such as AVIRIS are highly mixed in nature due to spatial resolution and other phenomena. For instance, it is very likely that the pixel labeled as 'vegetation' in Fig. 1 is actually composed of several types of vegetation canopies interacting at sub-pixel levels. The same comment applies to the 'soil' pixel, which may comprise different types of geological features. As a result, spectral unmixing is a very important task for hyperspectral data exploitation since the spectral signatures collected in natural environments are invariably a mixture of the pure signatures of the various materials found within the spatial extent of the ground instantaneous field view of the imaging instrument. Among several techniques designed to deal with the inherent complexity of hyperspectral images in supervised fashion [4], [5], linear spectral unmixing follows an unsupervised approach which aims at inferring pure spectral signatures, called *endmembers*, and their material fractions at each pixel of the scene.

The amount of hyperspectral image data sets have increased in the last years and a huge number of data sets have been collected of locations over the world, using a variety of instruments for Earth observation. Furthermore, the data sets which are available for public use are spread among different storage locations and present significant heterogeneity regarding the storage format, associated meta-data (if any), or ground-truth availability. At the moment there is no common repository of hyperspectral data intended to distribute and share hyperspectral data sets in the community, so that researchers who want to address their studies related to hyperspectral imaging find several problems for starting.

978-1-4799-3035-7/14 $31.00 © 2014 IEEE
DOI 10.1109/SYNASC.2013.68

In this paper, we describe a new digital repository for remotely sensed hyperspectral data with CBIR system functionality which takes advantage of seminal concepts from linear spectral unmixing concepts [6] to perform effective data retrieval. We use the information provided by spectral unmixing (i.e. the spectral endmembers) as effective metadata to develop a new CBIR system that can assist in the task of efficiently searching hyperspectral image instances in large data repositories.

The current system supports different spectral unmixing algorithms for estimation of number of endmembers such as Hyperspectral Signal Subspace Identification by Minimum Error (HySime) [7] or Virtual dimensionality (VD) [8], for endmember extraction such as Orthogonal subspace projection with Gram-Schmidt orthogonalization (OPS-GS) [9] or N-FINDR [10], and for abundances estimation such as Unconstrained least-squares (UCLS) algorithm for abundance estimation. Although our experiment includes only the algorithms more efficient in computational terms such as VD for estimation of number of endmembers, N-FINDR for endmember extraction and UCLS for abundances estimation.

In order to deal with the computational cost of extracting the information needed to catalog new hyperspectral images in our system, we resort to graphics processing units (GPUs) which have been successfully used to accelerate hyperspectral-related computations. There are a few works in the literature [11]–[15] dealing explicitly with the spectral information to guide the search and none of them are performed with GPUs. Furthermore, in order to relieve the load of the computational server, our system provides distributed computing management using tow clusters (with CPU and GPU architectures).

The proposed system is experimentally validated using both synthetic scenes constructed using fractals and a real hyperspectral data sets collected by NASA's Airborne Visible Infrared Imaging Spectrometer (AVIRIS) over the Cuprite Mining District in Nevada and over the World Trade Center (WTC) area in New York City on September 16. Our results indicate that the proposed system can efficiently retrieve hyperspectral images from a complex image database. The proposed system is expected to increase the value of the data acquired by airborne/satellite hyperspectral imaging instruments, and to improve the end-user services available in hyperspectral image databases.

The remainder of the paper is structured as follows. Section 2 describes the considered spectral unmixing methodology used to implement the core of our CBIR system. Section 3 describes the proposed CBIR system design. Section 4 describes the searching methodology. Section 5 assesses the performance of the system by comparing its retrieval accuracy using synthetic and real hyperspectral images with different noise levels and comparing the execution time of both serial and GPU algorithms. Section 6 concludes with some remarks and future research lines.

II. Spectral Unmixing Methodology

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:

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

where $\mathbf{M} = \{\mathbf{e}_i\}_{i=1}^{p}$ is a matrix containing p endmember signatures, $\alpha = [\alpha_1, \alpha_2, \cdots, \alpha_p]$ 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: 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.

In recent years, several techniques have been proposed to resolve the mixed pixel problem ([7]–[10] and others), in addition most of this techniques have been optimized in GPU implementations ([16]–[19]). In order to provide a CBIR system with high availability and quality of service, the system includes GPU implementation of the algorithms described above.

A. Unmixing Chain Algorithms

1) Virtual Dimensionality (VD) algorithm for estimation of the number of endmembers: Let us denote by $\mathbf{Y} \equiv [y_1, y_2, \cdots, y_N]$ 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/N$(\mathbf{Y}-\overline{\mathbf{Y}})^T(\mathbf{Y}-\overline{\mathbf{Y}})$ and the correlation matrix $\mathbf{R}_{L\times L}$ = $\mathbf{K}_{L\times L}$ + $\overline{\mathbf{Y}}\overline{\mathbf{Y}}^T$, 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 [8] 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 $\widehat{p}$ 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) N-FINDR algorithm for endmembers extraction: The N-FINDR algorithm [10] is one of the most widely used an successfully applied methods for automatically determining endmembers in hyperspectral image data without using a priori information. This algorithm looks for the set of pixels with the largest possible volume by *inflating* a simplex inside the data. The procedure begins with a random initial selection of

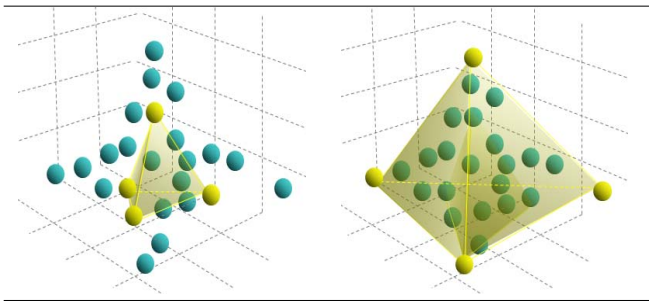
pixels (see Fig. 2). Every pixel in the image must be evaluated to refine the estimate of endmembers, looking for the set of pixels that maximizes the volume of the simplex defined by the selected endmembers. The mathematical definition of the volume of a simplex formed by a set of endmember candidates is proportional to the determinant of the set augmented by a row of ones. The determinant is only defined in the case where the number of features is $p-1$,p being the number of desired endmembers [20]. Since in hyperspectral data typically $n \gg p$, a transformation that reduces the dimensionality of the input data is required. In this work, we use the PCA [21] for this propose. The corresponding volume is calculated for every pixel in each endmember position by replacing that endmember and finding the resulting volume. If the replacement results in an increase of volume, the pixel replaces the endmember. This procedure is repeated in iterative fashion until there are no more endmember replacements. The method can be summarized by a step-by-step algorithmic description which is given bellow:

- *Feature reduction*. Apply the PCA [21] to reduce the dimensionality of the data from n to $d = p - 1$, where *p* is the number of endmembers to be extracted. Basically PCA is a statistical method for dimensionality reduction [18] which calculates the projection for representing the data, it is computed by performing the eigen-decomposition of the covariance matrix of the image.
- *Initialization*. Let $\{\mathbf{e}_1^{(0)}, \mathbf{e}_2^{(0)}, \ldots, \mathbf{e}_p^{(0)}\}$ be a set of endmembers randomly extracted from the input data.
- *Volume calculation*. At iteration $k \geq 0$, the volume defined by the current set of endmembers is calculated in volume calculation as follows.

$$V(\mathbf{e}_p^{(k)}, \mathbf{e}_p^{(k)}, \ldots, \mathbf{e}_p^{(k)}) = \frac{\left| det \begin{bmatrix} 1 & 1 & \ldots & 1 \\ \mathbf{e}_1^{(k)} & \mathbf{e}_2^{(k)} & \ldots & \mathbf{e}_p^{(k)} \end{bmatrix} \right|}{(p-1)!} \quad (2)$$

- *Replacement*. For each pixel vector x_j int he input hyperspectral data, we recalculate the volume by testing the pixel in all p endmember positions, i.e., first calculate $V(x_j, \mathbf{e}_2^{(k),\ldots,\mathbf{e}_p^{(k)}})$, then calculate $V(\mathbf{e}_1^{(k)}, x_j, \ldots, \mathbf{e}_p^{(k)})$, and so on until $V(\mathbf{e}_1^{(k)}, \mathbf{e}_2^{(k)}, \ldots, x_j)$. If none of the p recalculated volumes is greater than $V(\mathbf{e}_1^{(k)}, \mathbf{e}_2^{(k)}, \ldots, \mathbf{e}_p^{(k)})$, then no endmember is replaced. Otherwise, the combination with maximum volume is retained. Let us assume that the endmember absent in the combination resulting in the maximum volume is denoted by $\mathbf{e}_\mathbf{i}^{(\mathbf{k+1})}$. In this case, a new set of endmembers is produced by letting $\mathbf{e}_i^{(k+1)} = x_j$ and $\mathbf{e}_j^{(k+1)} = \mathbf{e}_j^{(k)}$ for $i \neq j$. The replacement step is repeated for all the pixel vector in the input data until all the pixels have been exhausted.

As a final comment, it has been observed that different random initializations of N-FINDR may produce different final solutions. Thus, our N-FINDR algorithm was implemented in iterative fashion, so that each sequential run was initialized with the previous algorithm solution, until the algorithm converges to a simplex volume that cannot be further maximized. Our experiments show that, in practice, this approach allows the algorithm to converge in a few iterations only.



a) N-FINDR initialed randomly (p=4) b) Final volume estimation by N-FINDR

Fig. 2. Graphical interpretation of the N-FINDR algorithm in a three-dimensional space.

3) 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 $\alpha = [\alpha_1, \alpha_2, \cdots, \alpha_p]$ 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:

$$\alpha = (\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T\mathbf{y}. \quad (3)$$

Two additional constrains can be imposed into the model described in (3), 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 it has been shown in practice to provide satisfactory results if the model endmembers are properly selected.

B. GPU implementation

In this subsection, we describe the GPU implementation of VD, N-FINDR and UCLS which are utilized in this work. They are carried out using the compute unified device architecture (CUDA) developed by *NVidia*^{TM}. As Fig. 3 shows, the architecture of a GPU 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 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. GPUs can be abstracted in terms of a stream model, under which all data sets are represented as streams (i.e. ordered data sets). 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 synchronization. The kernels can perform a kind of batch processing arranged in the form of a grid of blocks, where each block is composed by a group of threads that share data efficiently through the shared local memory and synchronize their execution for coordinating accesses to memory. With

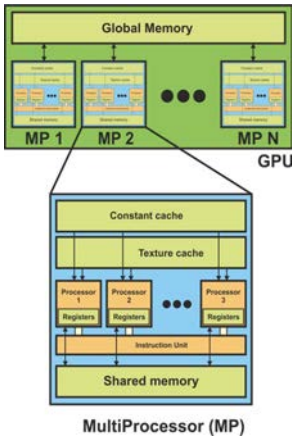


Fig. 3. Schematic overview of a GPU architecture, which can be seen as a set of multiprocessors (MPs).

the above ideas in mind, our GPU implementation of the hyperspectral unmixing chain comprises two stages: a) GPU implementation of VD; b) GPU implementation of N-FINDR; c) GPU implementation of UCLS.

1) 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 $\overline{\mathbf{Y}}$ of each band of the image and subtract 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 $\mathbf{L}$ 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 $\overline{\mathbf{Y}}$ 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 $(\mathbf{Y}-\overline{\mathbf{Y}})^T(\mathbf{Y}-\overline{\mathbf{Y}})$. 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 $\mathbf{K}_{l\times l}$ 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: $\mathbf{R}_{ij} = \mathbf{K}_{ij} + \overline{\mathbf{Y}}_i\overline{\mathbf{Y}}_j$. 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.

2) GPU Implementation of N-FINDR: Prior to the implementation of the GPU, a set of optimizations was performed. The most time-consuming computation in the N-FINDR algorithm is the calculation of the determinants. The determinant of a non-singular matrix $\mathbf{V}$ is usually obtained from the factorization $\mathbf{PV = LU}$ (where $\mathbf{P}$ is a permutation matrix, $\mathbf{L}$ is a unit lower triangular matrix, and $\mathbf{U}$ is an upper triangular matrix) as the product of the diagonal elements of $\mathbf{U}$. This decomposition is known as *Gaussian elimination* or LU factorization (with partial row pivoting). The repeated volume calculations of the N-FINDR algorithm can be reduced by exploiting some basic properties of the LU factorization and matrix determinants. Consider, i.e., the $p \times p$ and $p \times p-1$ matrices:

$$V_{\mathbf{M}}^{(1)} = \begin{bmatrix} 1 & \dots & 1 & 1 \\ \mathbf{e}_2^{(0)} & \dots & \mathbf{e}_p^{(0)} & \mathbf{x}_j \end{bmatrix}, and$$
$$\bar{V}_{\mathbf{M}}^{(1)} = \begin{bmatrix} 1 & \dots & 1 \\ \mathbf{e}_2^{(0)} & \dots & \mathbf{e}_p^{(0)} \end{bmatrix} \quad (4)$$

where $\mathbf{M}$ is the reduced version of the hyperspectral image with p components, obtained resulting from PCA transform which is also performed on GPU [18]. Assume that we have computed the LU factorization (with partial pivoting) $\mathbf{P_M}\bar{V}_{\mathbf{M}}^{(1)} = \mathbf{L_M U_M}$. Then, the LU factorization (with partial pivoting) of $V_{\mathbf{M}}^{(1)}$ is simply given by $\mathbf{P_M}V_{\mathbf{M}}^{(1)} = \left[\mathbf{U_M}(L_M^{-1}P_M^T\mathbf{x}_j)\right]$. Therefore, the LU factorizations required in the volume calculations of the N-FINDR algorithm can be all computed by simple forming the $p \times m$ matrix $\hat{\mathbf{M}} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ & & \mathbf{M}^T & \end{bmatrix}$, where $\mathbf{M}$ is the reduced hyperspectral image. Then, we need to compute $\mathbf{L_M^{-1}P_M^T}\hat{\mathbf{M}}$. This is one of the parts that we accomplished in the GPU by means of a `Volume-Calculation` kernel which obtains the volume of each pixel for one iteration. The *m* volumes required in the first iteration of teh N-FINDR algorithm are obtained from the product of the determinant of $\mathbf{U_M}$ times each one of the entries in the last row of ${\mathbf{L_M}}^{-1}{\mathbf{P_M}}^T\hat{\mathbf{M}}$. By means of a `ReductionVol` kernel, we get the value of the maximum volume and coordinates of the pixel that produces such volume. Given that $m \gg p$, this implies a significant reduction of the computational complexity of the original algorithm.

III. Proposed CBIR system

The proposed CBIR system for retrieval of hyperspectral imagery is based on the spectral unmixing methodology described in the previous section. Images are cataloged using the proposed unmixing chain and in order to accelerate the content-based searching process the extracted endmembers and their abundances are stored in the system database as binary content, so that, image retrieval is performed comparing endmembers and ground-truths restricted by a minimum abundance. In this section, we describe the system design including aspects such as description of the system architecture and the database design.

A. System Architecture

As shown by the architecture model described in Fig. 4, the proposed system can be described in different layers, which are defined and separated by roles.

1) Client layer: This layer defines the interactions between the user (through an Internet browser) and our system, and it is responsible for providing user remote access to the system.

2) Server layer: The system is considered a web service therefore the services provided by the system are managed and executed on the server layer, which is composed of several elements with different roles. As Fig. 4 shows, the

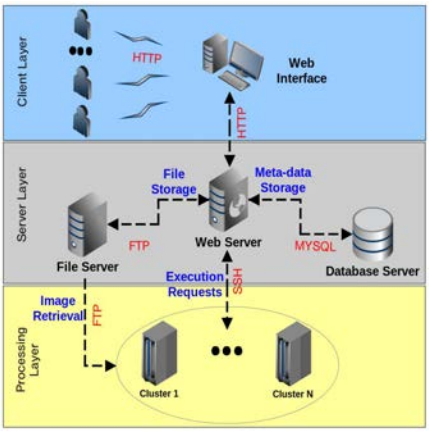


Fig. 4. Architecture

web server attends web interface requests and manages the system resources such as meta-data and file data management, in addition of handling algorithm executions. It could be considered the engine of the system since it manages and connects the components of our system. In the case of meta-data storage, *database server* stores image meta-data following the database schema describing in next subsection. *File storage server* is included for providing remote file access from any system layer, it is charged for uploading and downloading file data. Remote file storage allows to include distributed computing resources for processing high cost algorithms.

3) Processing layer: Although algorithms with low computing needs, such as query algorithms, are executed on the web server, the processing layer has been designed for executing algorithms with high computational cost, this layer relieves the web server load and provides high system availability. So that, the processing layer includes *distributed computing resources* (clusters), and the web server is charged for requesting algorithm executions and monitoring them using SSH communication. All the spectral unmixing chain algorithms are implemented in distributed resources. The current system is configured in order to support algorithm executing over two clusters with different architectures (CPU and GPU).

B. Database schema

The system consists of storing hyperspectral images and executing algorithms on these images, thus the database has been designed in order to allow storing relevant information for the hyperspectral data available through the community, in addition of information about the unmixing chain algorithms which are implemented for this work. So that, *Image, Type, Source and Publication* tables are charged for storing image features which are relevant for its analysis; *Algorithms, Type Algorithms, and Result Algorithms* tables contain information, respectively, about the supported algorithms, their place in the unmixing chain and the results of their different executions; and *Resource* table keeps the credential information of the supported clusters.

IV. Queries

The CBIR system allows an end-user to perform queries that compare the spectral endmembers associated to a given hyperspectral image with input ground truth signatures. Image endmembers are obtained through cataloguing each image with the desired extraction algorithm (OSP [9] or N-FINDR [10]), the number of endmembers to be extracted from the image is calculated using a specific estimation algorithm (HySime [7] or VD [8]), and the abundance of those extracted endmembers in the image is performed using the UCLS algorithm. The system allows to catalog images several times and every results are stored in the file storage server and their locations are referenced in the database, but in order to provide a fast content-based search system, the result with the most representative endmembers, user selected, is kept in the data base as binary content.

The *Spectral Angle Distance (SAD)* [22] is used to contrast the spectral signatures and it is a widely used metric in hyperspectral analysis. Our choice of SAD is mainly based on the fact that this distance is invariant to multiplicative scaling that may arise due to different illumination conditions and sensor observation angle. This algorithm calculates the spectral angle distance between two spectral signature vectors. So that, the angle provides a measure to compare two spectral signature vectors, useful for content-base search.

In order to evaluate the endmember abundance as percent of a given endmember in the image, we use a experimental approach which works as follows: at first the procedure we get the total sum adding all pixel values of all the abundance maps, as second step we get the partial sums adding separately all pixel values of each abundance maps, so that, we calculate the abundance percent with dividing between partial sums and the total sum.

V. Experimental results

The performance of the proposed unmixing-based CBIR system has been evaluated following two approaches, GPUs performance on catalog and matching accuracy. This section is organized as follows: First, we describe the hyperspectral data sets used in the experiments, then, we illustrate image retrieval accuracy, and finally, we analyze of GPU performance.

A. Hyperspectral data

1) Synthetic data: Since the unmixing algorithm accuracy is performance with ground truth, in order to validate the system a collection of synthetic hyperspectral images is included in the study. This collection are 35 images which are composed of known pure spectral signatures with different noise levels, spectral signatures source is the mineral spectral library from USGS Spectral Lab, version convolved to various remote sensing spectrometers such as the NASA/JPL Airborne Visual and Infra-Red Imaging Spectrometer (AVIRIS). The image collection consists in 5 types with different spectral signatures and for each type there are 7 noise levels (SNR-10, SNR-30, SNR-50, SNR-70, SNR-90, SNR-110 and no-noise), 35 images in total. In all case, the spectral resolution is of 221 narrow spectral bands between 0.4 and 2.5 micrometers, with 100×100 pixels. Fig. 5 shows those 5 types of synthetic images.

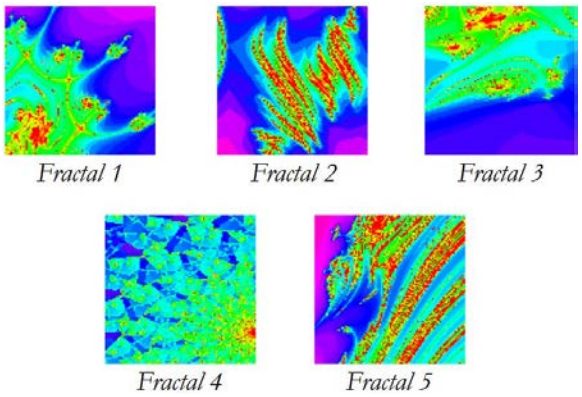


Fig. 5. Five types of synthetic hyperspectral images.

2) Real data: At the moment the repository has stored 42 images, which are 2 GigaBytes, and 7 of them are real image data sets but few of them have ground truth, thus two of the most well-known images have been included in our experiment. The first data set used in our experiments was collected by the AVIRIS sensor over the *Cuprite Mining District* in Nevada has 188 spectral bands in the range from 400 to 2500 nm and a total size of around 50 MB, the portion used in experiments corresponds to a 350×350-pixels subset. The second data set was collected by the AVIRIS sensor over the *World Trade Center (WTC)* area in New York City on September 16, which has 224 spectral bands in the range from 400 to 2500 nm and a total size of around 140 MB, the portion used in experiments corresponds to a 614×512 pixels subset.

B. Matching accuracy

Our CBIR system retrieves the images based on the matching results obtained from the ground truth and endmembers of those images. So that, in order to illustrate the performance of our CBIR, we specifically address a case study of accuracy of the matching results with images of different noise levels, using a collection synthetic and the Cuprite scene. We consider two metrics spectral angle using the spectral angle distance (SAD) for both real and synthetic images, in addition we use RMSE for evaluating the abundance maps of the synthetic images and the reconstructed image (combination of endmembers and abundances) of the real scene.

The metrics scores of the synthetic collection prove the matching accuracy of the system, and the Cuprite scene scores are relevant to illustrate the matching accuracy of the system on real images. Synthetic images are made up known pure spectral signatures which are present in the Cuprite scene, in order to simply the analysis, our experiment results are based on the first synthetic type of images, *Fractal 1*, which is quite representative and contains most of the Cuprite spectral signatures. Fig. 5 shows those 5 types of synthetic images, we can see at first position the no-noise *Fractal 1* image, which is composed of 9 spectral signatures: *KaolineiteKGA-l(wxyl), Dumortierite HS190.3B, Nontronite GDS41, Alunite GDS83 Na, Sphene HS189.3B, Pyrophyllite PYS1A fine, Halloysite NMNH10623, Muscovite GDS108 and Kaolinite CM9.*

The two accuracy metrics used in this study are spectral signature distance (SAD) and the root mean square error (RMSE) which is used to evaluate the quality of spectral unmixing results. Table I shows the spectral similarity scores and the error obtained from the abundance maps. In the case of AVIRIS Cuprite the table includes just the spectral similarity scores because we calculate the value of error map obtained after reconstructing the scene with the extracted endmembers and the abundance maps.

In the case of synthetic images the table results shows with the increase of noise, the spectral angles increase, further the smaller angles have angles close to zero. SAD [22] indicates values range between 0 and 90 grades, where 0 is the desired value, therefore the system accuracy is demonstrated. In addition, the error (RMSE) is bigger in images with highest noise level, although, we got some problems with the no-noise image results, due to the ground-truth spectral signatures are very similar and the N-FINDR method is confused with some endmembers. On the other hand, the real image (Cuprite) results have corresponding angle values between SRN-10 and SNR-30 of synthetic images and the error of the reconstructed image was 0.192, meaning real image results are quite good because the real image contains noise.

C. Analysis of GPUs performance

The proposed unmixing chain has been tested on two different platforms (GPU and core-processor):

- The GPU platform is the *NVidia*™ TESLA C2050[1], which features 448 streaming processor cores, with single precision floating point performance of 1.03 Tflops, double precision floating point performance of 515 Gflops, total memory dedicated 3 GB and memory bandwidth of 144 GB/sec. This GPU is connected to multi-core Quad Core Intel Xeon at 2.26 GHz with 4 physical cores, of which only one is used, and 24 GB of DDR3 SRAM memory. It is mounted on a Bullx R422[2].
- The second platform is a multi-core system which is used also in our experiment. It is made up of a Intel Xeon CPU X7550 at 2.00Ghz with 8 cores, of which only one is used, and 1 TeraByte of DDR3 RAM. It is mounted on a Bullx s6030 [3].

Before describing our results, it is important to emphasize that our GPU versions provides exactly the same results as the serial versions of the implemented, using the `gcc` (gnu compiler default) with optimization flag -O3. The serial algorithms were executed in one of the available cores of the multi-core system, and the GPU algorithms were executed in the used GPU architecture. For each experiment, ten runs were performed and the mean values were reported.

Usually, the delay for initialization and ignition of CUDA for the GPU device is not mentioned in the literature because CUDA is considered already ignition. But our experiment

[1]http://www.nvidia.co.uk/object/product_tesla_C2050_C2070_uk.html

[2]http://www.bull.com/catalogue/details.asp?tmp=bxs-rack-fr&opt=ns-r422e02&dt=ft&cat=bullx

[3]http://www.bull.com/catalogue/details.asp?tmp=bxs-node&opt=bullx-s6030e00&cat=bullx&dt=ft

TABLE I. MATCHING RESULTS: SPECTRAL ANGLE (IN DEGREES) AND RECONSTRUCTION ERROR (RMSE) OBTAINED BETWEEN THE GIVEN 9 USGS MINERAL SPECTRA AND THE EXTRACTED ENDMEMBERS FROM 5 SYNTHETIC IMAGES OF THE FIRST TYPE AND AVIRIS CUPRITE SCENE. ALGORITHMS USED WERE VD, N-FINDR AND UCLS ON GPU.

Spectral signatures		**Noise levels of synthetic image**					**Real image**
		10	**30**	**50**	**90**	**No-Noise**	**Cuprite**
Kaolinite CM9	**Angle**	12.345	3.421	2.844	1.365	1.765	5.251
	RMSE	0.195	0.214	0.206	0.222	1.409	-
Muscovite GDS108	**Angle**	18.847	1.682	0.236	0.169	0.169	3.618
	RMSE	0.207	0.047	0.117	0.004	0.003	-
Halloysite NMNH106236	**Angle**	24.206	2.556	0.365	0.276	0.277	16.677
	RMSE	0.237	0.065	0.013	0.053	0.002	-
Pyrophyllite PYS1A fine g	**Angle**	15.567	1.426	0.318	0.048	0.048	8.194
	RMSE	0.231	0.051	0.133	0.001	0.121	-
Sphene HS189.3B	**Angle**	26.077	3.537	0.766	0.482	0.348	4.480
	RMSE	0.279	0.151	0.042	0.004	0.319	-
Alunite GDS83 Na	**Angle**	14.746	1.529	0.157	0.066	0.065	8.299
	RMSE	0.174	0.126	0.029	0.119	0.449	-
Nontronite GDS41	**Angle**	38.714	1.605	0.223	0.135	0.135	13.710
	RMSE	0.168	0.045	0.006	0.053	0.401	-
Dumortierite HS190.3B	**Angle**	15.734	1.818	0.518	0.508	0.507	3.273
	RMSE	0.175	0.966	0.021	0.009	0.127	-
KaolineiteKGa-l (wxyl)	**Angle**	17.435	1.631	0.182	0.050	0.050	10.410
	RMSE	0.187	0.137	0.124	0.009	0.010	-

TABLE II. PROCESSING TIMES (IN SECONDS) AND SPEEDUPS ACHIEVED FOR GPU IMPLEMENTATION OF VD, N-FINDR AND UCLS ALGORITHMS, TESTED WITH CUPRITE AND WTC SCENES.

		VD			**N-FINDR**				**UCLS**		
		Initialization	**VD**	**Total**	**Initialization**	**PCA**	**N-FINDR**	**Total**	**Initialization**	**UCLS**	**Total**
AVIRIS CUPRITE	**CPU time**	0.367	16.110	16.477	0.108	9.073	2.209	11.391	0.173	1.500	1.167
	GPU time	3.582	0.127	3.709	3.545	0.074	0.252	3.871	3.574	0.057	3.631
	Speedup	-	126.850	4.442	-	122.608	8.765	2.942	-	26.316	0.321
AVIRIS World Trade Center	**CPU time**	0.802	56.274	57.076	0.325	35.301	20.158	55.784	0.517	16.457	16.974
	GPU time	3.743	0.282	4.025	3.635	0.220	2.227	6.082	3.746	0.209	3.955
	Speedup	-	199.553	14.180	-	160.459	9.665	9.172	-	78.741	4.292

works on a GPUs cluster and it is not likely that consecutive executions are performed in the same GPU device, so that, most of the times CUDA has to be initialized in each execution. We include the initialization times (GPU ignition and local data memory transfer times) in our experiment (see Table II) and we have observed these times are bigger in GPUs Cluster than in a simple GPU device [18], [19], likely because distribute resource communications delay the process. In our experiment the timing results in general were quite variable, the initialization results were the most unstable times which were in a range from 0.01 seconds to 9 seconds.

Table II summarizes the timing results and speedups measured after processing two real hyperspectral data sets by the C implementation and by the GPU implementation. The results are broken down in initialization and algorithm execution time, the speedups are calculated over the processing time and the total time (including initialization time). As shown by the table, with the increasing of the size of the image the speedup significantly increases, due to the time spent on CUDA ignition which ranges between 3 and 4 seconds. The best speedups are achieved for VD algorithm, it achieves 14.180 over the WTC scene and 4.442 over the CUPRITE scene, on the other hand, UCLS shows the worst results. All algorithms performance quite good results on GPUs and significant speedups with every scenes, even most of the speedups are high if we include the initialization time, excluding the UCLS algorithm, which produces good results only in big hyperspectral data sets as WTC. So that, GPUs algorithms provides high availability and quality of service to the CBIR system using huge images such as AVIRIS hyperspectral data sets.

VI. CONCLUSION AND FUTURE LINES

In this paper, we have developed an innovative hyperspectral image repository that allows uploading and sharing images, in addition a CBIR system for hyperspectral image retrieval based on spectral unmixing. The current implementation consists of a web application communicating with a server, in which the users can manage data through a web interface in visual way, while server manages the repository data base and algorithm executions, furthermore quality of service, time and availability are provided by GPUs algorithms. So that, this work has serial and GPU algorithm performances for single CPU and single GPU. In order to relieve the server load, distributed computing management is offered on two clusters (with CPU and GPU architectures).

The system has been implemented using well-known algorithms in the spectral unmixing community, such as VD [8]

for estimation the number of endmembers in a given scene, N-FINDR [10] for endmember extraction or UCLS for abundance estimation of which have been included both serial and GPUs implementations [18]. Our experimental results, conducted using both synthetic scenes constructed using fractals and a real hyperspectral data set collected by NASA's Airborne Visible Infrared Imaging Spectrometer (AVIRIS), indicate that the proposed CBIR system can accurately extract hyperspectral image instances from a complex image database with sub-pixel precision and quickly enough for practical use. This is accomplished by resorting to available parallel implementations of the considered spectral unmixing chain in different types of high performance computing architectures. As result, we believe that the proposed system can be a standardized hyperspectral data repository data intended to distribute and share hyperspectral data sets in the community.

As future extension of the system, we will implement Multi-GPUs algorithms for GPUs clusters. Furthermore, we plan develop adaptable algorithms to any GPUs platform that may provide better performance on execution of cataloging algorithms.

ACKNOWLEDGMENT

This work has been supported by the project AYA2011-29334-C02-02. West University of Timisoara supported the system implementation to execute algorithms in a distributed environment, in addition to provide computing resources. And 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.

REFERENCES

[1] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Fast dimensionality reduction and simple PCA," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1349–1380, 2000.

[2] C.-I. Chang, *Hyperspectral Imaging: Techniques for Spectral Detection and Classification*. Kluwer Academic/Plenum Publishers: New York, 2003.

[3] 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*, vol. 65, no. 3, pp. 227–248, 1998.

[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 Sensing of Environment*, vol. 113, pp. 110–122, 2009.

[5] G. Camps-Valls and L. Bruzzone, "Kernel-based methods for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, pp. 1351–1362, 2005.

[6] 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," *Journal of Geophysical Research*, vol. 91, pp. 8098–8112, 1986.

[7] J. M. Bioucas-Dias and J. M. P. Nascimento, "Hyperspectral subspace identification," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 8, pp. 2435–2445, 2008.

[8] C.-I. Chang and Q. Du, "Estimation of number of spectrally distinct signal sources in hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 3, pp. 608–619, 2004.

[9] J. C. Harsanyi and C.-I. Chang, "Hyperspectral image classification and dimensionality reduction: An orthogonal subspace projection," *IEEE Trans. Geosci. Remote Sens.*, vol. 32, pp. 779–785, 1994.

[10] M. E. Winter, "N-FINDR: An algorithm for fast autonomous spectral endmember determination in hyperspectral data," *Proceedings of SPIE*, vol. 3753, pp. 266–277, 1999.

[11] A. Plaza, J. Plaza, and A. Paz, "Content-based image retrieval at the end of the early years," *Concurrency and Computation: Practice and Experience*, vol. 9, pp. 1138–1159, 2010.

[12] M. Veganzones, J. Maldonado, and M. Grana, "On content-based image retrieval systems for hyperspectral remote sensing images," in *Computational Intelligence for Remote Sensing*, ser. Studies in Computational Intelligence, M. Graa and R. Duro, Eds. Springer Berlin Heidelberg, 2008, vol. 133, pp. 125–144.

[13] A. P. A. Plaza, J. Plaza and S. Blazquez, "Parallel cbir system for efficient hyperspectral image retrieval from heterogeneous networks of workstations," *Proc. Int. Symp. Symbolic Numeric Algorithms Sci. Computing*, pp. 285–291, 2007.

[14] M. Grana and M. A. Veganzones, "An endmember-based distance for content based hyperspectral image retrieval," *Pattern Recognition*, vol. 45, no. 9, pp. 3472 – 3489, 2012, best Papers of Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA 2011).

[15] M. A. Veganzones and M. Grana, "A spectral/spatial cbir system for hyperspectral images," *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, vol. 5, no. 2, pp. 488–500, 2012.

[16] F. L. A. P. A. Barberis, G. Danese and E. Torti, "Real-time implementation of the vertex component analysis algorithm on gpus," *IEEE Geoscience and Remote Sensing Letters*, vol. 10, pp. 251–255, 2013.

[17] S. Sanchez and A. Plaza, "Fast determination of the number of endmembers for real-time hyperspectral unmixing on gpus," *Journal of Real-Time Image Processing*, pp. 1–9, 2012. [Online]. Available: http://dx.doi.org/10.1007/s11554-012-0276-3

[18] S. Sanchez, R. Ramalho, L. Sousa, and A. Plaza, "Real-time implementation of remotely sensed hyperspectral image unmixing on gpus," *Journal of Real-Time Image Processing*, pp. 1–15, 2012. [Online]. Available: http://dx.doi.org/10.1007/s11554-012-0269-2

[19] S. Bernabe, S. Sanchez, A. Plaza, S. Lopez, J. Benediktsson, and R. Sarmiento, "Hyperspectral unmixing on gpus and multi-core processors: A comparison," *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, vol. PP, no. 99, pp. 1–1, 2013.

[20] A. Plaza, P. Martinez, R. Perez, and J. Plaza, "A quantitative and comparative analysis of endmember extraction algorithms from hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, pp. 650–663, 2004.

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

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