FAST SPATIAL-SPECTRAL PREPROCESSING FOR ENDMEMBER EXTRACTION AND SPECTRAL UNMIXING USING GRAPHIC PROCESSING UNITS

L. I. Jiménez[1], *G. Martín*[2], *S. Sánchez*[1], *J. Plaza*[1], *A. Plaza* [1]

[1] Hyperspectral Computing Laboratory, University of Extremadura, Cáceres, Spain
[2]Instituto de Telecomunicações, Lisbon, Portugal

ABSTRACT

Linear spectral unmixing consists on the identification of spectrally pure constituents, called *endmembers* and their corresponding proportions or *abundances* using a linear model. Traditionally, most of the attention has been focussed on the exploitation of spectral information when identifying a set of endmembers and, only recently, some techniques try to take advantage of complementary information such as the one provided by the spatial correlation of the pixels in the image. Computational complexity represents a major problem in most of these spatial-spectral based techniques, as hyperspectral images provide very rich information in both the spatial and the spectral domain. In this paper we provide a computationally efficient implementation of a spatial-spectral processing (SSPP) algorithm which can be used prior to endmember identification and spectral unmixing. Specifically we present an implementation optimized for commodity graphics processing units (GPUs), which is evaluated using two different GPU architectures from NVidia: GeForce GTX580 and GeForce GT740. Our experimental validation reveals that significant speedups can be achieved when processing hyperspectral images of different sizes.

Index Terms— Hyperspectral imaging, spatial-spectral preprocessing, graphics processing units (GPUs).

1. INTRODUCTION

The wealth of spectral information provided by hyperspectral instruments has promoted the application of hyperspectral imaging techniques in many different areas of interest, transforming it into a commodity product available to broad user community [1]. In hyperspectral imaging, endmember extraction is the process of selecting a collection of pure signature spectra of the materials present in a remotely sensed hyperspectral scene. These pure signatures are then used to decompose the scene into a ser of so-called abundance fractions by means of a spectral unmixing algorithm.

Linear mixture modeling is the most widely used spectral umixing approach. Several algorithms for automatic or semiautomatic spectral endmember identification have been developed over the last decade. Classic techniques include, among many others, the pixel purity index (PPI) [2], the vertex component analysis (VCA) [3], the orthogonal subspace projection (OSP) [4], the N-FINDR algorithm [5], or the iterative error analysis (IEA) [6]. A majority of algorithms have been developed under the pure signature assumption, i.e., they assume that the remotely sensed data contain one pure observation for each different material in the scene [7]. Most importantly, all the aforementioned algorithms rely exclusively on the exploitation of spectral information in order to select the final set of endmembers. With the aim of taking advantage also of spatial information, several techniques have also been proposed in the literature, such as the automatic morphological endmember extraction (AMEE) [8] or the spatial-spectral endmember extraction (SSEE) [9]. Further, several preprocessing algorithms have been developed that can be applied prior to any spectral endmember extraction technique like spatial preprocessing (SPP) [10] or spatial-spectral preprocessing (SSPP) [11]. The goal of these preprocessing methods is to guide the spectral endmember search procedure using spatial-based considerations. The spatial preprocessing algorithm adds extra computational cost to the unmixing process. The generation of efficient implementations for these techniques has become an important goal.

In this paper, we present a parallel design for commodity graphics processing units (GPUs) of the SSPP algorithm, which is one of the most successful spatial preprocessing strategies available in the literature [11]. Our implementation is evaluated on two different NVidia GPU architectures: GeForce GTX580 and GeForce GT740 using hyperspectral imagery with different spatial dimensions. Our experimental validation reveals that significant speedups can be achieved for the SSPP, enabling the possibility to embed it into a full hyperspectral unmixing chain with real-time expectations.

2. SPATIAL-SPECTRAL PREPROCESSING

In this section we describe the considered SSPP approach. As shown by the flowchart in Fig. 1, the proposed method consists of four steps that can be summarized as follows:

1. *Multi-scale Gaussian filtering*. This step takes as input the original hyperspectral image and returns a set of fil-

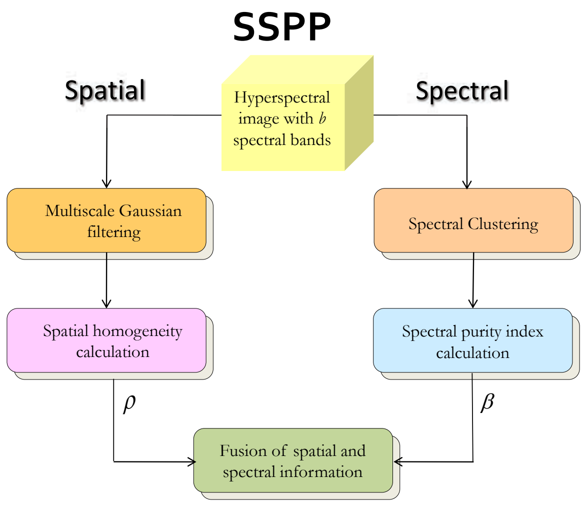
SSPP
Spatial
Spectral
Hyperspectral image with b spectral bands
Multiscale Gaussian filtering
Spectral Clustering
Spatial homogeneity calculation
Spectral purity index calculation
ρ
β
Fusion of spatial and spectral information

Fig. 1. Block diagram illustrating our spatial-spectral preprocessing (SSPP) method.

tered hyperspectral images. To perform this step, we first apply Gaussian filtering to each of the b spectral bands of the original hyperspectral image using different values of σ in the Gaussian filter, which results in different filtered versions of the original hyperspectral image.

2. *Spatial homogeneity calculation*. This step takes as input the set of filtered hyperspectral images obtained in the previous step and produces a spatial homogeneity index for each pixel in the original image. To perform this step, we first calculate the root mean square error (RMSE) [12] between the original hyperspectral image and each of the filtered images (the lower the RMSE score is in a pixel, the higher the similarity between the pixel in the original image and its neighbours). As a result, the RMSE can be used as a spatial homogeneity index for each pixel in the hyperspectral image.

3. *Spectral purity index calculation*. For this step we use a PCA-reduced version of the original hyperspectral image. First, we calculate the p first components and then calculate the maxima and minima projection values by means of a dot product computation, using the first principal components as the directions for which we identify the pixels with maxima and minima projection values. The pixels with maxima and minima projection are assigned a weight of 1. The weight of the mean value between the maxima and minima projection value is 0. A threshold value is also applied so that the weights lower than this threshold are assigned the value 0. Finally the spectral purity index will be the sum of all the weights over the firs p principal components.

4. *Spectral clustering*. In parallel to the previous steps, we perform a spectral-based unsupervised clustering of the original hyperspectral image. This step, which is applied in parallel to the first two steps, takes as input the original hyperspectral image and returns a set of clusters in which the original image is partitioned. Spectral-based unsupervised ISODATA algorithm [13] is used in [11], where the minimum number of classes was set to p and the maximum number of classes was set to $2p$. Resulting from this step, a number of clusters (comprised between p and $2p$) are identified in the original hyperspectral image.

5. *Fusion of spatial and spectral information*. This step takes as input the spatial homogeneity index calculated in the second step and the clusters calculated in third step, and returns a subset of pixels in the original hyperspectral image which will be used for endmember identification purposes (combining the spatial and the spectral information obtained in the previous steps). For each cluster in the spectral classification map, a subset of spatially homogeneous pixels is selected. On the other hand, a subset of spectrally pure pixels is selected for each cluster. For selection, pixels are ranked according to increasing values of their spatial homogeneity and spectral purity as calculated in the previous step. Finally, endmember extraction can be applied to the pixels retained after the procedure above.

3. GPU IMPLEMENTATION

In this section we describe the parallel implementation of the SSPP algorithm using NVidia Compute Unified Device Architecture (CUDA). We will focus only on steps 1-3 of the algorithm, as they are the most expensive from a computational point of view. It should also be noticed that, in order to achieve the optimal overlapping between CPU and GPU code, some parts of the parallel implementation will remain in the CPU (such as the filter creation or the calculation of the RMSE scores). The proposed parallel version of SSPP is based on six main CUDA kernels:

1. The first and second kernels compute the multi-scale Gaussian filtering (*convolutionRowSymmGPU* and *convolutionColSymmGPU*), calculating the image symmetric Gaussian filtering using the convolution of two 1-dimensional filters. Here the number of threads is set to 128 and the number of blocks equals the number of pixels divided by the number of threads. As mentioned before, Gaussian filtering is applied to each of the spectral bands of the image so each of the two kernels is b times.

2. The third kernel, *AvgXCUDA* calculates the normalized image obtained by subtracting the average pixel to the

original image using a reduction operation. The number of blocks is set to the number of bands b and the number of threads is set to the maximum allowed by the GPU.

3. We have also developed a kernel called *BitonicSort* to replace Quicksort algorithm (previously executed in the CPU) to execute a more effective bitonic sort in the GPU with very low latency [14], obtaining much better performance. In this case the number of threads is set to 256 and the number of blocks is calculated according to Eq. (1) below:

$$nblocks = \exp(\log_2 \lfloor \frac{npixels}{nthreads} \rfloor) \tag{1}$$

4. After executing the principal component analysis step, the rest of the spectral purity index calculation has been implemented though kernels *minimaxbands* and *weights*. The first one calculates the maxima and minima projections in the PCA domain, using a reduction operation. The second kernel obtains the weights of each minima and maxima projections on each principal component. The grid dimension of the *inimaxbands* kernel is set to the number of estimated endmembers (principal components), while the grid dimension in the *weights* kernel is obtained by dividing the number of pixels by the block size, which in both cases is set to the maximum allowed by the GPU architecture.

4. EXPERIMENTAL RESULTS

A collection of 24 synthetic hyperspectral images has been used for validation. These scenes are composed of known pure spectral signatures of different sizes, in the range between 10000 and 200000 pixels, synthesized from spectral signatures extracted from the USGS spectral library [15]. The procedure for constructing the images is described in 2 (more details about the procedure can be found in [11, 16]). In all cases, the images contain sets of (10, 20, 30) endmembers randomly selected from the USGS library, and their spectral resolution is of 221 narrow spectral bands between 0.4 and 2.5 micrometers per scene.

The GPU implementation of SSPP has been tested on two different architectures: a desktop computer (Intel core i7 920 CPU at 2.67 GHz and 6 GB of RAM) with a GPU NVidia GTX 580, which features 512 processor cores operating at 1.54 GHz, and a laptop computer (Intel core i7-4700MQ at 2.4 GHz and 8 GB of RAM) with a GPU Nvidia GT740, with 384 processor cores at 993 MHz.

The serial algorithm was executed in one of the available cores of computer, and the parallel times were measured in the considered GPU platform. The speedups are calculated between each pair CPU/GPU (regarding its counterpart). For each experiment, 10 runs were performed and the mean values are reported (these times were always very similar, with differences on the order of a few milliseconds only).

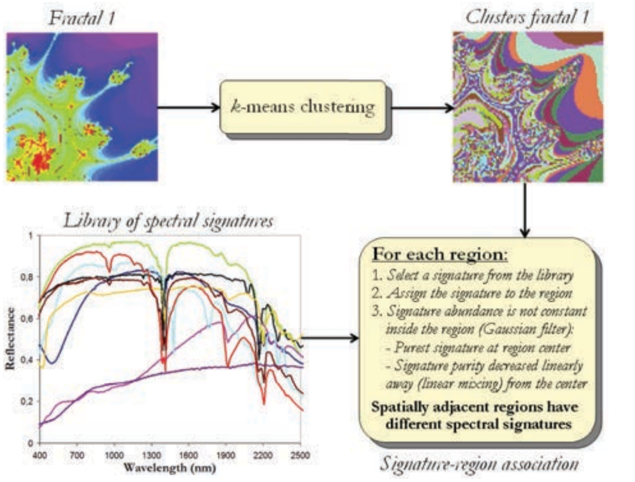
Fractal 1
k-means clustering
Clusters fractal 1
Library of spectral signatures
Reflectance
Wavelength (nm)
For each region:
1. Select a signature from the library
2. Assign the signature to the region
3. Signature abundance is not constant inside the region (Gaussian filter):
- Purest signature at region center
- Signature purity decreased linearly away (linear mixing) from the center
Spatially adjacent regions have different spectral signatures
Signature-region association

Fig. 2. Block diagram describing the procedure for generating synthetic hyperspectral images

Table 1. Mean execution times and speedups for the parallel version of the SSPP algorithm executed on the Nvidia GT740 GPU after 10 Monte-Carlo runs.

	# Endmembers					
	10		**20**		**30**	
Image size	**time**	**speedup**	**time**	**speedup**	**time**	**speedup**
100×100	0.283	2.634	0.273	2.783	0.272	2.836
200×100	0.382	3.706	0.363	4.020	0.374	3.973
300×100	0.466	4.510	0.454	4.731	0.462	4.755
400×100	0.570	4.869	0.556	5.123	0.551	5.270
100×500	0.667	5.187	0.672	5.293	0.656	5.546
200×500	1.117	6.124	1.116	6.289	1.127	6.381
300×500	1.568	6.518	1.562	6.753	1.563	6.918
400×500	2.002	6.839	1.992	7.045	1.991	7.245

Tables 1 and 2 show the execution times and speedups obtained by GPU implementation (executed in the GT740 and GTX580 respectively) regarding the C version (executed in one core of the laptop and desktop computer) considering a set of different size images containing different number of endmembers.

As revealed by Tables 1 and 2, it is easy to achieve a reasonable speedup when executing our parallel SSPP algorithm by simply combining the CPU core with the available GPU on each of the compared computers. Considering that the serial implementation needs 7.076 seconds in the desktop computer and 14.426 seconds in the laptop computer to process the largest synthesized image (400×500 pixels), we obtain an speedup of almost $10\times$. Last but not least, the real time requirements of a standard imaging spectrometer such as AVIRIS establish that we need to process a 614×512 pixels of 224 spectral bands in less than 5.09 seconds, which means

Table 2. Mean execution times and speedups for the parallel version of the SSPP algorithm executed on the Nvidia GTX580 GPU after 10 Monte-Carlo runs.

	# Endmembers					
	10		**20**		**30**	
Image size	**time**	**speedup**	**time**	**speedup**	**time**	**speedup**
100×100	0.213	1.965	0.215	1.945	0.211	1.997
200×100	0.244	3.107	0.234	3.239	0.236	3.260
300×100	0.267	4.085	0.262	4.215	0.256	4.321
400×100	0.300	4.795	0.296	4.896	0.287	5.100
100×500	0.317	5.626	0.311	5.812	0.309	5.922
200×500	0.426	8.238	0.424	8.491	0.419	8.570
300×500	0.548	9.550	0.540	9.757	0.539	9.815
400×500	0.734	9.309	0.754	9.342	0.753	9.392

that the proposed GPU implementation allow us to perform the spatial-spectral preprocessing of the image in real-time performance. Further, the SSPP method can also be integrated in a full hyperspectral unmixing chain without reducing the expectation of the full chain to be conducted also in real time.

5. CONCLUSIONS AND FUTURE LINES

In this paper, we have presented a new GPU implementation of a spatial-spectral preprocessing (SSPP) algorithm. The obtained experimental results indicate that it is possible to achieve significative speedups by overlapping the execution of the algorithm in CPU/GPU. Future work will be focused on the improvement of this implementation by studying different parallelization schemes, generating CUDA versions for the last steps of the SSPP algorithm. We also plan to develop other implementations of full spectral unmixing chain (including spatial preprocessing), using hardware devices such as field programmable gate arrays (FPGAs).

6. REFERENCES

[1] A. Goetz; G. Vane; J. Solomon and B. Rock, "Imaging spectrometry for earth remote sensing," *Science*, vol. 228, no. 4704, pp. 1147–1153, 1985.

[2] J. W. Boardman, F. A. Kruse, and R. O. Green, "Mapping Target Signatures Via Partial Unmixing of Aviris Data," *Proc. JPL Airborne Earth Sci. Workshop*, pp. 23–26, 1995.

[3] J. M. P. Nascimento and J. M. Bioucas-Dias, "Vertex Component Analysis: A Fast Algorithm to Unmix Hyperspectral Data," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 4, pp. 898–910, 2005.

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

[5] M.E. Winter, "N-FINDR: an algorithm for fast autonomous spectral end-member determination in hyperspectral data," in *Proceedings of SPIE*, 1999, vol. 3753, pp. 266–270.

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

[7] J. Plaza, E. M. T. Hendrix, I. Garcia, G. Martin, and A. Plaza, "On endmember identification in hyperspectral images without pure pixels: A comparison of algorithms," *Journal of Mathematical Imaging and Vision*, vol. 42, no. 2-3, pp. 163–175, 2012.

[8] A. Plaza, P. Martinez, R. Perez, and J. Plaza, "Spatial/spectral endmember extraction by multidimensional morphological operations," *IEEE Trans. Geosci. Remote Sens.*, vol. 40, pp. 2025–2041, 2002.

[9] D. M. Rogge, B. Rivard, J. Zhang, A. Sanchez, J. Harris, and J. Feng, "Integration of spatial–spectral information for the improved extraction of endmembers," *Remote Sens. Environ.*, vol. 110, no. 3, pp. 287–303, 2007.

[10] M. Zortea and A. Plaza, "Spatial preprocessing for endmember extraction," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 8, pp. 2679–2693, 2009.

[11] G. Martin and A. Plaza., "Spatial-spectral preprocessing prior to endmember identification and unmixing of remotely sensed hyperspectral data," *IEEE J. Sel. Topics Appl. Earth Observations Remote Sens.*, vol. 5, no. 2, pp. 380–395, 2012.

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

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

[14] I. Buck; T. Purcell, *GPU Gems*, chapter A toolkit for computation in GPU, pp. 621–636, Nvidia, 2004.

[15] R. N. Clark; G.A. Swayze; A. Gallagher; T.V.V. King, "The u. s. geological survey, digital spectral library: Version 1: 0.2 to 3.0 microns," Open File Report 93-592, U.S. Geological Survey, 1993.

[16] J. Sevilla; A. Plaza, "A new digital repository for hyperspectral imagery with unmixing-based retrieval functionality implemented on gpus," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 6, pp. 2297–2304, 2014.