PARALLEL IMPLEMENTATION OF VERTEX COMPONENT ANALYSIS FOR HYPERSPECTRAL ENDMEMBER EXTRACTION

José M. Rodríguez Alves[a], José M. P. Nascimento[a,b], José M. Bioucas-Dias[a,c], Vítor Silva[a,d], Antonio Plaza[e]

[a]Instituto de Telecomunicações, Lisbon, Portugal
[b]Instituto Superior de Engenharia de Lisboa, Lisbon, Portugal
[c]Instituto Superior Técnico, Lisbon, Portugal
[d]Instituto de Telecomunicações, DEEC, University of Coimbra, Coimbra, Portugal
[e]Hyperspectral Computing Laboratory, University of Extremadura, Cáceres, Spain

ABSTRACT

Vertex component analysis (VCA) has become a very popular and useful tool to linear unmix large hyperspectral datasets without the use of any a priori knowledge of the constituent spectra. Although VCA is fast method, many hyperspectral imagery applications require a response in real time or near-real time.

This paper proposes two different optimizations for accelerating the computational performance of VCA: the first one focus a parallel implementation based on graphics computing units (GPUs) to alleviate the VCA computational burden; The second one is focused on the development of a strategy to remove a large proportion of mixed pixels that play no effect on the VCA functioning.

Experiments are conducted using simulated and real hyperspectral datasets. These results reveal considerable acceleration factors, which satisfies the real-time constraints given by the data acquisition rate.

Index Terms— Hyperspectral Unmixing, Endmember Extraction, Vertex Component Analysis, Graphics Processing Unit, Parallel Methods.

1. INTRODUCTION

Remotely sensed hyperspectral images collect electromagnetic energy scattered within their ground instantaneous field of view in hundreds or thousands of nearly contiguous spectral bands with high spectral resolution [1, 2]. This technology provides enough spectral resolution for material identification, facilitating an enormous number of applications in the fields of military surveillance, target detection, environmental monitoring, oil spill and other types of chemical contamination detection, biological hazards prevention, and food safety [2].

This work was supported by FCT-IT under project PEst-OE/EEI/LA0008/2011.

Due to low spatial resolution provided by these devices and due to other effects (see [2, 3] for more details), several spectrally distinct materials (also called *endmembers*) can be found within the same pixel. Thus, each pixel can be viewed as a mixture of the endmembers signatures. Linear mixture model consider that these mixed pixels are a linear combination of the endmember signatures present in the scene weighted by the correspondent abundance fractions (*i. e.*, the percentage of each endmember). Consider a set of N observed vectors (pixels) of an hyperspectral image, where each pixel denoted by $\mathbf{y} \in \mathbb{R}^L$ (L is the number of bands) is given by

$$\mathbf{y} = \mathbf{M}\mathbf{s} + \mathbf{n}, \tag{1}$$

where $\mathbf{M} \equiv [\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_p]$ is a full-rank $L \times p$ mixing matrix ($\mathbf{m}_j$ denotes the jth endmember signature), p is the number of endmembers present in the covered area (with $p < L$), $\mathbf{s} = [s_1,\ s_2, \ldots,\ s_p]^T$ is the abundance vector containing the fractions of each endmember, and $\mathbf{n}$ is additive noise vector (notation $(\cdot)^T$ stands for vector transposed). To be physically meaningful [3], abundance fractions are subject to nonnegativity and full additivity constraints, thus, abundance fractions are in the $p-1$ probability simplex, *i.e.* $\{\mathbf{s} \in \mathbb{R}^p : s_j \geq 0, \sum_{j=1}^{p} s_j = 1\}$. Considering that the columns of $\mathbf{M}$ are affinely independent, the observed spectral vectors in a given scene are in a $p-1$ simplex in $\mathbb{R}^L$ whose vertices correspond to the endmembers.

Hyperpectral unmixing is a very important task in remotely sensed hyperspectral data exploitation. It amounts at estimating the number endmembers (p), their spectral signatures ($\mathbf{M}$), and their respective abundance fractions ($\mathbf{s}$) [2]. Over the last decade several approaches have been developed to automatically perform this task, exploiting geometrical and statistical concepts [4].

Geometrical methods are based on the fact that, under the linear mixing model, hyperspectral vectors belong to a simplex set whose vertices correspond to the endmembers signatures. Therefore, finding the endmembers is equivalent to identifying the vertices of the referred to simplex [1]. These

methods can be classified into pure-pixel-based and non-pure-pixel-based methods. In the former case, it is assumed the presence of at least one pure pixel per endmember in the dataset, meaning that there is at least one spectral vector on each vertex of the data simplex. This assumption, allows the design of very efficient algorithms from the computational point of view, some popular algorithms taking this assumption are *vertex component analysis* (VCA), [1], the *automated morphological endmember extraction* (AMEE) [5], the *pixel purity index* (PPI), [6], and the N-FINDR [7, 8].

VCA has become a very popular and useful tool to unmix large hyperspectral datasets in unsupervised fashion. VCA is very fast, light from the computational point of view, and accurate method that works with and without dimensionality reduction. This method, since its publication has had an enormous impact in the scientific community, being used on different areas such as band selection, clustering, and to initialize other unmixing methods. Recently, VCA has been optimized and implemented in different ways and in different platforms towards the real time or near-real time required by many of the above mentioned applications [9, 10, 11, 12, 13].

Thus, to met this requirement two different acceleration strategies are proposed in this paper: **i)** To alleviate the VCA computational burden, it is desirable to implement it in parallel. Recently, graphics computing units (GPUs) have become a topic of considerable interest due to their extremely high floating-point processing performance, huge memory bandwidth, compact size, and their comparatively low cost, which have made them appealing for implementation of hyperspectral imaging algorithms [14, 15]. In particular, they may be suitable in the future for real-time processing of hyperspectral data. This paper presents a GPU-based implementation of the VCA algorithm; **ii)** Another optimization strategy presented in this paper focuses on the fact that VCA computational complexity is proportional to the number of pixels of the dataset. It is found that not all the mixed pixels play a role to find the vertices of the simplex. Therefore, a new preprocessing technique is proposed to remove a large proportion of mixed pixels that play no effect on the VCA functioning. This procedure will cut down greatly the VCA computational complexity.

The remainder of this paper is organized as follows. Section 2 briefly describes the original VCA method, the pixel removal optimization and the new and fully optimized GPU implementation of VCA. Section 3 evaluates the proposed optimizations implementation in terms computational complexity. Section 4 outlines the conclusions of the paper with some remarks and future research lines.

2. VCA UNMIXING METHOD

VCA is an unsupervised method to unmix linear mixtures of hyperspectral datasets and is based on the geometry of convex sets. It exploits two facts: **1)** the endmembers are the vertices of a simplex and **2)** the affine transformation of a simplex is also a simplex. VCA is a fully automatic algorithm and it works with and without dimensionality reduction preprocessing step. Has mentioned before, there are many real situations where the hyperspectral vectors live in a subspace of very low dimension compared with the available number of bands ($p \ll L$) and, thus, it is advantageous, in terms of signal-to-noise ratio (SNR), memory usage, and computational complexity, to represent the spectral vectors in a signal subspace basis [16].

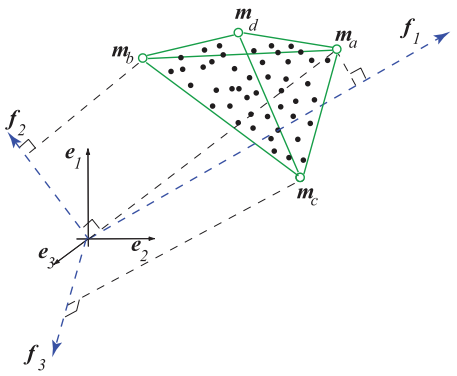

Fig. 1. Tree-dimensional diagram of an hyperspectral mixture of four endmembers illustrating the VCA algorithm. Endmembers (circles); mixed pixels (dots).

The VCA algorithm iteratively projects data onto a direction orthogonal to the subspace spanned by the endmembers already determined. The new endmember signature corresponds to the extreme of the projection. The algorithm iterates until all endmembers are exhausted.

Fig. 1 illustrates the VCA method working on a simplex defined by a mixture of four endmembers where circles and dots represent pure-pixels (endmembers signatures) and mixed pixels. In the first iteration, data is projected onto the first direction $\mathbf{f}_1$. The extreme of the projection corresponds to endmember $\mathbf{m}_a$. In the next iteration, endmember $\mathbf{m}_b$ is found by projecting data onto direction $\mathbf{f}_2$, which is orthogonal to $\mathbf{m}_a$. Then, a new direction $\mathbf{f}_3$, orthogonal to the subspace spanned by $\mathbf{m}_a$ and $\mathbf{m}_b$ is generated and the endmember $\mathbf{m}_c$ is found by seeking the extreme of the projection of the dataset onto $\mathbf{d}_3$. VCA algorithm iterates until all p endmembers are found (Full VCA algorithm can be found in [1]).

2.1. VCA Acceleration Strategies

2.1.1. Pixel Removal

Considering that VCA is applied for the dataset after the projection to the signal subspace [16], the computational complexity of VCA is $2p^2N$. Assuming that $\mathbf{Y} \equiv [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N]$ is a $p \times N$ matrix where each column corresponds to the projection of the spectral vectors onto the signal subspace and $\mathbf{f}_k$

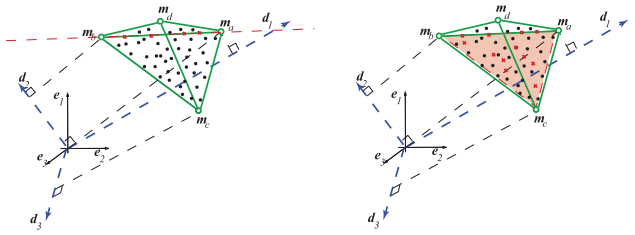

Fig. 2. Mixture of four endmembers illustrating the pixels removal. (a) third iteration; (b) fourth iteration. Endmebers (circles); mixed pixels (dots); removed pixels (crosses).

is the $p \times 1$ vector representing the direction on k-th iteration, the most consuming time operation is the projection of $\mathbf{Y}$ onto direction $\mathbf{d}_k$, *i.e.* the matrix-vector product $\mathbf{f}_k^T \mathbf{Y}$.

Since VCA computational complexity is proportional to the number of pixels of the dataset, one way to reduce the computational burden is to reduce the number of pixels to be projected on each iteration. It is found that on each iteration vectors that belong to the subspace spanned by the endmembers found so far have null projection onto the new direction, thus they can be removed since they are not candidates to be endmembers. Fig. 2 illustrates the pixel removal strategy functioning after two iterations. After the determination of the first two endmembers $\mathbf{m}_a$ and $\mathbf{m}_b$ the direction $\mathbf{f}_3$ is generated. Note that all pixels in the edge $\{\mathbf{m}_a, \mathbf{m}_b\}$ (denoted by red crosses on Fig. 2(a)) can be removed since they are orthogonal to direction $\mathbf{f}_3$. On the next iteration, the new direction will be orthogonal to subspace spanned by $\mathbf{m}_a$, $\mathbf{m}_b$ and $\mathbf{m}_c$, thus all pixels inside this facet of the simplex can be removed (illustrated by the shadow area on 2(b)).

2.1.2. Pixel Selection

Another acceleration strategy is based on the fact that endmembers have extreme values on one (or more bands) thus the scheme is based on the selection of those pixels that fall on the extreme ends of each band. This strategy will reduce drastically the number of pixels to be projected, reducing the computational complexity of VCA.

2.2. Parallel VCA

As referred in the previous section the computational intensive part of VCA is the the matrix-vector product $\mathbf{f}_k^T \mathbf{Y}$. To alleviate the computational burden it is desirable to implement it in parallel. It is worth noting that each pixel projection can be done independently from the other pixels projections. Recently, graphics computing units (GPUs) has become a topic of considerable interest due to their extremely high floating-point processing performance, huge memory bandwidth and their comparatively low cost [15].

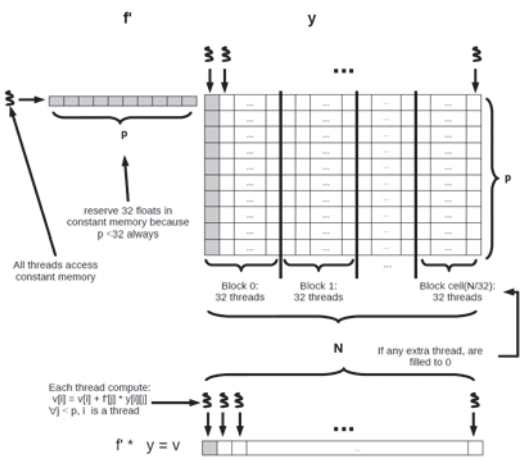

Fig. 3. Illustration of parallel VCA in the GPU.

The proposed implementation exploits the GPU architecture at low-level, using global memory to map the hyperspectral image. The scheme is as follows, on each iteration the generation of the direction $\mathbf{f}_k$ is performed on the CPU and transferred to the constant memory of the device, then a kernel is used to compute (in parallel) the pixel projection on the GPU. Finally, the result of the product is then processed by the CPU to select the new endmember. Fig. 3 illustrates the proposed parallel implementation, where the dataset is divided into blocks, and each block has 32 threads. Note that, to fully optimized the memory transfer procedure, we have used the *float4* variable data type and the minimum memory allocation is 32 float, filling the unused floats with zeros.

3. PERFORMANCE EVALUATION

In this section, we apply the sequential and the parallel VCA to simulated scenes and to the Cuprite dataset collected by the AVIRIS sensor. Several simulated scenes where created with different number of pixels and different number of endmembers. Each pixel is generated according to expression (1), the spectral signatures are selected from the USGS digital spectral library[1], containing 224 spectral bands covering wavelengths from 0.38 to 2.5 μm with a spectral resolution of $10\,nm$. The abundance fractions are generated according to a Dirichlet distribution which enforces positivity and full additivity constraints (see [17] for details). Regarding Cuprite subset it contains 350×350 pixels with 187 spectral bands.

Concerning the sequential VCA is implemented in C programming language running on a computer platform equipped with a Intel i7-2600, with 16 Gbyte memory and the parallel VCA is CUDA-based implementation on a GPU card equipped with a GTX-590 from NVidia[2] with 1024 CUDA

[1] http://speclab.cr.usgs.gov/spectral-lib.html
[2] www.geforce.com/hardware/desktop-gpus/geforce-gtx-590

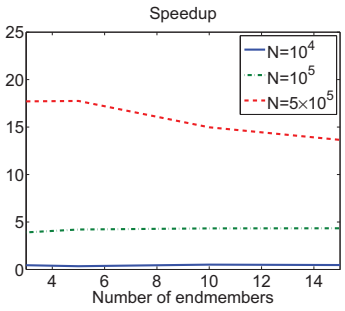

Fig. 4. VCA speedup as a function of the number of endmebers (for different number of pixels).

Table 1. Percentage of removed pixels for sequential VCA as a function of the iterations ($p = 10$).

iteration	1	2	3	4	5
removed pixels (%)	0.0	0.0	0.3	0.8	1.8
iteration	6	7	8	9	10
removed pixels (%)	4.3	6.9	12.0	15.3	25.0

cores and 3GB of memory. Figure 4 illustrates the achieved speedup as a function of the number of endmembers for different number of pixels. Note that the significant acceleration factors is higher as the number of pixels to be processed is larger. For the real dataset the speedup achieved is about 4. Table 1 presents the number of pixels removed by the strategy presented in Section 2.1.1 for a simulated dataset with ten endmembers. Notice that the the number of pixels removed increases with the iterations thus the computational complexity becomes smaller. For the Cuprite dataset, where the pixels are highly mixed [17], the total of removed pixels are less than 5%. Table 2 presents the number pixels selected by the strategy presented in Section 2.1.2 for the simulated datasets. It is worth mention that the percentage of selected pixels decreases as the total number of pixels become larger. Concerning the tests for the Cuprite dataset the selected pixels is 85, which is in accordance with the results for the simulated scenarios.

4. CONCLUSIONS

Vertex component analysis (VCA) [1] is one of the most used tool on hyperspectral imagery applications. Although VCA is a fast method, those applications very often require a response in real time or near-real time.

This paper proposes and develops two different optimizations strategies for accelerating the computational performance of VCA: the first implements a parallel VCA on graphics computing units (GPUs); The second proposes a strategy to remove a large proportion of mixed pixels that play no effect on the VCA functioning. The results obatined on the conducted experiments reveal considerable acceleration factors, which can satisfy the real-time requirements. Future work, include test on more recent GPU hardware and on the the design and implementation of the VCA on FPGAs.

Table 2. Number of selected pixels for sequential VCA as a function of the p and N.

	$p = 3$	$p = 5$	$p = 10$
$N = 10^3$	8	14	29
$N = 10^4$	18	23	40

5. REFERENCES

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

[2] J. Bioucas-Dias, *et.al.*, "Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches," *IEEE Journ. of Sel. Topics in Applied Earth Observ. and Rem. Sens.*, vol. 99, no. 1-16, 2012.

[3] José M. P. Nascimento and José M. Bioucas-Dias, "Does Independent Component Analysis Play a Role in Unmixing Hyperspectral Data?," *IEEE Trans. on Geo. Rem. Sens.*, vol. 43, no. 1, pp. 175–187, 2005.

[4] A. Plaza, *et.al.*, "Foreword to the special issue on spectral unmixing of remotely sensed data," *IEEE Trans. on Geo. and Rem. Sens.*, vol. 49, no. 11, pp. 4103–4110, 2011.

[5] A. Plaza, *et.al.*, "Spatial/Spectral Endmember Extraction by Multidimensional Morphological Operations," *IEEE Trans. on Geo. Rem. Sens.*, vol. 40, no. 9, pp. 2025–2041, 2002.

[6] J.W. Boardman, "Automating Spectral Unmixing of AVIRIS Data using Convex Geometry Concepts," in *Sum. of the 4th Annual JPL Airborne Geosci. Workshop, JPL Pub. 93-26.*, 1993, vol. 1, pp. 11–14.

[7] M. E. Winter, "N-FINDR: An Algorithm for Fast Autonomous Spectral End-member Determination in Hyperspectral Data," in *Proc. of the SPIE conf. on Imag. Spectrom. V*, 1999, vol. 3753, pp. 266–275.

[8] T.-H. Chan, *et.al.* "A simplex volume maximization framework for hyperspectral endmember extraction," *IEEE Trans. on Geo. Rem. Sens.*, vol. 19, no. 11, pp. 4177 - 4193, 2011.

[9] S. Lopez, *et.al.*, "A low-computational-complexity algorithm for hyperspectral endmember extraction: Modified vertex component analysis," *IEEE Geo. Rem. Sens. Let.*, vol. 9, no. 3, pp. 502 –506, 2012.

[10] J. M. Rodriguez Alves, *et.al.*, "Vertex Component Analysis GPU-Based Implementation for Hyperspectral Unmixing," in *4th IEEE WHISPERS*, 2012.

[11] J. M. P. Nascimento and J. M. Bioucas-Dias, "New developments on vca unmixing algorithm," 2008, vol. 7109, p. 71090F, SPIE.

[12] X. Zhu, *et.al.*, "Improved vca in hyperspectral image processing," 2010, vol. 7668, p. 76680Z, SPIE.

[13] J. Liu and J. Zhang, "A new maximum simplex volume method based on householder transformation for endmember extraction," *IEEE Trans. on Geo. Rem. Sens.*, vol. 50, no. 1, pp. 104 –118, jan. 2012.

[14] A. Plaza, *et.al.*, "High performance computing for hyperspectral remote sensing," *IEEE Journ. of Sel. Topics in Applied Earth Observ. and Rem. Sens.*, vol. 4, no. 3, pp. 528–544, 2011.

[15] S. Sanchez, *et.al.*, "Parallel unmixing of remotely sensed hyperspectral images on commodity graphics processing units," *Concurrency and Computation: Practice & Experience*, vol. 23, no. 13, pp. 1538–1557, 2011.

[16] J. M. Bioucas-Dias and J. M. P. Nascimento, "Hyperspectral Subspace Identification," *IEEE Trans. on Geo. Rem. Sens.*, vol. 46, no. 8, pp. 2435–2445, 2008.

[17] J. M. P. Nascimento and J. M. Bioucas-Dias, "Hyperspectral unmixing based on mixtures of dirichlet components," *IEEE Trans. on Geo. Rem. Sens.*, vol. 50, no. 3, pp. 863–878, 2012.