Parallel Processing of Hyperspectral Images

Antonio J. Plaza

Department of Technology of Computers and Communications
University of Extremadura, Avda. de la Universidad s/n, E-10071 Cáceres, SPAIN
Phone: +34 927 257195 – Fax: +34 927 257203 – E-mail: aplaza@unex.es
URL: http://www.umbc.edu/rssipl/people/aplaza
1. Introduction

2. Integration of spatial and spectral information
   - Extended mathematical morphology for hyperspectral processing
   - Spatial/spectral endmember extraction and spectral unmixing

3. Parallel implementations
   - Data partitioning strategies for parallel hyperspectral imaging
   - Parallel implementation of unsupervised spatial/spectral approaches

4. Case study: onboard implementation of PPI algorithm

5. Summary and future directions
1. Introduction

2. Integration of spatial and spectral information
   - Extended mathematical morphology for hyperspectral processing
   - Spatial/spectral endmember extraction and spectral unmixing

3. Parallel implementations
   - Data partitioning strategies for parallel hyperspectral imaging
   - Parallel implementation of unsupervised spatial/spectral approaches

4. Case study: onboard implementation of PPI algorithm

5. Summary and future directions
Levels of Spectral Information in Remote Sensing

- **Spectral mixture analysis**: Determines the abundance of materials (e.g. precision agriculture).
- **Characterization**: Determines variability of identified material (e.g. wet/dry sand, soil particle size effects).
- **Identification**: Determines the unique identity of the foregoing generic categories (e.g. land-cover or mineral mapping).
- **Discrimination**: Determines generic categories of the foregoing classes.
- **Classification**: Separates materials into spectrally similar groups (e.g., urban data classification).
- **Detection**: Determines the presence of materials, objects, activities, or events.
The spectral signature of a pixel is a combination of the reflected or emitted energy from all the features that fall within that pixel area.

Concept of hyperspectral imaging using NASA Jet Propulsion Laboratory’s Airborne Visible Infra-Red Imaging Spectrometer
Problem of mixed pixels in hyperspectral images

- Particularities not to be found in other types of image data:
  - *Mixed pixels* (due to insufficient spatial resolution and mixing effects in surfaces)
  - *Sub-pixel targets* (very important and crucial in many hyperspectral applications)
Standard hyperspectral analysis methodology

**INTRODUCTION**

Parallel Processing of Hyperspectral Images

First European School on Hyperspectral Imaging, Cáceres, October 29-31, 2007

**Pre-processing**

PCA, MNF, ICA

**Dimensional reduction**

**Reduced image**

FCLSU, LSU

**Abundance estimation**

Spectral unmixing

**Endmember selection**

Fractional abundance maps

endmembers

wavelength (nm)

reflectance
Challenges in hyperspectral image processing

• The special characteristics of hyperspectral data pose several processing problems:

1. The high-dimensional nature of hyperspectral data introduces important limitations in supervised classifiers, such as the limited availability of training samples or the inherently complex structure of the data.

2. There is a need to integrate the spatial and spectral information to take advantage of the complementarities that both sources of information can provide, in particular, for mixed pixel classification.

3. There is a need to develop parallel algorithm implementations, able to speed up algorithm performance and to satisfy the extremely high computational requirements of time-critical remote sensing applications.

• In this work, we have taken a necessary first step towards the understanding and assimilation of the above aspects in the design of last-generation hyperspectral image processing algorithms.
Why high performance computing is crucial?

Biomass Burning: Sub-pixel temperatures and extent, smoke, combustion products...

Environmental Hazards: Contaminants (direct and indirect), geological substrate...

Coastal and Inland Waters: Chemical and biological standoff detection, oil spill monitoring and tracking...

Ecology: Chlorophyll, leaf water, lignin, cellulose, pigments, structure, nonphotosynthetic constituents...

Commercial Applications: Mineral exploration, agriculture and forest status...

Military Applications: Detection of land mines, tracking of targets, decoys...

Others: Human infrastructure, Medical...
AVIRIS spectra were used to measure fire temperature, asbestos contamination, and debris spread.

**Fire Temperatures**

AVIRIS Estimate

Residual

WTC Hot Spot Area A

Hottest Spectrum

Temperature Estimate=928K

6% of the area

**Debris Composition**

**Asbestos**

Reference spectrum:

Chrysotile coating on Girder WTC01-08

(multiplied by 0.341)
1. Introduction

2. Integration of spatial and spectral information
   - Extended mathematical morphology for hyperspectral processing
   - Spatial/spectral endmember extraction and spectral unmixing

3. Parallel implementations
   - Data partitioning strategies for parallel hyperspectral imaging
   - Parallel implementation of unsupervised spatial/spectral approaches

4. Case study: onboard implementation of PPI algorithm

5. Summary and future directions
Pixel Purity Index (PPI) algorithm.
Why integrated spatial/spectral approaches?

- Much effort has been given to processing hyperspectral image data in *spectral* terms.
- Data analysis is carried out without incorporating information about *spatial* context.

- There is a need to incorporate the *image representation* of the data in the analysis.
- Most available approaches consider spatial and spectral information *separately*.
- *Mathematical morphology* provides a *remarkable framework* to achieve the *desired integration*.
• Nonlinear *spatial-based* technique that provides a framework to achieve the desired *integration* of spatial and spectral data:

**Binary erosion.**
• Nonlinear *spatial-based* technique that provides a framework to achieve the desired *integration* of spatial and spectral data:

**Binary dilation.**
• Opening and closing: shape-preserving operators.
• Excellent filtering properties:

Morphological opening (erosion + dilation)
Mathematical Morphology

- Grayscale morphology relies on a partial ordering relation between image pixels.

- Morphological operations for hyperspectral imagery require ordering of image pixels.
- Two strategies explored in the past: PCA-based ordering and vector-based ordering.
Vector-based ordering

- Based on a *spectral* distance function (SAD, SID) and a *cumulative* distance measure.

- The greatest element is the *most spectrally distinct* (pure) in the structuring element.
- The least element is the *most spectrally similar* (mixed) in the structuring element.
- *Extended dilation* has the effect of expanding pure spectral areas in the image.
- *Extended erosion* reduces pure spectral areas and expands mixed areas.
- Particularly suited for spatial/spectral endmember extraction.
Automated Morphological Endmember Extraction.

- Integration of spectral and **spatial** information (computation intensive)
- Selection of the most *spectrally pure* and the *most spectrally* mixed signatures.

\[ f : \mathbb{Z}^2 \rightarrow \mathbb{Z}^N \]

\[ (f \oplus K)(x, y) = \arg\text{Max}_{(s,t)\in \mathbb{Z}^2(K)} \{ D^+ (f(x,y)) \} \]

\[ MEI \]

\[ (f \otimes K)(x, y) = \arg\text{Min}_{(s,t)\in \mathbb{Z}^2(K)} \{ D^- (f(x,y)) \} \]
AVIRIS Data Over Cuprite, Nevada

RESULTS: ENDMEMBER EXTRACTION

Parallel Processing of Hyperspectral Images
Comparison of Endmember Extraction Algorithms.

RESULTS: ENDMEMBER EXTRACTION

Parallel Processing of Hyperspectral Images

First European School on Hyperspectral Imaging, Cáceres, October 29-31, 2007
1. Introduction

2. Integration of spatial and spectral information
   - Extended mathematical morphology for hyperspectral processing
   - Spatial/spectral endmember extraction and spectral unmixing

3. Parallel implementations
   - Data partitioning strategies for parallel hyperspectral imaging
   - Parallel implementation of unsupervised spatial/spectral approaches

4. Case study: onboard implementation of PPI algorithm

5. Summary and future directions
Parallel computers

- Parallel computer is a *collection of processing elements* that *cooperate* to *solve problems faster*.
- Hyperspectral imaging demands parallel computers to speed-up many applications.
- Speed-up ($p$ processors) = \( \frac{\text{Performance (}p\text{ processors})}{\text{Performance (1 processor)}} \)

**NASA Portable MiniCluster** (16 processors)  
**Earth Simulator** (5120 processors)
Data Partitioning for spatial/spectral approaches

**Spectral-domain partitioning:**
A single pixel vector (spectral signature) may be stored in different processing units and communications would be required for individual pixel-based calculations such as spectral angle computations.

**Spatial-domain partitioning:**
Every pixel vector (spectral signature) is stored in the same processing unit. This is beneficial for sliding window-based approaches in terms of low-level image processing and spatial/spectral data integration.
Parallel Framework for Morphological Methods

- The master processor is in charge of distributing the work among the workers.
- Each partition (PSSP) is processed independently, and the master gathers the result.
Border-Handling Data Strategy Adopted in Morphological Processing

Pixels which do not belong to the image domain are simply disregarded in the calculation of the morphological eccentricity index (MEI)
Kernel-based computations prevent exploitation of the concept of PSSP since communications are required for border pixels (simplified view).

PARALLEL IMPLEMENTATIONS

Handling Communications (I)
Overlapping scatter allows to process PSSPs independently through the introduction of redundant information (several ways to do this!)
RESULTS: CLUSTERS OF COMPUTERS

Thunderhead (NASA)
http://thunderhead.gsfc.nasa.gov

<table>
<thead>
<tr>
<th>Aggregate Specification</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of nodes</td>
<td>256</td>
</tr>
<tr>
<td>Total processors</td>
<td>512</td>
</tr>
<tr>
<td>Total memory (GB)</td>
<td>256</td>
</tr>
<tr>
<td>Total disk (GB)</td>
<td>20480</td>
</tr>
<tr>
<td>Interconnect 1</td>
<td>Myrinet 2000</td>
</tr>
<tr>
<td>Interconnect 2</td>
<td>Gigabit Ethernet</td>
</tr>
<tr>
<td>Total peak performance (Gflop/s)</td>
<td>2457.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node Specification</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Motherboard</td>
<td>Tyan Thunder 2720</td>
</tr>
<tr>
<td>Number of processors</td>
<td>Dual Intel 4 Xeon 2.4GHz 1</td>
</tr>
<tr>
<td>Memory (GB)</td>
<td>8GB</td>
</tr>
<tr>
<td>Local disk (GB)</td>
<td>Myrinet 2000</td>
</tr>
<tr>
<td>Interconnect 1</td>
<td>Gigabit 1000 Mbit/s</td>
</tr>
<tr>
<td>Interconnect 2</td>
<td></td>
</tr>
<tr>
<td>Peak performance (Gflop/s)</td>
<td>9.6</td>
</tr>
</tbody>
</table>

First European School on Hyperspectral Imaging, Cáceres, October 29-31, 2007
Performance of Morphological Endmember Extraction.-

- Algorithms were implemented in C++ using calls to Message Passing Interface (MPI).
- Using redundant computations versus communications was crucial.

### RESULTS: CLUSTERS OF COMPUTERS

**Parallel Processing of Hyperspectral Images**

<table>
<thead>
<tr>
<th>Number of processors</th>
<th>1</th>
<th>4</th>
<th>16</th>
<th>64</th>
<th>256</th>
<th>512</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time endmember extraction</td>
<td>6625</td>
<td>2859</td>
<td>650</td>
<td>156</td>
<td>42</td>
<td>22</td>
</tr>
<tr>
<td>Time unmixing</td>
<td>2840</td>
<td>1226</td>
<td>279</td>
<td>64</td>
<td>18</td>
<td>11</td>
</tr>
<tr>
<td>Time total</td>
<td>9465</td>
<td>4085</td>
<td>929</td>
<td>220</td>
<td>60</td>
<td>33</td>
</tr>
<tr>
<td>Speedup</td>
<td>1.0</td>
<td>2.3</td>
<td>10.1</td>
<td>43.1</td>
<td>157.7</td>
<td>356.34</td>
</tr>
</tbody>
</table>
1. Introduction

2. Integration of spatial and spectral information
   - Extended mathematical morphology for hyperspectral processing
   - Spatial/spectral endmember extraction and spectral unmixing

3. Parallel implementations
   - Data partitioning strategies for parallel hyperspectral imaging
   - Parallel implementation of unsupervised spatial/spectral approaches

4. Case study: onboard implementation of PPI algorithm

5. Summary and future directions
Algorithm 1 High-level implementation of the extreme projections step of the PPI algorithm

```plaintext
for (n = 0; n < N; n++) //N denotes the number of bands
{
    for (k = 0; k < K; k++) //K denotes the number of skewers
    {
        dp[k] = dot_product(pixels[n], skewers[k]);
        if (dp[k] < Min[k]) { Min[k] = dp[k]; Reg_Min[k] = n; }
        if (dp[k] > Max[k]) { Max[k] = dp[k]; Reg_Max[k] = n; }
    }
}
```
High-Level Description of PPI:

\[ N \]

\[ \text{skewer}_{1}^{(N)}, \ldots, \text{skewer}_{1}^{(i)} \]

\[ \text{skewer}_{2}^{(N)}, \ldots, \text{skewer}_{2}^{(i)} \]

\[ \text{skewer}_{3}^{(N)}, \ldots, \text{skewer}_{3}^{(i)} \]

\[ \text{skewer}_{K}^{(N)}, \ldots, \text{skewer}_{K}^{(i)} \]

\[ f_{1}^{(N)}, \ldots, f_{1}^{(i)} \]

\[ f_{2}^{(N)}, \ldots, f_{2}^{(i)} \]

\[ f_{3}^{(N)}, \ldots, f_{3}^{(i)} \]

\[ f_{T}^{(N)}, \ldots, f_{T}^{(i)} \]

\[ \text{min}_{1}, \text{min}_{2}, \text{min}_{3}, \ldots, \text{min}_{K} \]

\[ \text{max}_{1}, \text{max}_{2}, \text{max}_{3}, \ldots, \text{max}_{T} \]
Implementation issues (PPI).-

- The figure depicts an ideal systolic array in which \( T \) pixels and \( K \) skewers are processed.
- In a real systolic, \( T \) has to be divided by \( P \), the number of available processors.
- A similar comment applies to \( K \), the number of skewers.
- After \( T/P \) cycles, all dot nodes are busy.
- After \( K/P \) additional cycles, the first \( P \) pixel vectors are processed.

Implementation issues (LSU).-

- To obtain the endmember abundances, we multiply each \( f \) by \( (M^TM)^{-1}M^T \), where \( M = \{e_i\}_{i=1}^c \)
- This can be done using the same systolic architecture used for the PPI algorithm
CASE STUDY USING THE PPI ALGORITHM

PPI algorithm rewritten:

**Algorithm 1** High-level implementation of the *extreme projections* step of the PPI algorithm

```plaintext
for (n = 0; n < N; n++) // N denotes the number of bands
{
    par (k = 0; k < K; k++) // K denotes the number of skewers
    { dp[k] = dot_product(pixels[n], skewers[k]);
        if (dp[k] < Min[k]) { Min[k] = dp[k]; Reg_Min[k] = n; }
        if (dp[k] > Max[k]) { Max[k] = dp[k]; Reg_Max[k] = n; }
    }
}
```

**Algorithm 2** High-level implementation of the *extreme projections* step of the PPI algorithm, rewritten to be split into *P* algorithm passes

```plaintext
for (p = 0; p < P; p++) // P is the number of algorithm passes
{
    x = p × (K/P); // K denotes the number of skewers
    for (n = 0; n < N; n++) // N denotes the number of bands
    {
        par(k = 0; k < K/P; k++) // K denotes the number of skewers
        { dp[x + k] = dot_product(pixels[n], skewers[x + k]);
            if (dp[x + k] < Min[x + k]) { Min[x + k] = dp[x + k]; Reg_Min[x + k] = n; }
            if (dp[x + k] > Max[x + k]) { Max[x + k] = dp[x + k]; Reg_Max[x + k] = n; }
        }
    }
}
```
FPGA reconfigurable implementation of PPI:

- Xilinx reconfigurable Virtex-II field programmable gate array (FPGA) with 33,792 slices, 144 Select RAM Blocks and 144 multipliers (of 18-bit x 18-bit)
- One 3U Compact PCI card (weight below 1 lb) and power of approximately 25 Watts
- Complete system (systolic array plus PCI interface), implemented on XC2V6000-6 board, using different numbers of processors
- We measured an average PCI bandwidth of 15 Mbytes between the PC and the board
Application case study:

Data set owned by NASA/Jet Propulsion Lab

AVIRIS data over lower Manhattan (09/15/01)

Data set owned by U.S. Geological Survey

Spatial location of thermal hot spots in WTC area
Application case study:

Spectral angle-based spectral similarity scores between endmembers detected by PPI, N-FINDR, ATGP and the known ground targets. Single-processor times using one Thunderhead node are also given in the parentheses.
## Endmember extraction accuracy of FPGA version

### CASE STUDY USING THE PPI ALGORITHM

<table>
<thead>
<tr>
<th>ENVI’s PPI</th>
<th>PPI (C code)</th>
<th>FPGA-PPI</th>
<th>N-FINDR</th>
<th>ATGP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concrete (WTC01-37B)</td>
<td>0.027</td>
<td>0.028</td>
<td>0.028</td>
<td>0.031</td>
</tr>
<tr>
<td>Concrete (WTC01-37Am)</td>
<td>0.022</td>
<td>0.025</td>
<td>0.025</td>
<td>0.025</td>
</tr>
<tr>
<td>Cement (WTC01-37A)</td>
<td>0.021</td>
<td>0.022</td>
<td>0.022</td>
<td>0.045</td>
</tr>
<tr>
<td>Dust (WTC01-15)</td>
<td>0.019</td>
<td>0.020</td>
<td>0.020</td>
<td>0.020</td>
</tr>
<tr>
<td>Dust (WTC01-28)</td>
<td>0.017</td>
<td>0.019</td>
<td>0.019</td>
<td>0.025</td>
</tr>
<tr>
<td>Dust (WTC01-36)</td>
<td>0.031</td>
<td>0.031</td>
<td>0.031</td>
<td>0.033</td>
</tr>
<tr>
<td>Gypsum wall board</td>
<td>0.026</td>
<td>0.028</td>
<td>0.028</td>
<td>0.028</td>
</tr>
</tbody>
</table>
Validation on real Xilinx Virtex-II platform:

<table>
<thead>
<tr>
<th>Number of processors</th>
<th>Number of gates</th>
<th>Number of slices</th>
<th>Percentage of total</th>
<th>Operation frequency</th>
<th>Processing time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>97,443</td>
<td>1,185</td>
<td>3%</td>
<td>29,257</td>
<td>53.48</td>
</tr>
<tr>
<td>200</td>
<td>212,412</td>
<td>3,587</td>
<td>10%</td>
<td>21,782</td>
<td>22.65</td>
</tr>
<tr>
<td>400</td>
<td>526,944</td>
<td>12,418</td>
<td>36%</td>
<td>18,032</td>
<td>7.94</td>
</tr>
</tbody>
</table>

- An optimized C-based sequential implementation of PPI/LSU took 1163 seconds on a desktop PC with AMD Athlon 2.6 GHz processor and 512 MB of RAM.
- Our implementation was limited by the transfer rate: FPGA able to absorb a 40 Mbytes/second while PCI interface can only provide a flow of 15 Mbytes/second.
- We decided to report realistic experiments by resorting to a moderate amount of resources (gates) in the FPGA board (leaving room for implementation of additional algorithms on the same board, allowing for on-the-fly algorithm selection).
- Results not strictly in real-time (below 5 seconds) but already very close.

Appealing perspectives from an exploitation point of view: *on-the-fly selection*
Graphics Processing Units for Hyperspectral Imaging:

<table>
<thead>
<tr>
<th>Year</th>
<th>FX5950 Ultra</th>
<th>7800 GTX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture</td>
<td>2003</td>
<td>2005</td>
</tr>
<tr>
<td>Bus</td>
<td>NV38</td>
<td>G70</td>
</tr>
<tr>
<td>Video Memory</td>
<td>AGPx8</td>
<td>PCI Express</td>
</tr>
<tr>
<td>Core Clock</td>
<td>256MB</td>
<td>256MB</td>
</tr>
<tr>
<td>Memory Clock</td>
<td>475 MHz</td>
<td>430 MHz</td>
</tr>
<tr>
<td>Memory Interface</td>
<td>950 MHz</td>
<td>1.2 GHz GDDR3</td>
</tr>
<tr>
<td>Memory bandwidth</td>
<td>256-bit</td>
<td>256-bit</td>
</tr>
<tr>
<td>#Pixel shader processors</td>
<td>30.4 GB/s</td>
<td>38.4 GB/s</td>
</tr>
<tr>
<td>Texture fill rate</td>
<td>4</td>
<td>24</td>
</tr>
</tbody>
</table>

Performance Evolution

![Performance Chart](chart.png)
Summary

• The special characteristics of hyperspectral images pose new processing problems, not found in other types of remote sensing data.

• The integration of spatial and spectral information allows for the development of enhanced supervised/unsupervised analysis techniques.

• Endmember extraction and spectral unmixing can benefit from the incorporation of spatial information into the analysis.

• Most of the algorithms discussed in this work are very appealing for the design of parallel implementations.

• Standard algorithms such as the PPI (spectral) or the AMEE (spatial/spectral) fit very well into the framework of parallel computing and result in nice speedups.

• Commodity cluster and hardware-based implementations developed.

• Techniques developed in this work show the increasing sophistication of a field that is rapidly maturing at the intersection of many different disciplines.
Activities in this area


Parallel Processing of Hyperspectral Images

Antonio J. Plaza

Department of Technology of Computers and Communications
University of Extremadura, Avda. de la Universidad s/n, E-10071 Cáceres, SPAIN
Phone: +34 927 257195 – Fax: +34 927 257203 – E-mail: aplaza@unex.es
URL: http://www.umbc.edu/rssipl/people/aplaza