Efficient Segmentation and Classification of nDimensional Images in GPU for Real Time Processing

Pablo Quesada Barriuso

Centro Singular de Investigación en Tecnoloxías da Información
Universidade de Santiago de Compostela

Advisors: Dora B. Heras and Francisco Argüello

Table of Content

Table of Content

Motivation

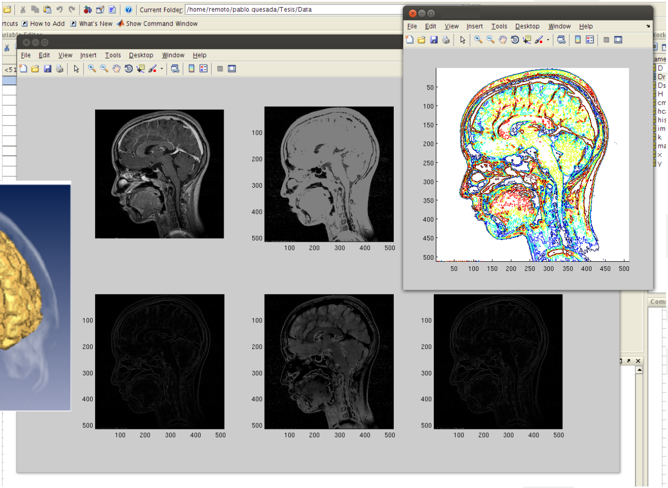
- Advances in sensor technology $\longrightarrow$ New challenges in image processing
- Segmentation and Classification are key operations
  - medical surgery, maritime rescue, object tracking...
- The complexity and time of processing these data increase
  - Hard to achieve real-time in commodity hardware
- High performance computing is commonly used

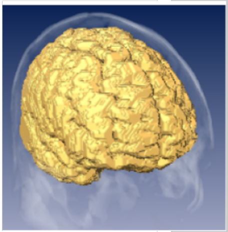
Objectives

- **Improving** the accuracy of techniques
- Achieving **real–time** execution
- **Eficiently exploiting** the computing capabilities of **GPUs**

2D and 3D Medical Images

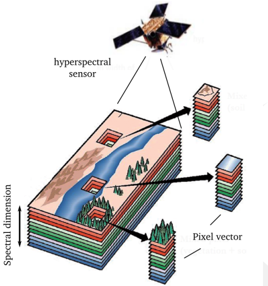
- Diagnosis, monitoring of patients, augmented reality in surgery, ...



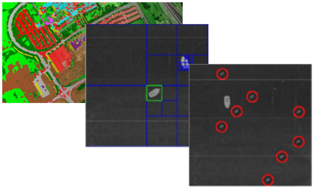


Multi– and Hyper–Spectral images

- Multi–Spectral
  - up to 10 spectral bands
- Hyper–Spectral
  - hundred of spectral bands



- Target detection
  - Maritime rescue, object tracking, ...
- Segmentation and Classification
  - Environment monitoring, change detection, under–surface gathering, ...



NVIDIA Compute Unified Device Architecture

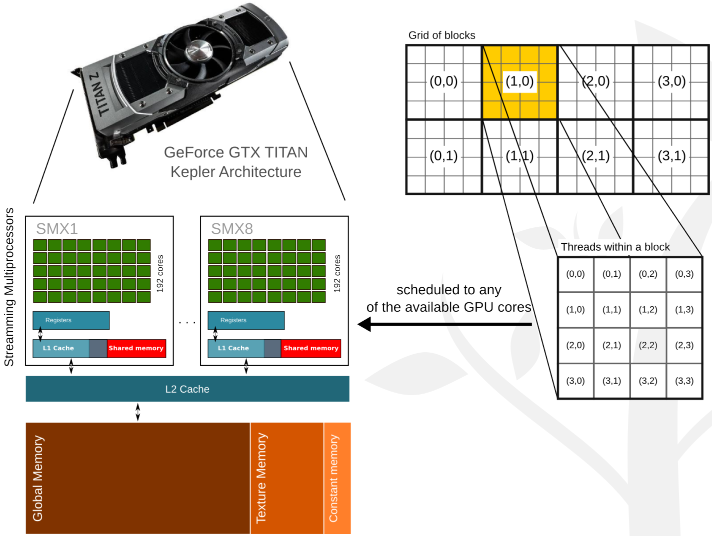
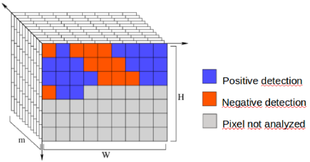
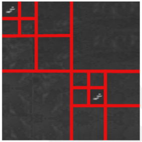


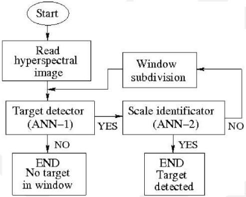
Table of Content

Target Detection based on Artificial Neural Networks

- Pixel level algorithm for target detection
- Multiresolution target detection algorithm



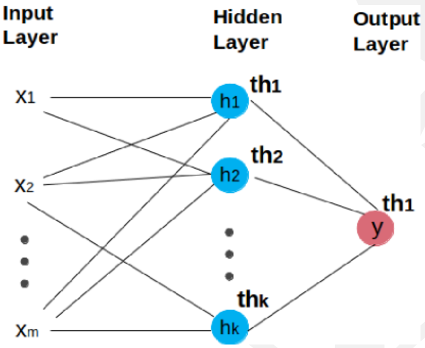




Target Detection

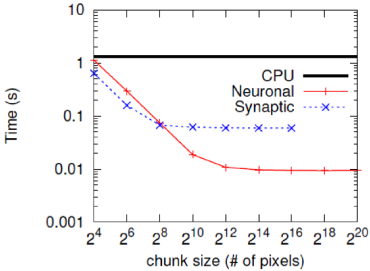
Parallelization strategies

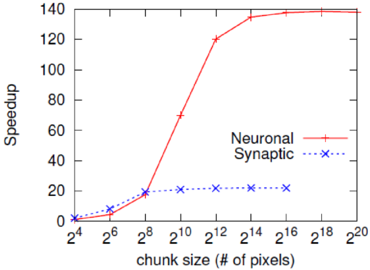
- Neuronal level parallelism
- Synaptic level parallelism



Results Execution Time

- Execution times and speedups for the GPU target detection algorithm at pixel level





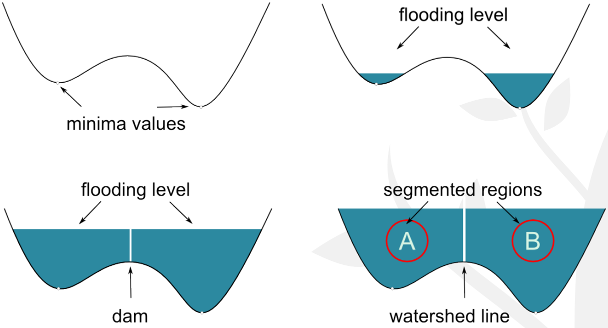
execution platforms

Intel Quad Core Xeon E5440 (2.83 GHz) with 6MB L2 cache per core pair and two 32KB L1 caches per core
NVIDIA GTX580 with 512 cores, and 1536 MB of Global Memory

Table of Content

Watershed Transform, Hill–Climbing algorithm

- Propagate the labels upwards, climbing up the hill
- Efficient implementations in CPU *use queues* $\longrightarrow$ bad for the GPU

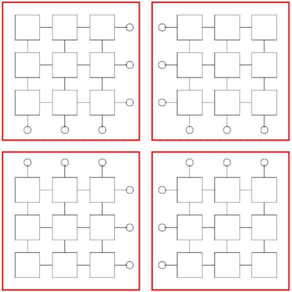


[1] Pablo Quesada-Barriuso, Julian Lamas-Rodríguez, Dora B. Heras y Francisco Argüello, "Influencia de las mesetas en la implementación de watershed sobre GPUs", XXIII Jornadas de Paralelismo, pp. 249-254. Elche (Spain). 2012
[2] Julián Lamas-Rodríguez, Pablo Quesada-Barriuso, Francisco Argüello, Dora B. Heras y Montserrat Bóo, "Proyección del método de segmentación del conjunto de nivel en GPU", XXIII Jornadas de Paralelismo, pp. 273-278. Elche (Spain). 2012

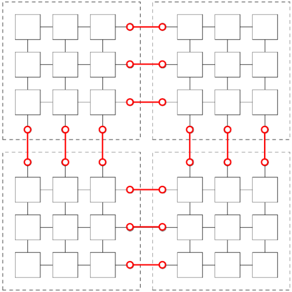
Block–Asynchronous Scheme

Watershed based on Cellular Automata

- Asynchronous cellular automaton
- **Hybrid** iterative **updating** process → It exploits the **shared memory**



Intra–block updating



Inter–block updating
Global Synchronization

Block–Asynchronous Scheme

Watershed based on Cellular Automata

- Intra–block updating $\longrightarrow$ Artifacts

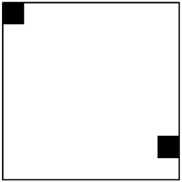
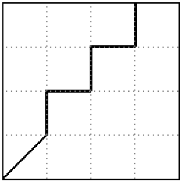
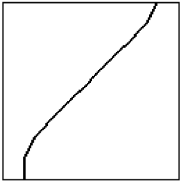


Image of
128 × 128 pixels



Artifacts produced by the
asynchronous computation
in blocks of 32 x 32 threads



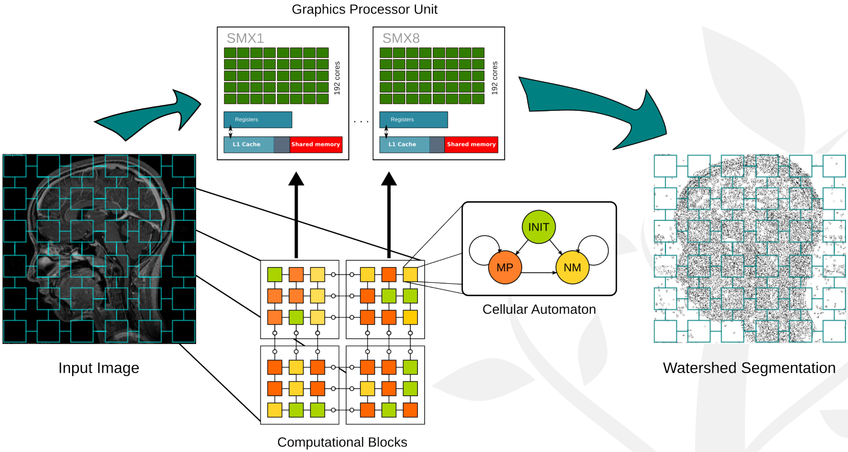
Correct
Watershed line

Artifacts–Free Solution

Wavefront technique $\longrightarrow$ based on counting the distance to the middle of a flat zone

GPU Watershed Image Segmentation

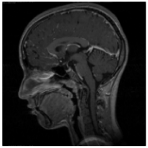
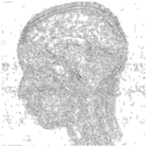
Watershed based on CA



[3] Pablo Quesada-Barriuso, Dora B. Heras, Francisco Argüello, “Efficient 2D and 3D Watershed on Graphics Processing Unit: Block-Asynchronous Approaches Based on Cellular Automata” Computers & Electrical Engineering, v.39, n.8, pp.2638-2655. 2013
[4] Pablo Quesada-Barriuso, Dora B. Heras, and Francisco Argüello, “Efficient GPU Asynchronous Implementation of a Watershed Algorithm Based on Cellular Automata” 10th IEEE Int. Symp. on Parallel and Distributed Processing with Applications, pp.79-86. Leganés. 2012

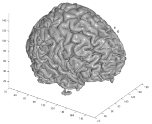
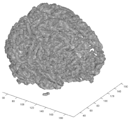
Results

2D CT Scan

	512 × 512		1024 × 1024		2048 × 2048	
OpenMP	0.499	1.0×	2.879	1.0×	15.092	1.0×
GPU Sync.	0.031	16.2×	0.144	20.1×	0.699	21.6×
GPU Async.	0.009	**53.1**×	0.038	**75.6**×	0.163	**92.7**×
GPU Artf–Free	0.013	39.2×	0.052	55.2×	0.235	64.1×

3D MRI Simulation

	45 × 54 × 45		90 × 108 × 90		181 × 217 × 181	
OpenMP	0.134	1.0×	2.261	1.0×	37.738	1.0×
GPU Sync.	0.008	15.9×	0.082	27.6×	1.123	33.6×
GPU Async.	0.004	**30.4**×	0.045	**50.1**×	0.509	**63.9**×
GPU Artf–Free	0.005	26.7×	0.054	41.8×	0.730	51.7×

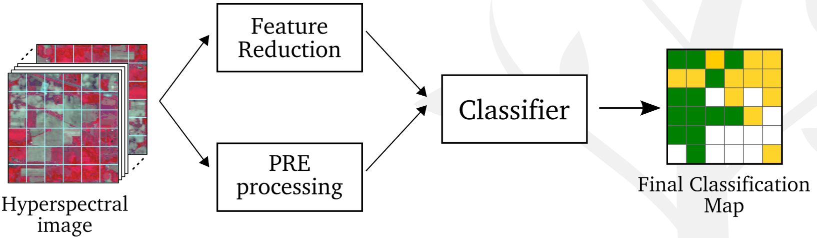
execution platforms

Intel Core i7 with four cores at 2,80 GHz and 8 GB of RAM using OpenMP (4 threads)
NVIDIA GTX580 with 512 cores, and 1536 MB of Global Memory

Table of Content

Standalone Spectral Classification

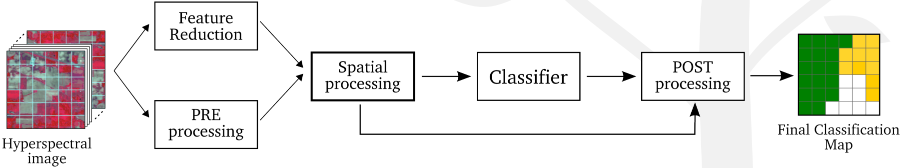
- Pre-processing is almost always required
  - *Denoising, Atmospheric correction, ...*
- Feature reduction is commonly applied
  - *PCA, ICA, Wavelets, ...*
- Variety of classification methods
  - *K-Means, ELM, SVM, ...*



[5] Dora B. Heras, Francisco Argüello, Pablo Quesada-Barriuso, "Exploring ELM-based spatial-spectral classification of hyperspectral images", International Journal of Remote Sensing, vol. 35, no. 2, pp. 401-423. 2014

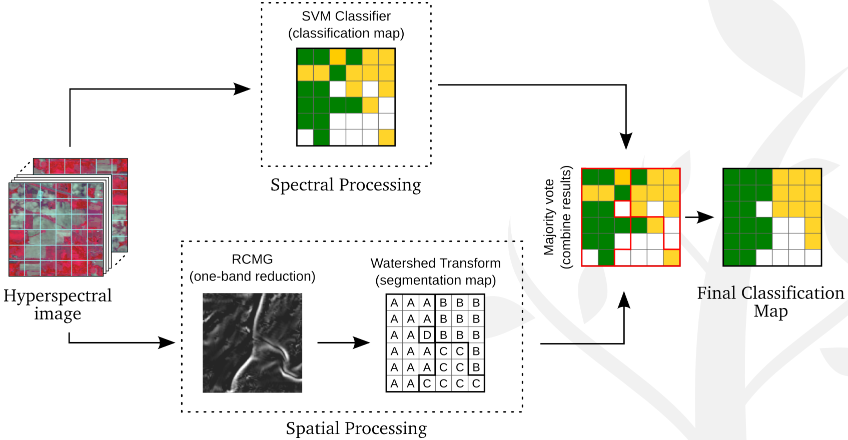
Spectral–Spatial Classification general scheme

- Spatial processing
  - Mathematical morphology
  - Segmentation
  - Morphological profiles
- Combining spectral and spatial information
  - Majority vote in an adaptative neighborhood
  - Stacked Vector



RCMG + Watershed + Majority Vote

- Segmentation and Classification



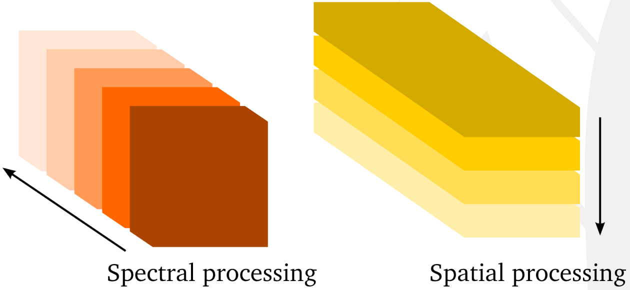
Tarabalka, Y., Chanussot, J., Benediktsson, J. A., Angulo, J., Fauvel, M., 2008. “Segmentation and classification of hyperspectral data using watershed”. In Proc. Of IGARSS ’08, Boston, USA.

GPU Workflow

RCMG + Watershed + Majority Vote

- Data are overlapped among blocks with one element apron
- Data are shared among the threads within the block
- The spectral processing needs less shared memory

$$\nabla(\mathbf{f})_{Robust} = \max_{i,j \in \chi - R_s} \{||\mathbf{x}_i - \mathbf{x}_j||_2\}$$



Results

Overall Accuracy

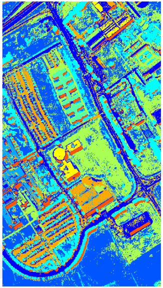
- University of Pavia 610 × 340 pixels × 103 bands

Input Image

RCMG

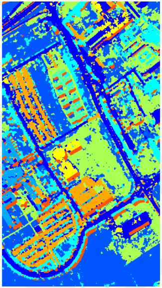
Watershed

SVM



Overall Accuracy 89,7%

Final Classification Map



Overall Accuracy 94,5%

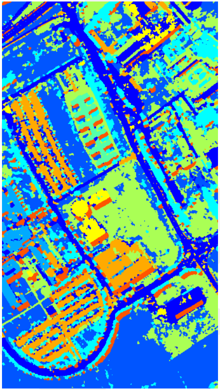
[6] Pablo Quesada-Barriuso, Francisco Argüello, and Dora B. Heras, “Efficient segmentation of hyperspectral images on commodity GPUs”, 16th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems, pp. 2130-2139. 2012

[7] Pablo Quesada-Barriuso, Francisco Argüello, Dora B. Heras, “Computing Efficiently Spectral-Spatial Classification of Hyperspectral Images on Commodity GPUs”, Recent Advances in Knowledge-based Paradigms and Applications, vol. 234, pp. 19-42. 2014

Results
Execution Time

- All computations were performed in double precision
- Time is shown in seconds

	OpenMP	GPU	Speedup
SVM	19.4713	3.1783	6.1×
RCMG	0.1547	0.0081	19.1×
CA-Watershed	0.0280	0.0010	28.0×
MV	0.0020	0.0003	6.6×
Total	19.6560	3.1877	6.2×

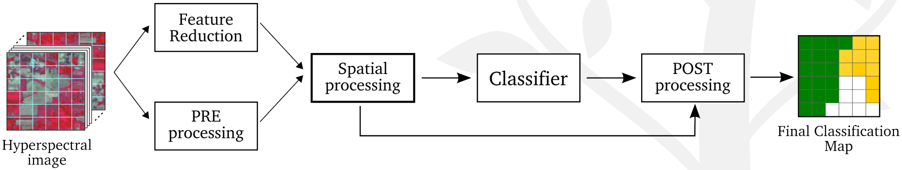


execution platforms

Intel Core i7 with four cores at 2,80 GHz and 8 GB of RAM using OpenMP (4 threads)
NVIDIA GTX680 with 1536 cores, and 2048 MB of Global Memory

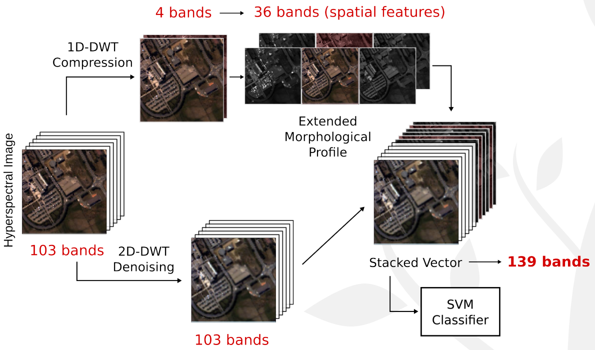
Spectral–Spatial Classification general scheme

- Spatial processing
  - Mathematical morphology
  - Morphological profiles
- Combine spectral and spatial information
  - Stacked Vector



Proposal for a new scheme spectral–spatial

- 1D-DWT, Mathematical Morphology, Denoising → Stacked Vector



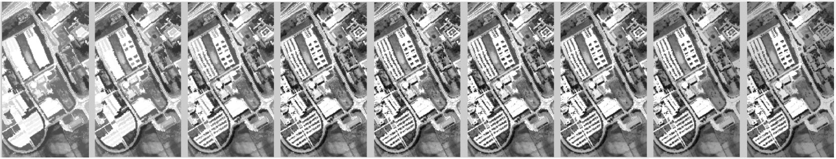
[8] Pablo Quesada-Barriuso, Francisco Argüello, and Dora B. Heras, "Spectral–Spatial Classification of Hyperspectral Images Using Wavelets and Extended Morphological Profiles," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 7, no. 4. 2014

[9] F. Argüello, D.B. Heras, M. Bóo, J. Lamas-Rodríguez, "The split-and-merge method in general purpose computation on GPUs," *Parallel Computing*, vol. 38, Issues 6–7, pp 277–288, 2012

GPU Workflow Wavelet + **Extended Morphological Profiles**

- Opening and Closing by recostruction are iterative processes
  - Capturing relevant structures at different scales
- Hybrid iterative updating process
  - Block–Asynchronous Scheme

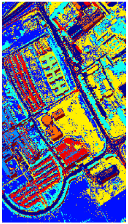
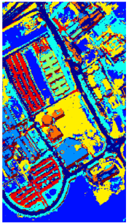
four opening by reconstruction fitrst band four closing by reconstruction



Results Overall Accuracy

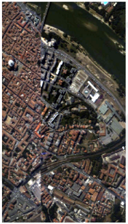
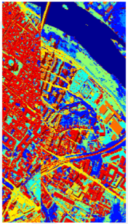
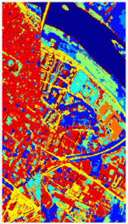
University of Pavia

610 × 340 pixels × 103 bands

Input Image	SVM	Final Classification Map
		
	Overall Accuracy 89,7%	Overall Accuracy 98,8%

City of Pavia

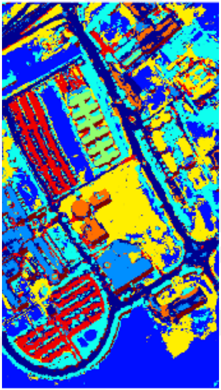
1096 × 715 pixels × 102 bands

Input Image	SVM	Final Classification Map
		
	Overall Accuracy 97,7%	Overall Accuracy 99,7%

Results Execution Time

- All computations were performed in double precision
- Time is shown in seconds

	OpenMP	GPU	Speedup
SVM	12.611	1.344	9.4×
1D-DWT	0.259	0.009	28.7×
EMP	0.724	0.104	6.9×
2D-DWT	0.748	0.221	3.4×
Total	14.342	1.678	8.5×



execution platforms

Intel Core i7 with four cores at 2,80 GHz and 8 GB of RAM using OpenMP (4 threads)
NVIDIA GTX TITAN with 2880 cores, and 6143 MB of Global Memory

Table of Content

Conclusions and future work

- **Real–time** execution was achieved
- The **accuracy** of the techniques was **improved**
  - Wavefront technique
  - New spectral–spatial scheme
- The computing capabilities of the **GPU** were **efficiently exploited**
  - Computation by blocks in shared memory
  - Hybrid iterative updating process
  - Data are packed, reused in shared memory and overlapped among blocks
- Future work
  - New segmentation techniques
  - Improving the classification process
  - New cases of study

Thanks for you attention

Questions, comments and feedbacks are welcome!

pablo.quesada@usc.es