Pragmatic Remote Sensing

A Hands-on Approach to Processing

J. Inglada[1] , E. Christophe[2]

[1]Centre d'Études Spatiales de la Biosphère, Toulouse, France

[2]Centre for Remote Imaging, Sensing and Processing,
National University of Singapore

Table of Contents

Pragmatic Remote Sensing

A Hands-on Approach to Processing

J. Inglada[1] , E. Christophe[2]

[1]Centre d'Études Spatiales de la Biosphère, Toulouse, France

[2]Centre for Remote Imaging, Sensing and Processing, National University of Singapore

Why?

Common problems

- Reading images
- Accessing metadata
- Implementing state of the art algorithms

$\Rightarrow$ to be able to extract the most information, we need to use the best of what is available: data, algorithms,...

What is Orfeo Toolbox (OTB)?

In the frame of CNES ORFEO Program

Goal

Make the development of new algorithms and their validation easier

- C++ library: provide many algorithms (pre-processing, image analysis) with a common interface
- Open-source: free to use, to modify, you can make your own software based on OTB and sell it
- Multiplatform: Windows, Linux, Unix, Mac

A bit of History

Everything begins (2006)

- Started in 2006 by CNES (French Space Agency), funding several full-time developers
- Targeted at high resolution images (Pleiades to be launched in 2010) but with application to other sensors
- 4 year budget, over 1,000,000€ recently renewed for 1 additional year (500,000€)

Moving to user friendly applications (2008)

- Strong interactions with the end-user community highlighted that applications for non-programmers are important
- Several applications for non programmers (with GUI) since early 2008
- Several training courses (3/5-day courses) given in France, Belgium, Madagascar, UNESCO and now Hawaii

Why doing that?

Is it successful so far?

- OTB user community growing steadily (programmers and application users)
- Presented at IGARSS and ISPRS in 2008, special session in IGARSS in 2009
- CNES is planning to extend the budget for several more years
- Value analysis is very positive (cf. Ohloh): re-using is powerful

Why make a multi-million dollar software and give it for free?

- CNES is not a software company
- One goal is to encourage research: it is critical for researchers to know what is in the box
- CNES makes satellites and wants to make sure the images are used
- if more people have the tools to use satellite images, it is good for CNES

How?

How to reach this goal?

Using the best work of others: do not reinvent the wheel

Many open-source libraries of good quality

- ITK: software architecture (streaming, multithreading), many image processing algorithms
- Gdal/Ogr: reading data format (geotiff, raw, png, jpeg, shapefile, . . .)
- Ossim: sensor models (Spot, RPC, SAR, . . .) and map projections
- 6S: radiometric corrections
- and many other: libLAS (lidar data), Edison (Mean Shift clustering), libSiftFast (SIFT), Boost (graph), libSVM (Support Vector Machines)

⇒ all behind a common interface

Application

Currently

- Image viewer
- Image Segmentation
- Image Classification (by SVM)
- Land Cover
- Feature Extraction
- Road Extraction
- Orthorectification (with Pan Sharpening)
- Fine registration
- Image to database registration
- Object counting
- Urban area detection
- more to come

Components available

Currently

- Most satellite image formats
- Geometric corrections
- Radiometric corrections
- Change detection
- Feature extraction
- Classification

Huge documentation available

- Software Guide (+600 pages pdf), also the online version
- Doxygen: documentation for developers

A powerful architecture

Modular

- Easy to combine different blocks to do new processing

Scalable

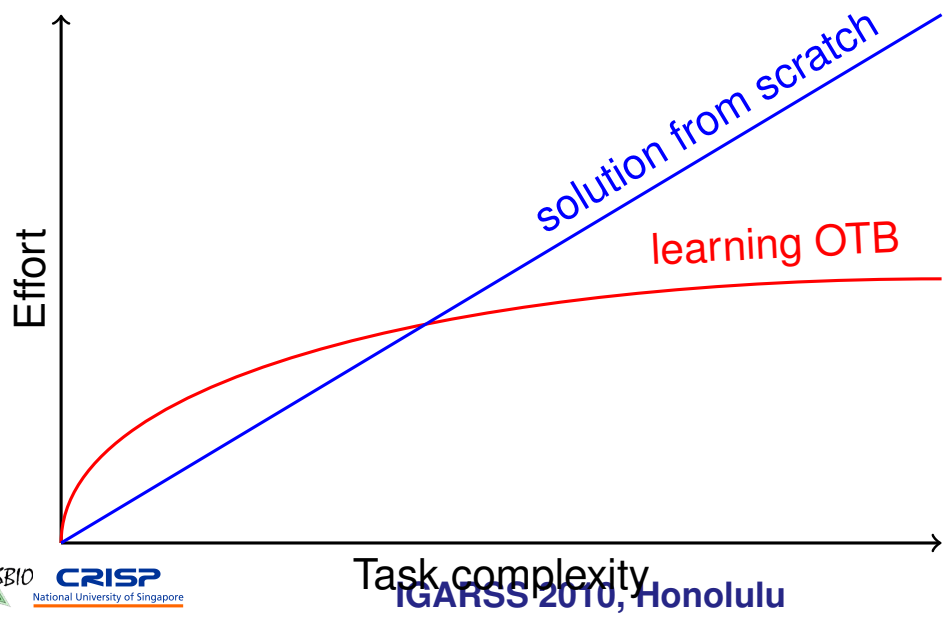
- Streaming (processing huge images on the flow) transparent for the user of the library
- Multithreading (using multicore CPUs) also

But a steep learning curve for the programmer

Advanced programming concepts

- Template metaprogramming (generic programming)
- Design patterns (Factory, Functors, Smart Pointers, ...)

Steep learning curve



Ask questions

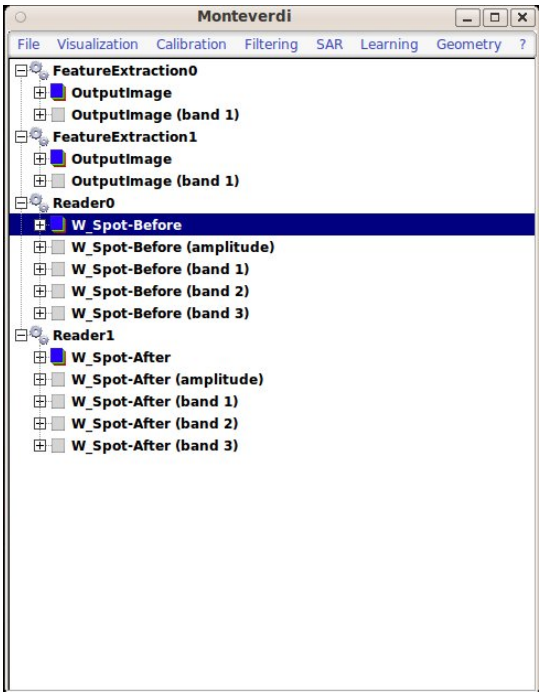
As for everything: easier when you're not alone

- Much easier if you have somebody around to help!
- We didn't know anything not so long ago...
- Not surprising that most software companies now focus their offer on support: help is important

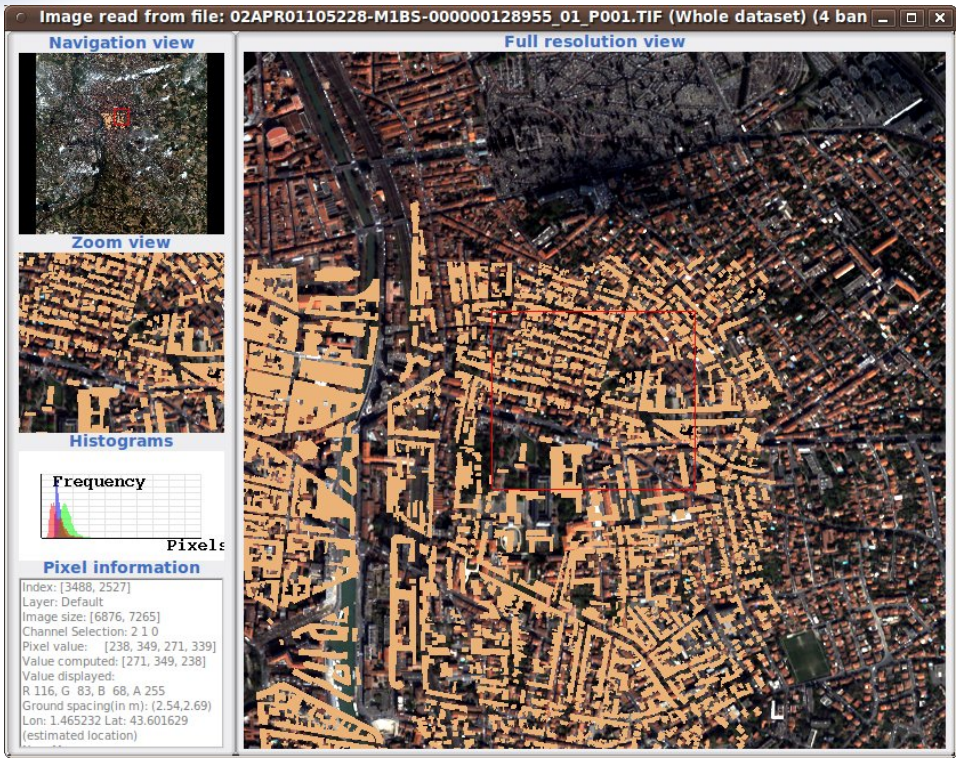
Making it easier for the users: Monteverdi

Module architecture

- Standard input/output
- Easy to customize for a specific purpose
- Full streaming or caching the data
- Each module follows a MVC pattern



Making it easier for the users: Monteverdi



Bindings: access through other languages

Not everybody uses C++!

- Bindings provide an access to the library through other languages
- Python: available
- Java: available
- IDL/Envi: cooperation with ITT VIS to provide a method to access OTB through idl/envi (working but no automatic generation)
- Matlab: recent user contribution (R. Bellens from TU Delft)
- Other languages supported by Cable Swig might be possible (Tcl, Ruby?)

Image radiometry in ORFEO Toolbox

From digital number to reflectance

J. Inglada[1] , E. Christophe[2]

[1]Centre d'Études Spatiales de la Biosphère, Toulouse, France

[2]Centre for Remote Imaging, Sensing and Processing, National University of Singapore

Introduction

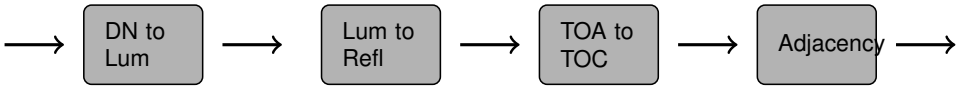
Purpose of radiometry

- Go back to the physics from the image

6S

- Using 6S library: http://6s.ltdri.org/
- Library heavily tested and validated by the community
- Translation from the Fortran code to C
- Transparent integration to OTB for effortless (almost!) corrections by the user

Atmospheric corrections: in four steps



Look like a pipeline

This is fully adapted to the pipeline architecture of OTB

Digital number to luminance

$$\mathbf{L^k_{TOA}} = \frac{X^k}{\alpha_k} + \beta_k$$

Goal

- Transform the digital number in the image into luminance

Using the class
otb::ImageToLuminanceImageFilter
filterImageToLuminance->SetAlpha(alpha);
filterImageToLuminance->SetBeta(beta);

- $\mathbf{L^k_{TOA}}$ is the incident luminance (in $W.m^{-2}.sr^{-1}.\mu m^{-1}$)
- $\mathbf{X^k}$ digital number
- α_k absolute calibration gain for channel k
- β_k absolute calibration bias for channel k

How to get these parameters?

From metadata

- Sometimes, the information can be present in the metadata but...
- the specific format has to be supported (Spot, Quickbird, Ikonos are and more on the way)

From an ASCII file

```
VectorType alpha(nbOfComponent);
alpha.Fill(0);
std::ifstream fin;
fin.open(filename);
double dalpha(0.);
for( unsigned int i=0 ; i < nbOfComponent ; i++)
{
    fin >> dalpha;
    alpha[i] = dalpha;
}
fin.close();
```


Luminance to reflectance

Goal

- Transform the luminance into the reflectance

Using the class otb::LuminanceToReflectanceImageFilter and setting parameters:

```
filterLumToRef->
SetZenithalSolarAngle(zenithSolar);
filterLumToRef-> SetDay(day);
filterLumToRef-> SetMonth(month);
filterLumToRef->
SetSolarIllumination(solarIllumination);
```

$$\rho^k_{TOA} = \frac{\pi.\mathbf{L^k_{TOA}}}{E^k_S.cos(\theta_S).d/d_0}$$

- $\mathbf{rho^k_{TOA}}$ reflectance
- θ_S zenithal solar angle
- E^k_S solar illumination out of atmosphere at a distance d_0 from the Earth
- d/d_0 ratio between Earth-Sun distance at the acquisition and average Earth-Sun distance

Top of atmosphere to top of canopy

Goal

- Compensate for the atmospheric effects

$$\rho_S^{unif} = \frac{\mathbf{A}}{1 + Sx\mathbf{A}} \qquad \mathbf{A} = \frac{\rho_{TOA} - \rho_{atm}}{T(\mu_S).T(\mu_V).t_g^{allgas}}$$

- ρ_{TOA} reflectance at the top of the atmosphere
- ρ_S^{unif} ground reflectance under assumption of a lambertian surface and an uniform environment
- ρ_{atm} intrinsic atmospheric reflectance
- t_g^{allgas} spherical albedo of the atmosphere
- $T(\mu_S)$ downward transmittance
- $T(\mu_V)$ upward transmittance

Top of Atmosphere to top of canopy

- Using the class `otb::ReflectanceToSurfaceReflectanceImageFilter`
  `filterToAtoToC->SetAtmosphericRadiativeTerms(correctionParameters);`
- These parameters are filtered by the `otb::AtmosphericCorrectionParametersTo6SAtmosphericRadiativeTerms` from the `otb::AtmosphericCorrectionParameter` class
- this later class has the methods:

```
parameters->SetSolarZenithalAngle();
parameters->SetSolarAzimutalAngle();
parameters->SetViewingZenithalAngle();
parameters->SetViewingAzimutalAngle();
parameters->SetMonth();
parameters->SetDay();
parameters->SetAtmosphericPressure();
parameters->SetWaterVaporAmount();
parameters->SetOzoneAmount();
parameters->SetAerosolModel();
parameters->SetAerosolOptical();
```

Adjacency effects

$$\rho S = \frac{\rho_S^{unif}.T(\mu_V) - <\rho S>.t_d(\mu_v)}{exp(-\delta/\mu_v)}$$

Goal

- Correct the adjacency effect on the radiometry of pixels

Using the class
`otb::SurfaceAdjacencyEffect6SCorrectionSchemeFilter`
with the parameters

```
filterAdjacency->SetAtmosphericRadiativeTerms();
filterAdjacency->SetZenithalViewingAngle();
filterAdjacency->SetWindowRadius();
filterAdjacency->SetPixelSpacingInKilometers();
```

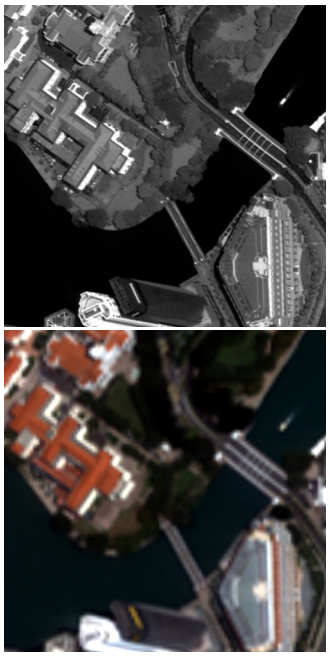
- ρ_S^{unif} ground reflectance assuming homogeneous environment
- $T(\mu_V)$ upward transmittance
- $t_d(\mu_S)$ upward diffuse transmittance
- $exp(-\delta/\mu_v)$ upward direct transmittance
- ρS environment contribution to the pixel target reflectance in the total observed signal

Hands On

1. Monteverdi: Calibration → Optical calibration
2. Select the image to correct
3. Look at the parameters that are retrieved from the image metadata
4. Apply the correction
5. Look at the different levels

Pansharpening

Bring radiometric information to high resolution image



Hands On

1. Monteverdi: Open two images, one PAN and one XS in the same geometry
2. Monteverdi: Filtering $\rightarrow$ Pan Sharpening

Image geometry in ORFEO Toolbox

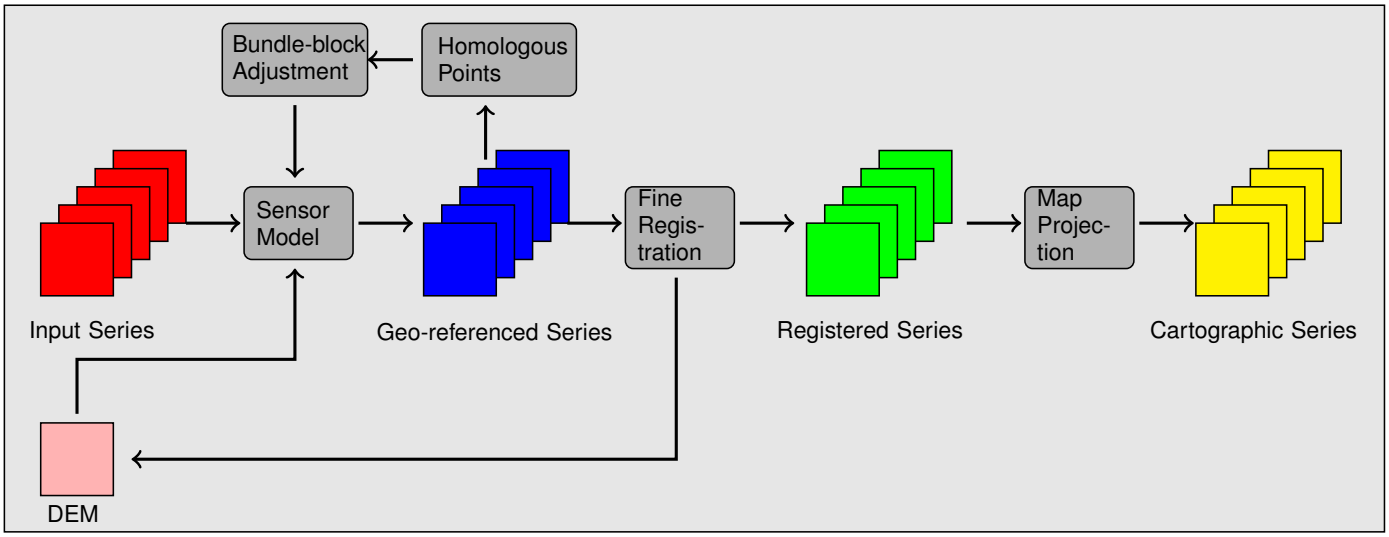
Sensor models and map projections

J. Inglada[1] , E. Christophe[2]

[1]Centre d'Études Spatiales de la Biosphère, Toulouse, France

[2]Centre for Remote Imaging, Sensing and Processing, National University of Singapore

Introduction



Outline of the presentation

Sensor models

Optimizations
- Bundle-block adjustment
- Fine registration

Map projections

Sensor models

What is a sensor model

Gives the relationship between image (l, c) and ground (X, Y) coordinates for every pixel in the image.

Forward

$$X = f_x(l, c, h, \vec{\theta}) \qquad Y = f_y(l, c, h, \vec{\theta})$$

Inverse

$$l = g_l(X, Y, h, \vec{\theta}) \quad c = g_c(X, Y, h, \vec{\theta})$$

Where $\vec{\theta}$ is the set of parameters which describe the sensor and the acquisition geometry.
Height (DEM) must be known.

Sensor models

Types of sensor models

- Physical models
  - Rigorous, complex, highly non-linear equations of the sensor geometry.
  - Usually, difficult to invert.
  - Parameters have a physical meaning.
  - Specific to each sensor.
- General analytical models
  - Ex: polynomial, rational functions, etc.
  - Less accurate.
  - Easier to use.
  - Parameters may have no physical meaning.

Sensor models

OTB's Approach

- Use factories: models are automatically generated using image meta-data.
- Currently tested models:
  - RPC models: Quickbird, Ikonos
  - Physical models: SPOT5
  - SAR models: ERS, ASAR, Radarsat, Cosmo, TerraSAR-X, Palsar
- Under development:
  - Formosat, WorldView 2

Hands On

1. Monteverdi: Open a Quickbird image in sensor geometry
2. Display the image
3. Observe how the geographic coordinates are computed when the cursor moves around the image

Sensor models

How to use them: ortho-registration

1. Read image meta-data and instantiate the model with the given parameters.
2. Define the ROI in ground coordinates (this is your output pixel array)
3. Iterate through the pixels of coordinates (X, Y):
   3.1 Get h from the DEM
   3.2 Compute $(c, l) = G(X, Y, h, \vec{\theta})$
   3.3 Interpolate pixel values if (c, l) are not grid coordinates.

Hands On

1. Monteverdi: Geometry $\rightarrow$ Orthorectification
2. Select the image to orthorectify
3. Set the parameters
4. Save the result
5. Repeat for the other image with the same parameters
6. Display the two images together

Sensor models

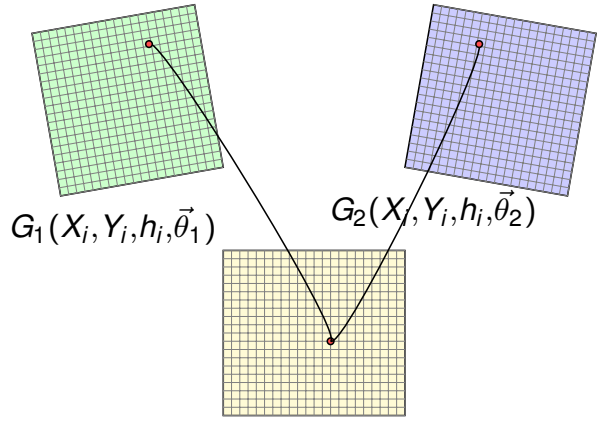
Limits of the approach

- Accurate geo-referencing needs:
  - Accurate DEM
  - Accurate sensor parameters, $\vec{\theta}$
- For time image series we need accurate co-registration:
  - Sub-pixel accuracy
  - For every pixel in the scene
- Current DEM's and sensor parameters can not give this accuracy.
- Solution: use redundant information in the image series!

Bundle-block adjustment

Problem position

- The image series is geo-referenced (using the available DEM, and the prior sensor parameters).
- We assume that homologous points (GCPs, etc.) can be easily obtained from the geo-referenced series : $HP_i = (X_i, Y_i, h_i)$
- For each image, and each point, we can write: $(l_{ij}, c_{ij}) = G_j(X_i, Y_i, h_i, \vec{\theta}_j)$



- Everything is known.

Introduction Models **Optimizations** Map projections **BBA** Fine registration

Bundle-block adjustment

Model refinement

- If we define $\vec{\theta}_j^R = \vec{\theta}_j + \Delta\vec{\theta}_j$ as the refined parameters, $\Delta\vec{\theta}_j$ are the unknowns of the model refinement problem.
- We have much more equations than unknowns if enough HPs are found.
- We solve using non-linear least squares estimation.
  - The derivatives of the sensor model with respect to its parameters are needed.

CESBIO CRISP National University of Singapore
IGARSS 2010, Honolulu

Hands On

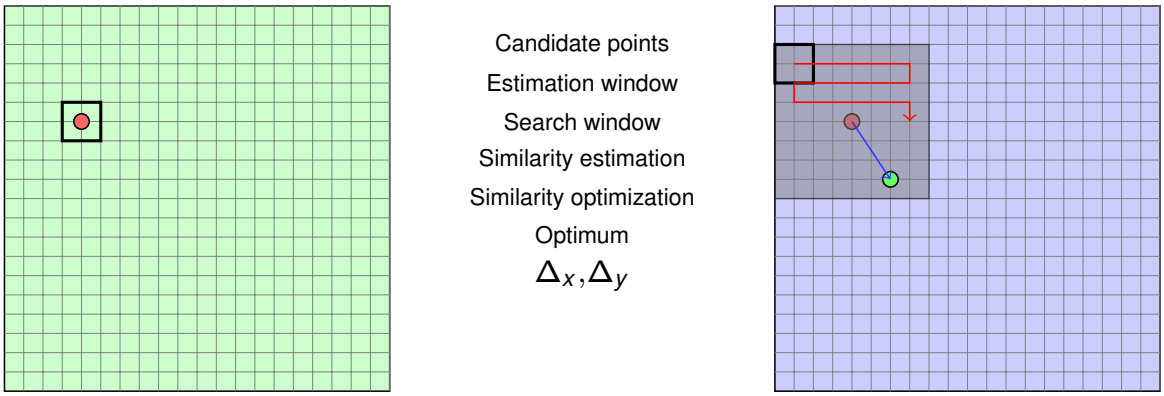
Manually register 2 images

- Monteverdi: Geometry → Homologous points extraction
- Select 2 images with a common region
- The GUI lets you select a transformation
- You can select homologous points in the zoom images and add them to the list
- When several homologous points are available, you can evaluate the transform
- Once the transform is evaluated, you can use the *guess* button to predict the position in the moving image of a point selected in the fixed image
- The GUI displays the parameters of the transform estimated as well as individual error for each point and the MSE
- You can remove from the list the points with higher error

Why fine registration?

- Homologous points have been used to refine the sensor model
- Residual misregistrations exist because of:
  - DEM errors
  - Sensor model approximations
  - Surface objects (high resolution imagery)
- We want to find the homologous point
  - of every pixel in the reference image
  - with sub-pixel accuracy
- This is called disparity map

Fine registration (under development)



Reference Image

Secondary Image

Map projections

What is a map projection

Gives the relationship between geographic (X, Y) and cartographic (x, y) coordinates.

Forward

$$x = f_x(X, Y, \vec{\alpha}) \quad y = f_y(X, Y, \vec{\alpha})$$

Inverse

$$X = g_X(x, y, \vec{\alpha}) \quad Y = g_y(x, y, \vec{\alpha})$$

Where $\vec{\alpha}$ is the set of parameters of a given map projection.

Map projections

Types of map projections

- OTB implements most of the OSSIM ones:
  - 30 map projections are available among which: UTM, TransMercator, LambertConformalConic, etc.
- For any Xyz projection the `otb::XyzForwardProjection` and the `otb::XyzInverseProjection` are available.
- Change of projections can be implemented by using one forward, one inverse and the `otb::CompositeTransform` class.
- One-step ortho-registration can be implemented by combining a sensor model and a map projection.

Hands On

Changing image projections

- Monteverdi: Geometry → Reproject Image
- Select an orthorectified image as input image
- Select the projection you want as output
- Save/Quit

Hands On

Apply the geometry of one image to another

- Monteverdi: Geometry → Superimpose 2 Images
- Select any as image image to reproject
- Select an orthorectified image as reference image
  - Make sure the 2 images have a common region!
- Use the same DEM (if any) as for the ortho-rectified image
- Save/Quit

As a conclusion

1. Use a good sensor model if it exists
2. Use a good DEM if you have it
3. Improve you sensor parameters after a first geo-referencing pass
4. Fine-register your series
5. Use a map projection

- All these steps can be performed with OTB/Monteverdi.
- Sensor model + map projection using a DEM are available as a one-step filter in OTB.

Pragmatic Remote Sensing

Features

J. Inglada[1] , E. Christophe[2]

[1]Centre d'Études Spatiales de la Biosphère, Toulouse, France

[2]Centre for Remote Imaging, Sensing and Processing, National University of Singapore

Features

Expert knowledge

- Features are a way to bring relevant expert knowledge to learning algorithms
- Different type of feature: radiometric indices, textures,. . .
- Important to be able to design your own indices according to the application
- See poster TUP1.PD.9 *"MANGROVE DETECTION FROM HIGH RESOLUTION OPTICAL DATA"*, Tuesday, July 27, 09:40 - 10:45 by Emmanuel Christophe, Choong Min Wong and Soo Chin Liew

Radiometric index

Most popular is the NDVI: Normalized Difference Vegetation Index [1]

$$\mathbf{NDVI} = \frac{L_{NIR} - L_r}{L_{NIR} + L_r} \tag{1}$$

J. W. Rouse. "Monitoring the vernal advancement and retrogradation of natural vegetation,"*Type ii report, NASA/GSFCT, Greenbelt, MD, USA*, 1973. 12.1.1

Vegetation indices 1/3

RVI	Ratio Vegetation Index [1]
PVI	Perpendicular Vegetation Index [2, 3]
SAVI	Soil Adjusted Vegetation Index [4]
TSAVI	Transformed Soil Adjusted Vegetation Index [6, 5]

R. L. Pearson and L. D. Miller. Remote mapping of standing crop biomass for estimation of the productivity of the shortgrass prairie, pawnee national grasslands, colorado. In *Proceedings of the 8th International Symposium on Remote Sensing of the Environment II*, pages 1355–1379, 1972.

A. J. Richardson and C. L. Wiegand. Distinguishing vegetation from soil background information. *Photogrammetric Engineering and Remote Sensing*, 43(12):1541–1552, 1977.

C. L. Wiegand, A. J. Richardson, D. E. Escobar, and A. H. Gerbermann. Vegetation indices in crop assessments. *Remote Sensing of Environment*, 35:105–119, 1991.

A. R. Huete. A soil-adjusted vegetation index (SAVI). *Remote Sensing of Environment*, 25:295–309, 1988.

E. Baret and G. Guyot. Potentials and limits of vegetation indices for LAI and APAR assessment. *Remote Sensing of Environment*, 35:161–173, 1991.

E. Baret, G. Guyot, and D. J. Major. TSAVI: A vegetation index which minimizes soil brightness effects on LAI and APAR estimation. In *Proceedings of the 12th Canadian Symposium on Remote Sensing, Vancouver, Canada*, pages 1355–1358, 1989.

Vegetation indices 2/3

MSAVI	Modified Soil Adjusted Vegetation Index [1]
MSAVI2	Modified Soil Adjusted Vegetation Index [1]
GEMI	Global Environment Monitoring Index [2]
WDVI	Weighted Difference Vegetation Index [3, 4]
AVI	Angular Vegetation Index [5]

J. Qi, A. . Chehbouni, A. Huete, Y. Kerr, and S. Sorooshian. A modified soil adjusted vegetation index. *Remote Sensing of Environment*, 47:1–25, 1994.

B. Pinty and M. M. Verstraete. GEMI: a non-linear index to monitor global vegetation from satellites. *Vegetatio*, 101:15–20, 1992.

J. Clevers. The derivation of a simplified reflectance model for the estimation of leaf area index. *Remote Sensing of Environment*, 25:53–69, 1988.

J. Clevers. Application of the wdvi in estimating lai at the generative stage of barley. *ISPRS Journal of Photogrammetry and Remote Sensing*, 46(1):37–47, 1991.

S. Plummer, P. North, and S. Briggs. The Angular Vegetation Index (AVI): an atmospherically resistant index for the Second Along-Track Scanning Radiometer (ATSR-2). In *Sixth International Symposium on Physical Measurements and Spectral Signatures in Remote Sensing, Val d'Isere*, 1994.

Vegetation indices 3/3

ARVI	Atmospherically Resistant Vegetation Index [1]
TSARVI	Transformed Soil Adjusted Vegetation Index [1]
EVI	Enhanced Vegetation Index [2, 3]
IPVI	Infrared Percentage Vegetation Index [4]
TNDVI	Transformed NDVI [5]

Y. J. Kaufman and D. Tanré. Atmospherically Resistant Vegetation Index (ARVI) for EOS-MODIS. *Transactions on Geoscience and Remote Sensing*, 40(2):261–270, Mar. 1992.

A. R. Huete, C. Justice, and H. Liu. Development of vegetation and soil indices for MODIS-EOS. *Remote Sensing of Environment*, 49:224–234, 1994.

C. O. Justice, et al. The moderate resolution imaging spectroradiometer (MODIS): Land remote sensing for global change research. *IEEE Transactions on Geoscience and Remote Sensing*, 36:1–22, 1998.

R. E. Crippen. Calculating the vegetation index faster. *Remote Sensing of Environment*, 34(1):71–73, 1990.

D. W. Deering, J. W. Rouse, R. H. Haas, and H. H. Schell. Measuring f̈orage productionöf grazing units from Landsat-MSS data. In *Proceedings of the Tenth International Symposium on Remote Sensing of the Environment. ERIM, Ann Arbor, Michigan, USA*, pages 1169–1198, 1975.

Example: NDVI

Soil indices

IR	Redness Index [1]
IC	Color Index [1]
IB	Brilliance Index [2]
IB2	Brilliance Index [2]

P. et al. Caracteristiques spectrales des surfaces sableuses de la region cotiere nord-ouest de l'Egypte: application aux donnees satellitaires Spot. In *2eme Journeees de Teledetection: Caracterisation et suivi des milieux terrestres en regions arides et tropicales*, pages 27–38. ORSTOM, Collection Colloques et Seminaires, Paris, Dec. 1990.

E. Nicoloyanni. Un indice de changement diachronique appliqué deux sècnes Landsat MSS sur Athènes (grèce). *International Journal of Remote Sensing*, 11(9):1617–1623, 1990.

Example: IC

Water indices

SRWI	Simple Ratio Water Index [1]
NDWI	Normalized Difference Water Index [2]
NDWI2	Normalized Difference Water Index [3]
MNDWI	Modified Normalized Difference Water Index [4]
NDPI	Normalized Difference Pond Index [5]
NDTI	Normalized Difference Turbidity Index [5]
SA	Spectral Angle

P. J. Zarco-Tejada and S. Ustin. Modeling canopy water content for carbon estimates from MODIS data at land EOS validation sites. In *International Geoscience and Remote Sensing Symposium, IGARSS '01*, pages 342–344, 2001.

B. cai Gao. NDWI - a normalized difference water index for remote sensing of vegetation liquid water from space. *Remote Sensing of Environment*, 58(3):257–266, Dec. 1996.

S. K. McFeeters. The use of the normalized difference water index (NDWI) in the delineation of open water features. *International Journal of Remote Sensing*, 17(7):1425–1432, 1996.

H. Xu. Modification of normalised difference water index (ndwi) to enhance open water features in remotely sensed imagery. *International Journal of Remote Sensing*, 27(14):3025–3033, 2006.

J. Lacauxa, Y. T. andC. Vignollesa, J. Ndioneb, and M. Lafayec. Classification of ponds from high-spatial resolution remote sensing: Application to rift valley fever epidemics in senegal. *Remote Sensing of Environment*, 106(1):66–74, 2007.

Example: NDWI2

Built-up indices

NDBI	Normalized Difference Built Up Index [1]
ISU	Index Surfaces Built [2]

Y. Z. J. G. S. Ni. Use of normalized difference built-up index in automatically mapping urban areas from TM imagery. *International Journal of Remote Sensing*, 24(3):583–594, 2003.

A. Abdellaoui and A. Rougab. Caractérisation de la reponse du bâti: application au complexe urbain de Blida (Algérie). In *Télédétection des milieux urbains et périurbains, AUPELF - UREF, Actes des sixièmes Journées scientifiques du réseau Télédétection de l'AUF*, pages 47–64, 1997.

Texture

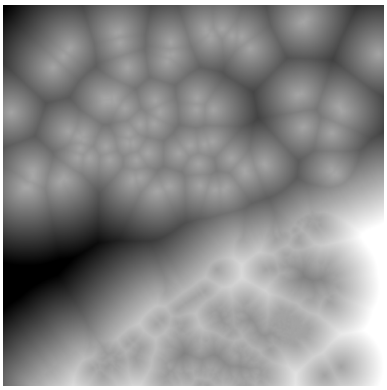
Energy	$f_1 = \sum_{i,j} g(i,j)^2$
Entropy	$f_2 = -\sum_{i,j} g(i,j) \log_2 g(i,j)$, or 0 if $g(i,j) = 0$
Correlation	$f_3 = \sum_{i,j} \frac{(i-\mu)(j-\mu)g(i,j)}{\sigma^2}$
Difference Moment	$f_4 = \sum_{i,j} \frac{1}{1+(i-j)^2} g(i,j)$
Inertia (a.k.a. Contrast)	$f_5 = \sum_{i,j} (i-j)^2 g(i,j)$
Cluster Shade	$f_6 = \sum_{i,j} ((i-\mu)+(j-\mu))^3 g(i,j)$
Cluster Prominence	$f_7 = \sum_{i,j} ((i-\mu)+(j-\mu))^4 g(i,j)$
Haralick's Correlation	$f_8 = \frac{\sum_{i,j}(i,j)g(i,j) - \mu_t^2}{\sigma_t^2}$

Robert M. Haralick, K. Shanmugam, and Its'Hak Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 3, no. 6, pp. 610–621, Nov 1973.

Example: Inertia on the green channel

Using these elements to build others

Example: distance to water



Hands On

1. Monteverdi: Filtering → Feature extraction
2. Select the image to extract the features
3. Choose the feature to generate
4. Try different parameters and look at the preview
5. Using the output tab, pick the relevant features
6. Generate the feature image

Pragmatic Remote Sensing

Image Classification

J. Inglada[1] , E. Christophe[2]

[1]Centre d'Études Spatiales de la Biosphère, Toulouse, France

[2]Centre for Remote Imaging, Sensing and Processing,
National University of Singapore

Outline of the presentation

What is classification

- Classification is the procedure of assigning a class label to objects (pixels in the image)
- Supervised
- Unsupervised
- Pixel-based
- Object oriented

What can be used for classification

- Raw images
- Extracted features
  - Radiometric indices: NDVI, brightness, color, spectral angle, etc.
  - Statistics, textures, etc.
  - Transforms: PCA, MNF, wavelets, etc.
- Ancillary data
  - DEM, Maps, etc.

Classification in a nutshell

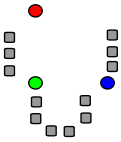
- Select the pertinent attributes (features, etc.)
- Stack them into a vector for each pixel
- Select the appropriate label (in the supervised case)
- Feed your classifier

Unsupervised classification

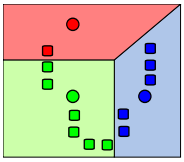
- Also called clustering
- Needs interpretation (relabeling)
  - Class labels are just numbers
- No need for ground truth/examples
  - But often, the number of classes has to be manually selected
  - Other parameters too
- Examples: k-means, ISO-Data, Self Organizing Map

Example: K-means clustering

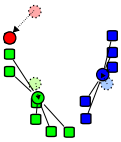
1. k initial "means" are randomly selected from the data set.



2. k clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.



3. The centroid of each of the k clusters becomes the new means.



Steps 2 and 3 are repeated until convergence has been reached.

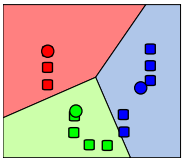
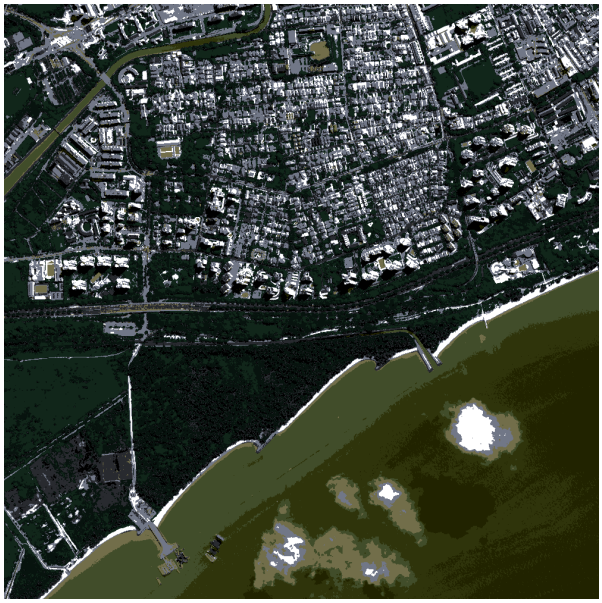


Image credits: Wikipedia

Example: 5 class K-means



Hands On

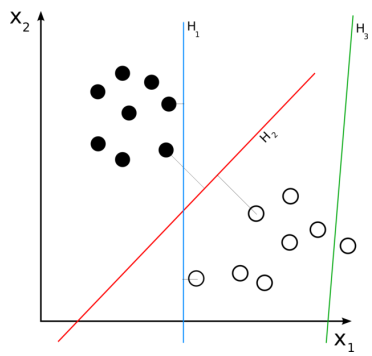
1. Monteverdi: Learning → KMeans Clustering
2. Select the image to classify
3. You can use only a subset of the pixels to perform the centroid estimation
4. Select an appropriate number of classes for your image
5. Set the number of iterations and the convergence threshold
6. Run

Supervised classification

- Needs examples/ground truth
- Examples can have thematic labels
  - Land-Use vs Land-Cover
- Examples: neural networks, Bayesian maximum likelihood, Support Vector Machines

Example: SVM

H3 (green) doesn't separate the 2 classes.
H1 (blue) does, with a small margin.
H2 (red) with the maximum margin.



Maximum-margin hyperplane and margins for a SVM trained with samples from two classes.
Samples on the margin are called the support vectors.

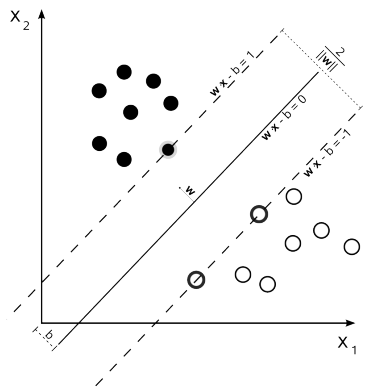
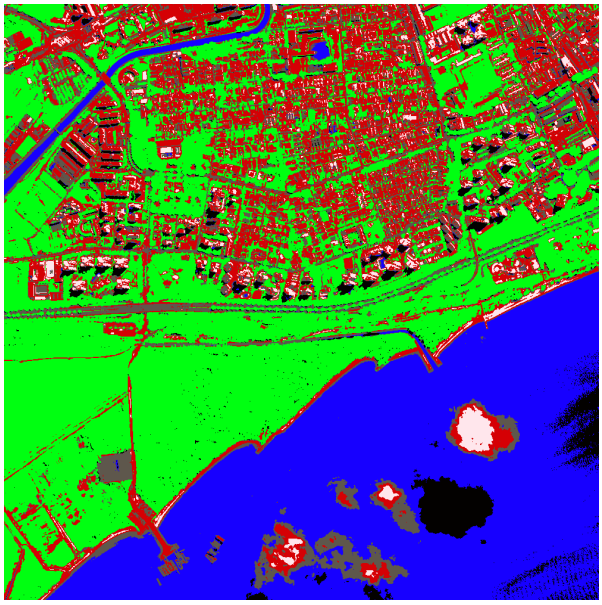


Image credits: Wikipedia

Example: 6 class SVM

Water, vegetation, buildings, roads, clouds, shadows



Hands On

1. Monteverdi: Learning → SVM Classification
2. Select the image to classify
3. Add a class
   - You can give it a name, a color
4. Select samples for each class
   - Draw polygons and use the *End Polygon* to close them
   - You can assign polygons to either the training or the test sets; or you can use the random selection
5. Learn
6. Validate: displays a confusion matrix and the classification accuracy
7. Display results

Object oriented classification

- Pixels may not be the best way to describe the classes of interest
  - shape, size, and other region-based characteristics may be more meaningful
- We need to provide the classifier a set of regions
  - Image segmentation
- And their characteristics
  - Compute features per region
- Since individual objects are meaningful, active learning can be implemented
  - See presentation WE2.L06.2 *"LAZY YET EFFICIENT LAND-COVER MAP GENERATION FOR HR OPTICAL IMAGES"*, Wednesday, July 28, 10:25 - 12:05, by Julien Michel, Julien Malik and Jordi Inglada.

No hands on

But a demo if the time allows it!

Pragmatic Remote Sensing

Change Detection

J. Inglada[1] , E. Christophe[2]

[1]Centre d'Études Spatiales de la Biosphère, Toulouse, France

[2]Centre for Remote Imaging, Sensing and Processing,
National University of Singapore

Outline of the presentation

Classical strategies for change detection

Available detectors in OTB

Interactive Change Detection

Possible approaches

- Strategy 1: Simple detectors
  Produce an image of change likelihood (by differences, ratios or any other approach) and thresholding it in order to produce the change map.
- Strategy 2: Post Classification Comparison
  Obtain two land-use maps independently for each date and comparing them.
- Strategy 3: Joint classification
  Produce the change map directly from a joint classification of both images.

Available detectors

- Pixel-wise differencing of mean image values:

$$I_D(i,j) = I_2(i,j) - I_1(i,j). \tag{1}$$

- Pixel-wise ratio of means:

$$I_R(i,j) = 1 - min\left(\frac{I_2(i,j)}{I_1(i,j)}, \frac{I_1(i,j)}{I_2(i,j)}\right). \tag{2}$$

- Local correlation coefficient:

$$I_\rho(i,j) = \frac{1}{N}\frac{\sum_{i,j}(I_1(i,j) - m_{I_1})(I_2(i,j) - m_{I_2})}{\sigma_{I_1}\sigma_{I_2}} \tag{3}$$

- Kullback-Leibler distance between local distributions (mono and multi-scale)
- Mutual information (several implementations)

Hands On

Displaying differences

1. Monteverdi: File → Concatenate Images
2. Select the amplitudes of the 2 images to compare and build a 2 band image
3. Monteverdi: Visualization → Viewer
4. Select the 2 band image just created
5. In the *Setup* tab, select *RGB composition mode* and select, for instance, 1,2,2.
6. Interpret the colors you observe
7. The same thing could be done using feature images

Hands On

Thresholding differences

1. Monteverdi: Filtering → Band Math
2. Select the amplitudes of the 2 images to compare and compute a subtraction
3. Monteverdi: Filtering → Threshold
4. Select the difference image just created and play with the threshold value
5. The same thing could be done using image ratios
6. The same thing could be done using feature images

Interactive Change Detection

- Generation of binary change maps using a GUI
- Uses simple change detectors as input
- The operator gives examples of the *change* and *no change* classes
- SVM learning and classification are applied

Hands On

Joint Classification

1. Monteverdi: Filtering → Change Detection
2. Select the 2 images to process
   - Raw images at 2 different dates
3. Uncheck *Use Change Detectors*
4. Use the *Changed/Unchanged Class* buttons to select one of the classes
5. Draw polygons on the images in order to give training samples to the algorithm
6. *End Polygon* button is used to close the current polygon
7. After selecting several polygons per class push *Learn*
8. The *Display Results* button will show the result change detection

Hands On

Joint Classification with Change Detectors

1. Monteverdi: Filtering → Change Detection
2. Select the 2 images to process
   - Raw images at 2 different dates
3. Make sure the *Use Change Detectors* check-box is activated
4. Proceed as in the previous case

Hands On

Joint Classification with Features

1. Monteverdi: Filtering → Change Detection
2. Select the 2 images to process
   - Feature images at 2 different dates
3. Proceed as in the previous cases