|  |  |
| --- | --- |
|  |  |

Manuel d’utilisation de la chaine Moringa

Raffaele GAETANO, Cirad UMR TETIS, Montpellier, France

Stéphane DUPUY, Cirad UMR TETIS, Saint-Pierre, Réunion, France

Audrey JOLIVOT, Cirad UMR TETIS, Montpellier, France

Monica LONDONO, Cirad UMR TETIS, Saint-Pierre, Réunion, France

Annelise TRAN, Cirad UMR TETIS, Saint-Denis, Réunion

Valentine LEBOURGEOIS, Cirad UMR TETIS, Montpellier, France

Ce rapport a été élaboré dans le cadre de…

Référence à citer :

Gaetano R., Dupuy S., Jolivot A.,Londoño Villegas M.M., Tran A., Lebourgeois V., 2019, Manuel d’utilisation de la chaine Moringa, Montpellier (France), 31 p.

© Cirad, Département Environnement et Société / Unité Mixte de Recherche TETIS, 2018.

SOMMAIRE

Table des matières

[1. Quelques remarques utiles 4](#_Toc9587878)

[2. Installation de la chaine Moringa 5](#_Toc9587879)

[a. Installation des modules python requis 5](#_Toc9587880)

[b. Clonage et mise à jour du dépôt GIT 5](#_Toc9587881)

[c. Installation des dépendances 7](#_Toc9587882)

[d. Réglages du shell OSGEO 7](#_Toc9587883)

[e. Utilisation des images S2 de niveau 2 diffusées sur PEPS 7](#_Toc9587884)

[f. Installation de Python FMASK et RIOS 7](#_Toc9587885)

[3. Prétraitements de l’image THRS 8](#_Toc9587886)

[a. Conversion en réflectance 8](#_Toc9587887)

[b. Orthorectification d’une image en utilisant une référence 9](#_Toc9587888)

[c. Création des mosaïques 11](#_Toc9587889)

[d. Pansharpening 13](#_Toc9587890)

[4. Téléchargement et prétraitements de la série temporelle 14](#_Toc9587891)

[a. Téléchargement des images 14](#_Toc9587892)

[b. Prétraitements 15](#_Toc9587893)

[5. Calcul des variables auxiliaires 17](#_Toc9587894)

[a. Textures 17](#_Toc9587895)

[b. Pentes 17](#_Toc9587896)

[c. Extraction des bandes de la THRS et des textures 18](#_Toc9587897)

[6. Partition de la base de données de vérité terrain 18](#_Toc9587898)

[7. Choix des paramètres de segmentation 19](#_Toc9587899)

[8. Lancement des étapes de la chaine 19](#_Toc9587900)

[a. Modification du fichier de configuration 19](#_Toc9587901)

[b. Mise en place des fichiers et des dossiers 21](#_Toc9587902)

[c. Lancement de la chaine 21](#_Toc9587903)

[9. Post-traitements 24](#_Toc9587904)

[a. Export de la classification format image 24](#_Toc9587905)

[b. Mosaïque des tuiles raster de la classification 24](#_Toc9587906)

[c. Coloration des classes 24](#_Toc9587907)

[d. Lissage par filtre majoritaire 24](#_Toc9587908)

[10. Validation 25](#_Toc9587909)

[11. Annexes 26](#_Toc9587910)

[Etat des dossiers au cours du traitement 26](#_Toc9587911)

RESUME

Ce manuel expose la démarche suivie pour produire une carte d’occupation du sol, à partir d’une image THRS (SPOT6/7 ou Pléiades), d’une série temporelle d’images Sentinel-2 et Landsat-8, et d’une base de données d’apprentissage (polygones dont l’occupation du sol est connue). La chaîne de traitement fait appel à des fonctions de l’Orfeo Tool Box (OTB), orchestrées par des scripts en python. Certaines étapes intermédiaires sont réalisées sous QGis.

La chaîne met à disposition des outils pour le téléchargement automatique et le prétraitement des données Sentinel-2 et Landsat-8.

La plupart des étapes se réalisent par ligne de commande, via le shell OSGeo, en se positionnant dans le dossier principal d’installation de la chaîne.

SUMMARY

1. Quelques remarques utiles

Ce document décrit l’utilisation de la chaine Moringa sous Windows. La chaine fonctionne également sur Linux.

Les algorithmes d’OTB lancés en ligne de commande sont exécutés sur la fenêtre de commandes d’OSGeo4W (disponible dans le dossier du logiciel QGis).

Si la fenêtre de commandes est bien configurée (Cf. chapitre relatif à l’installation), les lignes de commandes d’OTB peuvent être lancées sur la fenêtre d’OSGeo4W sur n’importe quel dossier. Dans le cas contraire, il faut se placer sur le dossier bin d’OTB.

Pour lancer les algorithmes de la chaîne de traitement sur la fenêtre d’OSGeo4W, il faut se placer sur le dossier moringa.

L’usage des algorithmes d’OTB peut être consulté en écrivant leur nom sur la fenêtre de commandes précédé de « otbcli\_ », par exemple otbcli\_superimpose + entrée.

Pour les algorithmes lancés depuis une fenêtre de commandes, il est préférable de mettre le chemin et le nom des fichiers entre guillemets. Cela est indispensable quand les noms des fichiers et/ou des dossiers contiennent des espaces (à éviter !!).

Sur ce document dans les exemples de commandes qui sont présentés, les explications sont placées entre < >. Il s’agit notamment d’identifier les fichiers et/ou le chemin vers le fichier.

Explication :

**otbcli\_bandmathx -il** <image en entrée : ImageTOA.tif> **-exp "10000\*im1" -out** <Image en sortie : TOAx10000.tif> **uint16**

Application :

**otbcli\_bandmathx -il** "D:\ImageTHR\ImageTOA\_MS\_bas.tif" **-exp "10000\*im1" –out** "D:\ImageTHR\TOAx10000\_MS\_bas.tif" **uint16**

Le chemin d’un dossier ou d’un fichier à renseigner sur les lignes de commande peut être copier en faisant clic droit sur celui-ci dans l’explorateur et en choisissant l’option "copier en tant que chemin d’accès" tout en maintenant appuyée la touche majuscule du clavier.

1. Installation de la chaine Moringa

Cette version de la notice s’applique aux systèmes WINDOWS x64 uniquement, pour des utilisateurs ayant un accès administrateur à la machine. La chaîne nécessite actuellement l’installation des dépendances suivantes :

* Une version récente de QGIS (>= 2.14) avec disponibilité du shell OSGEO (chercher « OSGeo4W Shell » dans les programmes installés)
* L’utilitaire **wget**, téléchargeable ici :
  + <https://eternallybored.org/misc/wget/releases/wget-1.19.1-win64.zip>
* Un build personnalisé de l’**Orfeo Toolbox (OTB)**, téléchargeable à l’adresse suivante :
  + <http://napoli.teledetection.fr/logiciels/otb_moringa_build_win_x64.zip>
* Le module pour le calcul du masque des nuages/ombres **python-fmask** compilé pour Win64, lui aussi disponible à l’adresse suivante, ainsi que sa dépendance **rios** (Raster IO Simplification) :
  + <http://napoli.teledetection.fr/logiciels/fmask_win_x64.zip>
  + <https://bitbucket.org/chchrsc/rios/downloads/rios-1.4.4.zip>
* Le programme **GIT** (pour le téléchargement de la chaîne et la synchronisation avec le dépôt) :
  + <https://github.com/git-for-windows/git/releases/download/v2.12.2.windows.2/Git-2.12.2.2-64-bit.exe>

## Installation des modules python requis

Pour la chaîne Moringa, le module python *scikit-learn* est requis. Pour l’installer, il est possible d’utiliser l’outil *pip* disponible sous le shell OSGeo4W.

Note du 26/07/18 : pour des problèmes de version de l’interprète Python et de compatibilité avec la version du module *numpy,* l’installation nécessite une petite manipulation supplémentaire. À terme il devrait être possible d’installer *scikit-learn* avec une seule simple commande.

Lancer le shell OSeo4W.

Désinstaller la version courante de *numpy* en lançant la commande suivante :

**python -m pip uninstall numpy**

Télécharger à partir du site web <https://www.lfd.uci.edu/~gohlke/pythonlibs/> la dernière version des packages *numpy+mkl* et *scikit-learn* pour Python 2.7 (fichiers contenant le mot clé ‘cp27’) et la plateforme x64. Ex. :

**numpy‑1.15.0+mkl‑cp27‑cp27m‑win\_amd64.whl  
scikit\_learn‑0.19.2‑cp27‑cp27m‑win\_amd64.whl**

Toujours depuis le shell OSGeo4W, lancer l’installation des deux module en lançant pour chacun la commande suivante :

**python -m pip install "<Dossier où se trouvent les whl>\<fichier .whl>"**

## Clonage et mise à jour du dépôt GIT

Premièrement, installer git.

Pour être sûr de disposer toujours de la version courante de la chaîne, il est conseillé de l’acquérir à l’aide du logiciel de contrôle de versions git installé. Normalement, une fois installé, l’outil git est disponible dans le path de système. Pour le vérifier il faut ouvrir la fenêtre Variables d’environnement en allant dans : panneaux de configuration/ Système / Paramètres système avancés / Variables d’environnement. Vérifier dans la partie basse « Variables système » si le path est bien renseigné pour GIT. Si ce n’est pas le cas ajouter le chemin en le séparant de ceux qui y sont déjà avec un « ; »

Si ce n’est pas le cas, ajouter au path le dossier avec le binaire git.exe installé, en utilisant l’interface Windows (Cf. figure1)

|  |  |
| --- | --- |
| Figure 1: Modification des variables système |  |

Démarrer ensuite une invité des commandes (saisir CMD dans l’interface windows), se placer dans le dossier (au choix) où la chaîne sera installé, et cloner le dépôt avec la commande suivante :

**git clone https://gitlab.irstea.fr/raffaele.gaetano/moringa.git**



Figure 2 : Commande pour cloner le dépôt

La chaîne sera donc copiée dans le dossier ./moringa

À partir de ce moment, depuis n’importe quel sous-dossier de ./moringa vous pouvez récupérer la dernière version de la chaine en lançant la commande git pull. Il est conseillé de le faire sur base régulière (par ex. chaque jour) pour être sûr de travailler sur la version la plus stable.

## Installation des dépendances

**OTB** : dézipper le paquet téléchargé dans un dossier au choix. Ex. D:\Dev\OTB

**WGET** : dézipper le paquet téléchargé dans un dossier au choix. Ex. D:\Dev\WGET

## Réglages du shell OSGEO

Dans l’arborescence de Moringa, récupérer le fichier install/moringa\_env.bat et le copier dans le sous-dossier etc/ini du dossier d’installation de QGIS, par ex. :

C:\Program Files\QGIS Essen\etc\ini

Editer ensuite ce fichier batch avec un éditeur de texte et modifier les lignes concernant les dossiers d’installation des dépendances comme dans l’exemple qui suit (attention, rajouter bien les \bin) :

* set OTB\_ROOT=D:\Dev\OTB
* set WGET\_BIN=D:\Dev\WGET
* set GIT\_BIN=C:\Program Files\Git\bin

## Utilisation des images S2 de niveau 2 diffusées sur PEPS

Si les données S2 de niveau 2 de THEIA sont disponibles sur la zone d’étude, il faut utiliser un script pour le téléchargement. Ce script est disponible ici : <https://github.com/olivierhagolle/peps_download>

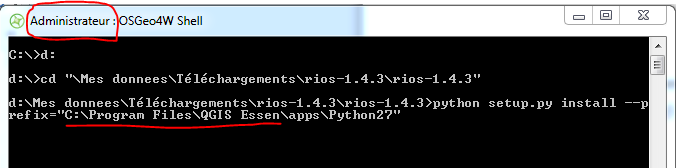
Il faut le télécharger et le copier dans un sous dossier de la chaine comme par exemple : **theia\_download-master**

## Installation de Python FMASK et RIOS

En attendant une solution plus élégante, l’outil python-fmask est à dézipper tout simplement dans le dossier principal du clone git (ex. D:\Dev\moringa).

Pour placer correctement les fichiers, veiller à que les scripts python de démarrage de fmask (le dossier *fmask* contenant les fichiers *fmask\_\*.py*) se trouvent bien au même niveau que ceux de la chaîne (ex. launchChain.py).

L’installation de *rios* se fait par le shell OSGeo (veillez à que sur la fenêtre s’affiche la mention *Administrateur*). Dézipper le contenu du zip dans un dossier au choix, puis s’y rendre et lancer la commande d’installation comme suit (changer le dossier de base de QGIS – C:\Program Files\ QGIS Essen dans l’exemple - avec celui où votre copie de QGIS est installé, et n’oubliez pas les guillemets) :



1. Prétraitements de l’image THRS

Si la zone d’étude est couverte par une seule image Spot 6/7 correctement orthorectifiée et se superposant parfaitement aux polygones de la BD d’apprentissage, il suffit de renseigner dans la chaine le chemin d’accès aux images panchromatique et multispectrale pour configurer la chaine (Cf. chapitre 8). Si elle est couverte par plus d’une image et/ou que cette image doit être recalée par rapport à une autre image (ex. l’image ayant servi de référence à la numérisation des polygones de la BD d’apprentissage), il faut prétraiter la THRS comme indiqué dans ce chapitre qui présente un exemple des traitements effectués à la Réunion où deux scènes SPOT 6 étaient nécessaires pour couvrir la zone d’étude. La Figure 3 résume la démarche suivie pour le prétraitement des images SPOT6 de 2016. Il s’agissait d’une image orthorectifié par Airbus et une image non orthorectifiée acquises le même jour et devant être assemblées.

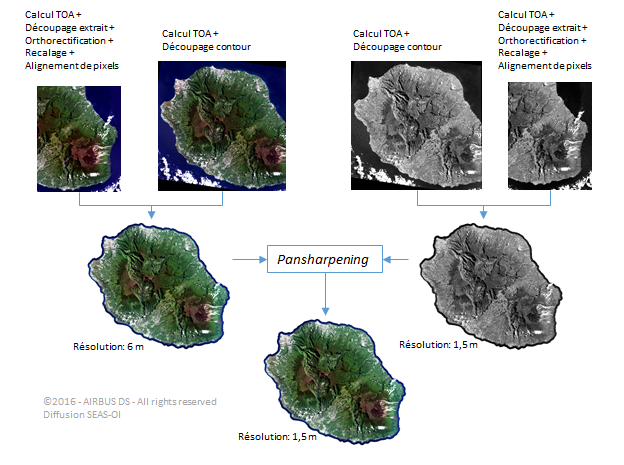


Figure 3 : prétraitements des images Spot6 sur l'ile de la Réunion

Le cas de plusieurs images à mosaïquer sera également traité dans le chapitre « Création des mosaïques » en page 12.

## Conversion en réflectance

La conversion en réflectance TOA a été faite pour les 2 couples d’images SPOT6 (panchromatique et multispectrale). On choisit d’enregistrer l’image en Float puis de la convertir en 16bits en multipliant des valeurs de réflectance par 10000 pour garder la profondeur de l’information.

* Conversion des valeurs en réflectance TOA

Outil : Optical Calibration sur MAPLA (il est également possible d’utiliser la commande otbcli\_OpticalCalibration en ligne de commande d’OSGeo4W)

Sur la fenêtre de dialogue :

* + Input : **< Image brute en entrée >**
  + Output **< Image en sortie : ImageTOA.tif MS ou PAN du haut ou du bas >** Type : float
  + Pixel size : taille de pixel de l’image en entrée
* Multiplication des valeurs par 10000 et conversion en UINT16

Outil : otbcli\_BandMathX d’OTB sur la fenêtre de commande d’OSGeo4W.

Ligne de commande :

***otbcli\_bandmathx -il*** *< image en entrée : ImageTOA.tif >* ***-exp "10000\*im1" -out*** *< Image en sortie : TOAx10000.tif >* ***uint16***

## Orthorectification d’une image en utilisant une référence

Dans notre cas, nous avons une image orthorectifiée et une autre qui ne l’est pas. Les 2 images ont une zone de recouvrement assez importante. Il est donc décidé d’effectuer une première orthorectification, puis d’utiliser la zone de recouvrement pour rechercher des points homologues afin d’effectuer une seconde orthorectification en utilisant un modèle de capteur modifié par le décalage, calculé par comparaison des coordonnées des points homologues localisés sur les 2 images. Cette technique peut également être appliquée si on bénéficie d’une image de référence sur le site d’étude.

* Création d’un extrait de l’image à corriger (facultatif)

Ce découpage doit être effectué dans le cas où la zone de superposition des images est importante et lorsque la trop grande quantité de points homologue complique la création du nouveau modèle de capteur. Dans notre cas, nous avons réduit la zone de chevauchement entre les images de l’Est et de l’Ouest.

Il s’agit de créer un extrait des images de l’Est (MS et PAN) qui comprend la zone manquante sur l’image de l’Ouest et une zone de chevauchement. D’abord l’empreinte est créée sur QGis pour qu‘elle soit utilisée pour faire l’extraction de l’image avec son fichier de métadonnées (.geom), sur OTB.

**Création de l’empreinte avec l’outil :** « découper » de QGIS (menu Raster : Extraction) :

Sur la fenêtre de dialogue :

* + Fichier source : **< Image en entrée : TOAx10000.tif MS ou PAN >**
  + Fichier en sortie : **< Image en sortie : ExtraitTOA.tif >**
  + Choisir l’option «Emprise » et dessiner l’emprise sur le canevas.

**Création de l’extrait avec l’outil :** ExtractROI d’OTB sur la fenêtre de commande d’OSGeo4W

Ligne de commande :

**otbcli\_extractROI -in** < image en entrée : TOAx10000.tif >**?&skipcarto=true -out** < Image en sortie : ExRoiTOAx10000.tif > **uint16 –mode fit –mode.fit.ref** < ExtraitTOA.tif >

Exemple :

***otbcli\_extractROI –in "D:\TOAx10000\_MS\_bas.tif?&skipcarto=true" -out "D:\ExRoiTOAx10000\_MS\_bas.tif" uint16 –mode fit –mode.fit.ref "D:\ExtraitTOA\_MS\_bas.tif"***

Une autre solution peut être de réduire la quantité maximale de points homologues à utiliser en tant que GCP (via l’argument « maxpoints » qui est par défaut égal à -1, mais que l’on peut fixer à 300 ou 500 par ex.) au moment de la coregistration, en utilisant l’outil coregister.py disponible dans la chaîne Moringa. Cet outil combine les différentes étapes de coregistration présentées ci-après (otbcli\_HomologousPointsExtraction, homoGeo2Pixel, otbcli\_GenerateRPCSensorModel, otbcli\_OrthoRectification) et s’utilise comme suit :

coregister.py [--band-ref <target ref band>] [--band-in <source band>] [--step <geobins initial step def. 256>] [--minstep <geobins minimum step def. 16>] [--minpoints <minimum # of homologous points def. 40>] [--maxpoints <maximum # of GCP points def. -1 unlimited> [--prec <colocalisation precision in pixels] <vhsr scene file> <stack to coregister>')

* Première orthorectification

Pour les extraits créés précédemment sur les images MS et PAN de l’Est.

Outil : otbcli\_OrthoRectification d’OTB sur la fenêtre de commande d’OSGeo4W.

Ligne de commande :

**otbcli\_orthorectification -io.in** <image en entrée : ExRoiTOAx10000.tif> **-io.out** < Image en sortie : OrthoExRoiTOAx10000.tif > **uint16 -map epsg -map.epsg.code 32740 -elev.dem** < dossier du SRTM > **-opt.gridspacing 3**

NB. Cet outil fait appel à l’utilisation d’un modèle numérique de terrain. Nous pouvons utiliser le DEM SRTM à 30m de résolution téléchargeable par exemple sur le site de l’USGS (https://earthexplorer.usgs.gov/). Télécharger les tuiles correspondant à votre zone d’étude, les dézipper et les placer dans un même dossier (non mosaïquées !)

* Extraction des points homologues par rapport à l’image de référence

L’extraction des points homologues a été faite seulement une fois sur l’image panchromatique de l’Est avec comme référence l’image panchromatique de l’Ouest. Ces mêmes points homologues sont utilisés aussi pour le recalage de l’image MS.

Outil : HomologousPointsExtraction d’OTB sur la fenêtre de commande d’OSGeo4W

Ligne de commande :

**otbcli\_homologouspointsextraction -in1** <image 1 : OrthoExRoiTOAx10000.tif de l’image PAN> **-in2** <Image 2 : TOAx10000.tif de l’image panchromatique du haut> **-out** <fichier texte en sortie des points homologues : PH.txt> **-outvector** <couche en sortie des points homologues : PH.shp> **-mode geobins -mode.geobins.binstep 100 -mode.geobins.binsize 100 -backmatching 1 -precision 10 -mfilter 1 -2wgs84 1 -elev.dem** <dossier SRTM> **-algorithm sift**

* Passage des points homologues à coordonnées pixel

Cela doit se faire de manière indépendante pour l’image panchromatique et multispectrale à partir du même fichier de points homologues.

Outil : mtdUtils.homoGeo2Pixel disponible sur le dossier de la chaîne de traitement.

Il faut d’abord se placer, dans l’invite de commandes d’OSGeo4W, sur le dossier de la chaine de traitement.

Ligne de commande :

**python -c "import mtdUtils; mtdUtils.homoGeo2Pixel('<PH.txt>','<OrthoExRoiTOAx10000.tif de l’image MS ou PAN, selon le cas>','<fichier en sortie des points homologues transformés : PHT.txt de la MS ou la PAN>')"**

***Exemple pour l’image MS:***

***python -c "import mtdUtils; mtdUtils.homoGeo2Pixel('"D:/PH.txt"','"D:/OrthoExRoiTOAx10000\_MS\_bas"','"D/PHT\_MS.txt>')"***

**Pour cette commande il faut changer les « \ » des chemins copiés des fichiers par « / ».**

* Création d’un nouveau modèle de capteur

Outil : GenerateRPCSensorModel sur la fenêtre de commande d’OSGeo4W

Ligne de commande :

**otbcli\_generaterpcsensormodel -inpoints** <PHT.txt de l’image MS ou PAN selon le cas> **-outgeom** <fichier du nouveau modèle du capteur NMC.geom> **-elev.dem** <dossier SRTM >

* Orthorectification avec le nouveau modèle de capteur

Outil : otbcli\_OrthoRectification d’OTB sur la fenêtre de commande d’OSGeo4W.

Ligne de commande :

**otbcli\_orthorectification -io.in "<OrthoExRoiTOAx10000.tif>?&skipcarto=true&geom=<NMC.geom>" -io.out** <Fichier en sortie : OrthoFinalExRoiTOAx10000.tif> **uint16 -map epsg -map.epsg.code 32740 -opt.gridspacing 3 -outputs.spacingx 1.5 -outputs.spacingy -1.5 -elev.dem <dossier SRTM>**

* Découpage final de l’extrait de l’image du bas

Ce découpage final est fait de manière similaire au premier découpage. Le but dans ce cas est d’éliminer la zone de chevauchement de l’extrait pour que l’image du haut soit celle prise en compte sur cette zone dans la mosaïque.

## Création des mosaïques

* **Cas 1 : utilisation de la mosaïque virtuelle de QGIS (VRT)**

Il faut faire tout le processus pour chacune des images : la panchromatique et la multispectrale

* + Découpage de l’image de l’Ouest

Outil : « Découper un raster selon une couche de masque » de GDAL, disponible sur la boîte à outils de traitements de QGis.

Sur la fenêtre de dialogue :

* + - Couche en entrée : <TOAx10000.tif de l’image du haut PAN ou MS, selon le cas>
    - Couche de masquage : <contour de La Réunion avec une zone tampon de 1000 m>
    - Cocher les options « Couper l’étendue du jeu de données cible selon l’étendue du trait de coupe » et « Garder la résolution du raster de sortie »
    - Paramètres avancés :
      * Type de raster en sortie : UInt16
      * Type de compression : NONE
      * Découpé : <Image du haut découpée : BaseMosaique.tif>
  + Alignement de la grille des pixels de l’extrait de l’image de l’Est

Outil : « Superimpose » d’OTB sur la fenêtre de commande d’OSGeo4W.

Ligne de commande :

**otbcli\_Superimpose.bat -inr** <image de référence : BaseMosaique.tif> **-inm** <image en entrée: OrthoFinalExRoiTOAx10000.tif> **-out** <image en sortie : SuperimposeOrthoFinalExRoiTOAx10000.tif> **uint16**

* Création du raster virtuel de la mosaïque

Outil : « Construire un raster virtuel » de QGis (menu raster : divers).

Sur la fenêtre de dialogue :

* + - Fichiers source : <BaseMosaique.tif, SuperimposeOrthoFinalExRoiTOAx10000.tif>
    - Vérifier que sur le cadre de texte que l’extrait de l’image du bas est placé en première>
    - Fichier en sortie : <preMosaique.vrt>
* Découpage de la mosaïque avec le contour

Outil : « Découper un raster selon une couche de masque ».

Sur la fenêtre de dialogue :

* Couche en entrée : <preMosaique.vrt >
* Couche de masquage : <contour de La Réunion avec une zone tampon de 1000 m>
* Cocher l’option « Garder la résolution du raster de sortie »
* Paramètres avancés :
  + - Type de raster en sortie : UInt16
    - Type de compression : NONE
    - Découpé : <Image du haut découpée : Mosaique.tif>
* **Cas 2 : utilisation de l’outil de mosaïque d’OTB**

La méthode mise en œuvre mobilise l’outil de mosaïque de l’OTB décrit dans cet article Cresson & Saint-Geours 2015[[1]](#footnote-1).

Cet outil permet de faire de l’égalisation radiométrique afin de limiter la différence entre les images. Comme nous souhaitons conserver les 4 bandes des images, nous avons privilégié une égalisation indépendante de chacune des bandes en utilisant une fonction de coût. Comme cette égalisation est sensible aux nuages, nous avons préalablement découpé chaque image en éliminant les zones nuageuses. Il est possible d’utiliser des zonages au format vecteur shape (dans le même ordre que les images en entrée) afin de déterminer d’une part les lignes de découpage et d’autre part les zones de calculs de statistiques qui permettent d’harmoniser les couleurs en évitant de découper les images. Dans notre cas, le résultat était meilleur en découpant les images. Attention lors de l’utilisation de vecteurs aux problèmes de topologie. Il est préférable de vérifier avant d’exécuter le script car en cas de problème de ce type il n’y a pas de message d’erreur et les masques ne sont pas utilisés.

Voici la commande utilisée : **otbcli\_Mosaic -il** <image1.tif> <image2.tif> <image3.tif> <image4.tif> **-vdcut** <vecteur decoupage1.shp> <vecteur decoupage2.shp> <vecteur decoupage3.shp> <vecteur decoupage4.shp> **-vdstats** <vecteur stat1.shp> <vecteur stat2.shp> <vecteur stat3.shp> <vecteur stat4.shp> **-out <resultat\_mosaique.tif> -harmo.method band -harmo.cost rmse**

Cette commande a été appliquée de la même manière sur les images multispectrales et sur les images panchromatiques. Attention l’ordre des vecteurs doit être le même que celui des images.

## Pansharpening

La chaine utilise l’algorithme Bayes pour la pansharpening. Dans le cas où on préfère produire cette image en dehors de la chaine, voici les traitements à effectuer

* Superimpose

La fonction superimpose sensor permet de re-échantillonner la taille des pixels d’une image par rapport à une image de référence. Ici cette étape est nécessaire avant de lancer le pansharp et elle s’applique sur l’image multispectrale en prenant comme référence l’image panchromatique du même bundle.

**otbcli\_Superimpose.bat -inr** <image panchromatique orthorectifiée avec le nouveau modèle de capteur au format .tif > **-inm** <image multispectrale orthorectifiée avec le nouveau modèle de capteur au format .tif > **--out** <image multispectrale en sortie au format .tif > **uint16**

* PANSHARP

Ci-dessous 2 méthodes pour obtenir une image fusionnée.

Bayes

**otbcli\_Pansharpening -inp** <image panchromatique orthorectifiée avec le nouveau modèle de capteur au format .tif > **-inxs** <image multispectrale issue du superimpose au format .tif > **-out** <image fusionnée en sortie au format .tif > **uint16 -method.bayes.lambda 0.99990 -method.bayes.s 1**

Bundle to perfect sensor

**otbcli\_BundleToPerfectSensor -inp** <image panchromatique orthorectifiée avec le nouveau modèle de capteur au format .tif > **-inxs** <image multispectrale issue du superimpose au format .tif > **-out** <image fusionnée en sortie au format .tif > **uint16**

1. Téléchargement et prétraitements de la série temporelle

Pour ces étapes il faut d’abord se placer, dans l’invite de commandes d’OSGeo4W, sur le dossier de la chaîne Moringa. L’usage de chaque algorithme est disponible en écrivant seulement son nom sur cette fenêtre.

## Téléchargement des images

* Cas d’un site non couvert par les images S2 de niveau 2 (Theia)

Pour plus de simplicité, il convient de lancer des requêtes de téléchargement à partir d’un shapefile contenant l’emprise de la zone d’étude (un polygone de forme quelconque). L’outil à disposition identifie automatiquement les tuiles, sélectionne les dates à télécharger selon des critères et génère un fichier batch (à lancer en différé) qui récupère toutes les images.

Deux outils python séparés sont à disposition pour générer les batchs de téléchargement des données S2 et L8, respectivement **genS2DownloadScript.py** et **genL8DownloadScript.py**.

Tous les outils de la chaîne ont une documentation (très minimaliste) qui s’affiche lorsque les scripts python sont lancés sans paramètres :

**D:\Dev\moringa> python genS2DownloadScript.py**

**Usage: python genS2DownloadScript.py [-o <output-dir>] [-c] [-r] [-m <bufsize>] [--min-coverage <perc>] [--max-clouds <perc>] <shapefile OR colon separated tile numbers> <start date> <end date>**

Les options ont la signification suivante :

* + -o <output-dir> : dossier racine où les données seront téléchargées
  + -c : permet de découper les images à partir du shapefile fourni
  + -r : efface les données sources après découpage
  + -m <bufsize> : applique un masque de nuage à partir des données ancillaires avec un buffer de bufsize en unités de la projection (deprecated)
  + --min-coverage <perc> : uniquement pour S2, exclut les images contenant seulement une portion de tuile inférieure au pourcentage saisi
  + --max-clouds <perc> : exclut les images contenant un pourcentage de nuages (sur la tuile entière) supérieure à celle saisie
  + shapefile : shapefile contenant l’emprise de recherche
  + start/end date : dates de début et fin de la recherche au format YYYYMMDD

Exemple d’utilisation :

**D:\Dev\moringa> python genS2DownloadScript.py –o D:\TEST –c –-min-coverage 90 –-max-clouds 50 D:\TEST\zone.shp 20160101 20161231**

Après exécution, qui peut durer quelques minutes si l’intervalle de dates est long et la connexion est lente, le script génère deux fichier, S2DownloadScript.bat, à lancer pour démarrer le téléchargement, et S2SearchLog.txt, qui illustre quels sont les produits trouvés et acceptés (NB. Vérifier le contenu du S2SearchLog.txt après la requête pour s’assurer que toutes les tuiles / dates y sont).

Les images seront téléchargées chacune dans un dossier type D:\TEST\S2\_<DATE>\_<TUILE>. Le fonctionnement pour Landsat 8 est analogue.

Il convient de vérifier après le téléchargement si toutes les tuiles présentes dans le SearchLog.txt ont bien été téléchargées (pour éviter les bugs de téléchargement pas toujours signalés).

* Génération du fichier de téléchargement
  + Sentinel 2

**python genS2DownloadScript.py -o** <dossier d’enregistrement des images> **-c -r --min-coverage 75 --max-clouds 50** <fichier .shp du contour de la zone d’étude> **AAAAMMJJ AAAAMMJJ**

* + Landsat 8

**python genL8DownloadScript.py -o** <dossier d’enregistrement des images> **-c -r --max-clouds 50** <fichier .shp du contour de la zone d’étude> **AAAAMMJJ AAAAMMJJ**

* Cas d’un site couvert par les images S2 de niveau 2 (Theia)

**Usage: python theia\_download.py -t** < numéro de tuile commeT40KCB> **-c SENTINEL2 -a config\_theia.cfg -d <date début YYYY-MM-DD> -f <date fin YYYY-MM-DD>**

Après exécution, le script génère un fichier, S2THEIAPreparation.bat, à lancer pour démarrer le téléchargement.

* Lancement du téléchargement des images

Sur la fenêtre de commandes d’OSGeo4W, dossier Moringa:

* + Sentinel-2 : **S2DownloadScript.bat**
  + Landsat-8 : **L8DownloadScript.bat**
  + Sentinel-2 THEIA : **S2THEIAPreparation.bat**

## Prétraitements

Tout comme pour les téléchargements et découpages, les prétraitements (conversion TOA et Pansharpening – pour L8, rééchantillonage des bandes à 10m et layer stack – pour S2, détection des masques de nuages et ombres pour S2 et L8) sont automatisés. Attention, cette partie crée des masques de nuages propres mais ne découpe pas les images avec (les masques sont utilisés ultérieurement pour le gap filling).

* Génération du fichier de prétraitement pour S2 de niveau 1 et L8

La préparation des prétraitements se fait par le script **genProcessScript.py** à la fois pour S2 et L8, en même temps si toutes les images se trouvent sous le même dossier racine. Encore une fois, cela ne lance pas les traitements, mais génère un batch que vous pouvez lancer ensuite (**ProcessScript.bat**).

**D:\Dev\moringa> python genProcessScript.py**

**Usage: python genProcessScript.py [--cloudmask] [--s2ref <ref\_band>] [--s2bandlist <colon separated band list>]** <images-dir>

Les options ont la signification suivante :

* + --cloudmask : déclenche l’algorithme de détection nuages/ombres (un seul masque binaire sera produit)
  + --s2ref <ref\_band> : uniquement pour S2, détermine la bande ayant la résolution spatiale de référence du stack final. Valeur par déf. B02 (à 10m)
  + --s2bandlist <list> : uniquement pour S2, détermine les bandes à inclure dans le stack final. Valeur par déf. B02:B03:B04:B05:B06:B07:B08:B8A:B11:B12 (toutes les bandes à 10 et 20m)
  + images-dir : dossier contenant les sous-dossiers avec les images (dans l’ex. précédent, D:\TEST)

Exemple d’utilisation :

D:\Dev\moringa> python genProcessScript.py –-cloudmask D:\TEST

Si les images Sentinel-2 et Landsat-8 sont placées sur des dossiers différents, il faut faire le processus pour chaque cas.

**python genProcessScript.py --cloudmask <Dossier qui contient les images téléchargées>**

***Attention le script* genProcessScript *ne traite que la première image d’un dossier : par exemple si les images entières et les images découpées sont dans le même dossier (option -r non choisie lors du téléchargement des images avec* genL8DownloadScript.py)*, il faut préalablement déplacer les images entières pour ne garder dans le <Dossier qui contient les images téléchargées> que les images à pré-traiter (vérifier l’état des dossiers au cours du traitement : annexe 10).***

Lancer ensuite le traitement avec : **ProcessScript.bat**

* Génération du fichier de prétraitement pour S2 de niveau 2 de THEIA

La préparation des prétraitements se fait par le script **S2THEIAPreparation.py**

**Usage : python prepareS2THEIA.py -o /tetis/data/GABIR\_2016/S2THEIA -c /tetis/data/GABIR\_2016/contour974\_epsg32740\_buffer1000m.shp /tetis/data/GABIR\_2016\_src/S2\_TOC/**

Lancer ensuite le traitement avec : **S2THEIAPreparation.bat**

* Cas des zones d’étude à cheval sur plusieurs tuiles
  + - Pour S2

Une fois les tuiles téléchargées dans un dossier père, on donne en entrée ce dossier et le script cherche automatiquement les sous-ensembles des tuiles acquises à max 3 jours d'écart (recherche en avant uniquement) et fait la mosaïque avec comme date de réf la date plus vieille de chaque lot.

**python genMosaicScript.py --onstack --extent <shp de l’emprise de la zone à mosaïquer> <dossier\_père>**

La commande lance directement le traitement.

* + - Pour L8

Même fonctionnement que pour S2.

1. Calcul des variables auxiliaires

## Textures

Les textures sont à calculer sur l’image Panchromatique SPOT6/7 (mosaïquée si vous avez plusieurs tuiles).

Outil : SelectiveHaralickTextures d’OTB sur la fenêtre de commande d’OSGeo4W

**otbcli\_SelectiveHaralickTextures.bat -in** <image panchromatique> **-parameters.xrad 11 -parameters.yrad 11 -parameters.min 307 -parameters.max 1680 -parameters.nbbin 64 -indices Contrast Correlation Energy Variance -out** <image en sortie avec les indices de texture en format tif>

* + - Les paramètres parameters.xrad et parameters.yrad déterminent la taille de la fenêtre glissante en pixels
    - Les paramètres parameters.min et parameters.max correspondent aux valeurs minimale et maximale de l’image panchromatique. (ils peuvent correspondre aux valeurs limites des classes qui veulent être différenciées à l’aide de la texture, dans ce cas c’est mieux de diminuer la valeur de –parameters.nbbin, par exemple à 16).
    - Les autres indices disponibles mais qui n’ont pas été inclus dans l’étude sont : correlation, mean, variance et dissimilarity

## Pentes

Télécharger les dalles SRTM à 30m de résolution spatiale sur le site https://earthexplorer.usgs.gov/

Mosaïquer les dalles à l’aide de la fonction otbcli\_mosaic

Pour cela, ouvrir une fenêtre OSGeo4W, se placer dans le dossier contenant les tuiles puis lancer la fonction.

Ex d’utilisation :

F:\Data\_CS\DEM>otbcli\_mosaic -il s18\_e047\_1arc\_v3.tif s18\_e048\_1arc\_v3.tif s19\_e045\_1arc\_v3.tif s19\_e046\_1arc\_v3.tif s19\_e047\_1arc\_v3.tif s19\_e048\_1arc\_v3.tif s20\_e045\_1arc\_v3.tif s20\_e046\_1arc\_v3.tif s20\_e047\_1arc\_v3.tif s20\_e048\_1arc\_v3.tif s21\_e045\_1arc\_v3.tif s21\_e046\_1arc\_v3.tif s21\_e047\_1arc\_v3.tif s21\_e048\_1arc\_v3.tif s22\_e047\_1arc\_v3.tif s22\_e048\_1arc\_v3.tif -out mosaic\_SRTM30m.tif

Sous QGIS, enregistrer la mosaïque de SRTM dans la bonne projection (ici EPSG32738), puis découper la mosaïque avec le shape de contour de la zone d’étude à l’aide de la fonction GDAL « Découper un raster à l’aide d’une couche de masque » (Cf. Figure 4).

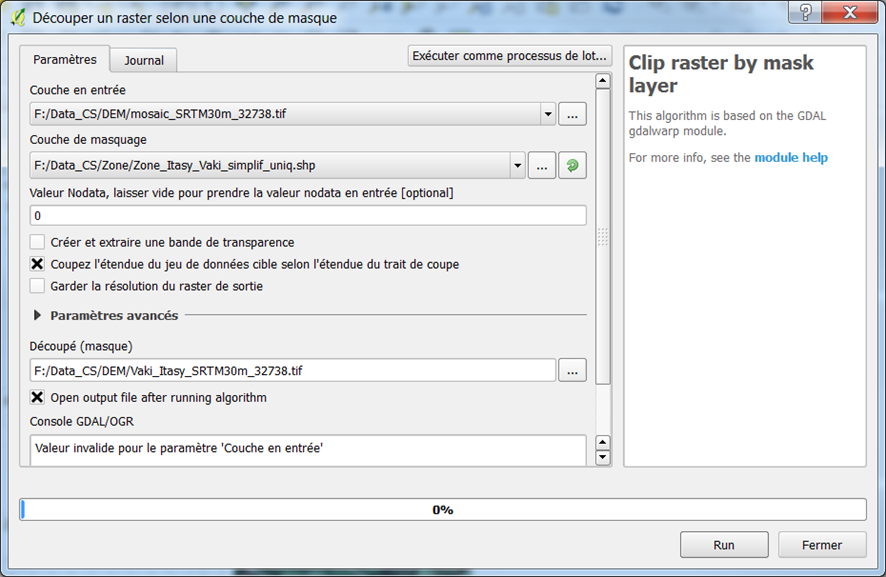


Figure 4 : Utilisation de l'outil de découpe d'un raster selon une couche de masque de QGIS

Calculer ensuite la pente.

Outil : « Pente » du menu « Raster : Analyse de terrain » de QGis.

## Extraction des bandes de la THRS et des textures

Outil : Bandmathx d’OTB sur MAPLA. Expressions : im1b1 (pour le rouge), im1b2 (pour le vert)

1. Partition de la base de données de vérité terrain

Moringa dispose d’un processus de validation croisée (K-fold) mais il est possible de mettre de côté une partie de la BD terrain pour procéder à une validation sur l’ensemble de polygones mis de coté et non utilisés par l’apprentissage de l’algorithme de classification. La base de données constituant la vérité terrain se compose de polygones dont l’OS est connue (selon une nomenclature donnée), et dont les contours ont été numérisés sur la THRS SPOT 6/7 ou Pléiades. Une partie de la base de données est utilisée pour faire l’entraînement de l’algorithme de classification, mais il faut extraire 20% des parcelles de chaque classe pour la validation

Outil : QGIS « Sélection aléatoire depuis des sous-ensembles », menu Vecteur : Outils de recherche.

Partitionner selon le niveau le plus fin (crop type ou simplified crop type si vous avez fait des regroupements de classes). Ne garder que les relevés d’OS dont la population est suffisante (ex. 20 parcelles), éliminer les autres relevés.

L’outil de sélection aléatoire va sélectionner les entités (20%). Il suffit ensuite « d’enregistrer sous » le shapefile en précisant de n’enregistrer que les entités sélectionnées (bd\_terrain\_validation.shp) puis inverser la sélection et re-sauver les entités sélectionnées pour garder les 80% restants dans un autre shape d’apprentissage (bd\_terrain\_train.shp)

1. Choix des paramètres de segmentation

Les tests doivent être effectués sur un extrait d’image d’environ 3000 X 3000 pixels en utilisant la version de base de l’algorithme de segmentation.

**otbcli\_GenericRegionMerging -in** <image\_à\_segmenter> **-out** <raster\_de\_segmentation> **int32 -threshold** <paramètre d’échelle> **-cw** <poids radiométrie> **-sw** <poids compacité>

Remarque importante: le cw est l'opposé du paramètre shape dans eCognition --> 1 veut dire qu’on donne une importance maximum à la radiométrie. La chaîne prend en compte cette inversion (1-x est appliqué avant de lancer l’algorithme) il convient donc de bien paramètrer le fichier de configuration en saisissant la valeur inverse par rapport aux tests : 0,9 dans le test devient 0,1 dans le fichier de configuration.

La segmentation est enregistrée en format raster. Pour passer en vecteur, afficher la couche dans QGIS et vectoriser avec l’outil basique de Qgis (raster/ conversion) afin de tester l’adéquation de la segmentation aux contours de parcelles et autres objets de la zone d’étude.

1. Lancement des étapes de la chaine

## Modification du fichier de configuration

Le fichier de configuration contient toutes les informations dont la chaine a besoin pour fonctionner. Il faut utiliser le modèle téléchargé automatiquement avec les autres outils de la chaine (dans le dossier moringa)  : chain\_setup\_win.cfg pour Windows ou chain\_setup\_lnx.cfg pour Linux.

Il est également important de mettre en place les données dans les dossiers adéquats pour qu’ils soient accessibles.

* + **[GENERAL CONFIGURATION]** 
    - **BASEFOLDER :** C’est le chemin du dossier de travail. Si par exemple, on souhaite utiliser 2 types d’images THRS (Pléiades et Spot6/7), à ce stade nous créons un dossier unique qui contiendra toutes les données (THRS, série temporelle…).
    - **EXTENT :** C’est le chemin et le nom de la couche du contour de la zone d’étude
    - **REFERENCE :** C’est le chemin de la couche qui contient les parcelles d’apprentissage (au format shapefile)
    - **CHAINNAME :** C’est le nom du dossier que les algorithmes vont créer, dans le dossier BASEFOLDER pour mettre les résultats (comme par exemple le dossier de la segmentation, la classification… Par exemple pour tester Pléiades, on choisit ici « RESULT\_PLEIADES ». Il faudra ensuite créer un autre fichier de config pour Spot6/7 dans lequel CHAINNAME s’appellera « RESULT\_SPOT67 »
    - **SETUPNAME :** il s’agit ici de faciliter les expérimentations sans dupliquer les données. Si on souhaite tester la chaine en utilisant les données S2 seules et comparer les résultats en utilisant S2+L8, on change le SETUPNAME et nous créons pour chaque cas un fichier de config different.
    - **CHAINMODE :** il faut préciser ici le type de résultat souhaité : soit la production de la validation seule, soit les cartes, soit les 2
    - **FORCESAMPLEGEN :** NO/YES si YES, en mode CrossValidation la chaine génère aussi les échantillons pour produire la carte
    - **VALIDATION :** C’est le chemin de la couche qui contient les parcelles de validation (au format shapefile), nécessaire si le chain mode prévoit la validation.
    - **VALIDMODE :** Il faut préciser ici le type de validation souhaitée (pixels ou surface). Il est donc possible de faire tourner la chaine jusqu’à l’étape de validation et de produire les cartes que si le résultat est satisfaisant.
    - **NPROCESS :** Pour les étapes de traitement parallélisables, spécifier ici le nombre de processus simultanés souhaités (typiquement dépendant du nombre de cœurs de la machine).
  + **[SENTINEL-2 CONFIGURATION] :** il s’agit ici de gérer l’utilisation des images Sentinel-2 de niveau TOA
    - **REFLECTANCE:** nom du préfixe des fichiers de réflectance produits par la chaine
    - **VALIDITYMASK:** nom du préfixe des fichiers de découpés en fonction des masques de nuage produits par la chaine
    - **GAPMASK:** nom du préfixe des fichiers produits par la chaine lors de l’étape de gapfilling
    - **FEAT :** bandes et indices que la chaine doit utiliser. Si on ne souhaite pas utiliser ce type d’images S2, il faut supprimer toutes les bandes et indices situées après « FEAT : » ou alors ne conserver que ceux qu’on souhaite utiliser.
    - **GAPFILLINGMODE:** type de Gapfilling à mettre en œuvre
    - **MINVIEWS:** Nombre minimum d’observation valides par pixels sur la série temporelle pour valider le gapfilling. Les pixels avec nombre de vues inférieur à ce seuil seront mis en no-data. Les objets se trouvant entièrement sur une zone de no-data seront attribués par un *rough fix* à la R.
  + **[SENTINEL-2 THEIA CONFIGURATION] :** il s’agit ici de gérer l’utilisation des images Sentinel-2 de niveau TOC fournies par THEIA
    - **REFLECTANCE:** nom du préfixe des fichiers de réflectance produits par la chaine
    - **VALIDITYMASK:** nom du préfixe des fichiers de découpés en fonction des masques de nuage produits par la chaine
    - **GAPMASK:** nom du préfixe des fichiers produits par la chaine lors de l’étape de gapfilling
    - **FEAT :** bandes et indices que la chaine doit utiliser. Si on ne souhaite pas utiliser ce type d’images S2, il faut supprimer toutes les bandes et indices situées après « FEAT : » ou alors ne conserver que ceux qu’on souhaite utiliser.
    - **GAPFILLINGMODE:** type de Gapfilling à mettre en œuvre
    - **MINVIEWS:** Comme pour S2.
  + **[LANDSAT-8 CONFIGURATION] :** il s’agit ici de gérer l’utilisation des images Landsat-8
    - **REFLECTANCE:** nom du préfixe des fichiers de réflectance produits par la chaine
    - **VALIDITYMASK:** nom du préfixe des fichiers de découpés en fonction des masques de nuage produits par la chaine
    - **GAPMASK:** nom du préfixe des fichiers produits par la chaine lors de l’étape de gapfilling
    - **FEAT :** bandes et indices que la chaine doit utiliser. Si on ne souhaite pas utiliser ce type d’images S2, il faut supprimer toutes les bandes et indices situées après « FEAT : » ou alors ne conserver que ceux qu’on souhaite utiliser.
    - **GAPFILLINGMODE:** type de Gapfilling à mettre en œuvre
    - **MINVIEWS:** Comme pour S2.
  + **[VHR SCENE CONFIGURATION] :** 
    - **PANFOLDER :** Chemin vers le dossier contenant l’image panchromatique de la THRS (format VRT)
    - **MSFOLDER :** Chemin vers le dossier contenant l’image multispectrale de la THRS (format VRT)
    - **TOA :** pour que la chaine applique la conversion TOA de la THRS : YES / NO
    - **PANSHARPENING:** pour que la chaine calcule le pansharpening : YES / NO. Si le pansharpening a été créé en dehors de la chaîne, il faut appeler le fichier PANSHARP.tif et le placer sur un dossier appelé aussi PANSHARP au même niveau que les dossiers PAN et MS. Écrire YES dans « PANSHARPENING » même si celui-ci a déjà été créé en dehors de la chaine car dans le cas contraire la segmentation sera calculée sur l’image multispectrale.
    - **SCALE :** paramètre d’échelle de la segmentation
    - **SHAPE :** paramètre shape de la segmentation
    - **COMPACTNESS :** paramètre de compactness de la segementation
  + **[VHR-SITS COREGISTRATION] :** gestion du recalage de HRS sur la THRS
    - **COREGISTRER :** il faut écrire YES si l’utilisateur nécessite que la série temporelle d’image soit recalée par rapport à l’image THRS. Par défaut la chaîne propose de réaliser cette étape.
    - **FORCE :** mettre à YES si l’on souhaite écraser une coregistration déjà faite.
    - **VHRBAND :** bande de la THRS utilisée pour la coregistration
    - **S2BAND :** bande S2 utilisée pour la coregistration
    - **L8BAND :** bande L8 utilisée pour la coregistration
    - **MAXPOINTS :**Nombre maximum de points d’amer à retenir pour la co-registration (nécessaire si le nombre total est trop élevé, causant des *warnings* sur le nombre maximum d’itérations pendant l’exécution de l’étape).
  + **[TRAINING CONFIGURATION] :** 
    - **CLASSCODE :** c’est le champ de la base de données d’apprentissage du (des) niveau(x) de la nomenclature sur lequel (lesquels) on veut faire la classification. Si on souhaite en mettre plusieurs, il faut les séparer par une « , »
    - **PARAMETERS :** il s’agit du modèle de classification à utiliser, ainsi que ses paramètres.
    - **RFIMPORTANCE : YES/NO pour calcul l’importance des variables**
    - **RFIMPTREE :** paramètre du nombre d’arbres lié à RandomForest
    - **RFIMPNODSIZE :** à compléter
    - **RFIMPMAXFEAT :** à compléter
    - **RFIMPNRUNS :** à compléter
  + **[CLASSIFICATION OUTPUT CONFIGURATION] :** format désiré pour la classification
    - **RASTERIZE :** Sélection dumode derasterisation. NO – seul le format vecteur est produit ; VRT – la rasterisation se fait par tuile de segmentation et le résultat est un catalogue virtuel ; FULL – toute la scène est rastérisée en un seul fichier.
  + **[ADDITIONAL FEATURES] :** gestion des variables auxiliaires choisies par l’utilisateur (pentes, MNT, textures…)
    - **USERFEATLEAST :** liste des chemins + noms des fichiers séparés par des virgules.

## Mise en place des fichiers et des dossiers

Quelques manipulations doivent être faites avant de lancer la chaîne de traitement :

* + - Placer dans le BASEFOLDER les dossiers des images Landsat-8 et Sentinel-2 dans des sous-dossiers appelés respectivement L8 et S2. Ne pas oublier de déplacer les fichiers sources S2 et L8 si une mosaïque a été effectuée.
    - Même si le pansharpening n’est pas créé avec la chaîne de traitement, les deux images THRS doivent être mises dans le BASEFOLDER, dans des sous-dossiers appelés PAN et MS. Il faut également créer un raster virtuel pour chacune des images dans leurs dossiers respectifs.

## Lancement de la chaine

Cela se fait sur la fenêtre de commandes d’OSGeo4W, sur laquelle il faut se placer sur le dossier de la chaîne Moringa.

Nous avons lancé chaque étape de manière indépendante. Cela est indiqué au travers de l’instruction « --single-step » (sauf pour les étapes 5 et 6).

L’étape 0 est disponible pour prétraiter la THRS dans les cas simples où il n’y a pas de prétraitements géométriques à réaliser (scène entière orthorectifiée en format supporté par l’OTB) :

**Python launchChain.py --runlevel 0 --single-step reunion\_test.cfg > Etape0.log**

* + Étape 1: segmentation

Ligne de commande :

**Python launchChain.py --runlevel 1 --single-step reunion\_test.cfg > Etape1.log**

L’instruction " > Etape1.log" permet d’enregistrer dans un fichier texte ce qui est affiché sur la fenêtre de commande, mais ce n’est pas obligatoire.

Lors de cette étape un dossier appelé « SEGMENTATION » est créé dans le CHAINNAME. Ce dossier contient les tuiles de la segmentation ainsi que la base de données d’apprentissage intersecté avec la segmentation dont les noms de fichiers commencent par GT.

Si l’étape de segmentation a été faite mais que vous souhaitez modifier la base de données d’apprentissage, il faut effacer les fichiers GT est relancer l’étape 1. Cela évite de refaire toute la segmentation, seule l’intersection entre la segmentation et la BD terrain sera effectuée. Cela évite de refaire toute la segmentation.

* + Étape 2: Recalage de la série temporelle

Ligne de commande : **Python launchChain.py --runlevel 2 --single-step reunion\_test.cfg > Etape2.log**

Les images recalées sont placées dans les dossiers L8 et S2. Elles correspondent aux fichiers qui dont les noms se terminent par COREG.

* + Étape 3: Interpolation des zones sous nuages

Ligne de commande : **Python launchChain.py --runlevel 3 --single-step reunion\_test.cfg > Etape3.log**

Dans cette étape un dossier FEAT est créé dans le BASEFOLDER. Dans ce dossier sont placées les piles d’images de la série temporelle (un fichier pour chaque bande et indices) avant interpolation des zones sous nuages mais aussi après interpolation (Gapfilling). Dans ce cas, le nom des images se terminent par GAPF correspondent au produit final : la concaténation des images de toutes les dates pour chacun des indices et des bandes, avec un remplissage des nodata provoqués par la présence de nuages (Gapfilling).

* + Étape 4: Statistiques zonales

Ligne de commande : **Python launchChain.py --runlevel 4 --single-step reunion\_test.cfg > Etape4.log**

Dans cette étape 3 nouveaux dossiers sont créés dans le dossier CHAINNAME pour ajouter les champs des variables utilisées pour la classification :

* + - GT\_SAMPLES : parcelles de la base de données d’apprentissage
    - SAMPLES : objets issus de la segmentation
    - VAL\_SAMPLES : parcelles de la base de données de validation
  + Étape 5 et 6: Modélisation de la classification

Ces deux étapes ont été lancées simultanément.

Ligne de commande : **Python launchChain.py --runlevel 5 reunion\_test.cfg > Etape5et6.log**

Dans ces étapes, deux nouveaux dossiers sont créés dans le dossier CHAINNAME. Les noms de ces dossiers se terminent par le SETUPNAME figurant dans le fichier de config:

* + - MODEL\_*SETUPNAME* : les fichiers produits par la modélisation
    - CLASSIFICATION\_*SETUPNAME* : contient les résultats de validation et les cartes aux formats raster et vecteur

La chaine enregistre la classification aux formats image et vecteur dans des dossiers séparé pour chaque niveau de la nomenclature.

Pour le format raster un fichier de mosaïque virtuelle est créé. Pour le format vecteur il s’agit de chaque tuile issue de la segmentation.

1. Post-traitements

## Export de la classification format image

À la fin du processus de classification, la chaine aura créé un raster virtuel (VRT) rassemblant toutes les tuiles de la classification. Si ce n’est pas le cas, il faudra le faire manuellement avec QGIS.

Il est ensuite possible de valider (Cf. Chapitre validation) ou d’exporter la classification au format TIF en utilisant QGIS. Attention lors de l’export au format TIF les limites des tuiles de la segmentation sont parfois visibles sous la forme de pixels codés en 0. Il est donc préférable de privilégier l’outil de mosaïque (Cf. chapitre b ci-dessous).

## Mosaïque des tuiles raster de la classification

La mosaïque des tuiles raster est identique à celle des images présentée dans le chapitre Création des mosaïques en page 13 sauf qu’il ne sera pas nécessaire d’utiliser d’égalisation radiometrique.

La commande à saisir sera :

**otbcli\_Mosaic -il** <tuile1.tif> <tuile2.tif> **-out** <Classif\_mosaic.tif**> uint8**

Le format de sortie en 8 bits est préférable pour ce type de donnée (sauf s’il y a plus de 255 classes dans la nomenclature !).

## Coloration des classes

Pour colorer les classes de la classification, il faut utiliser la fonction ColorMapping d’OTB. Attention cette manipulation augmente de façon importante la taille du fichier en sortie.

Pour cela, il faut préalablement créer un fichier texte formaté comme indiqué ci-dessous :

***# Habitat résidentiel mixte***

***1 134 102 114***

***# Habitat précaire***

***2 179 57 92***

Les lignes qui commencent par # sont du commentaire. Sur la seconde ligne, « 1 » correspond au numéro de la classe dans la nomenclature, « 134 » à la valeur dans le rouge, « 102 » à la valeur dans le vert et « 114 » à la valeur dans le bleu.

La commande à saisir sera :

**otbcli\_** **ColorMapping -in** <classification au format tif en entrée> **-out** < classification colorée au format tif en sortie> **uint8 -method custom -method.custom.lut** < fichier texte avec les codes de couleurs >

## Lissage par filtre majoritaire

Si nécessaire, il est possible sur la version raster de la classification d’appliquer un filtre majoritaire afin de lisser les contours et de supprimer des pixels isolés. Le choix de la taille de l’élément structurant (ip.radius) est importante ne pas trop dégrader la classification (un filtre de 1 correspond à 3x3 pixels)

La commande à saisir sera :

**otbcli\_ClassificationMapRegularization -io.in** <Classif.tif> **-io.out** <Classif\_liss.tif> **-ip.radius 3**

1. Validation

La chaine produit une validation de la classification. Il est toutefois possible de relancer la validation en utilisant la commande décrite ci-dessous.

Outil : ComputeConfusionMatrix d’OTB sur la fenêtre de commande d’OSGeo4W

otbcli\_ComputeConfusionMatrix.bat -in <vrt de la classification par exemple …/classif\_cod\_1.vrt> -out <fichier .csv en sortie qui contiendra la matrice> -ref vector –ref.vector.in <jeu de validation> -ref.vector.field <Champ, par exemple Cod\_1> > Fichier.log

L’instruction " > Fichier.log" dans cette étape est très importante parce qu’il crée le fichier texte avec le résultat des précisions des matrices. Le fichier .csv contient seulement les valeurs de la matrice de confusion.

1. Annexes

## Etat des dossiers au cours du traitement

*Exemple Burkina Sud*

|  |  |  |
| --- | --- | --- |
|  | **S2** | **L8** |
| **Téléchargement des images (et découpage de la zone d’étude d’après le shp + suppression des images d’origine)**  BASEFOLDER/S2 et L8 |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **S2** | | | **L8** |
| **Prétraitement (masque nuages, layer stack)**  BASEFOLDER/S2 et L8 |  | | |  |
| **Etape 0 : TOA + pansharp de la THRS**  (se trouve dans le dossier contenant l’image THRS) | P | |  | |
| MS | |  | |
| PANSHARP | |  | |
| **Etape 1 : Segmentation**  (se trouve dans le dossier contenant l’image THRS) | SEGMENTATION | |  | |
|  | **S2** | | | **L8** |
| **Etape 2 : Recalage série temporelle**  BASEFOLDER/S2 et L8 |  | | |  |
| **Etape 3 : Interpolation**  OUT/FEAT | **S2** |  | | |
| **L8** |  | | |
|  | OUT/GT\_SAMPLES | | | OUT/SAMPLES |
| **Etape 4 : Statistiques zonales** |  | | |  |
| **Etape 5 : Model**  OUT/MODEL |  | | | |
| **Etape 6 : Classification**  OUT/CLASSIFICATION |  | | | |
| **Validation**  OUT/CLASSIFICATION |  | | | |



Département Environnement et Société,

Unité Mixte de Recherche Territoire, Environnement Télédétection et Information Spatiale

Maison de la Télédétection, 500 rue Jean François Breton, TA C91 MTD, 34093 Montpellier Cedex 5

Tél. : +33 (0)4 67 54 87 54

www.cirad.fr

1. Natural Color Satellite Image Mosaicking Using Quadratic Programming in Decorrelated Color Space Cresson & Saint-Geours, July 2015, IEEE JSTARS Volume 8 Issue 8 [↑](#footnote-ref-1)