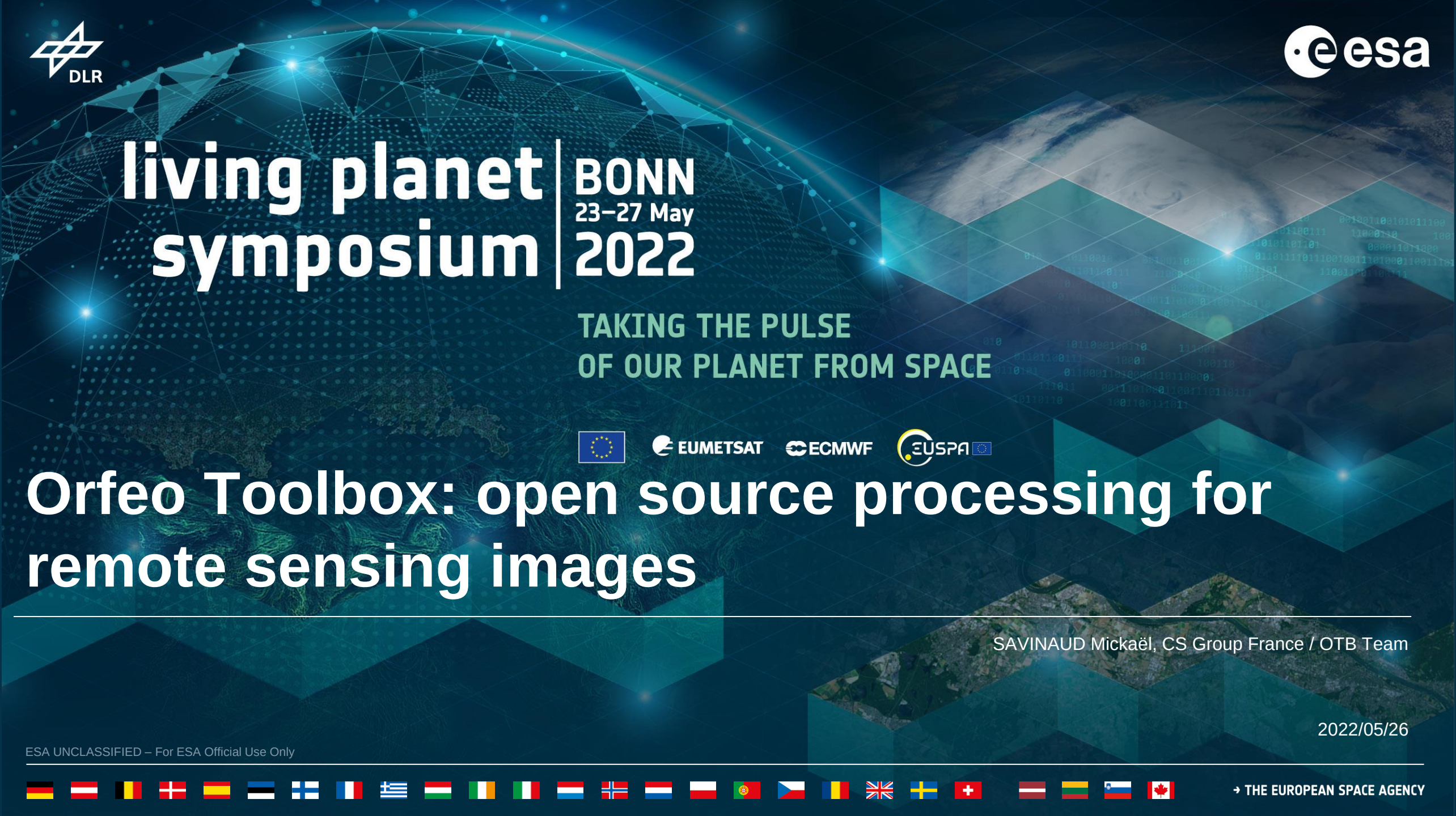


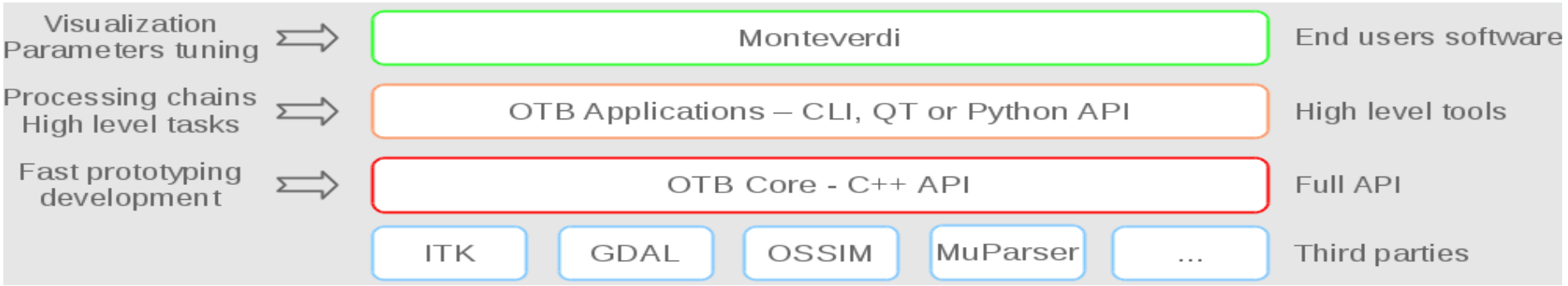
Orfeo Toolbox: open source processing for remote sensing images

SAVINAUD Mickaël, CS Group France / OTB Team

2022/05/26

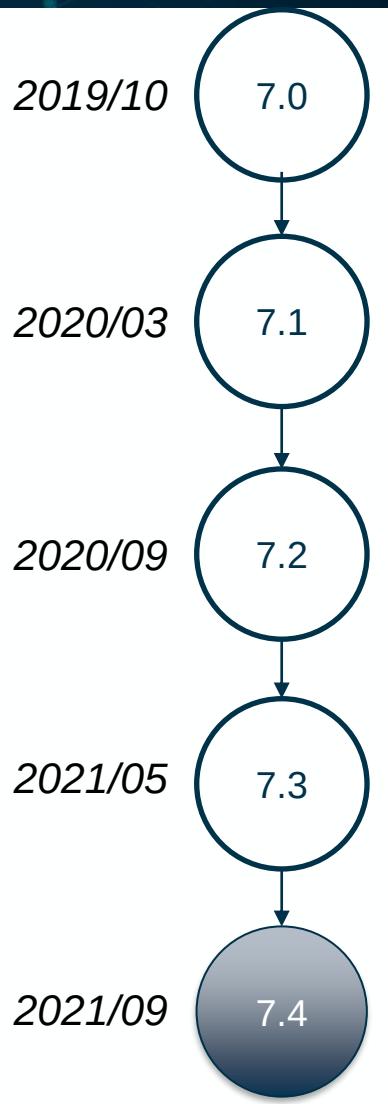
Short reminder

- Image processing library covers all needs in Remote Sensing: https://www.orfeo-toolbox.org/
- Funded and developed mainly by CNES: ORFEO program, SWOT Aval, Theia, …
- A 15-year-old story
- OSGEO Open-Source : Apache V2.0
- Maximum reach : for all kind of users, SIG, scientists… laptop to clusters computers
- Big data capable
- Streaming / pipeline



OTB since LPS 2019 - OTB 7.x

- OTB 7.x :
  - New set of applications:
    - Regression framework
    - Hyperspectral image processing
    - Image processing: FastNLMeans, …
    - Utility: Zonal statistics
  - Improve support for SAR image:
    - New SAR sensor models :
      - CosmoSkyMed, TerraSAR-X
    - Improvement of the Sentinel 1 model for S1Tiling and DiapOTB

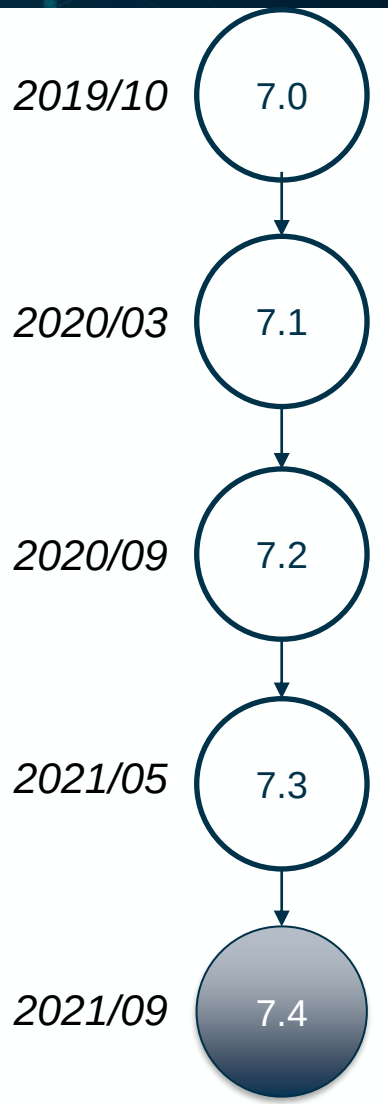


OTB since LPS 2019 – OTB 7.x

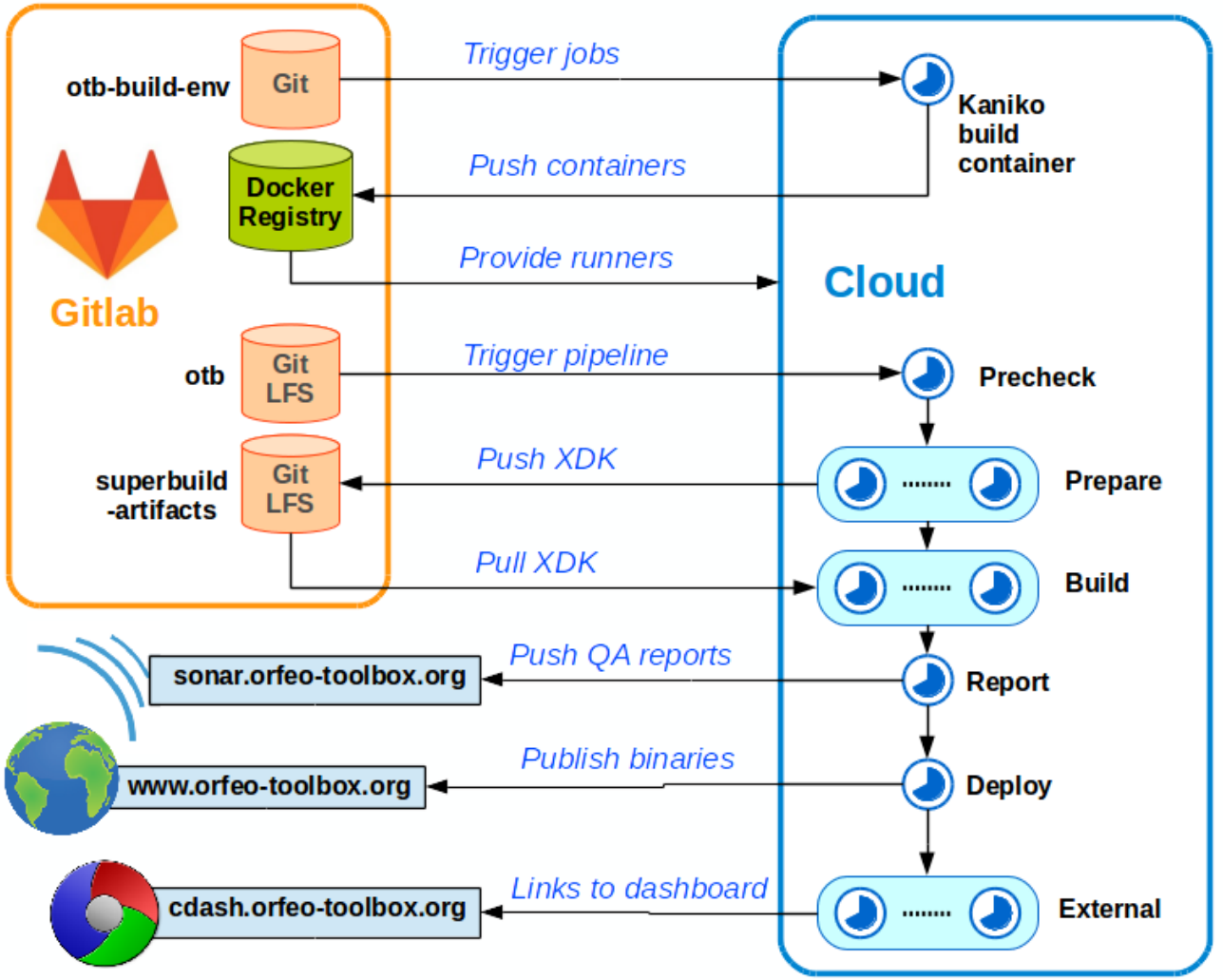
- Support for GDAL 3
- Switch from Python 2 to Python 3
- Logs for the Python wrapper
- NoData extended filename for output images:

  &nodata=(double) value

- The Java wrapper has been removed
- New CI platform

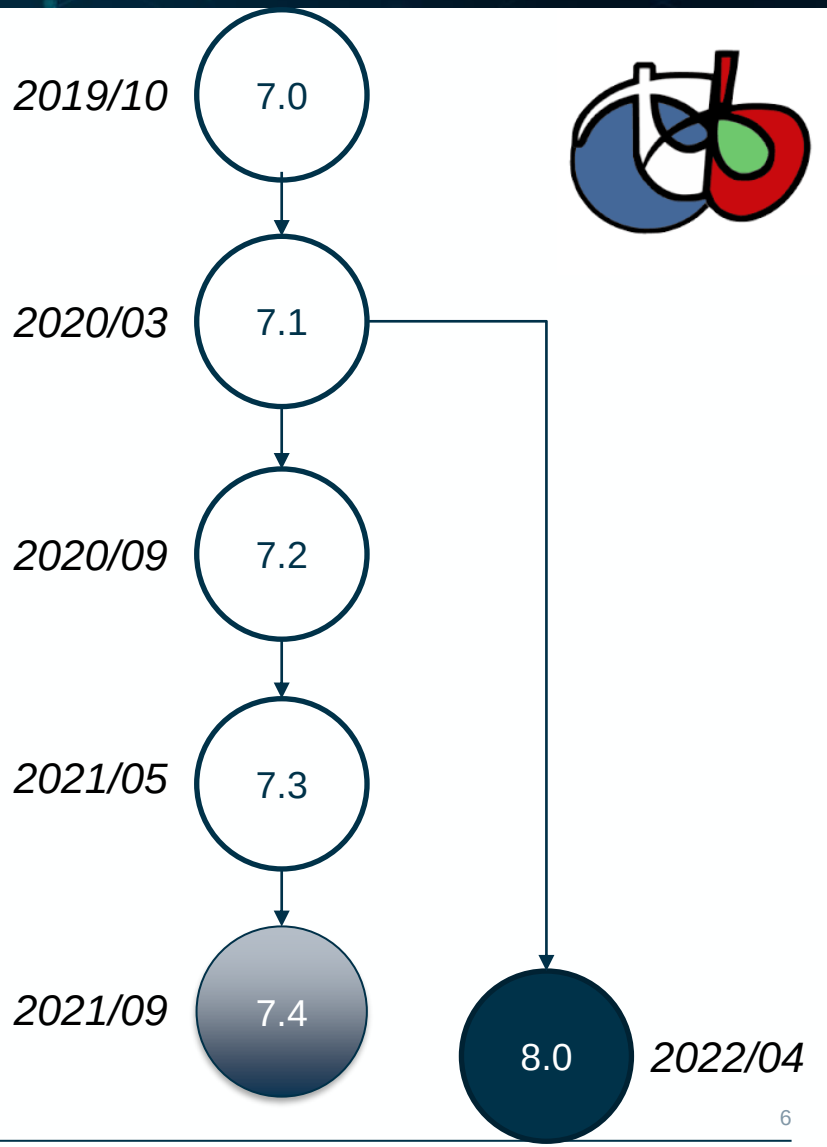


New OTB Continuous Integration platform

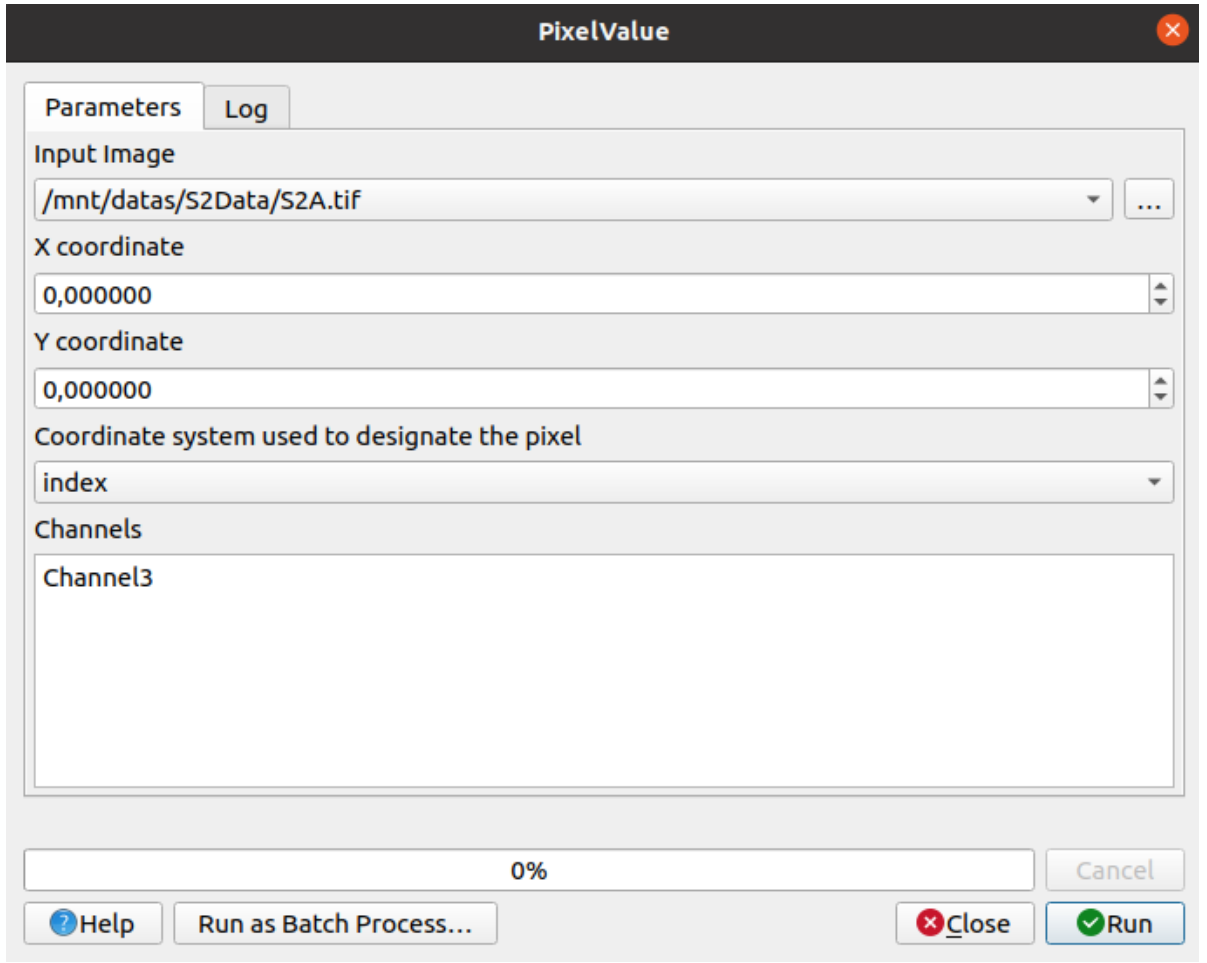


OTB since LPS 2019 - OTB 8

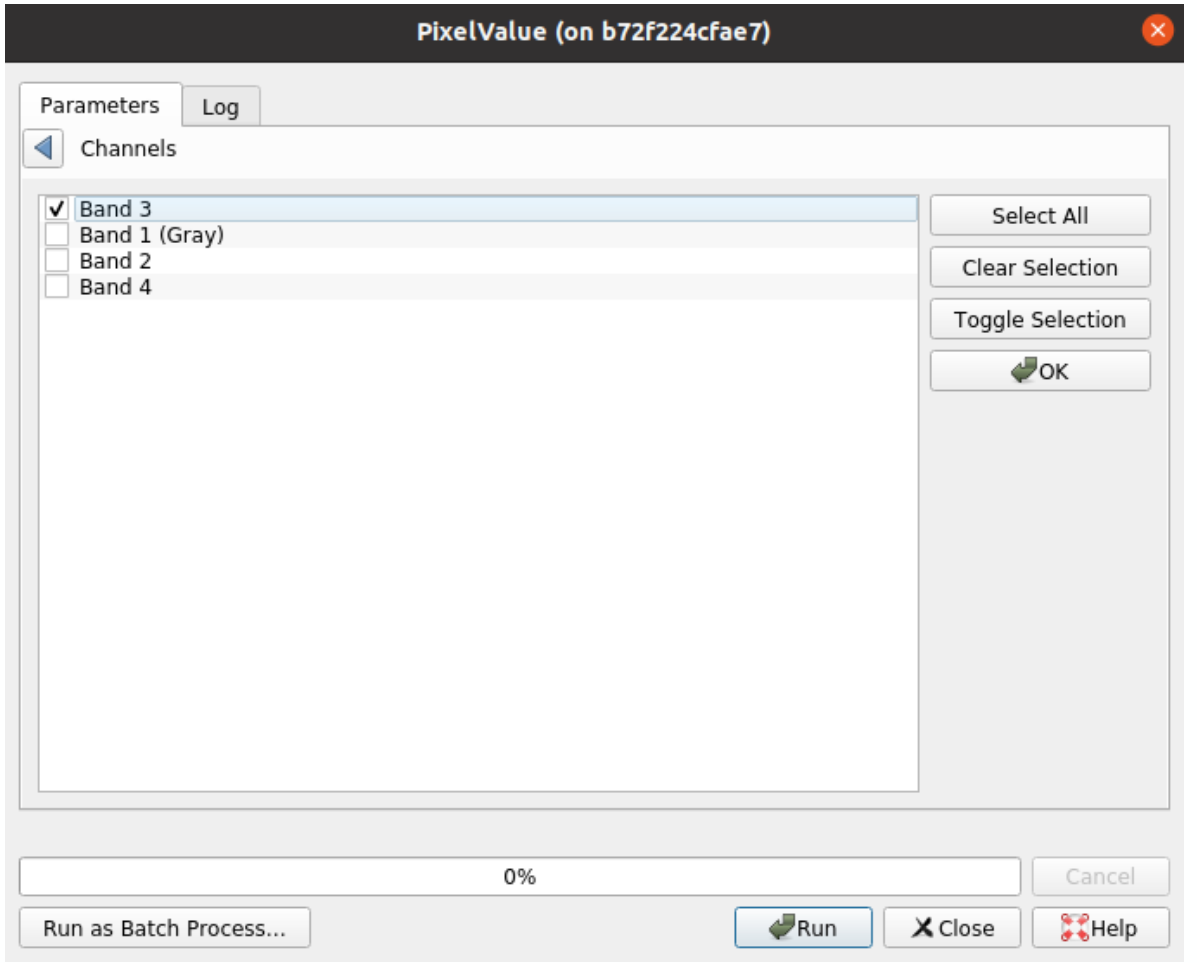
- Remove OSSIM
  - Why?
    - Hard to package (Debian, Conda, Superbuild) -> Python
    - Hard to follow Ossim development cycle
    - Many Ossim functionalities are also implemented in GDAL
  - Large impact:
    - Metadata parsing and management refactoring
    - DEM handler refactoring -> more flexibility
    - RPC handling
    - Reuse RPC parsing from GDAL and new RPC class
    - SAR model reimplementation
    - Modern time points and duration
- Improve OTB integration into QGIS



OTB 8 and QGIS – New interface

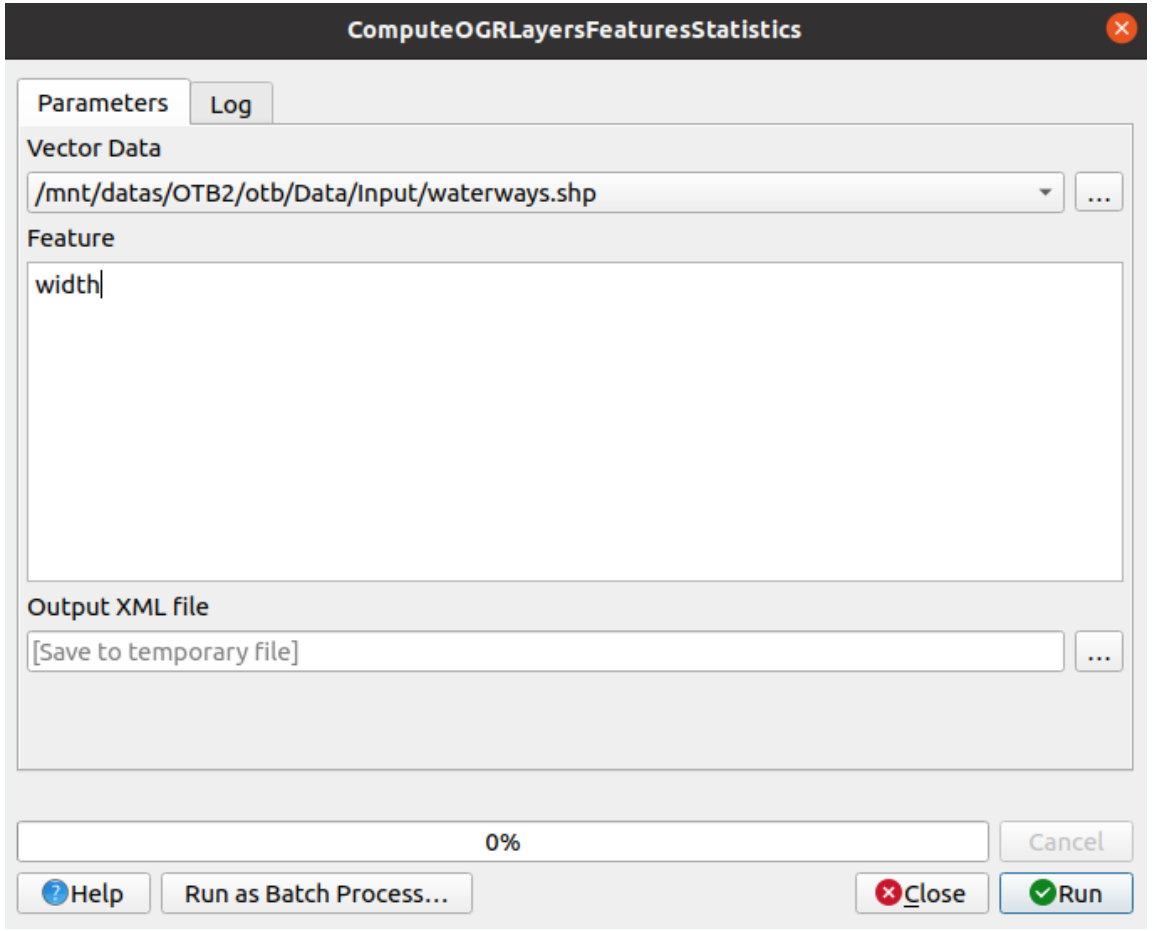


OTB 7.4

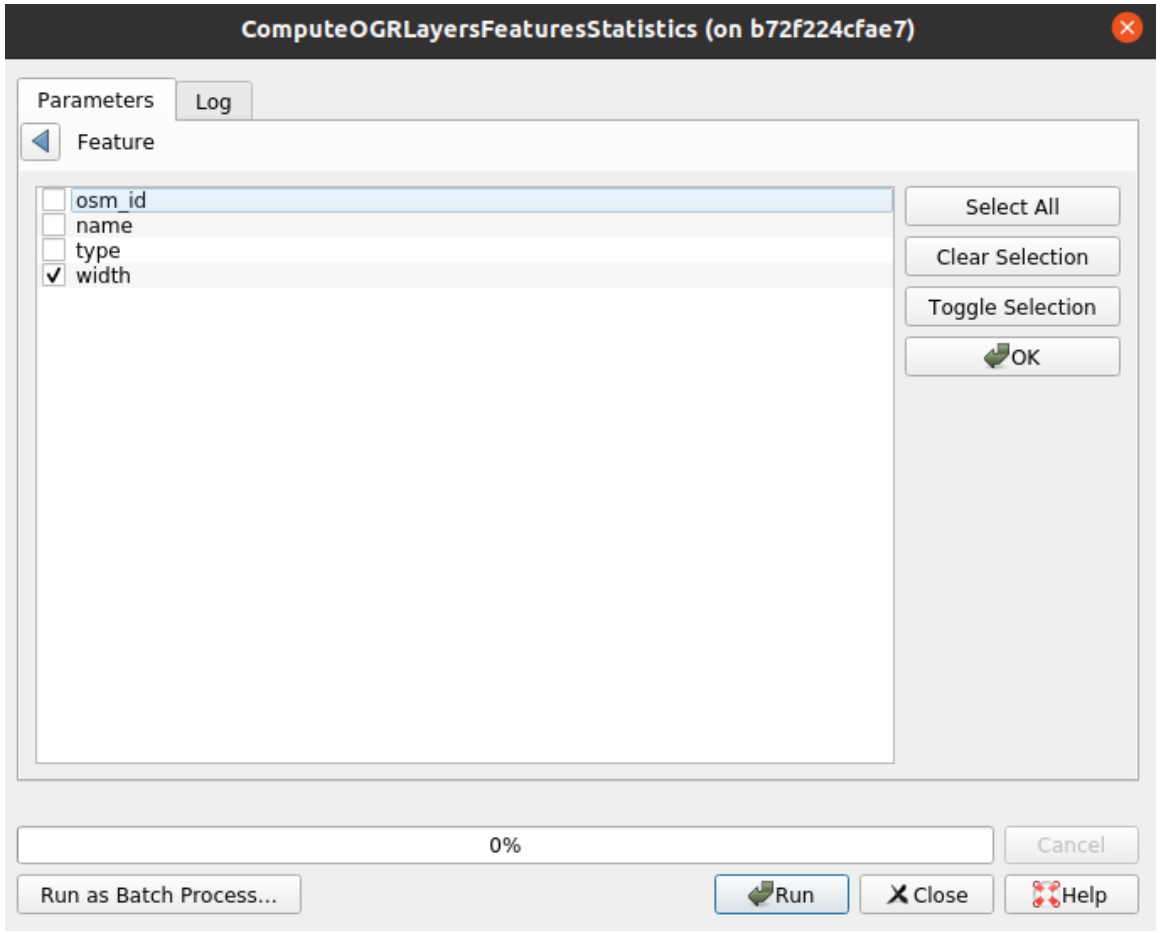


OTB 8.0

OTB 8 and QGIS – New interface

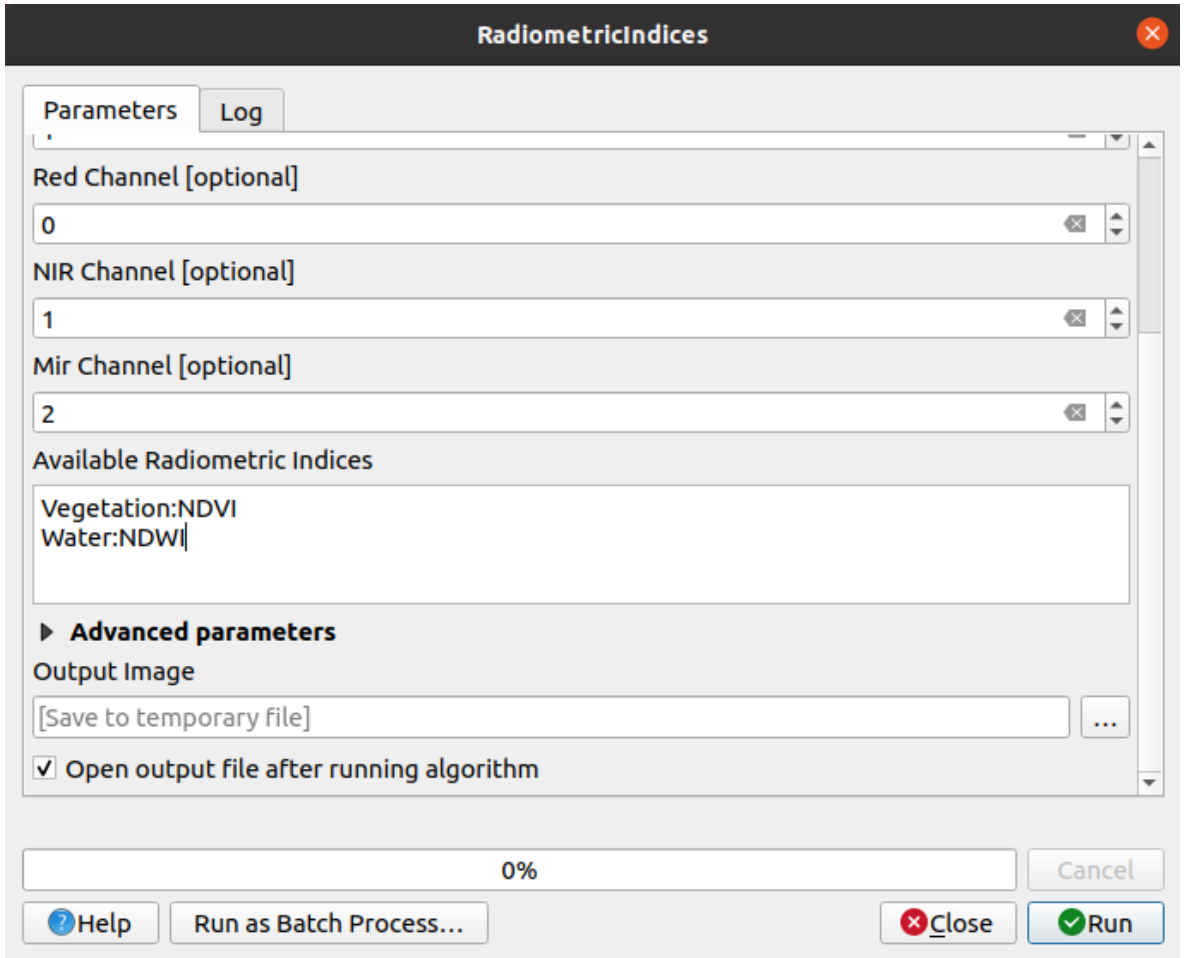


OTB 7.4

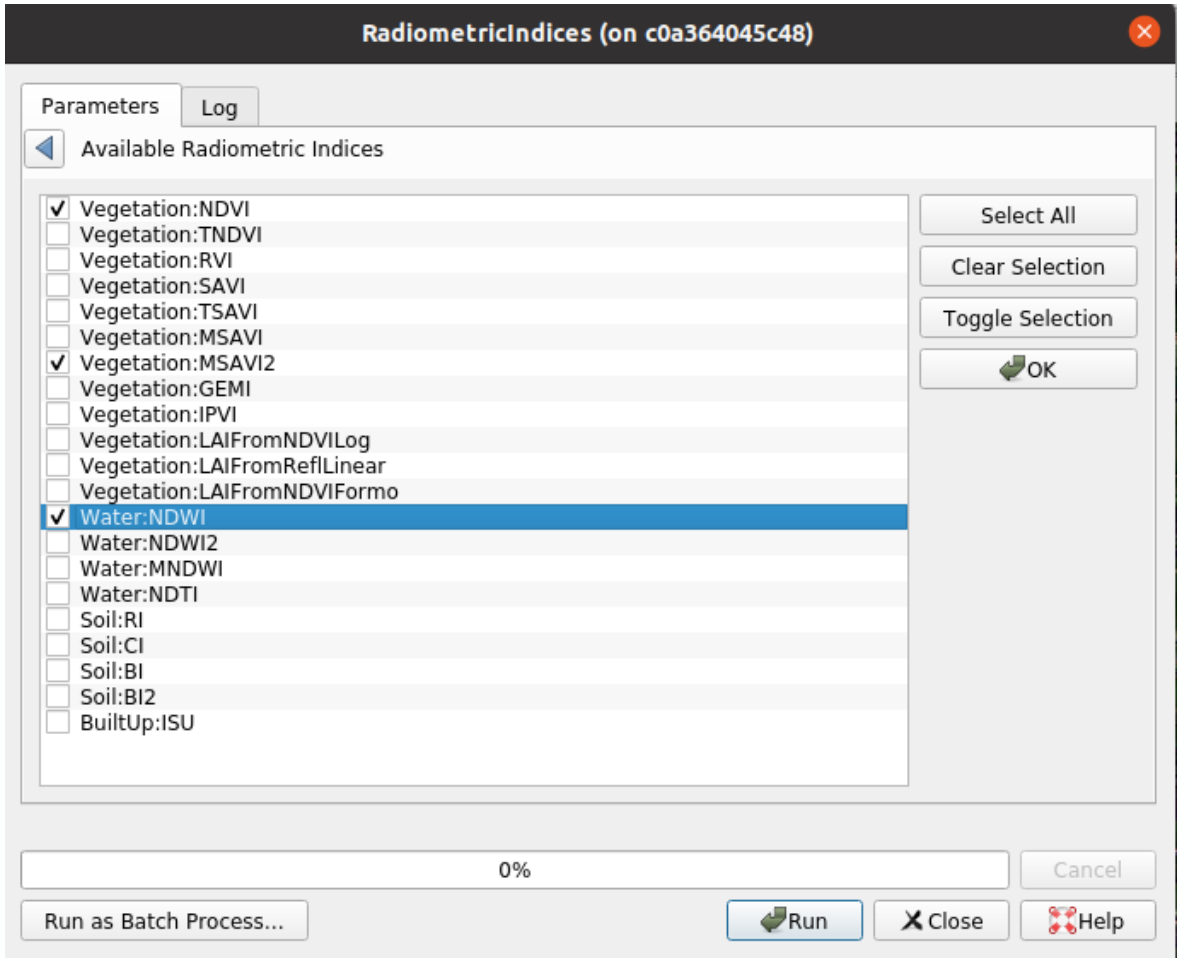


OTB 8.0

OTB 8 and QGIS – New interface



OTB 7.4



OTB 8.0

OTB User day in 2021

- OTB User day in November 2021
  - Discover new usage of OTB
  - Share experience of the community
  - Think to future
  - Video and presentations available

PyOTB - A PYTHONIC EXTENSION OF OTB

https://gitlab.orfeo-toolbox.org/nicolasnn/pyotb/

Using OTB Python API:

```
import otbApplication

resampled = otbApplication.Registry.CreateApplication('RigidTransformResample')
resampled.SetParameterString('in', 'my_image.tif')
resampled.SetParameterString('interpolator', 'nn')
resampled.SetParameterString('transform.type', 'id')
resampled.SetParameterFloat('transform.type.id.scalex', 0.5)
resampled.SetParameterFloat('transform.type.id.scaley', 0.5)
resampled.SetParameterString('out', 'output.tif')
resampled.ExecuteAndWriteOutput()
```

Using pyotb:

```
import pyotb

pyotb.RigidTransformResample({'in': 'my_image.tif', 'transform.type': 'id', 'transform.type.id.scaley': 0.5,
                              'transform.type.id.scalex': 0.5, 'interpolator': 'nn', 'out': 'output.tif'})
```

PyOTB - A PYTHONIC EXTENSION OF OTB

https://gitlab.orfeo-toolbox.org/nicolasnn/pyotb/

Prerequisite: having pyotb objects

```python
import pyotb

# transforming filepaths to pyotb objects
input1, input2, input3 = pyotb.Input('image1.tif'), pyotb.Input('image2.tif'), pyotb.Input('image3.tif')
```

Shape attribute

```python
print(input1.shape)  # (width, height, channels)
```

Slicing

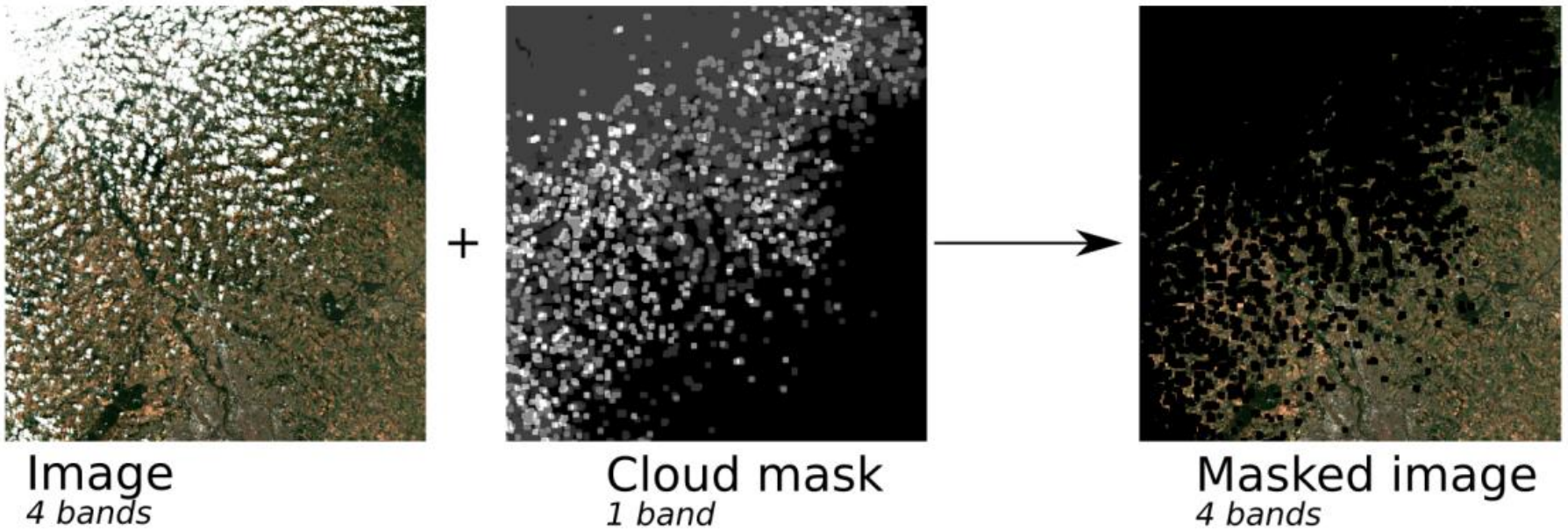
```python
input1[:, :, :3]  # selecting first 3 bands
input1[:, :, [0, 1, 4]]  # selecting bands 1, 2 and 5
input1[:1000, :1000]  # selecting 1000x1000 subset
```

```python
import pyotb
import numpy as np

input = pyotb.Input('image.tif')  # this is a pyotb object

# Creating a numpy array of noise
white_noise = np.random.normal(0, 50, size=input.shape)  # this is a numpy object

# Adding the noise to the image
noisy_image = np.add(input, white_noise)  # magic: this is a pyotb object that has the same georeference as input
noisy_image.write('noisy_image.tif')
```

PyOTB - A PYTHONIC EXTENSION OF OTB

https://gitlab.orfeo-toolbox.org/nicolasnn/pyotb/

Example: masking clouds on a satellite image

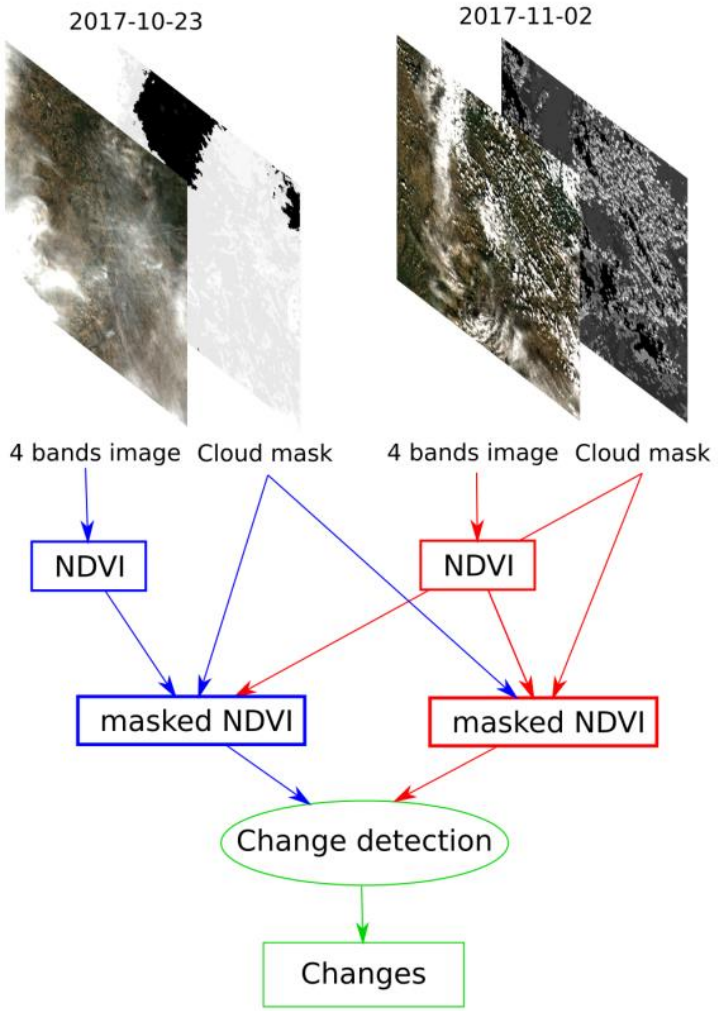



```
import pyotb

masked_image = pyotb.where(pyotb.Input('mask.tif') > 0, 0, 'image.tif')
masked_image.write('masked_image.tif', pixel_type='int16')
```

PyOTB - A PYTHONIC EXTENSION OF OTB

https://gitlab.orfeo-toolbox.org/nicolasnn/pyotb/

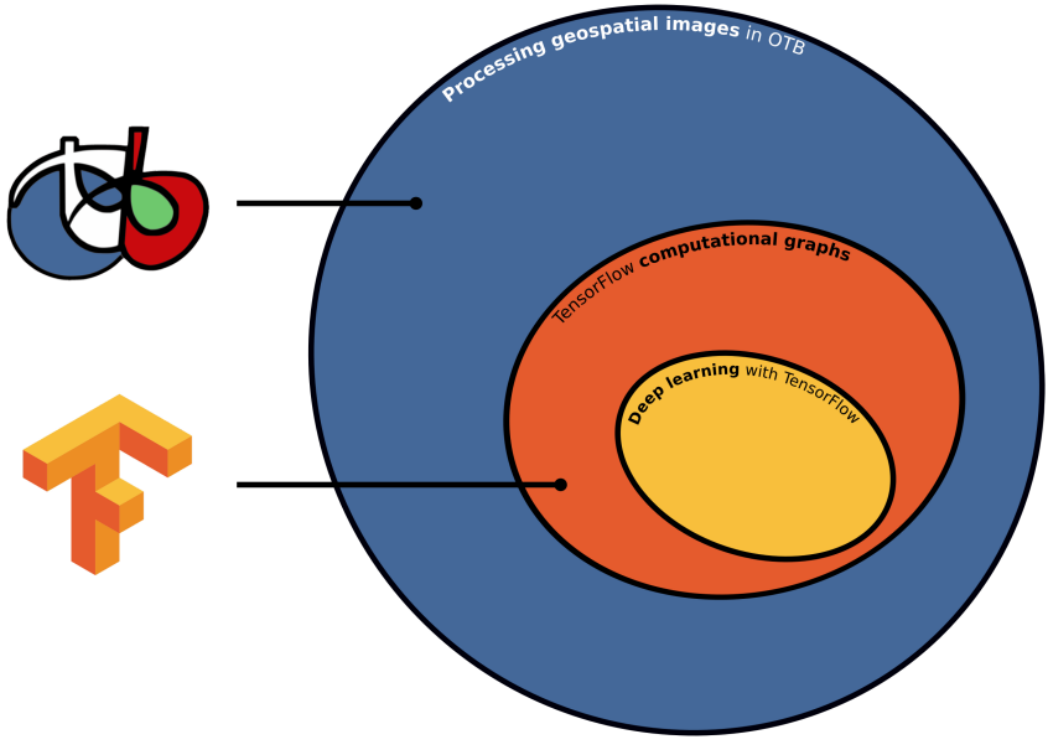
```python
import pyotb

image1 = pyotb.Input('SENTINEL2A_20171023-105154-753_L2A_T31TCJ_C_V2-2_FRE_10m.tif')
mask1 = pyotb.Input('SENTINEL2A_20171023-105154-753_L2A_T31TCJ_C_V2-2_CLM_R1.tif')
image2 = pyotb.Input('SENTINEL2A_20171102-105210-461_L2A_T31TCJ_D_V1-4_FRE_10m.tif')
mask2 = pyotb.Input('SENTINEL2A_20171102-105210-461_L2A_T31TCJ_D_V1-4_CLM_R1.tif')

ndvi1 = (image1[:, :, -1] - image1[:, :, 2]) / (image1[:, :, -1] + image1[:, :, 2])
ndvi2 = (image2[:, :, -1] - image2[:, :, 2]) / (image2[:, :, -1] + image2[:, :, 2])

merged_mask = (mask1 > 0) | (mask2 > 0)

masked_ndvi1 = pyotb.where(merged_mask == 1, -999, ndvi1)
masked_ndvi2 = pyotb.where(merged_mask == 1, -999, ndvi2)

pyotb.MultivariateAlterationDetector(in1=masked_ndvi1, in2=masked_ndvi2, out='changes.tif')
```



OTBTF – integration with TensorFlow

- Developed at INRAE to put in production research results
- Uses the C++ TensorFlow API
- Enables to apply TF models on raster in OTB filters and applications
- Preserves the streaming in OTB pipelines

https://github.com/remicres/otbtf

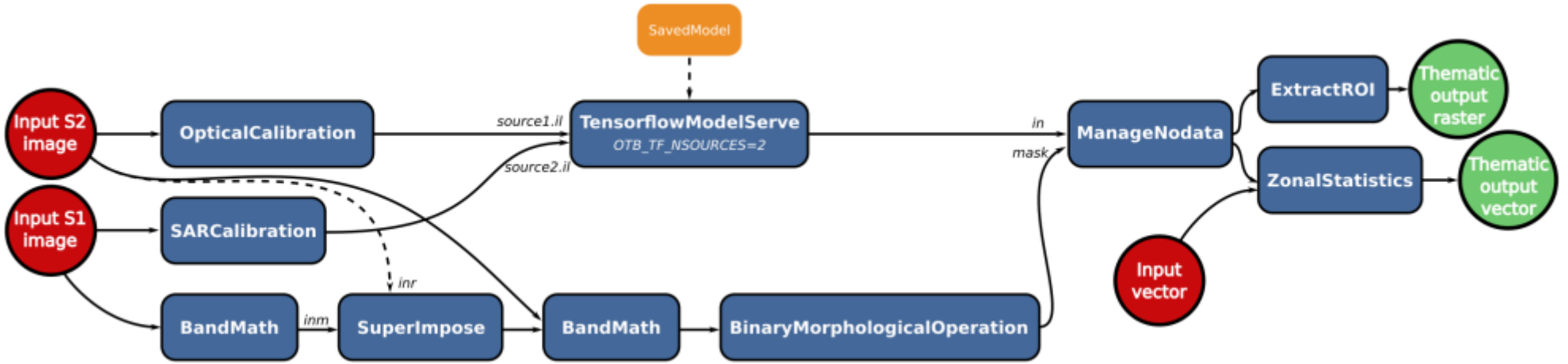


- Docker images at https://hub.docker.com/u/mdl4eo

OTBTF – integration with TensorFlow

New OTB Application:

- TensorflowModelServe: Inference
- TensorflowModelTrain: Training/validation (great for educational purpose)
- PatchesExtraction: extract patchs in one or multiple rasters
- TrainClassifierFromDeepFeatures: connects TensorflowModelServe to TrainImageClassifier
- ImageClassifierFromDeepFeatures: connects TensorflowModelServe to ImageClassifier
- PatchesSelection: for patches selection from rasters (experimental)
- LabelImageSampleSelection: select patches from a label image (experimental)
- DensePolygonClassStatistics: same as PolygonClassStatistics but faster (experimental)



OTBTF – integration with TensorFlow

- Perform SuperResolution with OTBTF
- https://github.com/remicres/sr4rs


```
# Download the pre-trained SR4RS model
wget https://tinyurl.com/sr4rsmodelv2
unzip sr4rsmodelv2
# Run SR4RS over a Sentinel-2 image (Bands 4, 3, 2, 8)
python sr4rs/code/sr.py \
--savedmodel sr4rs_sentinel2_bands4328_france2020_savedmodel \
--input S2_image_channels_4328_10m.tif \
--output S2_SR.tif
```

```
if __name__ == "__main__":
# Some secondary stuff to retrieve parameters values, etc
...
# The important stuff
infer =
otbApplication.Registry.CreateApplication("TensorflowModelServe")
infer.SetParameterStringList("source1.il", [params.input])
infer.SetParameterInt("source1.rfieldx", rfield)
infer.SetParameterInt("source1.rfieldy", rfield)
infer.SetParameterString("source1.placeholder",
constants.lr_input_name)
infer.SetParameterString("model.dir", params.savedmodel)
infer.SetParameterString("model.fullyconv", "on")
infer.SetParameterStringList("output.names", [ph])
infer.SetParameterInt("output.efieldx", efield)
infer.SetParameterInt("output.efieldy", efield)
infer.SetParameterFloat("output.spcscale", ratio)
infer.SetParameterInt("optim.tilesizex", efield)
infer.SetParameterInt("optim.tilesizey", efield)
infer.SetParameterInt("optim.disabletiling", 1)
infer.SetParameterString("out", out_fn)
infer.SetParameterOutputImagePixelType("out", encoding)
infer.ExecuteAndWriteOutput()
```

OTBTF – integration with TensorFlow

OTB in operational chains

BioPhy

Biophysical indices generation

MAJA

Atmospheric correction and cloud screening software

S1Tiling

Ortho-rectification of Sentinel-1 data on MGRS grid

LIS

Snow cover extent detection algorithm

WASP

Create monthly syntheses of cloud-free reflectances

DiapOTB

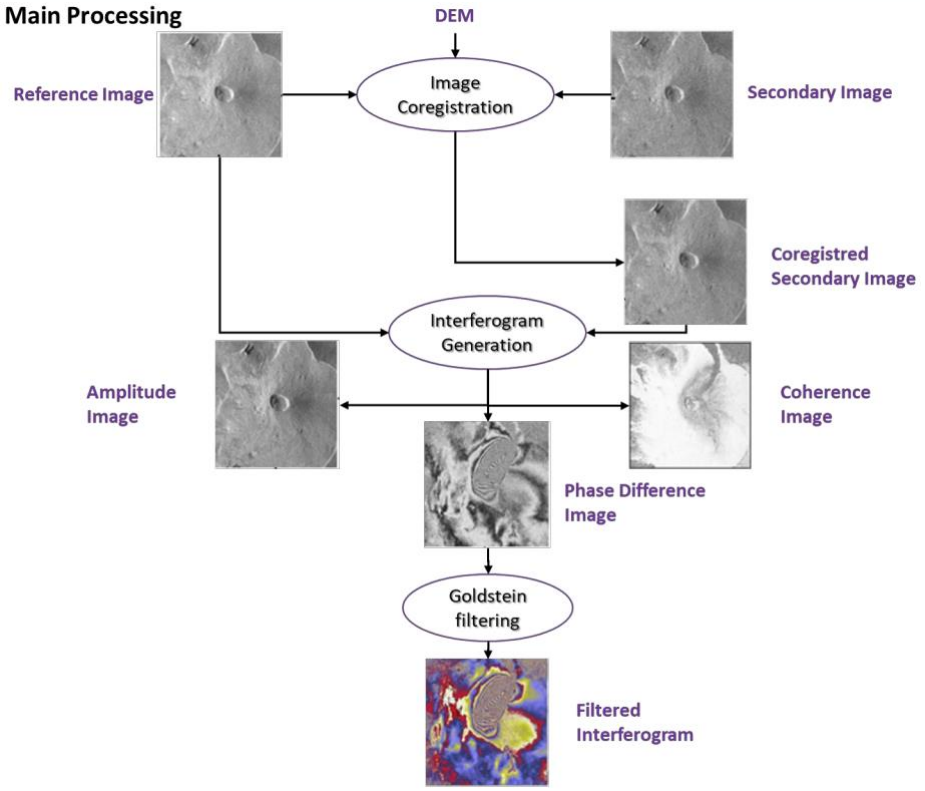
SAR Interferometric processing chain

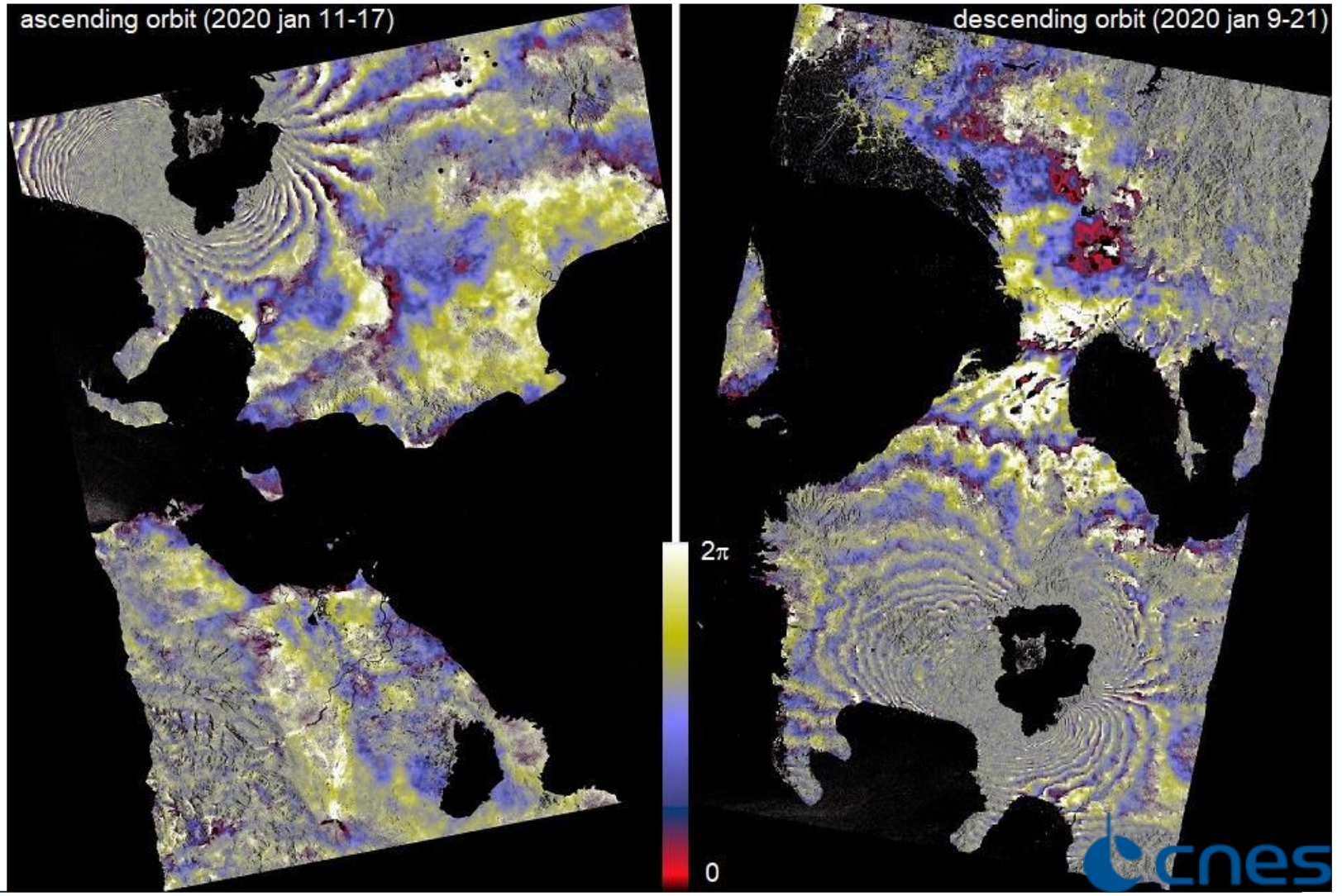
IOTA2

Large scale land cover mapping

DiapOTB – Results with Sentinel-1

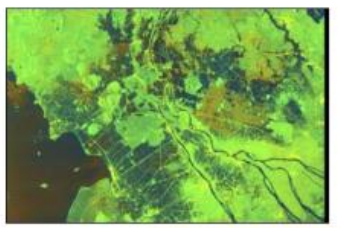
P. Durand, N. Pourthie, G. Usseglio and C. Tison, "DiapOTB: a new open source tool for Differential SAR interferometry," EUSAR 2021; 13th European Conference on Synthetic Aperture Radar, 2021, pp. 1-4.

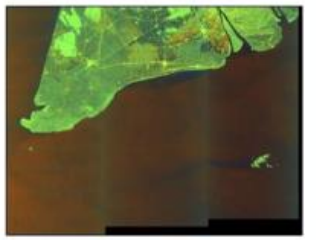


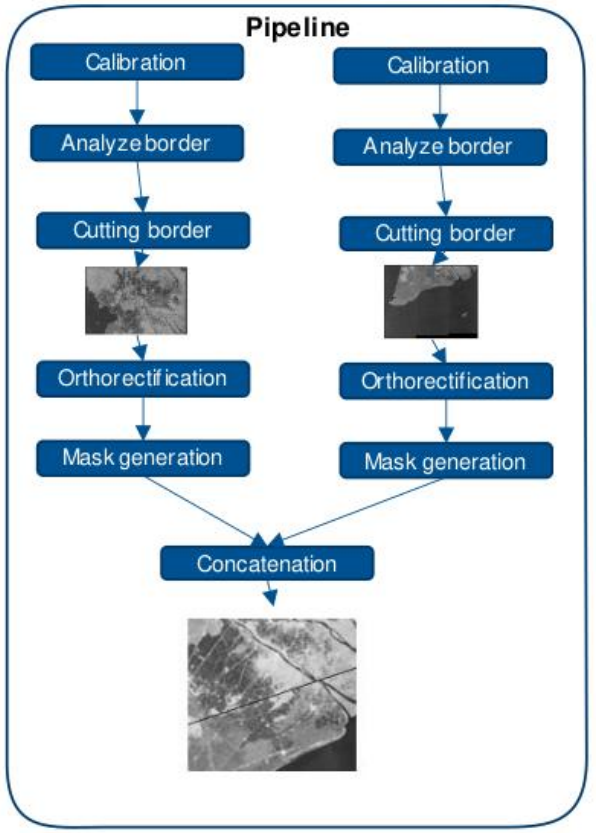


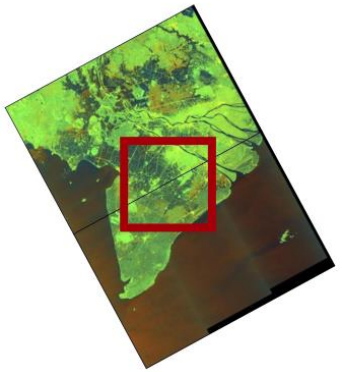
S1Tiling

- EODAG for data provider management
- OTB for building in memory pipelines
- DASK for running pipelines in parallel
- Sources: https:// gitlab.orfeo toolbox.org/s1 tiling/s1tiling
- Available on PiPy / Conda
- Documentation: https://s1tiling.pages.orfeotoolbox.org/s1tiling/latest
- Used in TropiSCO and WorldCereal projects at large scale



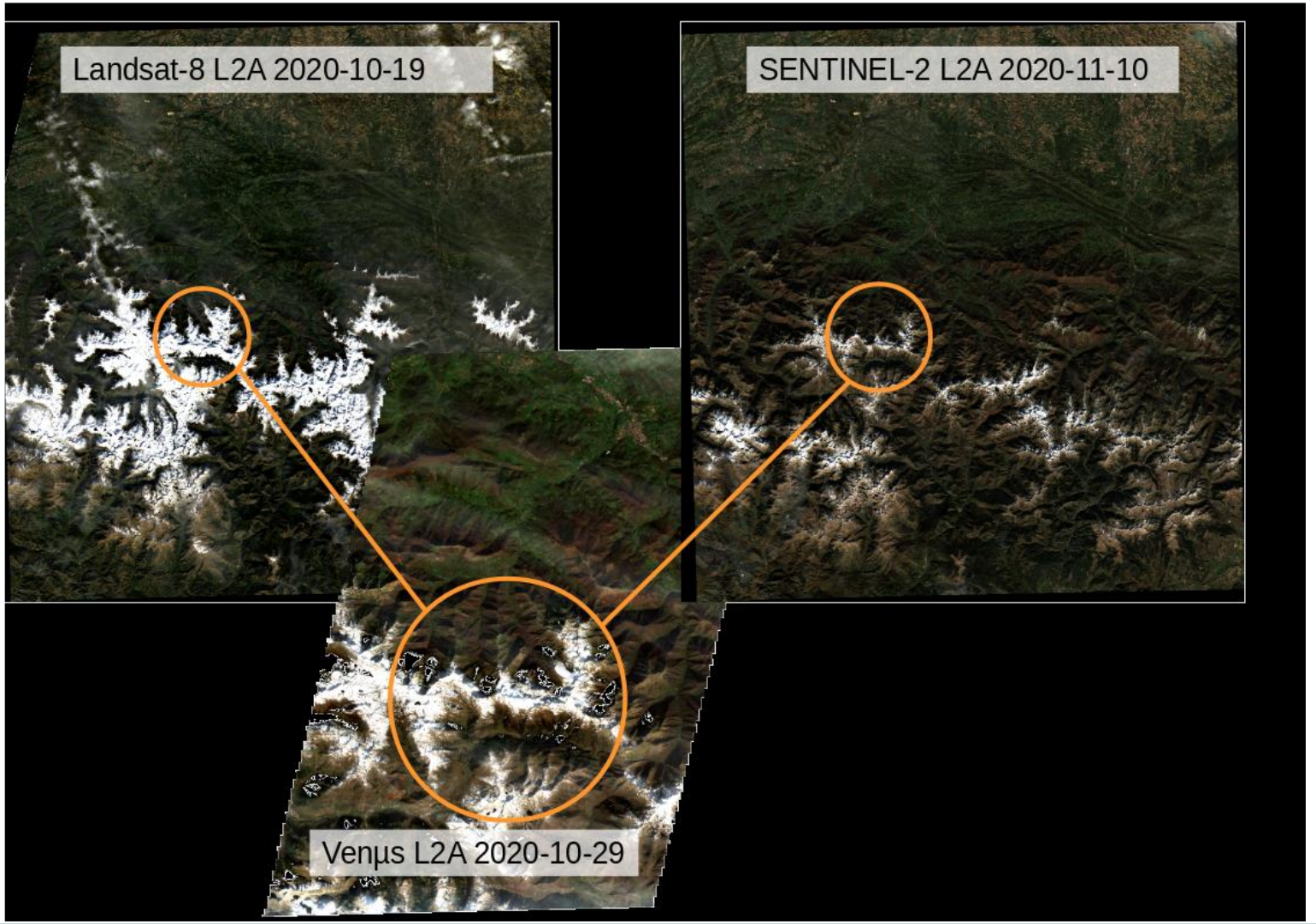






MAJA Processor

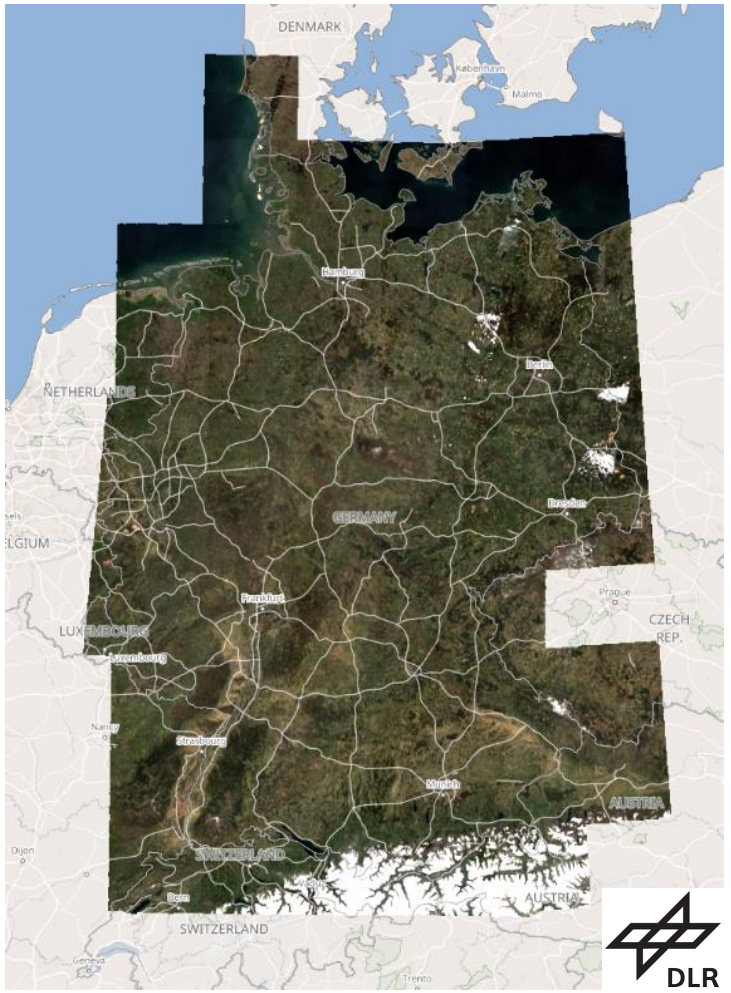
- FOSS L2A Processor:
- On-going research to improve and validate results
- MAJA to be in charge of L2A for the 6 VNIR/SWIR bands of Trishna
- Large production: 10 millions square kilometers via Theia initiative



Hagolle et al. (2015)
A Multi-Temporal and Multi-Spectral Method to Estimate Aerosol Optical Thickness over Land, for the Atmospheric Correction of FormoSat-2, LandSat, VENUS and Sentinel-2 Images
Remote Sensing 7(3), 2668 – 2691,
https://www.mdpi.com/2072-4292/7/3/2668.J.

WASP processor- L2A Temporal synthesis

Mosaic of WASP L3A



Norway Cloud-free mosaic 2021

Mosaic of WASP L3A September 2021

Conclusion – OTB Future

- OTB is alive and more and more used for operational use but less by researchers
- Roadmap to 9.0:
  - Remove GUI -> QGIS main interface
  - Drop support of MacOSX thanks to official docker image: https://hub.docker.com/r/orfeotoolbox/otb
  - Continue the Python integration
- And Next?
  - How to reached new contributor in C++?
  - Keep some machine learning module in OTB?
  - Focus on pre-processing to python array

- Your contribution are welcome:
  - on the OTB Forum: https://forum.orfeo-toolbox.org/
  - At FOSS4G this summer with one workshop and two presentations