Customer	**:**	**ESRIN**	**Document Ref**	**:**	**S2PAD-VEGA-SUM-0001**
Contract No	**:**	**21450/08/I-EC**	**Issue Date**	**:**	**13 April 2016**
WP No	**:**	**1.3**	**Issue**	**:**	**2.2**

Title : **Sentinel-2 MSI – Level-2A Prototype Processor Installation and User Manual**

Abstract : Sen2Cor is a prototype processor for Sentinel-2 Level 2A product formatting and processing. The processor performs the tasks of atmospheric-, terrain and cirrus correction and a scene classification of Level 1C input data. Level 2A outputs are: Bottom-Of-Atmosphere (BOA), optionally terrain- and cirrus corrected reflectance images, Aerosol Optical Thickness-, Water Vapour-, Scene Classification maps and Quality Indicators, including cloud and snow probabilities. The Level 2A Product Formatting performed by the processor follows the specification of the Level 1C User Product.

Author : Uwe Müller-Wilm
Digitally signed by Uwe Müller-Wilm
DN: cn=Uwe Müller-Wilm, email=uwe.mueller-wilm@telespazio-vega.de
Date: 2016.04.14 11:14:30 +02'00'
Uwe Müller-Wilm

Approval : Christian Laroque
Digitally signed by Christian Laroque
DN: cn=Christian Laroque, email=christian.laroque@telespazio-vega.de
Date: 2016.04.14 12:04:12 +02'00'
Christian Laroque

Accepted : Christine Dingeldey
Digitally signed by Christine Dingeldey
DN: cn=Christine Dingeldey, email=christine.dingeldey@telespazio-vega.de
Date: 2016.04.14 11:44:58 +02'00'
Christine Dingeldey
Quality Assurance Manager

Distribution :

Hard Copy File:
Filename: S2PAD-VEGA-SUM-0001-2.2.docx

Telespazio
A Finmeccanica/Thales Company

Telespazio VEGA Deutschland GmbH
Europaplatz 5, 64293 Darmstadt, Germany
Tel: +49 (0)6151 8257-0 Fax: +49 (0)6151 8257-799
www.telespazio-vega.de

This Page Is Intentionally Blank

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

AMENDMENT POLICY

This document shall be amended by releasing a new edition of the document in its entirety. The Amendment Record Sheet below records the history and issue status of this document.

AMENDMENT RECORD SHEET

ISSUE	DATE	DCI No	REASON
1.0	2.July 2012	-	Created
1.1	15 September 2012	-	Issue after S2PAD Phase 2 CDR. Updated according to ESA comments and discussion on CDR 02/08/2012
1.2	1.June 2013	-	Adaptation of installation procedures according to unification of environments and pre release of processor for QR
1.3	31. March 2014	-	New Section 2.2, updated Installation procedures for Windows, moved information for data IO into new created document [L2A-IODD]
1.4	27. June 2014	DCR-01, 02	Restructuring of sections 3.1, 3.2 to align the installation after upgrades of Anaconda and GDAL.
2.0	15.May 2015	-	Integration into Sentinel-2 Toolbox, Version 2.0 Complete Improvement of Installation Procedure: Integration of CONDA Packages for GDAL and GLYMUR Upgrade to PSD V12 Upgrade of JPEG-2000 Readers to OpenJPEG 2.1.0 instead of Jasper Fixes of SPRs according to Release note for Version 2.0
2.1	10.02.2016		Parallelisation on tile base implemented Upgrade to PSD V13.1
2.2	13.04.2016		Integration of Look Up Tables Automated Aerosol determination Automated Ozone selection New description for DEM selection Various improvements for command line Handling

DOCUMENT CHANGE RECORD

		DCR No	001
		Date	01. July 2012
		Originator	Uwe Müller-Wilm
		Approved by	Christian Laroque
1. Document Title:		Sentinel-2 Level-2A Prototype Processor Installation and User Manual	
2. Document Reference Number:		S2PAD-VEGA-SUM_SD-17	
3. Document issue / revision number:		1.0 DRAFT	
4. Page	5. Paragraph	6. Reason for change	
All	All	Created new	

		DCR No	002
		Date	15 Sept 2012
		Originator	Uwe Müller-Wilm
		Approved by	Christian Laroque
1. Document Title:		Sentinel-2 Level-2A Prototype Processor Installation and User Manual	
2. Document Reference Number:		S2PAD-VEGA-SUM_SD-17	
3. Document issue / revision number:		1.1	
4. Page	5. Paragraph	6. Reason for change	
	REF	SUM_01	
	REF	SUM_02	
	2.1	SUM_03	
	2.2	SUM_04	
	table 9	SUM_05	
	Operation	SUM_06	

		DCR No	003
		Date	01 June 2013
		Originator	Uwe Müller-Wilm
		Approved by	Christian Laroque

1. Document Title:		Sentinel-2 Level-2A Prototype Processor Installation and User Manual
2. Document Reference Number:		S2PAD-VEGA-SUM_SD-17
3. Document issue / revision number:		1.2
4. Page	5. Paragraph	6. Reason for change
	1.3.1 – 1.3.2	Consolidation of references with glossary
	2.5	Alignment with processor development
	3.1	Alignment with processor development
	3.2	Alignment with processor development
	3.3	Alignment with processor development
	3.4	Alignment with processor development

	DCR No	004
	Date	31. March 2014
	Originator	Uwe Müller-Wilm
	Approved by	Christian Laroque
1. Document Title:		Sentinel-2 Level-2A Prototype Processor Installation and User Manual
2. Document Reference Number:		S2PAD-VEGA-SUM
3. Document issue / revision number:		1.3
4. Page	5. Paragraph	6. Reason for change
	2.2.1 – 2.2.6	New Sections on processors purpose and use
	3.1.2.2.3	moved information for data IO into new created document [L2A-IODD]
	3.1	added Installation procedures for Windows,
	3.2	Added configuration procedure for Windows.
	DCR No	004
	Date	31. March 2014
	Originator	Uwe Müller-Wilm
	Approved by	Christian Laroque
1. Document Title:		Sentinel-2 Level-2A Prototype Processor Installation and User Manual
2. Document Reference Number:		S2PAD-VEGA-SUM
3. Document issue / revision number:		1.3
4. Page	5. Paragraph	6. Reason for change

	2.2.1 – 2.2.6	New Sections on processors purpose and use
	3.1.2.2.3	moved information for data IO into new created document [L2A-IODD]
	3.1	added Installation procedures for Windows,
	3.2	added configuration procedure for Windows.

		DCR No	004
		Date	31. March 2014
		Originator	Uwe Müller-Wilm
		Approved by	Christian Laroque
1. Document Title:		Sentinel-2 Level-2A Prototype Processor Installation and User Manual	
2. Document Reference Number:		S2PAD-VEGA-SUM	
3. Document issue / revision number:		1.4	
4. Page	5. Paragraph	6. Reason for change	
	3.1	Restructuring of sections 3.1 to align the installation after upgrades of Anaconda and GDAL.	
	3.2	Restructuring of sections 3.2 to align the configuration after upgrades of Anaconda and GDAL.	

		DCR No	005
		Date	10. Februar 2016
		Originator	Uwe Müller-Wilm
		Approved by	Christian Laroque
1. Document Title:		Sentinel-2 Level-2A Prototype Processor Installation and User Manual	
2. Document Reference Number:		S2PAD-VEGA-SUM	
3. Document issue / revision number:		2.1	
4. Page	5. Paragraph	6. Reason for change	
	2.2.6	added description for usage of Planet DEM	
	2.6	new chapter on parallelisation.	
	3.2.1	command line interpreter upgraded to parallelisation.	

		DCR No	006
		Date	13. April 2016
		Originator	Uwe Müller-Wilm
		Approved by	Christian Laroque
1. Document Title:		Sentinel-2 Level-2A Prototype Processor Installation and User Manual	
2. Document Reference Number:		S2PAD-VEGA-SUM	
3. Document issue / revision number:		2.2	
4. Page	5. Paragraph	6. Reason for change	
	2.1	Moved and updated Processor Architecture	
	2.2.2.1	Look up Table Generation	
	2.2.6	Surface reflectance retrieval	
	2.2.7	Usage of Digital Elevation Maps	
	2.4.2	Improvement of the Processing Routines	
	3.1.1.1	Anaconda Upgrade	

This Page Is Intentionally Blank

1. INTRODUCTION

1.1 Purpose and Scope

This document is the Software *Installation and User Manual (SUM)* for the Sentinel-2 Level-2A Prototype Processor, which is labelled ***Sen2Cor*** for ***Sen***tinel ***2*** (atmospheric) ***Cor***rection.

The prototype implementation for the Level 2A processing of Sentinel-2 imagery over land is a combination of state-of-the-art techniques for performing Atmospheric Corrections (AC, including Cirrus clouds and terrain correction), tailored to the Sentinel-2 environment and a Scene Classification (SC) module.

Sen2Cor performs a pre-processing of Level-1C (L1C) Top of Atmosphere (TOA) image data, and applies a scene classification an atmospheric correction and a subsequent conversion into an ortho-image Level-2A Bottom-Of-Atmosphere (BOA) reflectance product. Outputs are an Aerosol Optical Thickness (AOT) map, a Water Vapour (WV) map and a Scene Classification map together with Quality Indicators data. Details of the products and its contents is provided in [L2A-PDD] of section 1.3.

Level-2A (L2A) products are re-sampled as L1C products with a constant GSD (Ground Sampling Distance) of 10m, 20m and 60 m according to the native resolution of the different spectral bands. If applicable, Level-2A products are provided for each MSI channel at coarser resolution (i.e. 20 m and 60 m) as well.

A large database of look-up tables (LUTs) has been compiled using an atmospheric radiative transfer model based on libRadtran[1]. The LUTs are generated for a wide variety of atmospheric conditions, solar geometries, and ground elevations and are calculated with a high spectral resolution of 0.6 nm. This database has been subsequently resampled with the Sentinel-2 spectral responses, in order to obtain the sensor-specific functions needed for the atmospheric correction.

This Installation and User Manual mainly addresses the following aspects of the software:

1. Introduction into the main functionality of the system
2. Description of the system functionality, operations, architecture and modules
3. Overview of the Sen2Cor system and its operational environments:
   - Description of the Software Development Environment (SDE)
   - Description of the Source Distribution (SCD)
   - Description of the Runtime Environment (RTE)
4. Installation, configuration and operation of SDE, SCD and RTE
5. Creation of distribution media for all environments

   The configuration parameter and their descriptions are now part of the IODD [L2A-IODD]

[1] https://www.libradtran.org

Restrictions:

1. The Look Up tables are only compiled for the rural aerosol and the mid-latitude summer (MS) atmosphere with its corresponding ozone column (331 DU for sea level. Other Look up Tables can be generated on request but are not part of the current baseline.

2. The current version of the processor will not give validated results over water or on coastal regions. The design of the current implementation does not prevent such an extension, but this is not part of the current baseline.

1.2 Document Overview

The configuration and user manual consists of the following chapters and sections:

Chapter, Section	Description
1	this chapter
2	Introduces the Sen2Cor system. What is the general purpose of the application, how is it structured, what are the processing schemes. It lists the general system architecture modules and functionality and gives a brief overview on its functionality and operation.
3	Introduces the two different environments (runtime and development) of the system in general and the source code distribution.
3.1	How to install the processor software and the according runtime environment.
3.2	How to configure the processor software and the according runtime environment.
3.3	How to run the processor software in the according runtime environment.
3.4	Describes the setup configuration and operation of the software development environment, set up on top of Eclipse and PyDev. Contains also a detailed overview on all used third party tools.

With these sections the configuration and user manual enables developers to upgrade and maintain the software and users of the software to operate the system within their specific hardware environment.

What this document will not provide is the scientific background of the application. This is part of the corresponding ATBD [L2A-ATBD], see below. Also the configuration of the processor has been moved into [L2A-IODD]. In order to avoid redundancies and inconsistencies between the different project documents, this content will thus not be repeated here. If it is necessary for the understanding of the operation, this SUM will refer to the according chapters of [L2A-IODD], [L2A-PDD] and [L2A-DPM].

1.3 Documentation and Definitions

The reference list of all project related documents with their full version numbers and issue dates is given in:

[L2A-GLODEF] S2PAD Project Glossary, S2PAD-VEGA-GLO-0001, version 3.3, 31.03.2014

1.3.1 Normative Reference Documents

[L2A-IODD] Sentinel-2 MSI – Input Output Data Definition

[L2A-PFS] Sentinel-2 MSI – Product Format Specification

[L2A-PDD] Sentinel-2 MSI – Level-2A Product Definition Document

[L2A-ATBD] Sentinel-2 MSI – Level 2A Products, Algorithm Theoretical Basis

[L2A-PERF] Sentinel 2 MSI – Performance Tests on 64 and 32 bit Python

1.3.2 Informative Reference Documents

Document

[L2A-DPM] Sentinel-2 MSI – Level 2A Detailed Processing Model

This Page Is Intentionally Blank

2. FUNCTIONALITY AND OPERATION

Sen2Cor is a prototype processor for Sentinel-2 Level 2A product formatting and processing. The processor performs the tasks of atmospheric-, terrain and cirrus correction and a SC of Level 1C input data. Level 2A outputs are: Bottom-Of-Atmosphere (BOA), optionally terrain- and cirrus corrected reflectance images, AOT-, WV-, SC maps and Quality Indicators, including cloud and snow probabilities. The Level 2A Product Formatting performed by the processor follows the specification of the Level 1C User Product. Details are given in [L2A-PFS].

The Sentinel-2 Multi-Spectral Instrument (MSI) consists of 13 spectral bands with three different resolutions (10m, 20m and 60m) as shown in Figure 2-1. The instrument covers a 290 km swath. The Level-1C image product, which serves as the input for the Level-2A processing consists of a series of n tiles, each with a 100 km square. Each tile consists of thirteen compressed JPEG-2000 images, each image representing one single band. The thirteen bands have three different resolutions (10m, 20m and 60m) which lead to different image dimensions of the Level-1C input product. These details are given in [L2A-PDD] and [L2A-IODD].

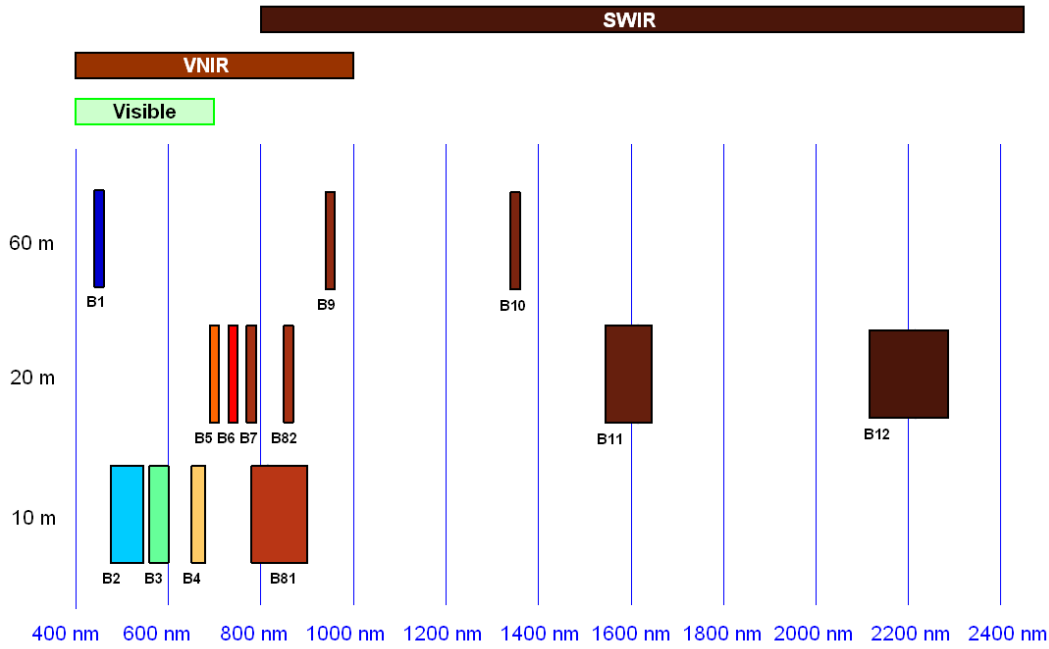

Figure 2-1 – Sentinel-2 Spectral Bands and Resolutions

2.1 Level-2A Processor Architecture

It follows a high level description of the processor architecture:

The Sen2Cor application is designed by the following ten essential modules (classes):

1. **L2A_Process:** the general operator module, which coordinates the interaction between the other modules and creates the skeleton L2A product structure of the metadata.
2. **L2A_Schedule:** a scheduler module, coordinating the parallel execution of 1..n processing modules, working on tile base (L2A_ProcessTile).
3. **L2A_ProcessTile:** a single processing module, executing the tasks of scene classification, atmospheric correction and the creation of metadata on tile base.

4. **L2A_SceneClass:** performs the coarse classification of the input images into their different contents like clouds, snow, water, soil etc. and provides statistical analysis.
5. **L2A_AtmCorr:** transforms the input from top of atmosphere (TOA) to bottom of atmosphere (BOA) representation and performs the atmospheric correction of the input.
6. **L2A_Config:** a helper class providing the configuration parameters to all other modules listed above.
7. **L2A_Tables:** a helper class, managing the conversion of the JPEG-2000 based input data to an internal format (and vice versa) and providing a high performance access to the data for the processing modules (see section 2.4.1). It uses its own private L2A_Config instance.
8. **L2A_Manifest:** a class specialized for the generation of the manifest on product level.
9. **L2A_XmlParser:** a utility class for parsing the metadata and GIPP files on demand.
10. **L2A_Library:** a collection of common tools used by all classes on demand.

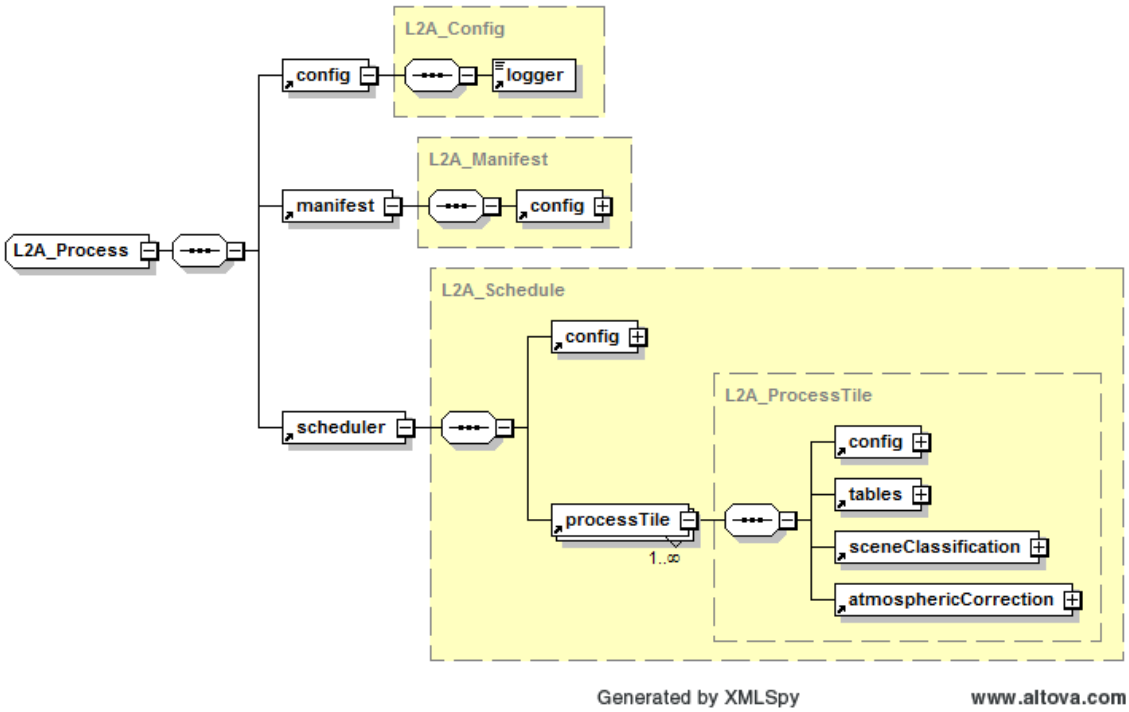

Figure 2-2 – High Level Processor Architecture

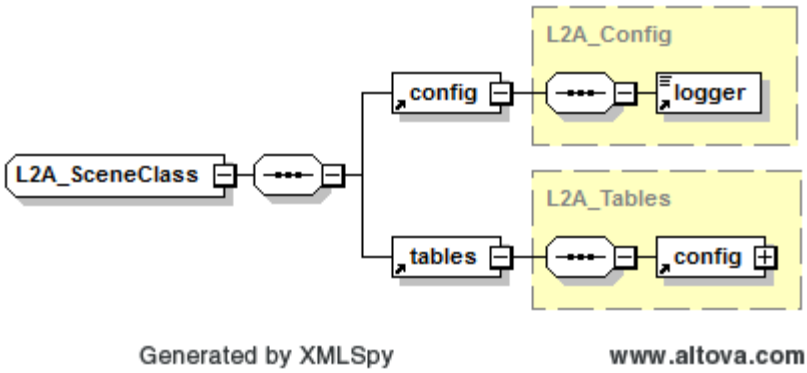

Figure 2-3 – Scene Classification Module

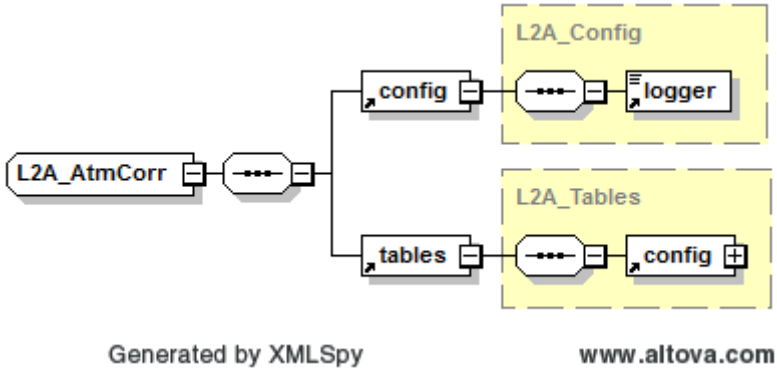

Figure 2-4 – Atmospheric Correction Module

2.2 Workflow

Figure 2-5 below shows the main processing workflow. After reading and processing the input parameter and data the main processing module triggers the creation of an internal temporary database, which is then used by the SC and the AC module to retrieve and to store the data and intermediate products. The processing can act in a loop, dependent on the number of different product resolutions to be generated.

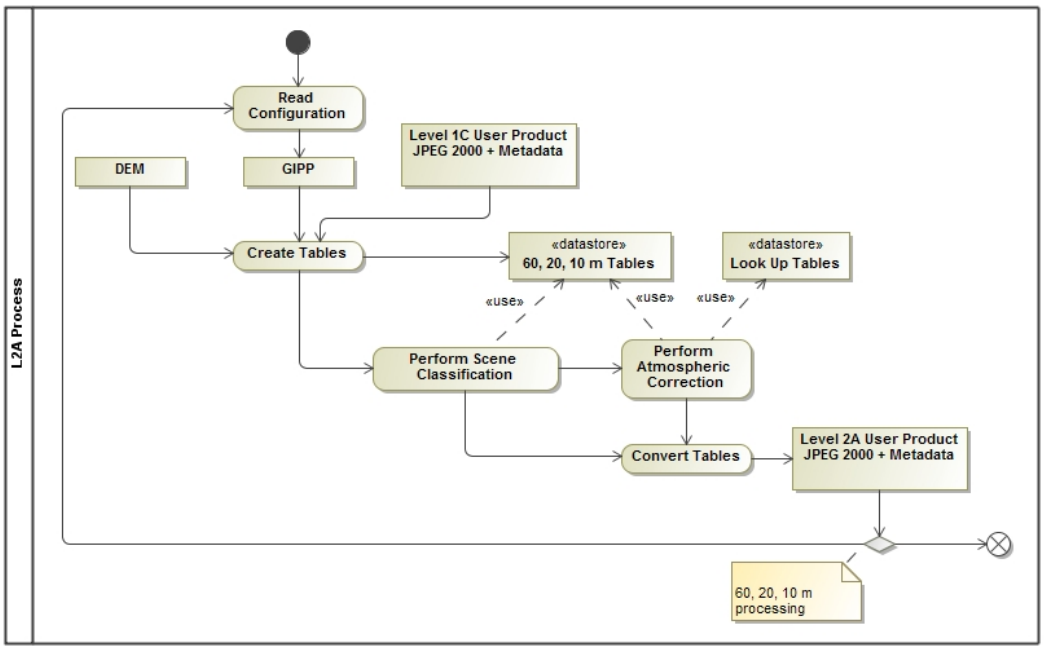

Figure 2-5 – Processing Flow, Overview

2.2.1 Scene Classification (L2A_SceneClass)

The SC algorithm allows to detect clouds, snow and cloud shadows and to generate a classification map, which consists of 4 different classes for clouds (including cirrus), together with six different classifications for shadows, cloud shadows, vegetation, soils / deserts, water and snow. The algorithm is based on a series of threshold tests that use as input top-of-atmosphere reflectance from the Sentinel-2 spectral bands. In addition, thresholds are applied on band ratios and indexes like the Normalized Difference Vegetation - and Snow Index (NDVI, NDSI [3]). For each of these thresholds tests, a level of confidence is associated. At the end of the processing chain a probabilistic cloud mask quality map and a snow mask quality map is produced. The algorithm uses the reflective properties of scene features to establish the presence or absence of clouds in a scene. Cloud screening is applied to the data in order to retrieve accurate atmospheric and surface parameters, either as input for the further processing steps below or for being valuable input for processing steps of higher levels. Figure 2-5 below shows the results of a SC (right side) based on modified AVIRIS test data (left side). Twelve different classifications are provided.

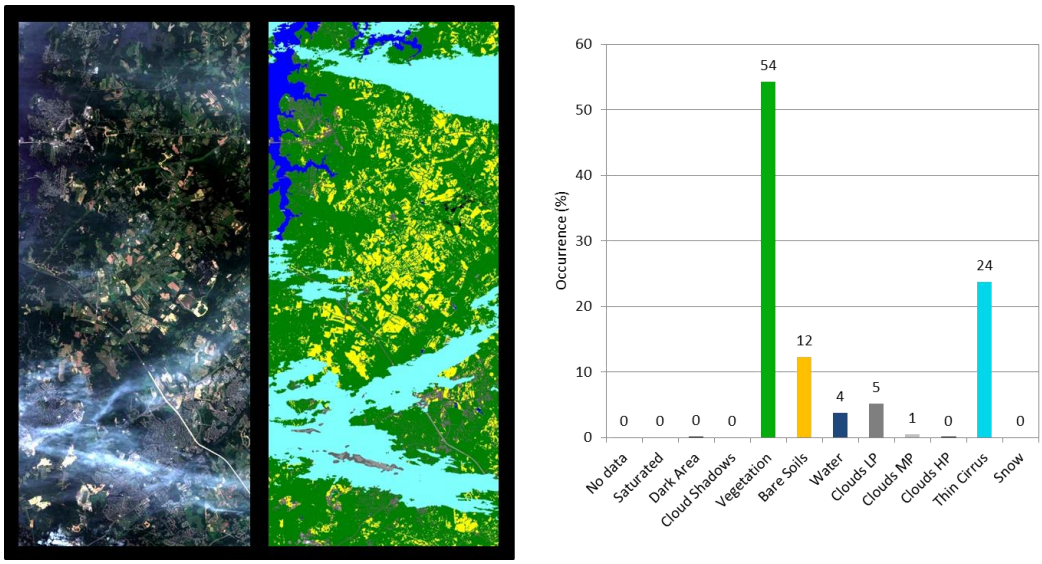

Figure 2-6 – Scene Classification

The generated classification map is specified as follows:

Table 2-1 – Classification Map

Label	Classification
0	NO_DATA
1	SATURATED_OR_DEFECTIVE
2	DARK_AREA_PIXELS
3	CLOUD_SHADOWS
4	VEGETATION
5	BARE_SOILS
6	WATER
7	CLOUD_LOW_PROBABILITY
8	CLOUD_MEDIUM_PROBABILITY
9	CLOUD_HIGH_PROBABILITY
10	THIN_CIRRUS
11	SNOW

Associated quality indicators on snow and cloud probability are generated from the results. These Quality indicators calculate the probability (0-100%) that the earth surface is obstructed by clouds or optically thick aerosol (ice or snow).

The SC processing consists of six different steps:

1. Snow detection;
2. Cloud detection;
3. Cloud shadow detection;
4. Cirrus detection;
5. Classification map generation.

The processing is shown in the flow diagram below.

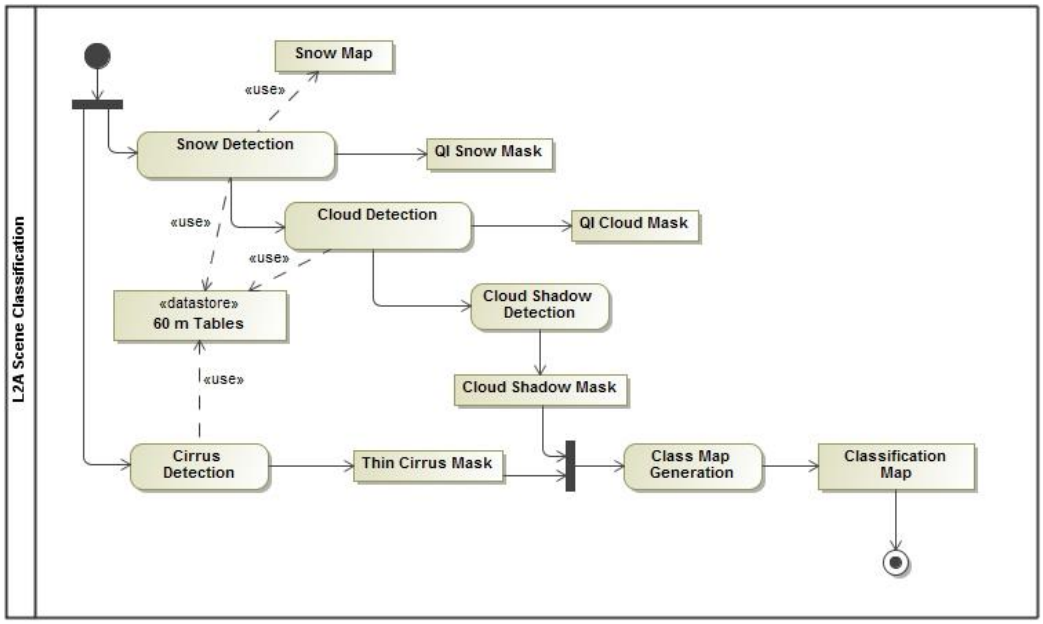

Figure 2-7 – Scene Classification, Processing Flow

Details of this algorithm, especially on the different threshold conditions are given in chapter 3 of [L2A-ATBD] and chapter 2.5 of [L2A-DPM]

2.2.2 Atmospheric Correction (L2A_AtmCorr)

The AC processing consists of a set of four different subtasks, (AOT, WV and terrain retrieval (optional with terrain and cirrus correction, having three different user products as output: AOT and WV tables on pixel level and the BOA corrected reflectance images for all bands measured. Figure 2-8 below shows the processing flow for the atmospheric correction module.

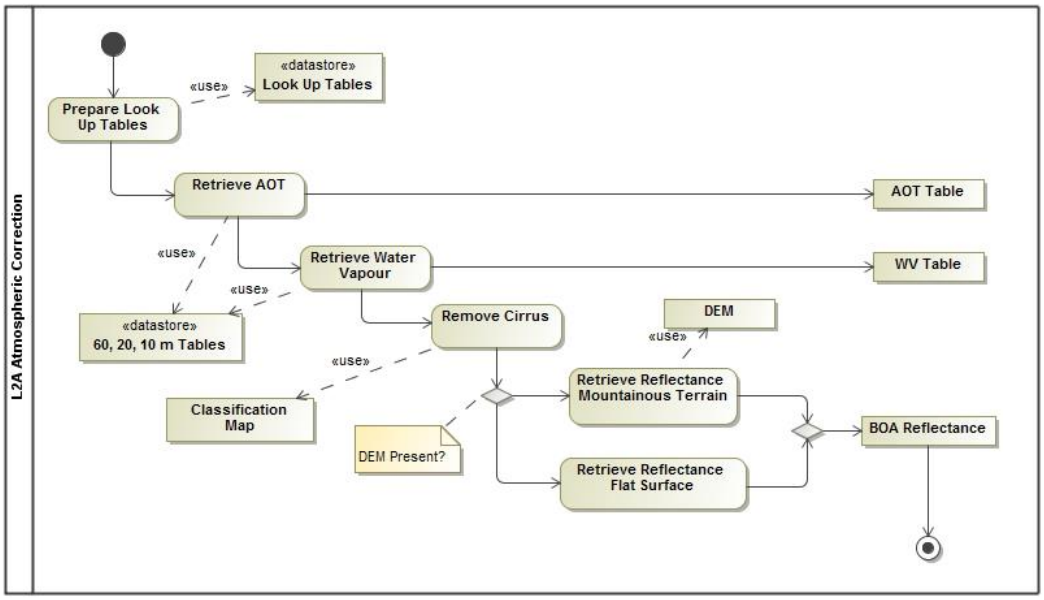

Figure 2-8 – Atmospheric Correction, Processing Flow

2.2.2.1 Look Up Table Generation

The atmospheric model of SEN2COR (L2A_AtmCorr) is dependent on the calculation of radiative transfer functions for different sensor and solar geometries, ground elevations, and atmospheric parameters. The following list presents a 6-dimensional parameter space and the grid spacing for each parameter. The processor reads the parameter in form of Look Up Tables (LUTs) pertaining to this parameter space and interpolates, if required. The LUTs have been generated via libRadtran, a library for the calculation of solar and thermal radiation in the Earth's atmosphere.

Table 2-2 – Parameter space for atmospheric correction

Parameter	Range	Increment / grid points
Solar zenith angle	0 -70°	10°
Sensor view angle	0 -10°	10°
Relative azimuth angle	0 -180°	30°(180°= backscatter)
Ground elevation	0 -2.5 km	0.5 km
Visibility	5 -120 km	5, 7, 10, 15, 23, 40, 80, 120 km
Water vapour, summer	0.4 -5.5 cm	0.4, 1.0, 2.0, 2.9, 4.0, 5.0 cm
Water vapour, winter	0.2 -1.5 cm	0.2, 0.4, 0.8, 1.1 cm

Starting with SEN2COR release 2.2.0 the user can select between four atmospheric models: a set of 24 LUTs has been integrated to cover most of atmospheric conditions on Earth for the Sentinel-2 mission. In the SEN2COR context, a set of LUTs is composed by 6 or 4 LUT files depending on the total water vapour columns content of the atmosphere. Different LUTs are calculated for the mid-latitude summer and mid-latitude winter atmospheres, with 6 different (sea level) ozone contents, a rural and a maritime aerosol, 6 or 4 different sea level water vapour columns. For each supported water vapour level, the ground-to- space water vapour column depending on elevation according to the atmosphere temperature / humidity vertical profile is provided.

SEN2COR LUTs are calculated for:

- 2 different types of aerosols (rural and maritime)
- 2 different types of atmospheres (mid latitude summer and mid latitude winter)
- 6 different types of ozone concentrations (depending on summer or winter case)
- 6 or 4 different amounts of water vapour column (depending on summer or winter)

2.2.2.2 User configuration

The LUT selection is configurable via the user configuration file (L2A_GIPP.xml) located inside the cfg folder of the directory where the $SEN2COR_HOME environment variable points to. In the Look_UP_Tables selection of the configuration file, three entries: Aerosol_Type, Mid_Latitude and Ozone_Content can be set. The water vapour columns are set internally.

Default processing via configuration is the rural (continental) aerosol type with mid latitude summer and an ozone concentration of 331 Dobson Units.

2.2.2.2.1 Setting of Automated Ozone Input

If the Ozone_Content is set to '0' by the user, it will be determined automatically by the processor. In that case, the measured ozone concentration is read from the L1C input product (located in the AUX_DATA folder of each tile) and the LUT with the best fit for the measured ozone concentration is used. Other parameters possible are referenced in the configuration file.

2.2.2.2.2 Setting of Automated Aerosol / Atmosphere Determination

If the Aerosol_Type and / or Mid_Latitude are set to 'AUTO' by the user, it will be determined automatically by the processor. In that case the processor will process two (aerosol only) or four test trials before the final processing of the atmospheric correction takes place. After calculation of the scene path radiance in the blue and red region (as total minus reflected radiance, using the average values obtained for the dark reference pixels) the ratio of the path radiance for the blue channel scene compared to the read channel scene be compared to the corresponding ratio for the existing aerosols (RURAL, MARITIME) or atmospheres (SUMMER, WINTER) as contained in the LUTs.

The aerosol type for which the double ratio (dp) is closest to 1 is the best approximation for the scene and will be selected ad used in all subsequent measures for the corresponding tile.

2.2.3 Aerosol Optical Thickness

AOT retrieval provides a measure for the visual transparency of the atmosphere. It is derived using the DDV (Dense Dark Vegetation) algorithm [5], using the short wave infrared (SWIR) band 12 and correlates its reflectance with bands 4 (red) and 2 (blue). The algorithm requires that the scene contains reference areas of known reflectance behaviour, preferably Dark Dense Vegetation (DDV) and/or dark soil and water bodies.

The algorithm starts with a user-defined visibility (default: 20 km) as input. If the scene contains no dark vegetation or soil pixels, the surface reflectance threshold of band 12 will be successively iterated in order to include medium brightness reference pixels in the sample. If the scene contains no reference and no water pixels the scene is processed with the start visibility instead. The algorithm delivers an AOT map as shown in Figure 2-9 below.

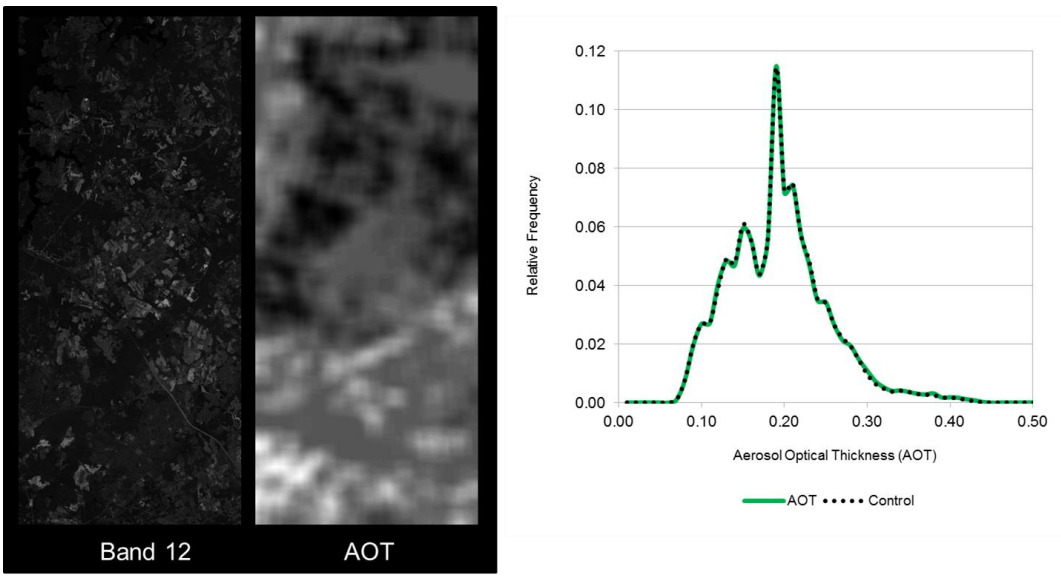

Figure 2-9 – AOT Retrieval using Band 12

2.2.4 Water Vapour Retrieval

WV retrieval over land is performed with the Atmospheric Pre-corrected Differential Absorption algorithm (APDA, [6]) which is applied to the two Sentinel-2 bands B8a, and B9 (Fig. 4). Band 8a is the reference channel in an atmospheric window region. Band B9 is the measurement channel in the absorption region. The absorption depth is evaluated by calculating the radiance for an atmosphere with no WV, assuming that the surface reflectance for the measurement channel is the same as for the reference channel. The absorption depth is then a measure of the WV column content.

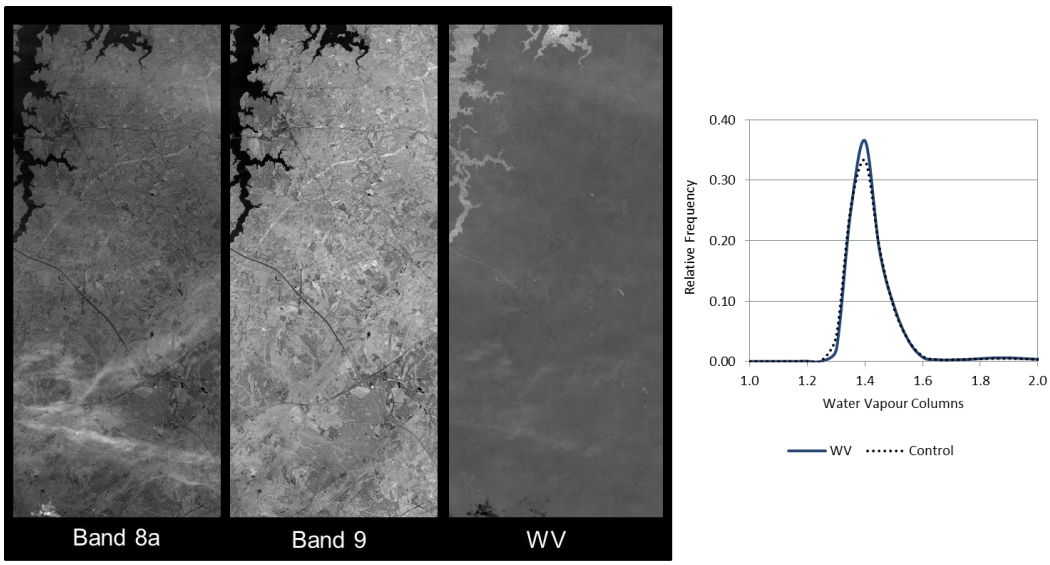

Figure 2-10 – WV Retrieval using Bands 8a and 9

2.2.5 Cirrus Correction

The Cirrus Correction algorithm uses the sentinel (cirrus) channel 10. Thin cirrus clouds affect the visible, near- and shortwave infrared spectral regions. They are partially transparent and thus difficult to detect with broad-band multispectral sensors, especially over spatially inhomogeneous land areas.

WV, in contrast, dominates in the lower troposphere of 0-5 km. A narrow spectral band in a spectral region of very strong WV absorption (Band 10) will thus absorb the ground reflected signal, but will receive the scattered cirrus signal.

Cirrus reflectance of band 10 can therefore be correlated with other bands in the VNIR and SWIR region and the cirrus contribution can thus be removed from the radiance signal to obtain a cirrus-corrected scene. This is shown in Fig. 5 below as a qualitative result.

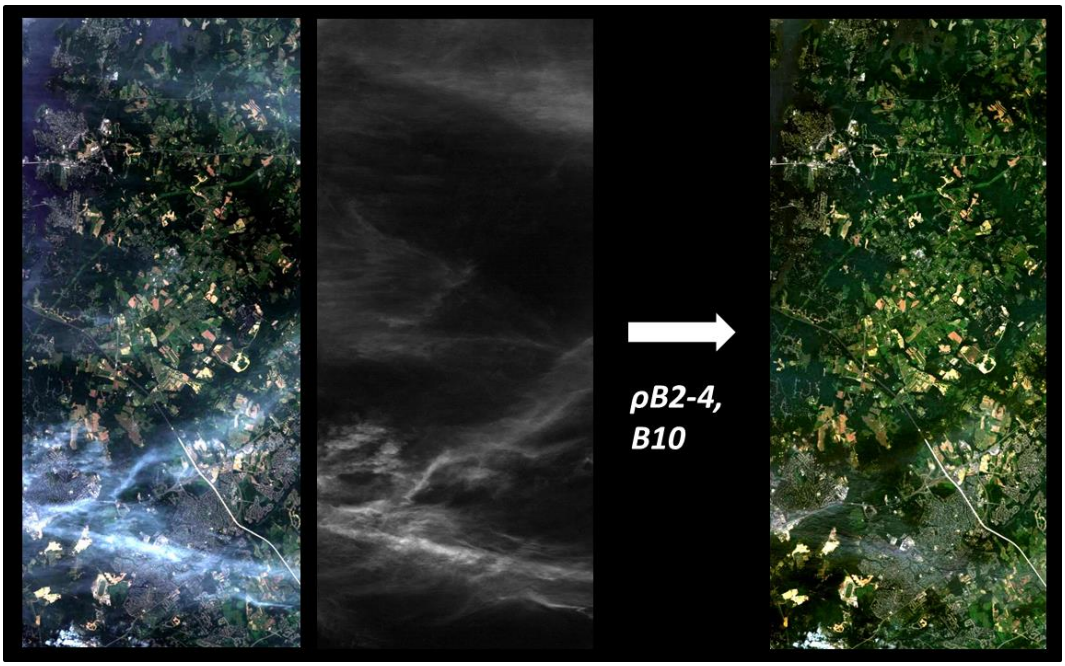

Figure 2-11 – Cirrus Correction, Bands 2-4 with Band 10

2.2.6 Surface Reflectance Retrieval

Surface Reflectance retrieval is performed for each sequential Band B1 – B12. Figure 2-9 below shows the Level 1C input data and the corresponding Level 2A output after atmospheric correction from a scene of La Paz, retrieved on 28.03.2016.

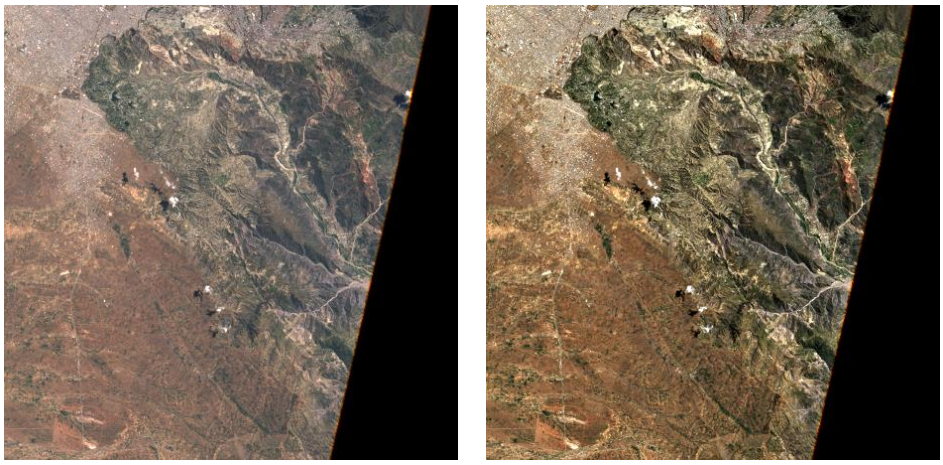

Figure 2-12 – Left: Level 1C Input, Bands 2-4; right: Level 2A Output, Bands 2-4, RGB composite images, scene from La Paz on

2.2.7 Usage of Digital Elevation Maps

Since the release of SEN2COR 2.2 Digital Elevation Maps can be used for two purposes:

1. Improvement of the scene classification: previous versions of the processor had a tendency of false classification of water pixels inside of cloud borders and the correct discrimination between topographic and cloud shadow pixels. This has now been improved by taking the height information of an (optional) digital elevation map (DEM) as an additional

input. To use this feature it is necessary to activate the reading of an appropriate DEM as is explained below.

2. Improvement of the terrain correction for rugged terrains: the algorithm for rugged terrain requires the existence of an appropriate Digital Elevation Map (DEM) and a set of derived products like slope, aspect and terrain shadow maps. The retrieval of the water vapour map also includes this terrain elevation.

SEN2COR is currently prepared to make use of two different DEM's. The first one is the 90m SRTM Digital Elevation Database from CGIAR-CSI. Please read carefully the Disclaimer given on:

http://www.cgiar-csi.org/data/srtm-90m-digital-elevation-database-v4-1

before you decide to make use of this database.

The second supported format is the commercial 90m DTED-1 Format from PlanetDEM:

http://www.planetobserver.com/products/planetdem/planetdem-90/

We do not activate the usage of any of these databases by default. It has to be set manually. The user can specify a DEM in the L2A_GIPP.xml configuration file by setting the parameter for <DEM_Directory> to a relative path, instead of NONE. The DEM will then be located in the specified folder in the sen2cor home directory.

Example: <DEM_Directory>dem/srtm</DEM_Directory>

The Planet DEM is not downloadable for free and must be purchased. Therefore, the processor expects the appropriate DEM data having the form of: ***eXXX_nYY.dt1*** in the specified local folder and uses them, if available. If no appropriate DEM is found in this folder it tries to retrieve it from CGIAR-CSI:

For the CGIAR-CSI DEM, the processor will start automatically to download the DEM's from the database referenced by:

<DEM_Reference>

http://data_public:GDdci@data.cgiar-csi.org/srtm/tiles/GeoTIFF/

</DEM_Reference>

If a relative path for DEM_Directory is addressed. If an already downloaded DEM can be found in the local folder referenced by DEM_Directory, the download will not take place and the local archive is used instead. If the download fails and no local DEM can be found, the processor continues with a flat surface calculation.

The geo-coordinates and angular information are obtained from the Level-1C metadata. The area of interest is created and fitted to the input images according to the information retrieved from the metadata and the conversion is performed using the GDAL libraries from OSGEO.

The algorithm reformats resamples and assembles the DEMs based on the geo locations obtained either from the Level-1C metadata or from the JPEG-2000 images. Finally it creates the according slope, aspect and a hill shadow map using the GDAL tools.

The retrieval of the spectral reflectance cube consists of the following steps:

1. iterations for terrain reflectance;

2. (optional) empirical BRDF correction;
3. adjacency correction;
4. spherical albedo correction.

Figure 2-10 below shows a comparison of the Level 2A output without usage of a DEM and the corresponding Level 2A output when a 90 m CGIAR DEM is applied before terrain correction. The same scene as for Figure 2.9 is shown.

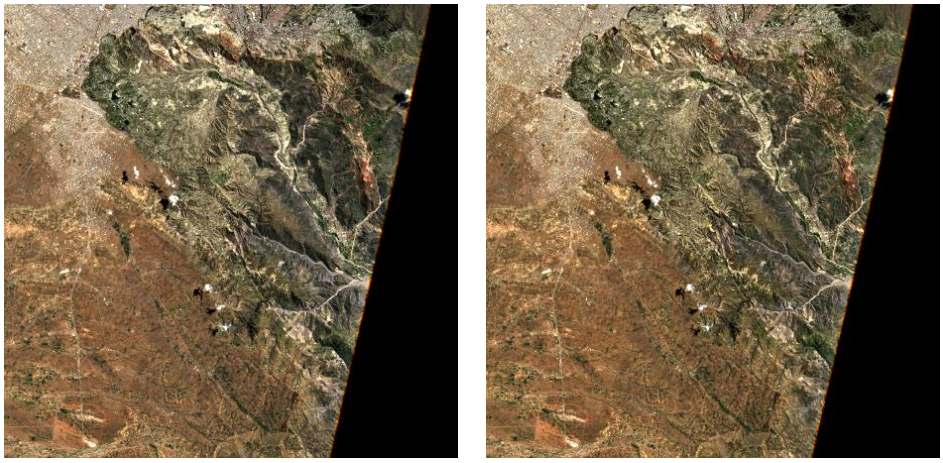

Figure 2-13 – Terrain correction; Left: Level 1C Input, Bands 2-4, RGB composite image, middle: DEM, reshaped to according scene, right: Level 2A Output, Bands 2-4, RGB composite image.

2.3 Product Formatting

The Level 2A Product format is closely related to the Level 1C Top-of-Atmosphere (TOA) reflectance product which serves as an input to the processor. It consists of 13 JPEG-2000 images, associated to the 13 Sentinel-2 spectral bands at three different spatial resolutions with a ground sampling distance of 10, 20, and 60m.

The generated Level 2A BOA reflectance output images are resampled and generated with an equal spatial resolution for all bands, based on three user selectable resolutions of 10, 20 and / or 60m.

The tile level contains different components, based on the user selected resolution:

- a 10 m resolution product contains spectral bands 2, 3, 4 and 8 and an AOT map resampled from 20m
- a 20 m product contains band 2 – 7, the bands 8a, 11 and 12 and an AOT and WV map
- a 60m product contains all components of the 20m product and additionally the 60m bands 1 and 9

The Cirrus band 10 will be omitted in the Level 2A output, as it does not contain surface information. Fig. 2 shows the Level 2A user product on tile level.

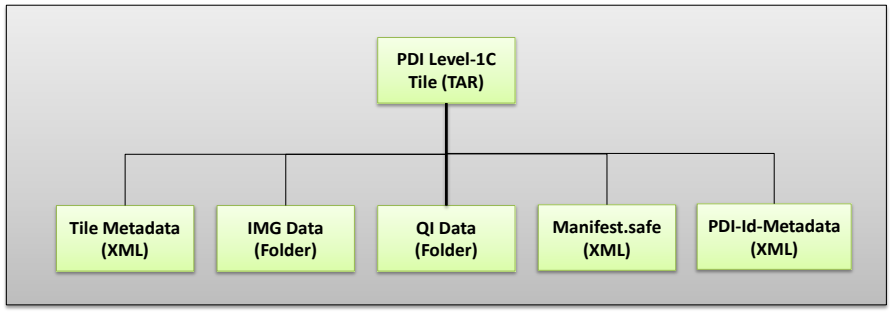

Figure 2-14 – Filing structure of Level-1C product on tile level

The complete specifications for all inputs are provided in [L2A-PFS] and [L2A-IODD]. Level-2A Inputs are derived from the Level-1C data schemes as described in [S2-PFS].

2.4 Runtime Configuration

Ground Image Processing Parameter (GIPP) are configured in an XML file named L2A_GIPP.xml, located in the <cfg> subdirectory of the processor (see example in the appendix), where they can be configured by the user. For each processed Level 2A tile the GIPP xml file will be renamed to S2A_USER_GIP_L2A_TL_<TILE_ID> (see section A.2.3 of [L2A_PFS]) and subsequently copied into the AUX_DATA subfolder of the corresponding granule. The GIPP are listed in their current context. These parameters, together with all input, output and other auxiliary data are comprehensive listed in the corresponding [L2A-IODD] document and thus not repeated here.

2.4.1 Pre-processing (L2A_Tables)

The Level-1C input data is expected to be present in a folder structure at product level, as specified in section 3.1 of [L2A-PDD]. The location of the Level-1C input archive can be specified via the command line argument.

Due to the three different resolutions of the Level-1C input images, conversion routines will serve for an up- and down-sampling of parts of the images into the appropriate resolution as well as the transfer from JPEG-2000 into a different internal format (see below). This has been implemented using the OpenJPEG library, which will be installed during the Sen2Cor setup.

Due to the relative huge size of the image data (especially for the high resolution 10 bands B02 – B04 and B08) data in- and output has been shown as rather time and memory consuming.

An intermediate data conversion thus was selected which converts the JPEG-2000 images and the necessary metadata (like the DEM) into an internal format based on HDF5. HDF5 is a data model, library, and file format for storing and managing data. It is especially designed for flexible and efficient I/O and for high volume and complex data. HDF5 is portable and extensible, allowing applications to evolve in their use and platforms to be supported.

The interfacing with the internal data format is implemented via PyTables. PyTables is a package for managing hierarchical datasets, designed to efficiently cope with extremely large amounts of data. It is built on top of the HDF5 library, the Python language and the NumPy package. PyTables and HDF5 are described in section 3.3.1.6

2.4.2 Improvement of the Processing Routines

Starting with SEN2COR 2.2.0 the routines for generating user products of different resolutions have been decoupled and generally improved:

- SEN2COR will now also work on existing L2A user products. This allows the generation of different resolutions in subsequent steps.
- If the user is specifying no resolution at all on the command line, all resolutions will be generated in three subsequent steps. This is the default. The selection of a 60 m resolution or 20m resolution via command line will only generate the resolution specified.
- If a resolution has already been processed for a given L2A product, sen2cor will not start regeneration but will inform the user that this product already exists.
- The selection of a 10 m resolution requires the generation of a 20 m product. If a 20m product already exists from a previous processing, this will be taken as input, otherwise it will be generated in a first step, before the 10 m resolution product is processed. If the user wants to start from scratch, he must move, delete or rename the existing product.
- It is also possible to remove only single tiles of an existing product. These tiles will then be regenerated; the other existing tiles will be left untouched.

2.4.3 The 60 m Product Processing

The 60m product processing takes the three 60m bands B01, B09 and B10 and a 60m DEM as inputs and converts them directly into the internal HD5 format. The three 10m bands B02-B04 will first be down-sampled to 60m. The same is true for the 20m bands B05 – B8A and B11, B12. The 10m band B08 is not used for the 60m processing.

Beneath the twelve optical channels (the cirrus channel B10 is excluded as it does not represent surface information) the 60m product processing generates a SC map according to the classification below, quality indicators for presence of snow and clouds (in percentage), an Atmospheric Optical Thickness (AOT), a WV map, a Visible Index file and a preview image, covering the three visible bands 2-4 within an JPEG-2 compressed image, having a 320m resolution.

2.4.4 The 20 m Product Processing

The 20m product processing takes the 20m bands B05 – B8A and B11, B12 and a 20m DEM as inputs and converts them directly into the internal HD5 format. The three 60m bands B01, B09 and B10 will first be up-sampled to 20m, the three 10m bands B02-B04 will be down-sampled to 20m. The 10m band B08 is not used for the 20m processing.

The 20m product processing covers nine optical channels a 20m AOT and a WV and a Visibility Index file corresponding to the AOT. The three resampled 60m bands B01, B09 and B10 are omitted in order to avoid spectral artefacts due to mixed signatures and resampling.

Starting with se2cor 2.2.0 it is no longer required, that a 60 m product will be processed first.

2.4.5 The 10 m Product Processing

Inputs for the 10m product processing are the four 10m bands B02-B04 and B08, an optional 10m DEM an up-sampled Scene Classification and a Visibility Index map, both up-sampled from 20m. As the WV influence is very small, only the scene-average WV needs to be used for the surface reflectance retrieval. The 10m product processing covers thus only four optical channels. The other channels, not used for the calculation

will be omitted. If a 20 m processing has already been performed in ahead, these data will be used as input. Else, a 20 m resolution will be performed first, in order to have access to the input data.

2.4.6 Post-Processing

The OpenJPEG Library is used for generating the final L2A-Product, transferring the internal HDF5 based tables back into the JPEG-2000 format. It keeps for all generated products the final (resampled) resolution.

Post-processing also packs the folder back to a Level-2A archive which is shown in its unpacked version in Figure 2-15. This archive will be placed in parallel to the Level-1C input archive. The location of the Level-2A output archive can be configured in the L2A_GIPP.xml configuration file, which can be found in the path where the SEN2COR_HOME environment variable points to.

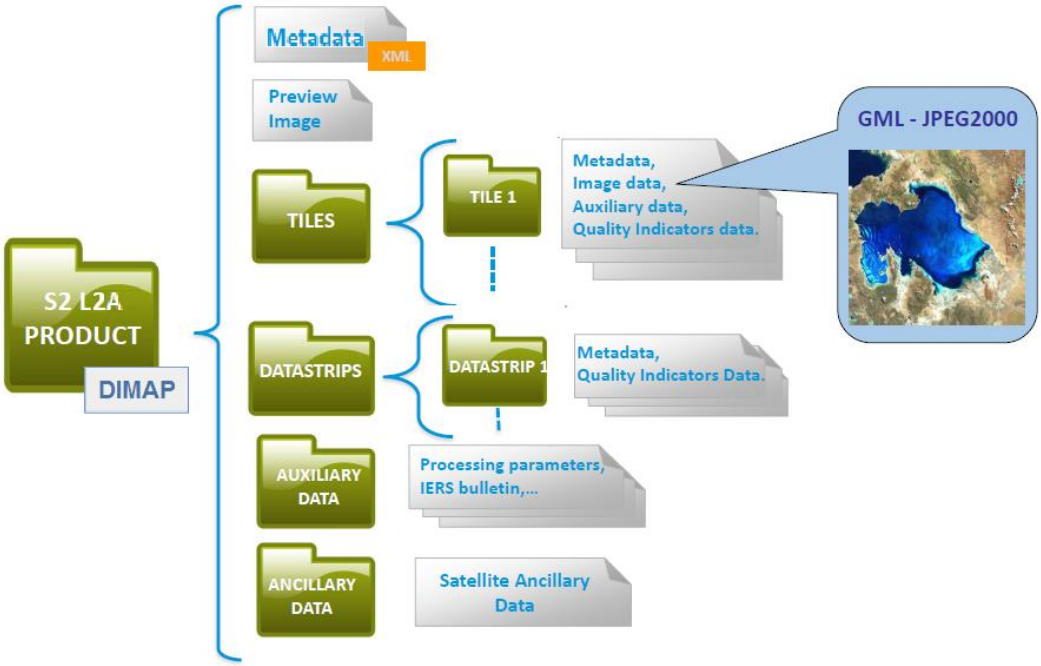

Figure 2-15 – Level 2A product, physical format Configuration (L2A_Config)

2.5 Parallel Processing

Since version 2.2.0 of sen2cor a parallel processing on tile base is implemented.

The maximum of processes can be determined via the L2A_GIPP.xml input configuration:

```
<Nr_Processes>AUTO</Nr_Processes>
```

Nr_Processes can be an unsigned integer value specifying the number or processes intended to operate in parallel or: AUTO. If AUTO is chosen, the processor determines the number of processes automatically, using the CPU info of the given architecture.

If L2A_Process is started with a L1C User product as the given directory argument, a series of tiles will be processed in parallel, according to the given Nr. of Processes in the L2A_GIPP above. If the Nr of Processes is 1, only a sequential processing will take place.

If L2A_Process is started with a L1C Tile as an argument, only the selected tile will be processed. The Nr. of processes will be temporary set to 1.

The given directory argument can be absolute or relative (see also section 3.2.1).

Memory requirements: the atmospheric correction processing for 10m resolutions uses a huge amount of memory due to the 10.000 x 10.000 pixel for each image. Multiple images must be kept at certain intervals completely in memory for performing correlations. Thus, for each processing at 10m resolution, as a general rule of thumb, a resource of **4 GB of memory per process** should be available. Thus, unless you have a machine with sufficient memory, leave the Nr. of processes to single processing (1), otherwise memory overflow errors will occur during processing. On a 2 Core MAC / Linux workstation with 8 GB RAM, 2,8 GHz Intel i5 and 500 GB SSD, a 4 tiles parallel processing for 10m resolution images has been successfully performed during the tests. In systems which are equipped with hard disks, however more memory might be needed, due to swapping effects.

2.5.1 Interface changes:

Command line: a new feature for the processing of single tiles has been added. The processing of a complete product can be used as in previous versions.

Configuration: the L2A_GIPP.xml has been rearranged. Only user specific configuration parameters are presented. Expert configuration parameters have been moved into 2 configuration files which are located under:

<anaconda_home>/lib/python2.7/site-packages/sen2cor-2.2.0-py2.7.egg/sen2cor/cfg

L2A_CAL_SC_GIPP.xml: contains the calibration parameter for the scene classification which should only be changed by expert users. Thus this file is no member of the user configuration.

L2A_CAL_AC_GIPP.xml: contains the calibration parameter for the atmospheric correction which should only be changed by expert users. Thus this file is no member of the user configuration.

Other Changes:

Commands can now be executed repeatedly on an already existing User Product. The current product will then be overwritten with the new calculation.

- A selected resolution of 60 m will only process the 60 m resolution.
- A selected resolution of 20 m will only process the 20 m resolution.
- A selected resolution of 10 m will process the 20 m resolution first, if not exist (as it is the base for the 10 m. If it exists, only the 10 m resolution will be processed.

Interface to the PDGS

The current xml interface from sen2Cor embedded in the PDGS was agreed during May 2015:

```
<Processor_List>
<Hp_Sc_Name step_id="1" wall_time="00:10:00" mem="2">L2A_Process <param1>
<param2><param3></Hp_Sc_Name>
<Hp_Sc_Version>xx.yy.zz</Hp_Sc_Version>
</Processor_List>
```

Sen2Cor creates a bash script as below:

```
# BEGIN
# L2A_Process.bash:
#
export PATH="/<the anaconda installation directory>/anaconda/bin":$PATH
source ./L2A_Bashrc
L2A_Processor-$1 $2 $3 $4
# END
```

Where:

```
$1: a version string
$2: the resolution (60 <default>, 20, 10)
$3: the L2A Product ID (a string)
$4: optional parameter, currently not used.
# BEGIN
# L2A_Process.bash:
#
export PATH="/<the anaconda installation directory>/anaconda/bin":$PATH
source ./L2A_Bashrc
L2A_Process-$1 $2 $3
# END
```

Where:

```
$1: a version string
$2: the resolution (60 <default>, 20, 10)
$3: the L2A Product directory,
```

$3 (the directory argument can be used in two different modes.

Examples:

```
<the_L1C_product_directory>/S2A_OPER_PRD_MSIL1C_PDMC_20150630T084936_R062
_V20150627T103515_20150627T103515.SAFE

<the_L1C_product_directory>/S2A_OPER_PRD_MSIL1C_PDMC_20150630T084936_R062
_V20150627T103515_20150627T103515.SAFE/GRANULE/S2A_OPER_MSI_L1C_TL_MTI__2
0150627T180307_A000062_T32TNR_N01.00
```

When the second argument (on tile base) is used, the processor switches automatically in the single process mode and generates only the given tile. With this, the processor can also be used in combination with an external (higher level) orchestrator.

2.6 Logging (Logger)

The module L2A_Config keeps an instance of a logger object. Each of the processing modules writes its own diagnosis and status messages into a common log-file, located in the HTML folder at the top of the User product.

The log level can be configured in the GIPP (L2A_GIPP.xml) as:

```
DEBUG, INFO, WARNING, ERROR, CRITICAL, NOTSET
```

The log-level is hierarchical: if e.g. set to ***DEBUG***, all higher messages such as ***INFO***, ***WARNING***, ***ERROR*** or ***CRITICAL*** will be displayed as well. Setting the log-level to ***ERROR*** displays only ***ERROR*** and ***CRITICAL*** messages.

Beneath the message itself, the logger displays the system time stamp when the message was generated, the log-level, the module which has generated the message and the function and code line of the module which has generated the message.

3. CONFIGURATION AND INSTALLATION

The Sen2Cor software is a command-line oriented application, written in the programming language python. It consists of a set of modules, as listed in section 2.3. Due to license restrictions the module for the atmospheric correction is a binary library, containing those parts of the algorithm not public available.

3.1 Installation

In order to avoid dependency problems between the three different platforms supported and its according python packages, the Sen2Cor application works under the umbrella of the cross platform Python distribution Anaconda:

https://store.continuum.io/cshop/anaconda/

Currently Python version 2.7 is used as a runtime environment.

Two important Remarks:

- **It is strongly advised** to follow the recommendations choosing a **local Anaconda installation** on your machine, in order not to run into conflicts with the default python interpreter at your platform.
- **Avoid any trials to install the Sen2Cor application under a different python distribution outside of Anaconda.** There are numerous dependencies to distinct versions of GDAL, pyTables and the OpenJPEG library and to its configurations, which will very probable not be fulfilled by your default python installation.

The installation of the whole system is then performed in two steps:

3.1.1 Setting up the Runtime Environment

3.1.1.1 Anaconda Upgrade

If you have already installed Anaconda on you platform, then perform the following command via the command line:

```
C:\>conda update conda
Using Anaconda Cloud api site https://api.anaconda.org
Fetching package metadata: ....
```

it should end with displaying the following information:

```
conda                     4.0.5                    py27_0
```

```
C:\>conda update anaconda
Using Anaconda Cloud api site https://api.anaconda.org
Fetching package metadata: ....
```

it should end with displaying the following information:

```
anaconda                  4.0.0               np110py27_0
```

Check the proper installation with:

```
C:\>python
Python 2.7.11 |Anaconda 4.0.0 (64-bit)| (default, Feb 16 2016, 09:58:36) [MSC
v.1500 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://anaconda.org
>>>
```

You can skip the next section and continue with the setup of SEN2COR.

3.1.1.2 Anaconda installation from scratch

If you are new to Anaconda, then follow the steps below:

Download the recent version of the Anaconda python distribution for your operating system from:

http://continuum.io/downloads

and install it according to the default recommendations of the anaconda installer, which can be found here:

http://docs.continuum.io/anaconda/install.html

If you are working under Linux you should add the anaconda binary directory to your PATH environment variable at the end of the installation process. For the other installations (Windows and Mac) you will be asked if you want to allow this modification automatically. Please read the Anaconda installation procedure for your platform carefully before performing the installation.

If you already work with a different python interpreter outside of this new setup, you should add the PATH information for anaconda/bin to your environment only temporary. This is done as follows:

```
export PATH=<your installation directory>/anaconda/bin:$PATH
```

If you have no other python installed, you can add this line permanently, either to your .bashrc or .profile script in your home directory.

make sure that the the anaconda/bin directory is finally added to your PATH and check the proper installation by typing `conda -V`. If the command answers with the display of its version number, the installation was successful (the version number can be different). Example:

```
$: conda –V

conda 4.0.5
```

If this command can be found, you are ready for the second step, the installation of the processor. Continue with the next section.

The installation of the runtime environment is a one pass process. For subsequent installations of Sen2Cor you do not have to repeat the anaconda installation.

3.1.2 Sen2Cor Installation

For Windows:

Download the archive:

http://s2tbx.telespazio-vega.de/sen2cor/sen2cor-2.2.0.zip

and extract it with an unzip utility.

Step into the folder ***sen2cor-2.2.0***, type "***python setup.py install***" and follow the instructions. The setup will install the Sen2Cor application and all its dependencies under the Anaconda python distribution.

At the end of the installation you can select your home directory for the Sen2Cor configuration data. This is by default:

"***C:\Users\<your user account>\documents\sen2cor-<version>***"

The setup script generates the following three environment variables:

SEN2COR_HOME: this is the directory where the user configuration data are stored (see above). This can be changed later by you in setting the environment variable to a different location.

SEN2COR_BIN: this is a pointer to the installation of the Sen2Cor package. This is located in the "site-packages" folder of Anaconda. **Do not change this.**

GDAL_DATA: this is a pointer to the directory where the GDAL coordinate system info is stored. This is located in the ***"sen2cor\cfg\gdal_data***" folder of the SEN2COR site_package installation inside of Anaconda. **Do not change this.**

Open a new command line window, to be secure that your new environment settings are updated. From this new command line window, perform the following test:

Call the processor via "L2A_Process --help". This will give you a list of possible further options.

If no errors are displayed, your installation was successful.

Data sets can be downloaded from the Sentinels Data Hub located under:

https://scihub.copernicus.eu/

Problems **with long pathnames on Windows:** windows had a path length limitation of 260 characters in the past. Unfortunately, Windows Explorer on Windows 7 x64 is still subject to this path limit. As a consequence, the very long pathnames of a typical Level-1C or Level-2A user product can sometimes not be handled correctly.

Since release 2.10 sen2cor is adapted to handle long path names on windows correctly. However there might still be problems that generated Level 2A User products (as well as the L1C input folders) cannot be removed entirely using the Windows explorer. If this is the case, the path names must be manually shortened to a length below 260 characters before removing can take place. **Please note - this is a windows limitation and not in any case related to sen2cor.**

For Linux and Mac:

Download the archive:

http://s2tbx.telespazio-vega.de/sen2cor/sen2cor-2.2.0.tar.gz

and extract it with ***tar –xvzf sen2cor-2.2.0.tar.gz***.

Step into the folder ***sen2cor-2.2.0***, type ***python setup.py install*** and follow the instructions. The setup will install the Sen2Cor application and all its dependencies under the Anaconda python distribution.

At the end of the installation you can select your home directory for the Sen2Cor configuration data. By default this is the directory where your ***$HOME*** environment variable points to.

The setup script generates a script called ***L2A_Bashrc*** and places it into your home directory. It contains the following three environment variables:

SEN2COR_HOME: this is the directory where the user configuration data are stored (see above). This can be changed later by you in setting the environment variable to a different location.

SEN2COR_BIN: this is a pointer to the installation of the Sen2Cor package. This is located in the "site-packages" folder of Anaconda. Do not change this.

GDAL_DATA: this is a pointer to the directory where the GDAL coordinate system info is stored. This is located in the ***sen2cor/cfg/gdal_data*** folder inside the site_packages folders of Anaconda. Do not change this.

These settings are necessary for the execution of the processor. There are two possibilities how you can finish the setup:

You can call this script automatically via your ***.bashrc or .profile*** script (OS dependent). For this purpose, add the line:

source <location of your script>/L2A_Bashrc

to your script.

You can call this script also manually via ***source L2A_Bashrc*** every time before starting the processor or include it into a script of your choice.

Finally, to check the installation after sourcing the ***L2A_Bashrc***, call the processor via ***L2A_Process --help***. This will give you a list of possible further options how to operate.

3.1.3 Configuration Files

Located under ***$SEN2COR_HOME/cfg*** the GIPP file of the application (which is an xml formatted file) can be configured by the user for individual purposes. If a different configuration shall be used,'***$SEN2COR_HOME*** directory can be reconfigured to a different directory:

export SEN2COR_HOME' = <directory of your choice>

This allows the operation with multiple configuration settings.

3.2 Operation

The processor can be operated in two different ways:

- either as a purely command line driven application;

- or from the Sentinel-2 Toolbox
- as a command line driven application inside the PDGS

3.2.1 Command line interpreter

Calling the script L2A_Process with the option '-h' via command line displays the following menu:

```
W:\testdata>L2A_Process --help
usage: L2A_Process-script.py [-h] [--resolution {10,20,60}] [--sc_only]
                             directory

Sentinel-2 Level 2A Prototype Processor (Sen2Cor). Version: 2.2.0, created:
2016.02.08, supporting Level-1C product version: 13.1.

positional arguments:
  directory             Directory where the Level-1C input files are located

optional arguments:
  -h, --help            show this help message and exit
  --resolution {10,20,60}
                        Target resolution, must be 10, 20 or 60 [m]
  --sc_only             Performs only the scene classification at 60m
                        resolution
```

The ***<directory>*** argument can be either a relative or an absolute pathname.

If a relative pathname is given, it is expected that the user is calling sen2cor from inside a parent directory. Sen2cor will expand the absolute pathname for that directory.

The pathname can either point to a L1C user product or to a single tile of the user product.

If the argument points to a user product, all tiles of the user product will be processed subsequently. If the argument points to a single tile, only that tile will be processed. In the latter case, the configuration settings for parallel processing will be ignored.

Sen2cor will use the L1C user product identifier for generating a subsequent L2A product. For this purpose, the L1C source directory must start with an identifier like **'S2A_????_???_???L1C*'** which is the standard, if you download a L1C user product from the Sentinel 2 data hub. The generated product will get the identifier **'S2A_????_???_???L2A*'**, everything else will be inherited from the L1C source.

The output directory for the L2A data can be configured in the L2A_GIPP.xml configuration file. Here, the user can specify the target directory where the L2A user product will be created. By default, the processor will create it in the same directory where the L1C user product is created, but replacing "L1C_" with "L2A_".

--resolution is the target resolution for the product to be processed. See section 2.4.1 for details. If resolution is omitted, the product will be processed with the lowest resolution, which is 60 m.

On L1C User Product Level:

Absolute path:

```
L2A_Process
</the_L1C_product_directory>/S2A_OPER_PRD_MSIL1C_PDMC_20150630T084936_R06
2_V20150627T103515_20150627T103515.SAFE --resolution=60
```

Relative Path (the command must be called from inside the user product directory):

```
L2A_Process
S2A_OPER_PRD_MSIL1C_PDMC_20150630T084936_R062_V20150627T103515_20150627T1
03515.SAFE --resolution=60
```

On L1C Tile Level:

Absolute path:

```
L2A_Process
</the_L1C_product_directory>/S2A_OPER_PRD_MSIL1C_PDMC_20150630T084936_R06
2_V20150627T103515_20150627T103515.SAFE/GRANULE/S2A_OPER_MSI_L1C_TL_MTI__
20150627T180307_A000062_T32TNR_N01.00 -resolution=20
```

Relative Path (the command must be called from inside the GRANULE folder):

```
L2A_Process
S2A_OPER_MSI_L1C_TL_MTI__20150627T180307_A000062_T32TNR_N01.00 --
resolution=20
```

3.2.2 Integration into the Sentinel-2 Toolbox

A detailed instruction the installation of sen2cor under the umbrella of the Sentinel-2 Toolbox is given under:

https://www.youtube.com/channel/UCPnL3aynCQxTOjPttxMiS3Q

3.3 The Software Development Environment

This chapter describes all necessary steps to install, configure and operate the necessary tools in order to extend or maintain the prototype processor software. The content of this chapter is normally not needed by normal users of the application.

3.3.1 Requirements and Third Party Software

The Sen2Cor development distribution contains an archive of essential open source tools, which provide the development environment for the processor. The following table specifies a comprehensive overview on all required items, including the runtime environment. The IDL runtime environment is only needed in that case, that a 1:1 testing is required with the initial ATCOR code, as this is written in IDL. The original ATCOR IDL code itself is not part of the development distribution and can only be requested at ESA by maintainers of the S2PAD project.

Table 3-1: Third Party products for the SDE

Operating System			
Product	Name	Version	Source
Linux (alternatives)	CentOS RHEL Ubuntu	>= 6.0 6.0 12.04.2 LTS	http://www.centos.org/ http://www.de.redhat.com/products/rhel/desktop/ http://www.ubuntu.com/download/server
Software Development Environment			
Product	Name	Version	Source
IDE	Eclipse Classic	>= 3.7.1	http://www.eclipse.org/downloads/packages/
Python IDE for Eclipse	PyDev	>= 2.5.0	http://pydev.org/
XML Editor for Eclipse	Rinzo	1.4.0	http://editorxml.sourceforge.net/
Operational Tools and Libraries (see run_time_environment)			
Product	Name	Version	Source
Python Distribution Package (including Python 7.2, NumPy + SciPy, PyTables, Cython, Distutils and more …	Anaconda	>= 4.0.0	https://store.continuum.io

3.3.1.1 Eclipse, PyDev and Rinzo

Eclipse Classic is used and distributed as the standard development environment for the processor. Together with PyDev, it provides a complete development environment, including editor, syntax checker, debugger and runtime environment for python. Rinzo is a plugin for the syntax check of XML files, which are created and used as the standard I/O data format for the processor. For installation, it is referred to the installation procedures of the different tools as can be found in the references given in Table 3-1 above.

3.3.1.2 Anaconda

The Sen2Cor application uses a huge amount of python libraries, where “numpy” and “scipy” are only the most prominent ones. One requirement of this project was, to keep the installation overhead as low as possible. Thus, it was decided to use a free python distribution. We decided for Anaconda after several tests with different Python distributions. Anaconda is a completely free enterprise-ready Python distribution for large-scale data processing, predictive analytics, and scientific computing. Anaconda comes with its own installers, which facilitates the setup of the development environment considerably. Anaconda is able to support other operating systems (Mac OS and Windows) and thus avoiding the common problem of version mismatches between the different operating systems.

3.3.1.3 Cython

Cython is a python compatible language, which enables to convert native python code into C, so that the code can be compiled into a shared library and can be executed without exposing the readable python code. This technique is used here in order to protect the licensed ATCOR algorithm in the runtime distribution from being read and analysed. Cython is an integrated part of the Anaconda distribution described above.

3.3.1.4 Distutils

The Distutils are the Python standard for producing distribution packages. It is used here for generating both, the development and the runtime installation package. Distutils is an integrated part of the Anaconda distribution described above.

3.3.1.5 GDAL

GDAL is a translator library for raster geospatial data formats that is released under an X/MIT style Open Source license by the Open Source Geospatial Foundation. As a library, it presents a single abstract data model to the calling application for all supported formats. It also comes with a variety of useful commandline utilities for data translation and processing, which is used here (among others) for preparing the Digital Elevation Map.

3.3.1.6 PyTables

PyTables is a package for managing hierarchical datasets and designed to efficiently and easily cope with large amounts of data.

PyTables is built on top of the HDF5 library, using the Python language and the NumPy package. It features an object-oriented interface that, combined with C extensions for the performance-critical parts of the code (generated using Cython), makes it a fast, yet extremely easy to use tool for interactively browse, process and search very large amounts of data. One important feature of PyTables is that it optimizes memory and disk resources so that data takes much less space (specially if on-flight compression is used) than other solutions such as relational or object oriented databases. PyTables is an integrated part of the Anaconda distribution described above.

3.3.2 Installation

The software development distribution (SDD) is currently only provided for Linux. Baseline is CentOS 6.0, resp. RHEL 6.0. It is in principle possible to setup the complete SDE for other Linux Distributions or other operating systems such as Mac OSX or Windows. However, the SDD is not tested extensively on these platforms, and thus not recommended. The SDD is provided in form of the original archives, to be found in the project subdirectory labelled ‘development_environment’.

For the installation of the development environment two different installation scenarios are possible:

Global installation (default): Eclipse will be installed within a common directory like e.g. ***/opt*** on the target system. For this installation root access on the target system is needed.

Local installation (optional): Eclipse will be installed under a certain user account (e.g. /home/s2l2app on the target system. For this installation a corresponding user account must exist and the installation shall be performed under that account.

Although Eclipse and PyDev are part of the distribution packages, it is recommended to install these third party tools from the original software providers, following the corresponding installation steps as given in the following links:

- Eclipse: http://wiki.eclipse.org/Eclipse/Installation#Install_a_JVM
- PyDev: https://wiki.appcelerator.org/display/tis/PyDev+Install

3.3.3 Configuration

3.3.3.1 Configure Python

In order to select Anaconda as the Python Interpreter of your choice, within Eclipse, first click on ***/Project/Properties/PyDev – Interpreter***, second, click on ***Configure and Interpreter not listed.***

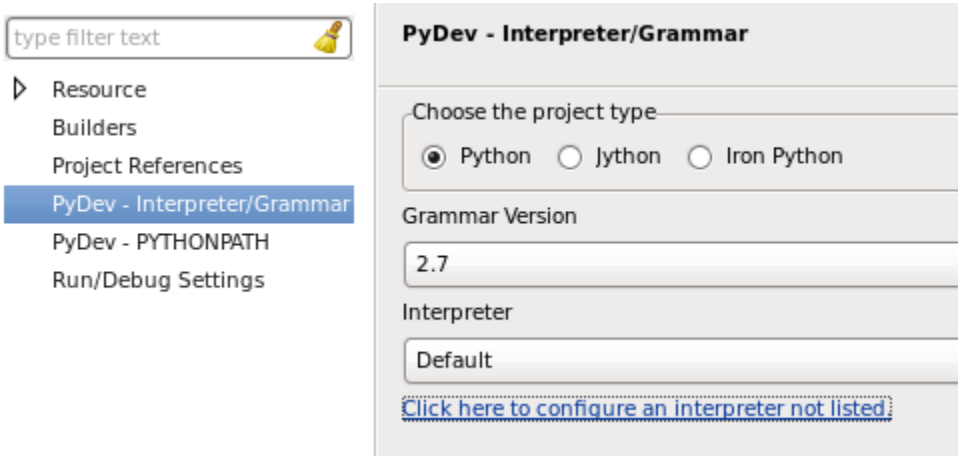

Figure 3-1 – Configuration of Python interpreter within Eclipse I

Then click on ***New*** and select the Python Interpreter in the folder where Anaconda was installed. An example installation is shown below:

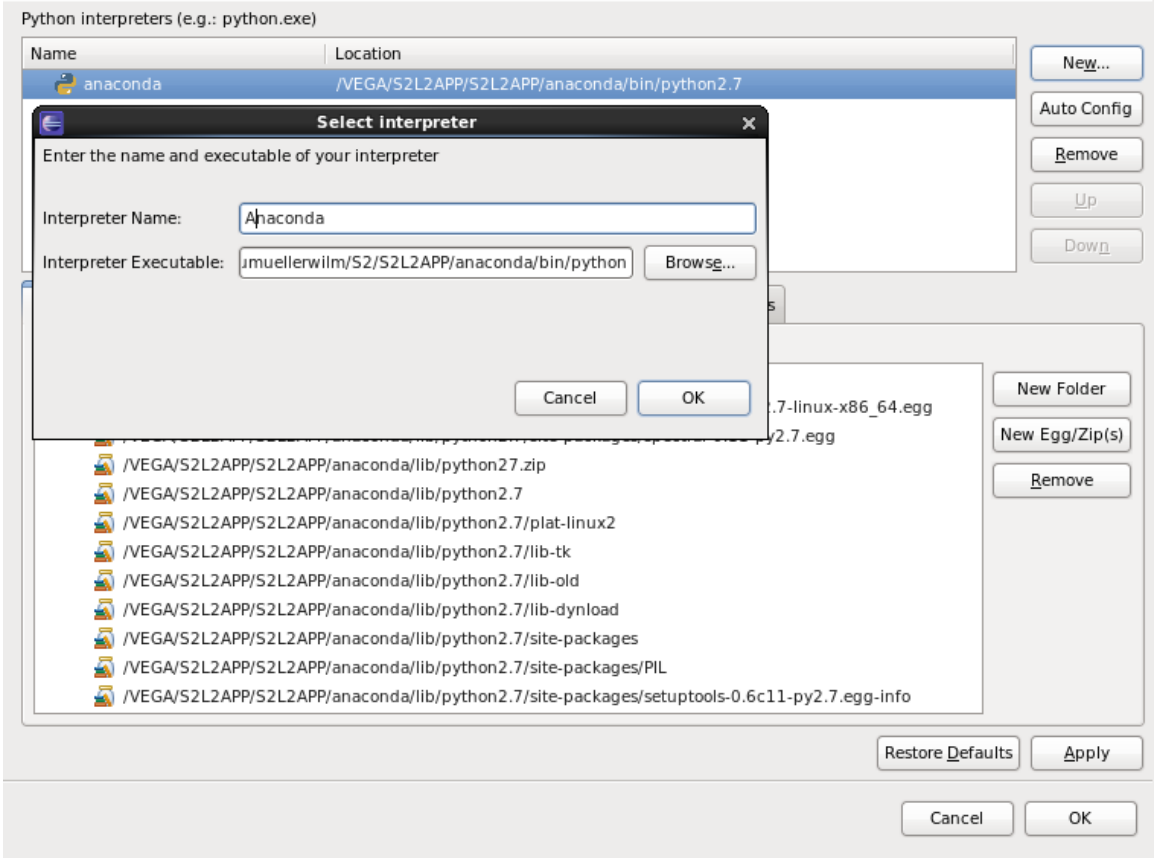

Figure 3-2 – Configuration of Python interpreter within Eclipse II

3.3.3.2 Environment Settings

Two environment variables are necessary to be set up in order to obtain a full configured system:

$S2L2APPHOME: this environment variable named is used to configure the root directory of the Sen2Cor application package. Example: if the Sen2Cor application package is installed ***under /home/s2l2app***, ***$S2L2APPHOME*** shall be assigned to:

export S2L2APPHOME = /home/s2l2app

The settings depend on your local setup.

A second, optional environment variable can be set up in order to change the location of the configuration directory. By default this is located under ***$S2L2APPHOME/cfg*** but can be overwritten by setting:

S2L2APPCFG = <directory of your choice>

This allows the operation with multiple configuration settings or the usage of the processor within different environments. See also section **Error! Reference source not found.** ff. on this issue.

These two environment variables can be configured inside of the Eclipse workspace via ***/Project/Properties/PyDev – Interpreter***, as shown below:

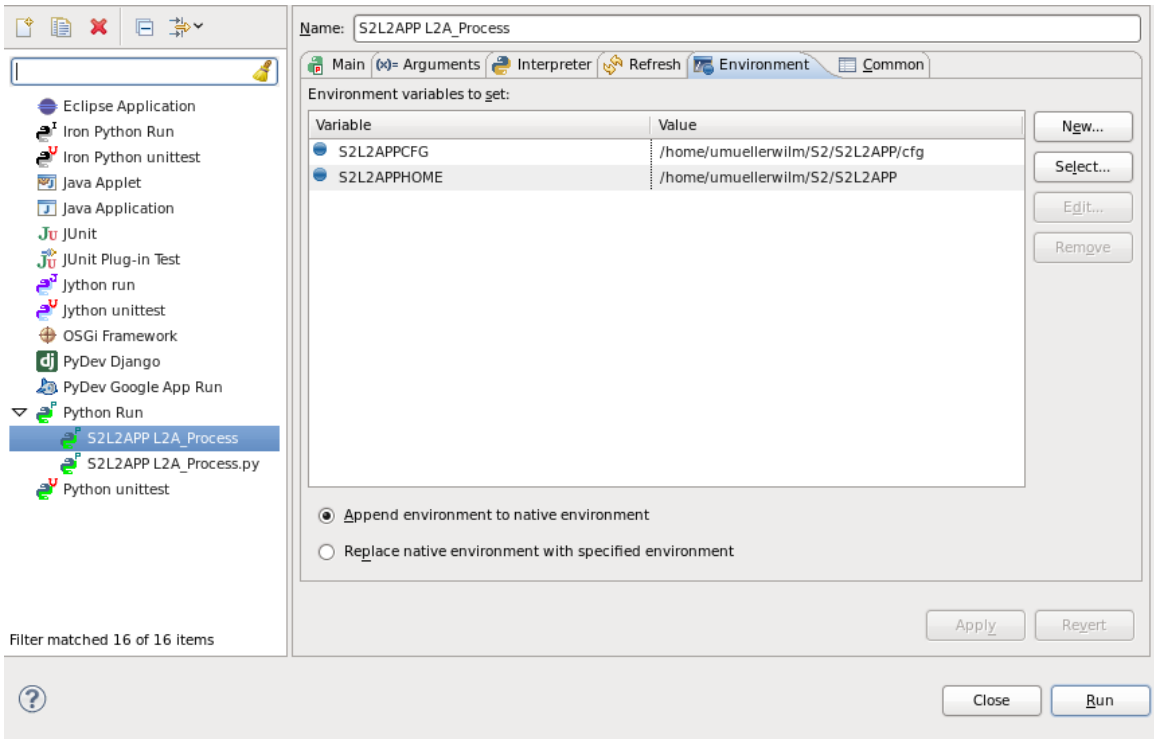

Figure 3-3 – Environment setting via Eclipse

Details on the RTE configuration are given in section **Error! Reference source not found.**.

3.3.4 Operation

3.3.4.1 Running the processor within Eclipse

Within Eclipse, create a new project using the root of your project (*$S2L2APPHOME*)

Within Eclipse perform a refresh. The directory will show your project.

The main() operation of the Sen2Cor is located in the L2A_Process module. For testing purposes within Eclipse the whole application can be operated in two different ways:

1. Runtime Mode: */Run/Run*
2. Debug Mode*: /Run/Debug*

In order to simulate a scenario for testing purposes a command line can be configured within Eclipse via:

/Run/Run Configurations

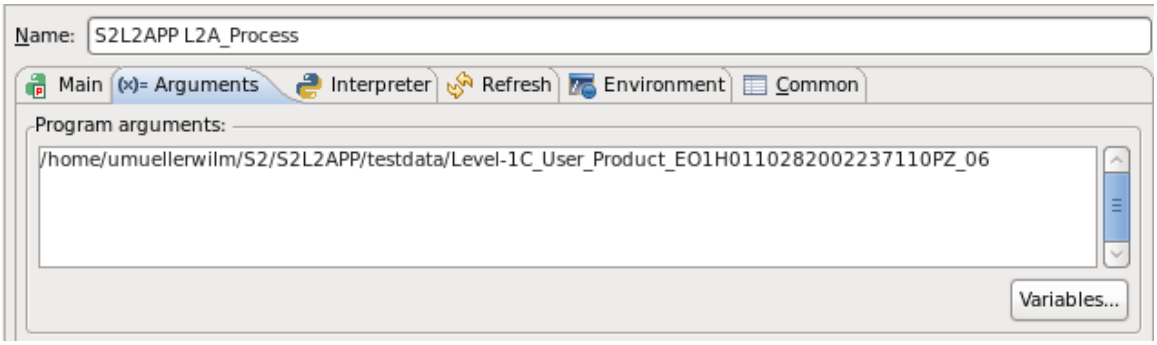

Figure 3-4 – SCD command line arguments

All other operational issues are equivalent with the scenario described for section 3.2.1.

3.3.4.2 Generating a Source Distribution

From the Eclipse workspace or via command line shell execute:

python setup.py s2l2app_SCD.py

This will generate an archive of the form:

s2l2app_SCD_<VERSION>.tar.gz

which then can be used for redistribution. The output will be generated in the distribution folder of the SDE.

Please be aware that the source redistribution is only allowed to be delivered to ESA itself or such ESA contractors, being in charge for the development or maintenance of the processor. In case of doubts, contact the corresponding Technical Officer of the project for clarification.

3.3.4.3 Generating a Build Distribution

The source code of the original ATCOR software is protected by a license agreement (see appendix). As a consequence, the Python source code of the processor core algorithm cannot be distributed to the end user. Thus, Cython is used as an intermediate development step. Cython is is able to generate C source code from python code. The generated C code can then be compiled platform specific on each target platform and can finally be redistributed in a non-human readable shared object or dynamic link library. Following target platforms are currently tested and will be supported:

Linux x86, 64 bit (CENTOS 6.0, alternatively RHEL 6.0)

MacOSX, x86, 64 (Mountain Lion)

Windows, 64 bit (Tested on Windows 7 Enterprise)

Generation of a build distribution therefore is performed in two steps:

Compilation of python source file into a target specific runtime library. This is done via the command ***python setup.py build_ext*** and must be executed on the target platform for which the distribution package shall be created.

Finally, the platform specific library together with all provided sources, test data, configuration and documentation is archived into platform specific distribution packages, which are named:

Sen2Cor_RTD-x.y.z-CentOS-x86_64.tar.gz

Sen2Cor_RTD-x.y.z-MacOSX-x86_64.tar.gz

Sen2Cor_RTD-x.y.z-Windows-x86_64.tar.gz

The generated packages can then be burned and delivered via a distribution medium.

4. REFERENCES

1. Richter, R., Wang, X., Bachmann, M. and Schlaepfer, D. (2011). Correction of cirrus effects in Sentinel-2 type of imagery. *International Journal of Remote Sensing,* **32***,* 2931-2941.

2. Louis, J., A. Charantonis, A. and Berthelot, B (2010). Cloud Detection for Sentinel-2. *Proceedings of ESA Living Planet Symposium.*

3. Salomonson V.V., I. Appel I. (2004). Estimating fractional snow cover from MODIS using the normalized difference snow index. *Remote Sensing of Environment* **89**, 351–360.

4. Vane, G., Green, R. O., Chrien, T. G., Enmark, H. T., Hansen, E. G., and Porter, W. M. (1993). The airborne visible / infrared imaging spectrometer (AVIRIS). *Remote Sens. Environ.* **44**, 127-143.

5. Kaufman, Y., et al. (1997). The MODIS 2.1 µm channel – correlation with visible reflectance for use in remote sensing of aerosol. *IEEE TGRS,* **35,** 1286 – 1298.

6. Schläpfer, D. et al. (1998). Atmospheric precorrected differential absorption technique to retrieve columnar water vapour. *Remote Sens. Environ.,* **65**, 353-366.

5. APPENDIX

5.1 Licenses

Anaconda:

http://docs.continuum.io/anaconda/eula.html

GDAL/OGR Licensing

==========

GDAL/OGR General

GDAL/OGR is licensed under an MIT/X style license with the following terms:

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included

in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

5.2 Example L2A_GIPP.xml

```
<?xml version="1.0"  encoding="UTF-8"?>
<Level-2A_Ground_Image_Processing_Parameter
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="L2A_GIPP.xsd">
  <Common_Section>
    <Log_Level>INFO</Log_Level>
    <!-- can be: NOTSET, DEBUG, INFO, WARNING, ERROR, CRITICAL -->
    <Nr_Processes>1</Nr_Processes>
    <!-- can be an unsigned integer value specifying the number or
processes you intend to operate in parallel or: AUTO. If AUTO is chosen,
the processor determines the number of processes automatically, using
cpu_count() -->
    <Target_Directory>DEFAULT</Target_Directory>
    <!-- should be either a directory or 'DEFAULT'. If default, target
will be created at root of L1C product -->
    <DEM_Directory>NONE</DEM_Directory>
```

```
    <!-- should be either a directory in the sen2cor home folder or
'NONE'. If NONE, no DEM will be used -->
    <DEM_Reference>http://data_public:GDdci@data.cgiar-
csi.org/srtm/tiles/GeoTIFF/</DEM_Reference>
    <!-- will be ignored if DEM is NONE. A SRTM DEM will be downloaded
from this reference, if no local DEM is available -->
    <UP_Scheme_1C>S2-PDGS-TAS-DI-PSD-V13.1_Schema/S2_User_Product_Level-
1C_Metadata.xsd</UP_Scheme_1C>
    <UP_Scheme_2A>S2-PDGS-TAS-DI-PSD-V13.1_Schema/S2_User_Product_Level-
2A_Metadata.xsd</UP_Scheme_2A>
    <Tile_Scheme_1C>S2-PDGS-TAS-DI-PSD-V13.1_Schema/S2_PDI_Level-
1C_Tile_Metadata.xsd</Tile_Scheme_1C>
    <Tile_Scheme_2A>S2-PDGS-TAS-DI-PSD-V13.1_Schema/S2_PDI_Level-
2A_Tile_Metadata.xsd</Tile_Scheme_2A>
    <DS_Scheme_1C>S2-PDGS-TAS-DI-PSD-V13.1_Schema/S2_PDI_Level-
1C_Datastrip_Metadata.xsd</DS_Scheme_1C>
    <DS_Scheme_2A>S2-PDGS-TAS-DI-PSD-V13.1_Schema/S2_PDI_Level-
2A_Datastrip_Metadata.xsd</DS_Scheme_2A>
    <GIPP_Scheme>L2A_GIPP.xsd</GIPP_Scheme>
    <SC_Scheme>L2A_CAL_SC_GIPP.xsd</SC_Scheme>
    <AC_Scheme>L2A_CAL_AC_GIPP.xsd</AC_Scheme>
  </Common_Section>
  <Scene_Classification>
    <Filters>
      <Median_Filter>0</Median_Filter>
    </Filters>
  </Scene_Classification>
  <Atmospheric_Correction>
    <Look_Up_Tables>
      <Aerosol_Type>RURAL</Aerosol_Type>
      <!-- RURAL, MARITIME, AUTO -->
      <Mid_Latitude>SUMMER</Mid_Latitude>
      <!-- SUMMER, WINTER, AUTO -->
      <Ozone_Content>h</Ozone_Content>
      <!-- 0, f-k, t-y -->
      <!-- The atmospheric temperature profile and ozone content:
       "0" means: get best approximation from metadata (this is smallest
difference between metadata and column DU)

        For midlatitude summer atmosphere:
        "f" 250 DU
        "g" 290 DU
        "h" 331 DU (standard MS)
        "i" 370 DU
        "j" 410 DU
        "k" 450 DU

        For midlatitude winter atmosphere:
        "t" 250 DU
        "u" 290 DU
        "v" 330 DU
        "w" 377 DU (standard MW)
        "x" 420 DU
        "y" 460 DU
       -->
    </Look_Up_Tables>
    <Flags>
      <WV_Correction>1</WV_Correction>
```

```
            <!-- 0: No WV correction, 1: only 940 nm bands, 2: only 1130 nm
        bands , 3: both regions used during wv retrieval, 4: Thermal region -->
            <VIS_Update_Mode>1</VIS_Update_Mode>
            <!-- 0: constant, 1: variable visibility -->
            <WV_Watermask>1</WV_Watermask>
            <!-- 0: not replaced, 1: land-average, 2: line-average -->
            <Cirrus_Correction>0</Cirrus_Correction>
            <!-- 0: no, 1: yes -->
            <BRDF_Correction>0</BRDF_Correction>
            <!-- 0: no BRDF correction, 1: , 2: ,11, 12, 22, 21: -->
            <BRDF_Lower_Bound>0.22</BRDF_Lower_Bound>
            <!-- In most cases, g=0.2 to 0.25 is adequate, in extreme cases of
        overcorrection g=0.1 should be applied -->
          </Flags>
          <Calibration>
            <DEM_Unit>0</DEM_Unit>
            <!-- (0=[m], 1=[dm], 2=[cm]) -->
            <Adj_Km>1.000</Adj_Km>
            <!-- Adjancency Range [km] -->
            <Visibility>23.0</Visibility>
            <!-- visibility (5 <= visib <= 120 km) -->
            <Altitude>0.100</Altitude>
            <!-- [km] -->
            <Smooth_WV_Map>100.0</Smooth_WV_Map>
            <!-- length of square box, [meters] -->
            <WV_Threshold_Cirrus>0.25</WV_Threshold_Cirrus>
            <!--  water  vapor  threshold  to  switch  off  cirrus  algorithm
        [cm]Range: 0.1-1.0 -->
          </Calibration>
        </Atmospheric_Correction>
</Level-2A_Ground_Image_Processing_Parameter>
```