Final Report for the Study Project
Copernicus Earth Observation Infrastructure II

Group B – Detection of Surface Water from Sentinel-1 Data

Summer Semester 2018

Akhil Patil, Raphael Witt, Shenaha Sivakumar, Christopher Rohtermundt

Institute for Geoinformatics

Introduction

Copernicus - “Europe’s eyes on Earth” is a European Union (EU) Programme which aims to develop information services based on satellite Earth Observation as well as on in situ data sources. The Sentinels are new fleet of ESA satellites that are delivering vast amounts of global data, freely and openly, accessible to all of its users and imagery that are central to Europe’s Copernicus programme. By offering a set of key information services for a broad range of applications, this global monitoring programme makes a steep positive change in the way we manage our environment, understand and tackle the effects of climate change, and safeguard everyday lives. Sentinel-1 is the first in the series and it carries a 12 m-long advanced synthetic aperture radar (SAR), working in C-band. Due to the penetration capacity of Synthetic Aperture Radar (SAR) data through clouds and hazy atmospheric circumstances like fog, smog, light rain, mist etc., it has ability to continuous observation of flood events for producing accurate, rapid and cost effective flood mapping. This feature is of significant importance to the choice of Sentinel-1 image for this project. Even during search and rescue missions in poor weather conditions. SAR looks specular reflection in the flooded areas (dark pixels) [2].

Sentinel-1 is a phase-preserving dual polarization SAR system. It can transmit and receive signals in both horizontal (H) and vertical (V) polarization. We tried to bring this feature to use in this project combining both the bands in processing the satellite image [1]. Based on the information provided by ESA , the commonly used polarization schemes are (1) HH-HV or HH polarization for the monitoring of polar environments, sea-ice zones and (2) VV-VH or VV polarization for all other observation zones (with an exception for the Baltic Sea observed partially in HH-HV during northern winter, on descending orbits) [3].

As a constellation of two satellites orbiting 180° apart, the mission images the entire Earth every six days benefiting services like monitoring of the Arctic sea-ice extent, routine sea-ice mapping, surveillance of the marine environment, monitoring land-surface for motion risks, water and soil management and mapping to support humanitarian aid and crisis situations. For this project we are focusing on Sentinel-1A and Sentinel-1B which were launched on 3rd April 2014 and 25th

April 2016, respectively [4]. The Copernicus Earth Observation Infrastructure II project in the summer semester 2018 was based on the work done in the first installment of the Copernicus EOI project carried out in the summer semester 2017. This year the Sentinel group (B) is focused on the work done by the Group B from last year which included tasks like preprocessing of Sentinel-1 data and working on water detection. The group will also work on some part done by the Ingestion group (A) from last year.

Aim

The aim of group B was to develop an automated detection of water bodies using real time Sentinel-1 images within a cloud infrastructure (Open Telekom Cloud infrastructure). It builds on top of last year's project where students implemented this workflow on AWS (Amazon Web Services). Technically, the main goal was to shift the working infrastructure (searching, downloading, ingesting, pre-processing, visualizing) from AWS to the OTC platform.

Team & Task distribution

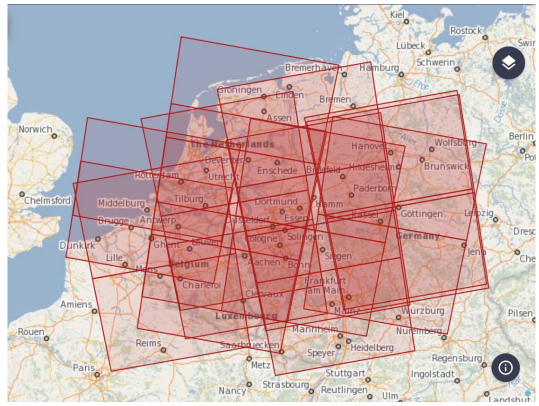
For this year's project the full study-project team consisted of 7 students out of which we 4 students were assigned to the Sentinel data processing. As the volume of task to be done by this group was big, we decided to divide the group and assign each task to a pair. The idea behind this was to have two people working on one topic. This would enhance the speed of work and would overcome any obstacle that generally occurs when one person works on a topic and gets stuck. Also an important benefit was in case of availability of a person of the group due to any unavoidable reason, then the partner would be present to carry the work forward. This way we made sure that the project activities never get halted.

In the early stages, activities involving document reading, getting hands-on and trying out mockups was assigned per person. But here as well, related things were designated to a pair with the reason that they will share their learnings and difficulties in the pair.

The raw sentinel images should be preprocessed using SNAP functions and further ISO-clustering classification performed on the pre-processed images. The result should be displayed with the help of an ArcGIS Image Server. The processes should get automated using scheduler tasks within the OTC environment.

Study Areas and Datasets

Sentinel-1 images from North Rhine-Westphalia (NRW) region are chosen for the project. The Sentinel-1 imagery include pre-event, during-event, or post-event images collected from the European Space Agency (ESA) Sentinels Scientific Data Hub. Level-1 GRD, Interferometric Wide (IW) swath mode products are being downloaded. Each image has a spatial resolution of about 20 × 22 m with double polarization (VV and VH).



https://scihub.copernicus.eu/dhus/#/home

https://www.kfz.net/zulassungen/nordrhein-westfalen/

Fig. 1 NRW region; Possible Sentinel-1 images filtered for NRW.

Architecture

ArcGIS stack installation

The ArcGIS Enterprise stack, consisting out of ArcGIS Server, Portal for ArcGIS, ArcGIS Datastore and the ArcGIS Web Adaptor, had to be installed in the OTC environment.

Fig. 2 ArcGIS Enterprise stack.

https://www.esri.com/

We bought the domain "www.copernicus-ssp.eu" which is serving as the entry point to the capabilities we are using of the ArcGIS stack. The masked water bodies should be published as mosaic datasets via ArcGIS Portal to the ArcGIS Server which has been authorized as an Image Server. The extracted water bodies could then be viewed using the possibilities within the portal or could be used as a WMS in a different web application. During the installation we encountered several problems, some of them have been very time consuming (e.g. authorization of ArcGIS portal to our domain).

In this project we could not implement the visualization of the automatically created output due to time constraints. See “Future work” for more information regarding this.

Automation process

The automatic process to detect water bodies from Sentinel 1 images is divided into four steps. They are explained more detailed in the following paragraphs.

Step 0: OTC setup

The complete OTC setup created is given in figure 3 below.

For the purpose of keeping track of the products/scenes downloaded from Sci-hub/Code-DE repository, we created a network of two Distributed Message Services (DMS) queues. Each queue served a specific purpose and were named as SP-idavailable and SP-iddownloaded. For both the queues every individual message holds the id of a Sentinel-1 product.

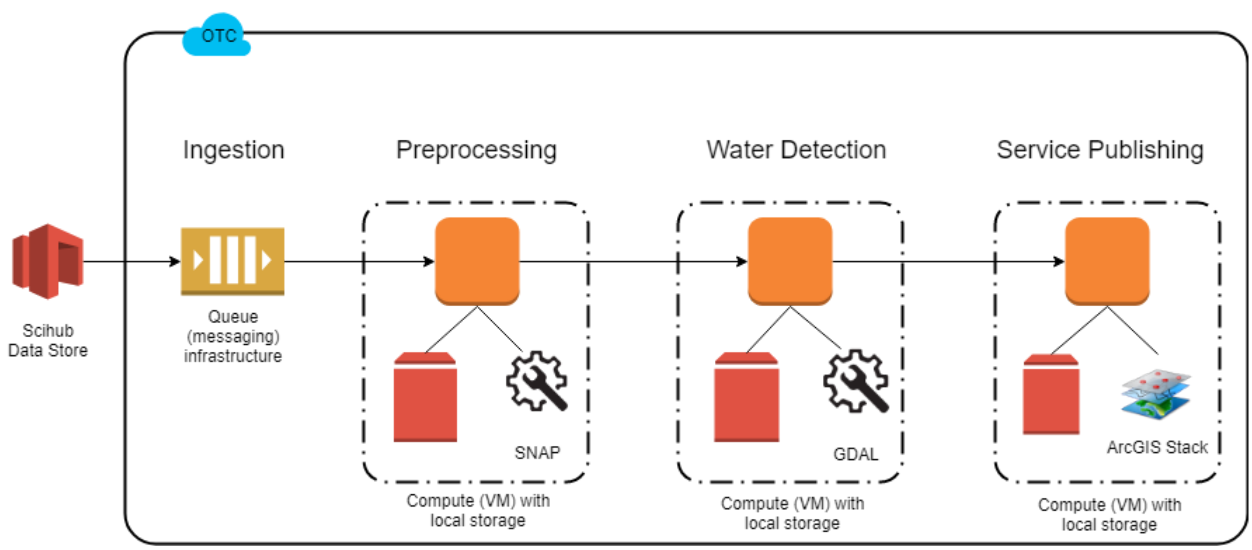


Fig. 3 System architecture (with OCT configuration).

We used a compute/VM named as ‘SP-ecs-arcserver’ (specifications: ECS type - General-purpose | Flavor - s2.xlarge.8 | vCPUs - 4 | RAM - 32 GB) for execution of all the python scripts and installation of the softwares (like SNAP, ArcGIS Enterprise stack, ArcGIS Pro) needed for this project. Due to time constraints we could not implement a connection of ECS with an Object Storage Service (OBS) unit. Instead we performed the operations on a single ECS and linked multiple Elastic Volume Service disks to it. The first disk was the default with 40 GB space and the additional disk of 200 GB was added later to cope up with the issue of limited disk space for storing downloaded products and final outputs. However, going forward the best option would be to create and connect an OBS instance that can hold greater amount of storage.

As explained in Fig. 3, the setup is prominently divided in 4 subdivisions. The DMS queue interacts with ESA Sci-hub for inputs and on the other side collaborates with the OTC compute.

As mentioned above, we use a single compute for all the activities. The three distinct representations of the compute in Fig. 3 is only for the sake of understanding and clarity.

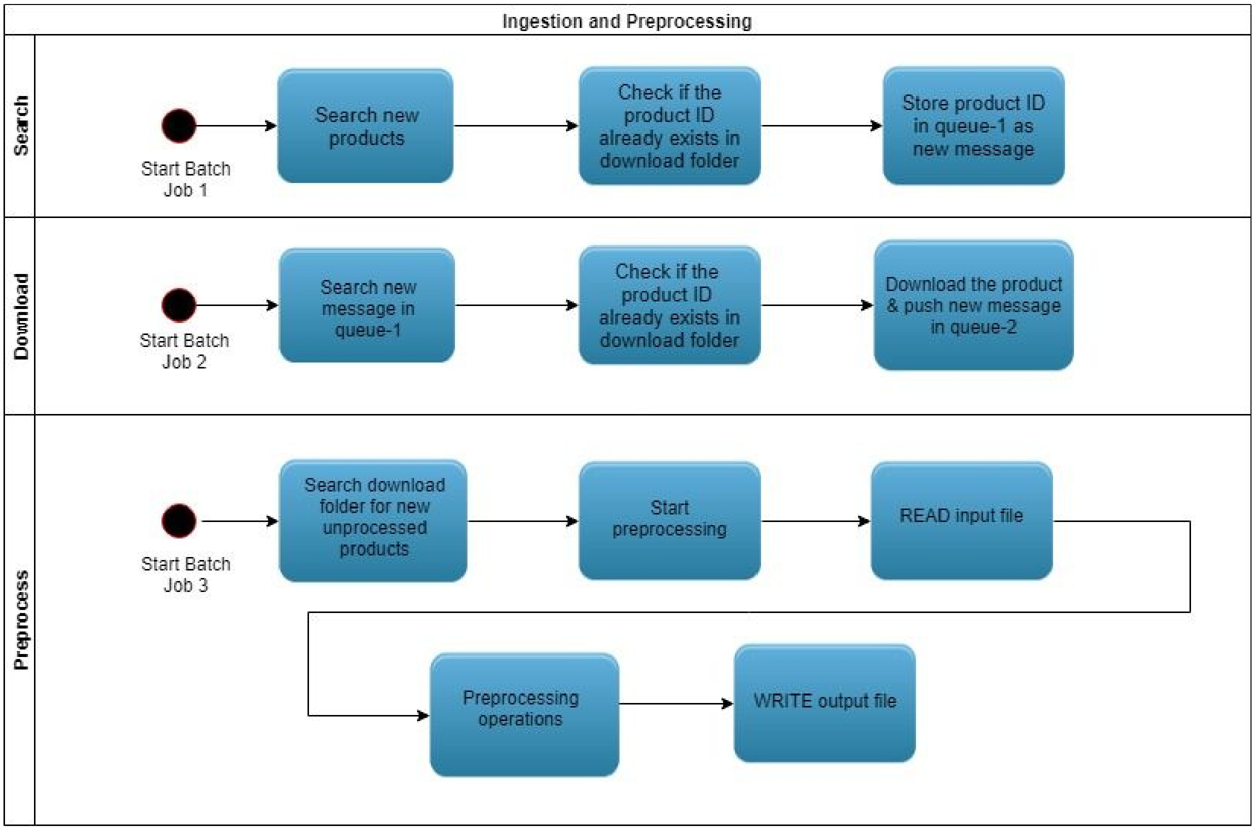


Fig. 4 Batch scheduler flow.

Step 1: Searching Sentinel 1 Data

To find Sentinel 1 images that are suitable for the use case they first have to be identified. This step is done using a python script that is querying against the Code-DE data hub to get product IDs of images that are covering North Rhine Westphalia. The script is called through a batch file which is executed at regular intervals of time by setting up of a Windows Task Scheduler “SP-TS-initiate”. If the product ID already exists in the download folder, it is not putting the ID into the queue. If new products are identified their ID is being sent to the queue “SP-idavailable” in the OTC.

Step 2: Ingesting Sentinel 1 Data

In this step the products are being downloaded in the VM where SNAP is installed. It is done this way since the final outputs are very small so no additional bucket for storing is needed (this can be adjusted). For this process a second batch file is executed with the help of a windows task scheduler “SP-TS-downloader”. A python script checks if the products of the queue “SP-idavailabe” already have been downloaded into a folder and if not it downloads the product. The IDs of products downloaded are then sent to the queue “SP-iddownloaded” in the form of a new message.

Step 3: Pre-processing Sentinel 1 Data

Pre-processing is being executed on satellite images for image restoration and rectification that are intended to correct for sensor- and platform-specific radiometric and geometric distortions of data. To run the Python script with pre-processing operations, we have created the schedular “SP-TS-preprocessing” which runs every 5 minutes and executes a batch file which in turn calls the pre-processing script. Radiometric correction is necessary due to variations in scene illumination and viewing geometry, atmospheric conditions, and sensor noise and response. Each of these will vary depending on the specific sensor and the platform used to acquire the data and the conditions during data acquisition. SAR has the advantage of working without solar illumination and in all weather conditions but it is affected by speckle, a multiplicative noise that gives the image a grainy appearance and makes it hard to interpret. For better interpretation we use preprocessing. The preprocessing procedure further enhances the image's appearance by removing speckles and transforms their raw coordinates to terrain coordinates which are related to a geodetic datum. To simplify mutual comparison of images, we ensure that centres of pixels are always at the same coordinates. The last pre-processing step consists of topographic corrections with a Shuttle Radar Topography Mission (SRTM) Digital Elevation Model (DEM). We correct the foreshortening and layover effects as well as different lighting of slopes. Applying water detection methods on a enhanced image will produce better results.

To carry out pre-processing we make use of SNAP (Sentinel Application Platform), SNAP GPT (Graph processing tools) and GDAL (Geospatial Data Abstraction Library). All data products downloaded are in the Sentinel Standard Archive Format for Europe (SAFE) which are read using SNAP. GPT is a module provided by the SNAP and allows creation of graphs that aimlessly execute several operations like read, write, geometric corrections, radiometric corrections , masking etc. Graphs are written as XML files and executed on the command shell using the GPT commands [5] [6] [7]. We use two SNAP .xml graphs, one for pre-processing and another for converting BEAM-DIMAP to GeoTIFF-BigTIFF. Combining the two graphs is more time consuming hence we use two separate graphs as you can seen in Fig 5.

GDAL Processing

After obtaining a .tif end product from SNAP graph (2) we further carry on GDAL processes using gdal_translate [9], gdaladdo [10] and gdalwarp [11]. One of the major inquietude with satellite images are the NO-DATA values that are present within the images even after preprocessing. These NO-DATA has very low backscatter values which is similar to the backscatter values of the water bodies. Hence, during classification NO-DATA get falsely identified as water which in turn will have a declining effect in the percentage of accuracy. In Order to avoid such instance we remove nodata with the help of Gdal_translate utility. GdalWarp utility is used to reproject cleaned image to web mercator projection. For the purpose of easy visualisation gdaladdo utility is used to build or rebuild overview images for supported file formats and also to produce the smallest possible JPEG-In-TIFF overviews (Fig.6).

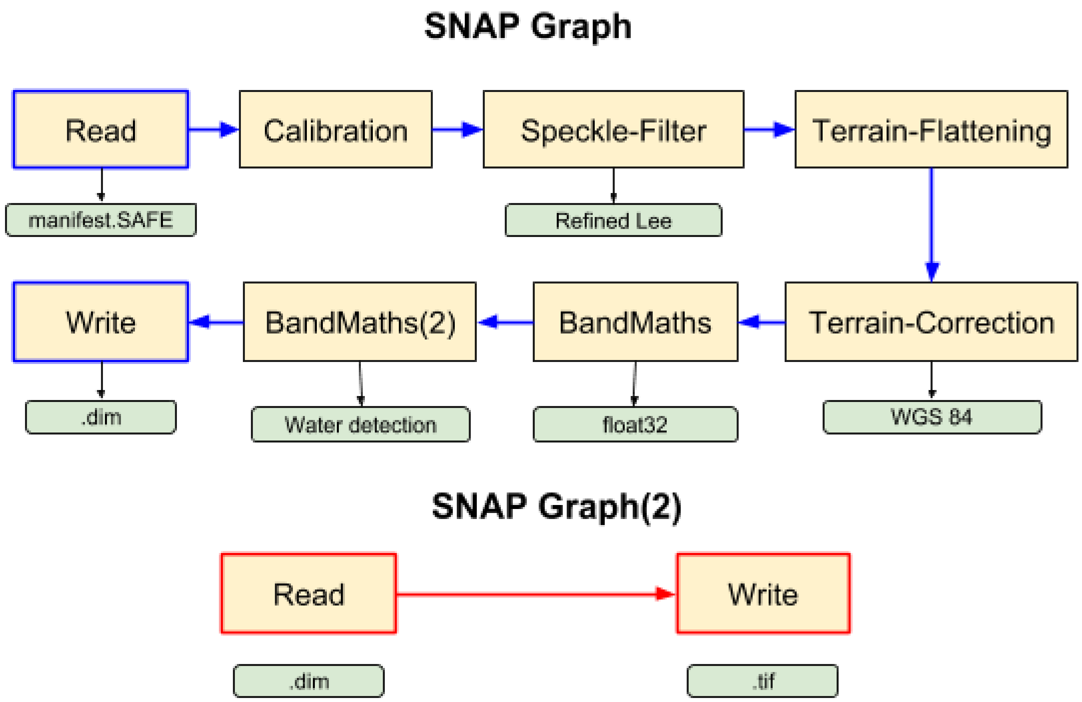


Fig. 5 SNAP Graph:Pre-processing of Sentinel-1 image graph. SNAP Graph(2): Writing BEAM-DIMAP images to Geotiff-Bigtiff graph.

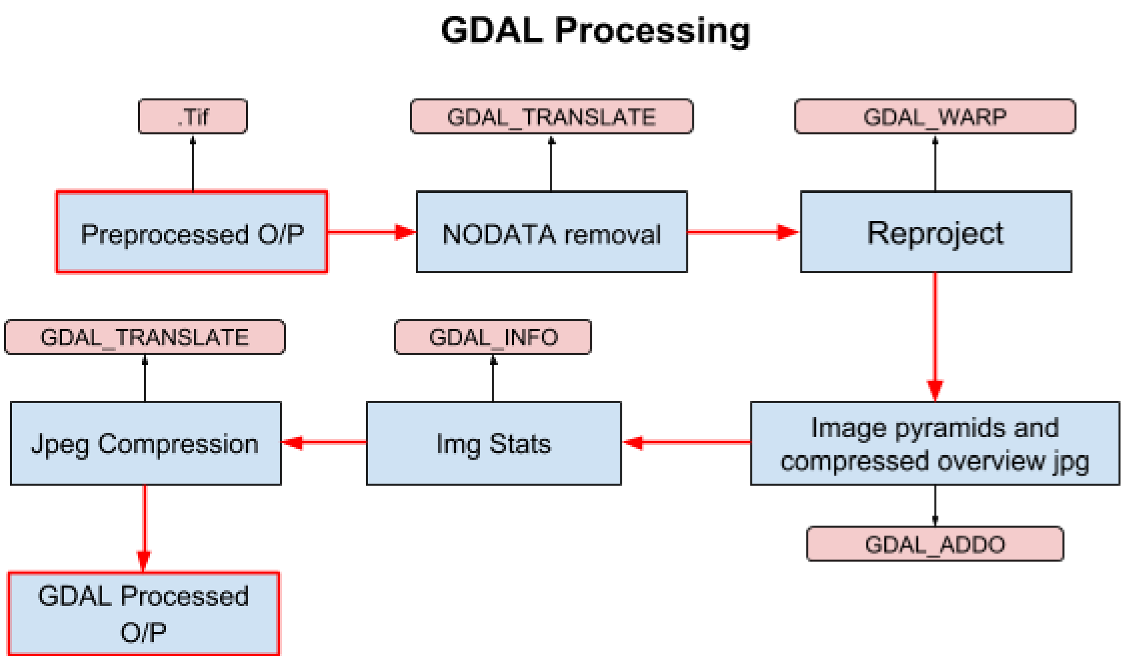
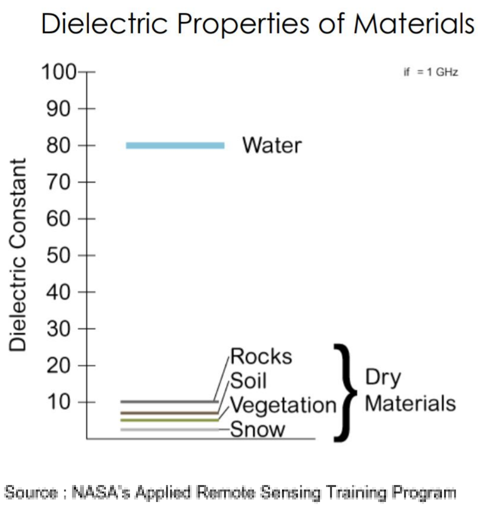


Fig. 6 Gdal processing graph.

Step 4: Water Detection - Appearance of water in SAR images

Water detection from satellite borne Satellite aperture Radar (SAR) images is an increasingly used tool in environmental monitoring applications such as flood detection, flood extent mapping

or coastline extraction [12]. Water surface in SAR image appears dark when compared to the land areas (except wetland areas). This is because water surface behaves as a specular or smooth reflectors where major parts of the incident radar energy is reflected back and thus appears as low intensity areas in SAR images contrasting with the brighter intensities consisting of rougher surrounding terrain. This contrast depends on the polarization and the angle of incident of the waves. Vertical polarization VV and VH was majorly used to detect water bodies. If the wind is faster than 10 km/hour, roughening of the water due to the wind influences the radar backscattering resulting in a diminished water contrast [8]. Hence, for more accurate mapping, wind factor needs to be considered.
To identify water bodies in SAR images, several algorithms have been proposed from which we have tried to incorporate some methods to detect water bodies.



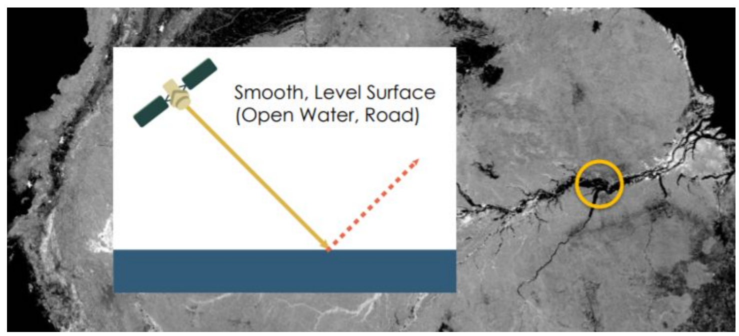


Fig. 7a Dielectric property of water. Fig. 7b Example of radar interaction with water surface near South American region.

Water detection using Bandmath (.xml)
Initially, a bandmath formula is used to convert float32 to unsigned int8. The type conversion is necessary when the data types of the operand cannot be used directly in the expression for obtaining better and accurate outputs. Further, the bands in the Sentinel-1 images are divided into mainVV, mainARI2, mainVH2, sideVV, sideARI2 and sideGEO2. The expressions within these bands include multiplying gamma amplitude band of VH and VV with integer values and then rounding them to compare with the max band value. The output of the result has a main and a side file image.The final .tif in the side folder is the final output, this is obtained from the .tif file in the main folder using bandmath expression. The final .tif is a RGB composite output (Fig. 11).

ISO Cluster Unsupervised Classification
Another approach we tried for water detection was the ESRI ISO Cluster Unsupervised Classification tool [13]. ICUC performs unsupervised classification on a series of preprocessed input raster bands using the Iso Cluster and Maximum Likelihood Classification tools. Iso Cluster performs clustering of the multivariate data combined in a list of input rasters. The

resulting signature file can be used as the input for a classification function, such as Maximum Likelihood classification, that produces an unsupervised classification raster. The minimum valid value for the number of classes is two. There is no maximum number of clusters [14].
In general, the more number of classes the better the solution. The ICUC classifies the input images based on the increasing values of the backscattered values, the first class having the lowest backscattered values when compared to the last class. The number of classes was decided based on a trial and error method on the output image. 40 classes have been taken as the suitable class number for the study. Last year, class-2 was taken as water bodies but we improvised by combining class-2 and class-5 in the python script and got better results. The output is displayed in Fig.11 below.

Automatic Thresholding

We understood that water can be detected using the histogram of the filtered backscatter coefficient and applying a threshold. The water pixels can be separated from the non-water pixels. Accurate statistics are created for the satellite image with which two peaks histogram with one small and high peak is obtained. The high peak indicates non-water regions and the low peak comprises of water region. We think of this as a form of unsupervised classification.
After determining a good threshold value (in Fig.8 the red line represents the threshold value) in raster math calculation type in 255*(sigma0_VV < The point between two curves). This expression will return a 1 or true for pixel values less than the threshold and 0 or false for values higher than the threshold. This is then multiplied by 255 to get the binary black and white image. This step was included along in the GPT graph. Since we did not have much time automating this thresholding method, we performed it only in SNAP. We skipped the automation process.

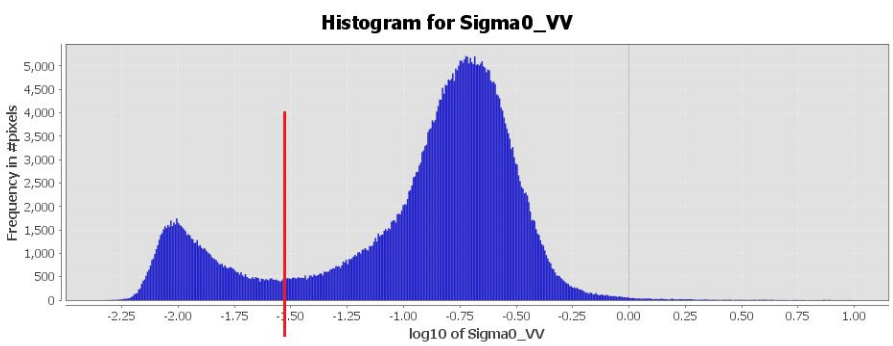


Fig. 8 Histogram statistics for Sigma0_VV.

Water Mask

Water Mask Algorithm, an inbuilt option from Sentinel-1 toolbox, takes the geographic bounds of the input product and creates a new product covering the same area. The output product contains a single band, which indicates if a pixel is water. For each pixel, it contains the fraction of water: 0.0 indicates land, 100.0 indicates water and every value in between indicates a mixed pixel. The Fractional Land/Water Mask operator generates a raster with the same extent than the input raster, but all pixel values are taken from an external data source (kind of coarse permanent water bodies), i.e. the mask pixel values you see are in no aspect related to the input raster pixels. Trying it as an option, we decided that this operator cannot be used for our use case.

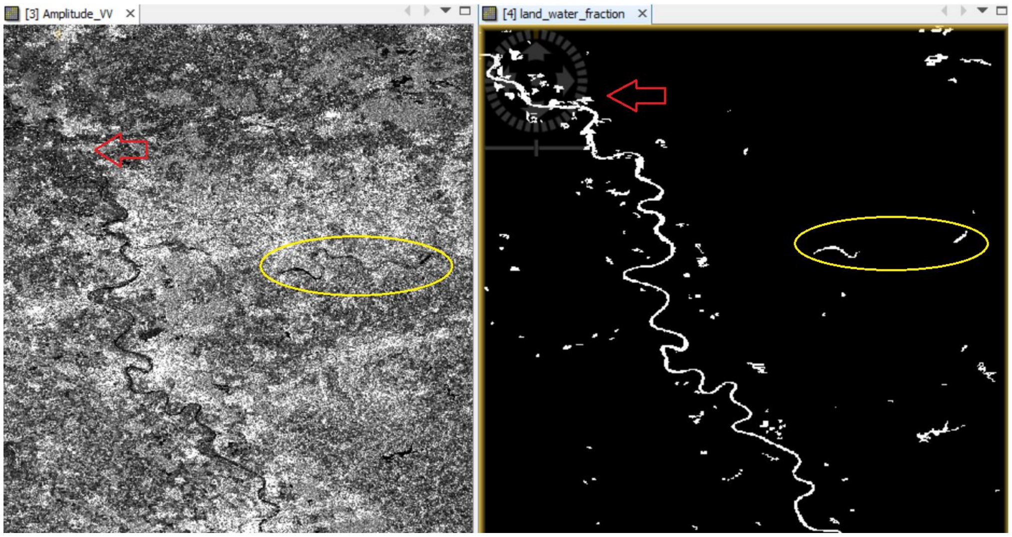


Fig. 9 Comparing Raster image and Water mask Output.

As you can see in Fig. 9, the large water bodies are clearly displayed as compared to the small water bodies, the yellow circles, where parts of the small water body are missing. In Fig. 10 satellite images before projection and after projection are displayed and we can see that a higher number of nodata values are included with the bottom right images when compared to the top right image. Also, since the terrain corrected image is tilted and the resulting water mask tries to preserve the tilted features in the horizontal image and also stretches the result image to a small degree. Including this will also affect the accuracy of the final output.

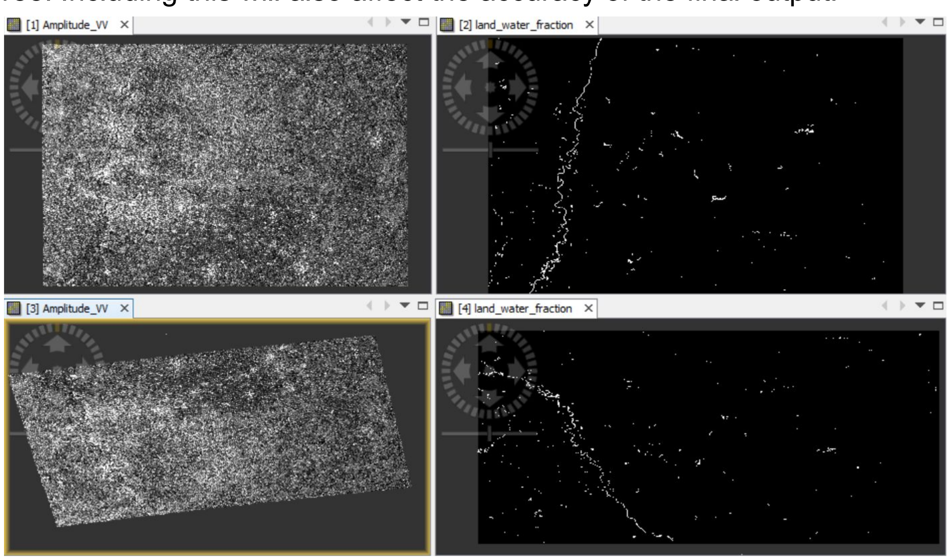


Fig. 10 Difference between water mask O/P before and after projection.

Results

The result from the Bandmath, Iso Clustering and Water mask are displayed below in Fig. 11b, Fig. 11c and Fig. 11d, respectively. Fig. 11a displays the actual raster image. All the four images have been reduced to the same area of extent for better comparison.

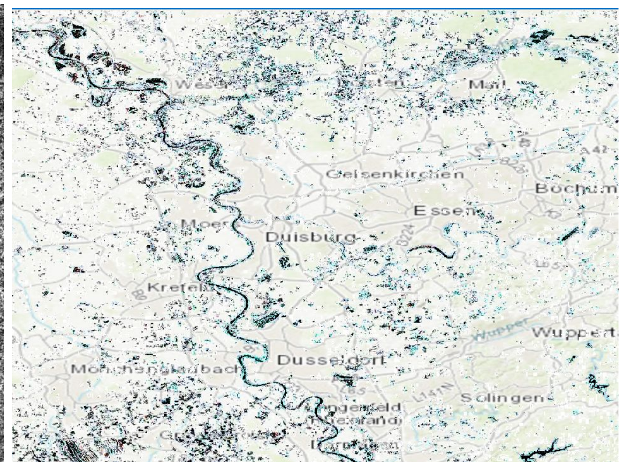
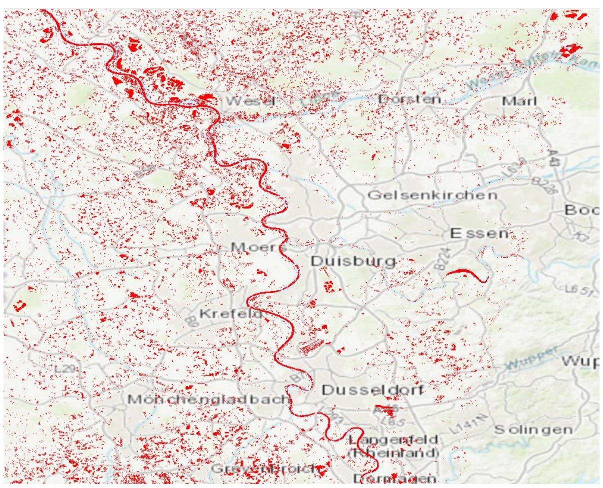


Fig. 11a Actual Raster Sentinel-1 Image.

Fig. 11b Result from BandMath.



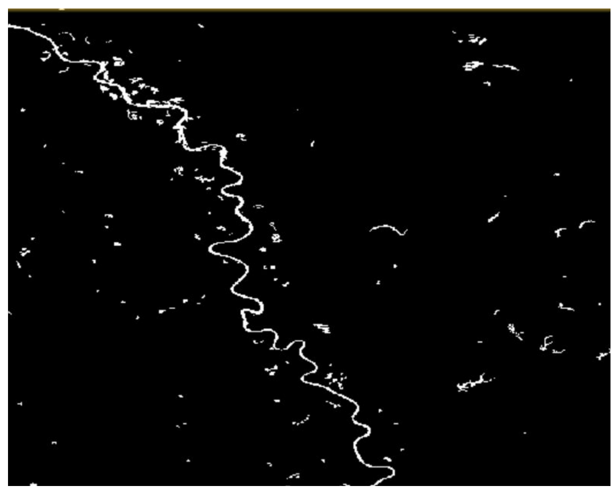


Fig. 11c Result from Iso Clustering.

Fig. 11d Result from Water Mask.

Experiences Gained & Discussion

The overall learnings from this project were the same for all the group members because of the similar technical background and limited previous experience about Remote Sensing or cloud computing. We learned how having knowledge of prerequisites before starting a project can accelerate the work. We got a good hands-on with technologies like cloud infrastructure, SNAP tool and programming in Python language. Apart from this we got a taste of agile development because of the weekly review meetings and had a glimpse of conducting small teamwork for a project with time limitations. The results obtained could have been achieved earlier if the OTC access was available right from the start. Moreover, a crucial part is the individual contribution of every team member to the whole project, since in a group of just four students it makes a huge difference if someone is not working. Also, organizing the work gets harder because one cannot know if every team member is working on a task if they do not take part in meetings. In general, though, we learned *a lot of different things* since we have been a small team. Nonetheless, the workload could not have been bigger because of missing workforce.

Future work

As mentioned above, there are certain key areas where there is a good prospect to carry the development further. One of such important points is the visualization of the output. The connection between the existing workflow and ArcGIS Image Server can be made. By referencing the output folder with the result images to the Image Server running on the VM the extracted water bodies are added to a mosaic dataset. This mosaic dataset contains just the water bodies that have been automatically created. It could be viewed via the ArcGIS Portal at the domain: *https://www.copernicus-ssp.eu.* There is also an option of publishing it as a WMS, so others could implement it in a different web service, e.g. in an emergency service.
Going forward, focus can be made on involving OBS instances in the process flow. This will help to avoid the memory conflict. Multiple compute instances can be used to operate the Sentinel-1 images. This will also help to observe the scalability of the current architecture. In the water detection segment, other water detection algorithms can be used to detect water as part of next phase of development. This year we used a different approach to last year's project. We performed the water detection for one image as a whole unlike last year, where the image was broken in multiple parts and operated separately and then combined back.

Reference

1. https://sentinel.esa.int/web/sentinel/user-guides/sentinel-1-sar/product-overview/polarimetry
2. https://gisgeography.com/synthetic-aperture-radar-examples/
3. https://sentinel.esa.int/web/sentinel/missions/sentinel-1/observation-scenario.
4. https://m.esa.int/Our_Activities/Observing_the_Earth/Copernicus/Sentinel-1/Introducing_Sentinel-1
5. ESA NEST Documents. Available from: https://earth.esa.int/web/nest/documents
6. ESA. Sentinel-1 Toolbox Tutorials, Snap Command Line Tutorial. Available from: http://step.esa.int/main/doc/tutorials/sentinel-1-toolbox-tutorials
7. https://sentinel.esa.int/documents/247904/1653440/Sentinel-1_Data_Access_and_Products
8. https://crisp.nus.edu.sg/~research/tutorial/sar_int.htm
9. https://www.gdal.org/gdal_translate.html
10. https://www.gdal.org/gdaladdo.html
11. https://www.gdal.org/gdalwarp.html
12. S.Solbo and I.Solheim, "Towards operational flood mapping with satellite SAR",Proceedings of the 2004 Envisat & ERS Symposium (ESA SP-572). 6-10 September 2004, Salzburg, Austria
13. ESRI. ISO Cluster Unsupervised Classification Tool. Available from: http://desktop.arcgis.com/en/arcmap/10.3/tools/spatial-analyst-toolbox/iso-cluster-unsupervisedclassification.htm#S_GUID-474C588B-B6F7-410D-9383-9E6C3D9247CC
14. http://edndoc.esri.com/arcobjects/9.2/net/shared/geoprocessing/spatial_analyst_tools/iso_cluster.htm