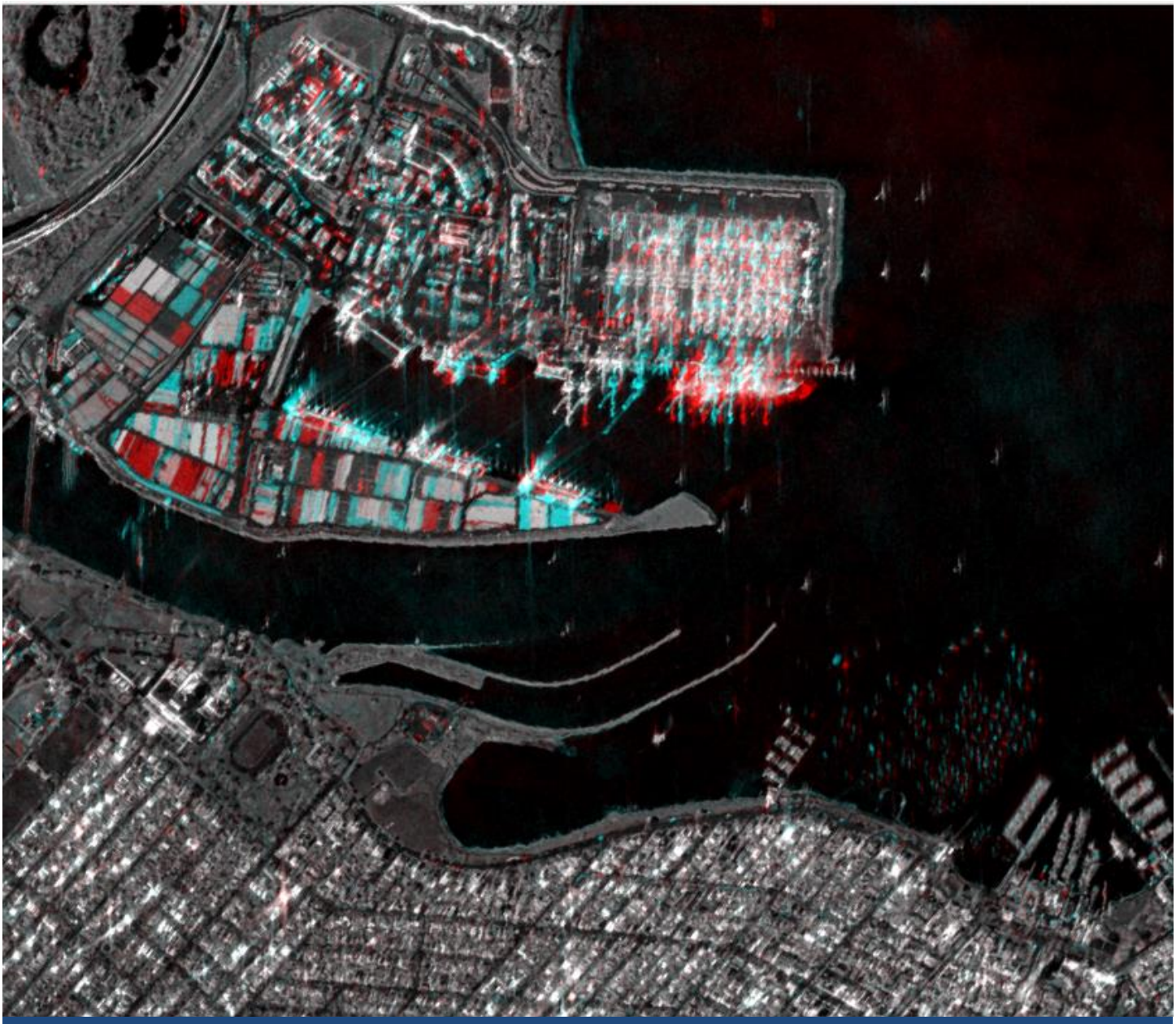

# Change Detection in Melbourne using UMBRA Open Data

Pratanu Maity

2/17/25

# Contents

**Amplitude Change Detection (ACD) Workflow:** ..... 1

**Tools:** ..... 2

**Details of Amplitude Change Detection (ACD) Workflow:** ..... 2

**Step 1: Optimal Image Pair Selection** ..... 2

**Step 2 : Speckle Filtering** ..... 4

**Step 3: Coregistration** ..... 4

**Step 4: Stack Creation and RGB Composite Generation** ..... 5

**Results and Discussion:** ..... 5

**Areas for future R&D** ..... 7

**Appendix** ..... 8

# Introduction

Change detection using Synthetic Aperture Radar (SAR) imagery requires careful consideration of acquisition parameters. Ideally, SAR images used for this purpose should be acquired under identical illumination conditions (orbit direction, look direction, and incidence angle) and from near-identical positions in space. This minimizes differences in scattering properties and parallax, ensuring that observed changes are truly representative of surface alterations rather than variations in viewing geometry. While ideal acquisitions would be from the same point in space with the same illumination, real-world SAR data, even Ground Track Repeat (GTR) products, inevitably exhibit offsets in spatial position and some variation in illumination (incidence angle, squint angle). These discrepancies introduce parallax between images, which can complicate change detection. However, even with relatively large baseline offsets, change detection with SAR amplitude is feasible under nearly identical illumination. Therefore, prior to performing change detection, image co-registration is a crucial processing step to mitigate the effects of these geometric differences and ensure accurate results. This document details a amplitude change detection analysis of Melbourne, Australia, using a SAR data pair from the UMBRA open-data catalogue, acknowledging the importance of co-registration and outlining a pathway for more robust analysis in future work.

# Amplitude Change Detection (ACD) Workflow:

The following steps were undertaken for this ACD task:

1. **Optimal Image Pair Selection:** From a large stack of SAR images, an optimal pair was identified. The primary criterion for selection was near-identical illumination conditions between the two acquisitions, minimizing the influence of varying viewing geometries.

2. **Speckle Filtering:** Both selected images were processed using a speckle filter. This step was crucial for improving the accuracy of the subsequent coregistration process by reducing noise and enhancing feature clarity.

3. **Coregistration:** The speckle-filtered image pair was precisely coregistered. This process aligns the images geometrically, ensuring accurate pixel-to-pixel correspondence for reliable change detection.

4. **Stack Creation and RGB Composite Generation:** A stack of the coregistered image pair was generated and subsequently converted to an RGB composite, yielding the final amplitude change detection results.

# Tools:

This analysis utilized Python, PCI Geomatica, GDAL, Excel and the AWS CLI.

# Details of Amplitude Change Detection (ACD) Workflow:

## Step 1: Optimal Image Pair Selection

To ensure accurate change detection results, eligible image pairs within the stack covering the port areas were required to meet specific criteria regarding imaging geometry. Specifically, selected pairs needed to share the same orbit direction (ascending or descending), the same look direction (left or right), and near-identical incidence and squint angles (differences of less than 1 degree).

1. **Metadata Acquisition:** The imaging geometry information for each image is contained within its respective metadata file, provided in JSON format. As the UMBRA open data is stored in an AWS bucket, all metadata JSON files for Melbourne were downloaded using the following AWS CLI command:

```
aws s3 cp --no-sign-request "s3://umbra-open-data-catalog/sar-
data/tasks/Melbourne, Australia" "G:/SATELLITE_IMAGERY/Umbra/All_Json/Melbourne,
Australia" --recursive  --exclude "*.*" --include *.json*
```

2. **Metadata Processing :**The downloaded JSON metadata files were parsed and their contents consolidated into a single CSV file. This conversion facilitated the identification of image pairs with similar imaging geometry using Python, leveraging the pandas, os, and json packages. The python script is shown in Appendix

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| collect_id | taskId | revisitId | startAtUTC | endAtUTC | radarBand | radarCent | polarizati | angleAzim | angleGraz | angleIncidenceDegrees | angleSquintDegrees | slantRang | antennaG | satelliteTrack | observationDirectior |
| 4849abe7- | 86fa9f6b-d8fb-44a2-b | | 2023/02/0 | 2023/02/0 | X | 9.8E+09 | (VV) | -80.2282 | 69.63007 | 20.36992588 | -1.390966329 | 572832.9 | 0 | ASCENDING | LEFT |
| a1d71564- | fc0f8b14-2c3f-48dc-a | | 2023/02/0 | 2023/02/0 | X | 9.6E+09 | (VV) | -103.335 | 59.01532 | 30.98467881 | -1.194708925 | 628395.5 | 0 | DESCENDING | RIGHT |
| 27577ee1- | 6e81b26d-a23c-418e | | 2023/04/0 | 2023/04/0 | X | 9.8E+09 | (VV) | 83.48694 | 68.14486 | 21.85514113 | 4.539289975 | 566538.6 | 0 | ASCENDING | LEFT |
| 731dece2- | 7871f99d-5f86-416a-( | | 2023/02/1 | 2023/02/1 | X | 9.6E+09 | (VV) | -87.0208 | 42.22279 | 47.77721491 | -18.93643737 | 771586.6 | 0 | DESCENDING | RIGHT |
| f696be3c- | cf4b602d-716a-4190- | | 2023/02/1 | 2023/02/1 | X | 9.6E+09 | (VV) | -104.62 | 47.71178 | 42.28821815 | -0.920367891 | 710644.7 | 0 | DESCENDING | RIGHT |
| fb064380-: | 74459934-6fba-4008- | | 2023/08/2 | 2023/08/2 | X | 9.6E+09 | (VV) | 120.3372 | 67.36106 | 22.63894021 | 135.8344116 | 576411.8 | 0 | ASCENDING | LEFT |
| 907c366c- | 8e37f492-a6f4-49d9-( | | 2023/04/0 | 2023/04/0 | X | 9.8E+09 | (VV) | -94.638 | 65.86834 | 24.131664 | -15.55832058 | 566646.9 | 0 | ASCENDING | LEFT |
| 069892e2- | 9a73c975-279b-4b8b | | 2023/02/1 | 2023/02/1 | X | 9.6E+09 | (VV) | 147.4076 | 59.18525 | 30.81474937 | -43.56508633 | 625440.7 | 0 | DESCENDING | RIGHT |
| f75004ed-( | dbdd0598-5f12-4d86- | | 2023/02/1 | 2023/02/1 | X | 9.6E+09 | (VV) | 106.8833 | 43.75988 | 46.24012277 | -0.891807654 | 755542.4 | 0 | DESCENDING | RIGHT |
| 11065c98- | d50429ed-a97c-4199 | | 2023/02/2 | 2023/02/2 | X | 9.6E+09 | (VV) | 106.7836 | 42.53719 | 47.46281212 | -0.645047927 | 770461.4 | 0 | DESCENDING | RIGHT |
| ac517ed2- | 99631618-5454-41af- | | 2023/06/1 | 2023/06/1 | X | 9.6E+09 | (VV) | 109.2022 | 58.12421 | 31.8757913 | 150.2981262 | 629528.6 | 0 | ASCENDING | LEFT |
| b83cb880- | 199bc874-3107-4c9b | | 2023/03/0 | 2023/03/0 | X | 9.6E+09 | (VV) | 105.0688 | 56.54116 | 33.45883992 | -0.363247831 | 637224.1 | 0 | DESCENDING | RIGHT |
| 10f592f9-c | d547ea8f-e521-4206- | | 2023/03/0 | 2023/03/0 | X | 9.6E+09 | (VV) | 104.788 | 59.71753 | 30.28247086 | -0.346316169 | 617134 | 0 | DESCENDING | RIGHT |
| 0dd3a1fd-( | 10cfea01-9087-4bcd- | | 2023/03/1 | 2023/03/1 | X | 9.6E+09 | (VV) | 107.6531 | 49.4984 | 40.50160252 | -2.337890531 | 685412.7 | 0 | DESCENDING | RIGHT |
| 2b09d715- | 344de480-99f0-401f-( | | 2023/03/1 | 2023/03/1 | X | 9.6E+09 | (VV) | 107.255 | 32.31759 | 57.68240609 | 0.297871244 | 919308.1 | 0 | DESCENDING | RIGHT |
| 6224d365- | 52d4467c- | 241f-4af7-( | 2023/03/1 | 2023/03/1 | X | 9.6E+09 | (VV) | 105.2139 | 52.92511 | 37.07488576 | -0.236750617 | 652505.2 | 0 | DESCENDING | RIGHT |
| cd485989- | 7621c5eb-5f8d-4208- | | 2023/03/1 | 2023/03/1 | X | 9.6E+09 | (VV) | 131.6825 | 59.56693 | 30.43307039 | -27.54346571 | 608905.4 | 0 | DESCENDING | RIGHT |
| 9f78c0c5-( | 9b0c87e5-2c14-4e89 | | 2023/03/1 | 2023/03/1 | X | 9.6E+09 | (VV) | 107.9012 | 39.53939 | 50.46061199 | -1.477845411 | 791944.3 | 0 | DESCENDING | RIGHT |
| 80e8103e- | 39e06a7c-332d-4aa9 | | 2023/03/0 | 2023/03/0 | X | 9.6E+09 | (VV) | -79.968 | 56.82434 | 33.17565548 | -0.185231244 | 635473 | 0 | ASCENDING | LEFT |
| 241272ba- | 1a9da087-1ec5-40b2 | | 2023/03/2 | 2023/03/2 | X | 9.6E+09 | (VV) | 108.0899 | 37.93259 | 52.0674078 | -1.482940591 | 809302.6 | 0 | DESCENDING | RIGHT |
| 7f4233c6-( | dad3793b-dc5a-4974 | | 2023/03/3 | 2023/03/3 | X | 9.6E+09 | (VV) | -85.6054 | 44.81923 | 45.18077029 | -19.96780147 | 722802.6 | 0 | DESCENDING | RIGHT |
| 4675b40d- | b90c0aa0-ff9b-480c-( | | 2023/04/0 | 2023/04/0 | X | 9.6E+09 | (VV) | 106.8457 | 35.54942 | 54.45058477 | 0.071961724 | 841820.6 | 0 | DESCENDING | RIGHT |
| 9e16f3d4-( | 29354f68-4bf8-42d5- | | 2023/02/1 | 2023/02/1 | X | 9.6E+09 | (VV) | -79.8934 | 55.91023 | 34.089772 | -0.03463741 | 644736.7 | 0 | ASCENDING | LEFT |
| a6ed2f24-( | c7095082-2e8b-48f6- | | 2023/07/2 | 2023/07/2 | X | 9.6E+09 | (VV) | 122.9992 | 53.97085 | 36.02915093 | 135.2036743 | 651151.1 | 0 | ASCENDING | LEFT |
| f949e2b1-( | 430e1bfa-5d2e-4f00-( | | 2023/04/0 | 2023/04/0 | X | 9.6E+09 | (VV) | -87.3866 | 55.63004 | 34.36995658 | -17.25471427 | 629248.8 | 0 | DESCENDING | RIGHT |
| aa2f8732-( | 86816e7c-d8f3-4e32- | | 2023/02/1 | 2023/02/1 | X | 9.6E+09 | (VV) | -81.5078 | 53.04379 | 36.95620714 | -1.397268948 | 665197.8 | 0 | ASCENDING | LEFT |
| af8af597-3 | 3e8851d3-9874-4ad0 | | 2023/03/2 | 2023/03/2 | X | 9.6E+09 | (VV) | -81.3697 | 51.36129 | 38.63871263 | -1.146378922 | 661049.4 | 0 | ASCENDING | LEFT |

Fig - Snapshot of the Output CSV File Containing Metadata Information for the Image Stack

3. **Image Pair Selection:** Using Excel functionalities, an optimal image pair for change detection was selected from the generated CSV file based on similarity in imaging geometry. The key imaging geometry parameters for the chosen pair are illustrated in the figure below in red rectangle, and detailed information about each image is presented in the accompanying table.

| startAtUTC | endAtU | ra | radar | polariza | angleAzimuth | angleGrazingD | angleIncidenceDegree | angleSquintDegrees | slantRa | satelliteTrack | observationDirectio |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2023/10/05 23:33:59+00 | 2023/10/0 | X | 1E+10 | (VV) | -37.75889656 | 68.18657199 | 21.81342801 | 135.8695374 | 570581.3 | DESCENDING | LEFT |
| 2023/10/04 23:34:59+00 | 2023/10/0 | X | 1E+10 | (VV) | -36.69213432 | 64.82525761 | 25.17474239 | 135.701889 | 578597.4 | DESCENDING | LEFT |
| 2024/04/26 23:44:16+00 | 2024/04/2 | X | 1E+10 | (VV) | 235.4092464 | 58.36538348 | 31.63461652 | -137.5752716 | 547627.4 | DESCENDING | LEFT |
| 2024/03/06 00:35:50+00 | 2024/03/0 | X | 1E+10 | (VV) | 328.7889983 | 57.72116068 | 32.27883932 | 127.7563248 | 644549 | DESCENDING | LEFT |
| 2024/02/12 12:19:25+00 | 2024/02/1 | X | 1E+10 | (VV) | 327.9463012 | 57.69662715 | 32.30337285 | 128.6091156 | 623691 | DESCENDING | LEFT |
| 2024/06/24 23:51:23+00 | 2024/06/2 | X | 1E+10 | (VV) | 321.0975518 | 57.23883166 | 32.76116834 | 135.5882568 | 510575 | DESCENDING | LEFT |
| 2024/04/02 12:17:14+00 | 2024/04/0 | X | 1E+10 | (VV) | 325.6869872 | 57.10615096 | 32.89384904 | 131.1541595 | 613755.8 | DESCENDING | LEFT |
| 2024/02/01 23:34:30+00 | 2024/02/0 | X | 1E+10 | (VV) | 317.8777642 | 56.81206177 | 33.18793823 | 138.5827942 | 651093.2 | DESCENDING | LEFT |
| 2024/05/11 23:30:56+00 | 2024/05/1 | X | 1E+10 | (VV) | 292.236484 | 56.7242921 | 33.2757079 | 158.1297302 | 635243.9 | DESCENDING | LEFT |
| 2023/10/10 23:37:39+00 | 2023/10/1 | X | 1E+10 | (VV) | -35.5108334 | 56.3961957 | 33.6038043 | 135.2765961 | 622653.7 | DESCENDING | LEFT |
| 2023/10/16 23:37:44+00 | 2023/10/1 | X | 1E+10 | (VV) | -35.85699834 | 56.35541405 | 33.64458595 | 135.6439362 | 622477.3 | DESCENDING | LEFT |
| 2024/05/29 23:28:04+00 | 2024/05/2 | X | 1E+10 | (VV) | 313.1597734 | 56.22830824 | 33.77169176 | 143.8140411 | 641890.8 | DESCENDING | LEFT |
| 2024/04/23 23:46:51+00 | 2024/04/2 | X | 1E+10 | (VV) | 304.5566511 | 56.1241485 | 33.8758515 | 152.165741 | 560783.3 | DESCENDING | LEFT |
| 2024/05/08 12:20:53+00 | 2024/05/0 | X | 1E+10 | (VV) | 290.5855608 | 55.35336329 | 34.64663671 | 165.9203186 | 614220 | DESCENDING | LEFT |
| 2024/03/27 12:23:13+00 | 2024/03/2 | X | 1E+10 | (VV) | 284.2574195 | 55.04777041 | 34.95222959 | 173.7015076 | 630024.6 | DESCENDING | LEFT |
| 2024/01/13 12:23:31+00 | 2024/01/1 | X | 1E+10 | (VV) | -36.20009397 | 54.19408927 | 35.80591073 | 135.2456818 | 651789.4 | DESCENDING | LEFT |
| 2023/11/13 23:38:21+00 | 2023/11/1 | X | 1E+10 | (VV) | -20.36796005 | 43.58593328 | 46.41406672 | 120.0912018 | 726919.8 | DESCENDING | LEFT |
| 2023/11/18 23:43:14+00 | 2023/11/1 | X | 1E+10 | (VV) | -31.76109575 | 40.68157039 | 49.31842961 | 130.9310303 | 760847 | DESCENDING | LEFT |
| 2023/09/27 23:51:10+00 | 2023/09/2 | X | 1E+10 | (VV) | -62.54447464 | 40.19915243 | 49.80084757 | 160.6848602 | 783898.8 | DESCENDING | LEFT |

Fig – Selected optimum image pair for change detection

| Parameters | Reference Image | Secondary Image |
|---|---|---|
| TaskId | b042ee5a-8334-4fb9-9904-660f988b48d4 | f8e1ecb5-dbbd-46ef-8c92-7c3aaf894b9f |
| FileName | 2023-10-10-23-37-38_UMBRA-04_GEC | 2023-10-16-23-37-43_UMBRA-04_GEC |
| Acquisition Date | 2023-10-10 | 2023-10-16 |
| Polarization | VV | VV |
| Orbit Direction | DESCENDING | DESCENDING |
| Look Direction | LEFT | LEFT |
| Incidence Angle | 33.6038043 | 33.64458595 |
| Squint Angle | 135.2765961 | 135.6439362 |
| Azimuth Resolution | 0.736897092 m | 0.729425425 m |
| Range Resolution | 0.736897706 m | 0.729424813 m |

Table – Descriptions of the image pair

## Step 2 : Speckle Filtering

To enhance pixel matching during the subsequent coregistration process, speckle filtering was applied to both SAR images. This technique reduces the inherent grainy noise characteristic of SAR data. Using the GUI toolbox within PCI Geomatica, a Lee filter with a 7x7 window was implemented on each image.

## Step 3: Coregistration

Sub-pixel accurate alignment was performed between the two images using the Python API functions of PCI Geomatica. This coregistration process involved two primary steps:

1. **Pixel Matching and Offset Calculation:** Corresponding pixels in both images were identified, and the offsets between these matched pixels were calculated.
2. **Image Alignment:** These calculated offsets were then applied to the secondary image, aligning it precisely with the reference image. The Python script used for this process is provided below.

```
from pci import algo

# path of secondary image
sec_img = r"E:\Pratanu\COREG\2023-10-16-23-37-43_UMBRA-04_GEC_spk.pix"

# path of reference image
ref_img =r"E:\Pratanu\COREG\2023-10-10-23-37-38_UMBRA-04_GEC_spk.pix"

# path of offset file
offset = r"E:\Pratanu\COREG\offsets.pix"

# path of output registered image
reg_image = r"E:\Pratanu\COREG\2023-10-16-23-37-43_UMBRA-04_GEC_spk_reg.pix"

# pixel matching and offsets file generation
algo.supermatch(fili=sec_img, dbic=[1], filref=ref_img, dbic_ref=[1], searchr=[30], searchun="METER", fftsize=[64], minscore=[0.8], pntgrid=[25], pntstrat="FIRST", pntclean="MED", filo=offset)

# Applying offsets on secondary image
algo.superapply(fili=sec_img, filoff=offset, dbdc=[1, 2], filo=reg_image)
```

## Step 4: Stack Creation and RGB Composite Generation

A multi-band raster was created by stacking the reference image (Band 1) and the coregistered secondary image (Band 2) using PCI Geomatica's GUI toolbox. This stacked image was then converted into an RGB composite. This conversion was accomplished using GDAL commands, assigning Band 1 to the red channel, Band 2 to the green channel, and Band 2 also to the blue channel.

```
gdal_translate -of COG
G:\SynspectiveTask\Umbra_Melbourne\OptimumPair\COREG\Stacked.tif
G:\SynspectiveTask\Umbra_Melbourne\OptimumPair\COREG\Stacked _rgb.tif -b 1 -b 2
-b 2 -a_nodata 0
```

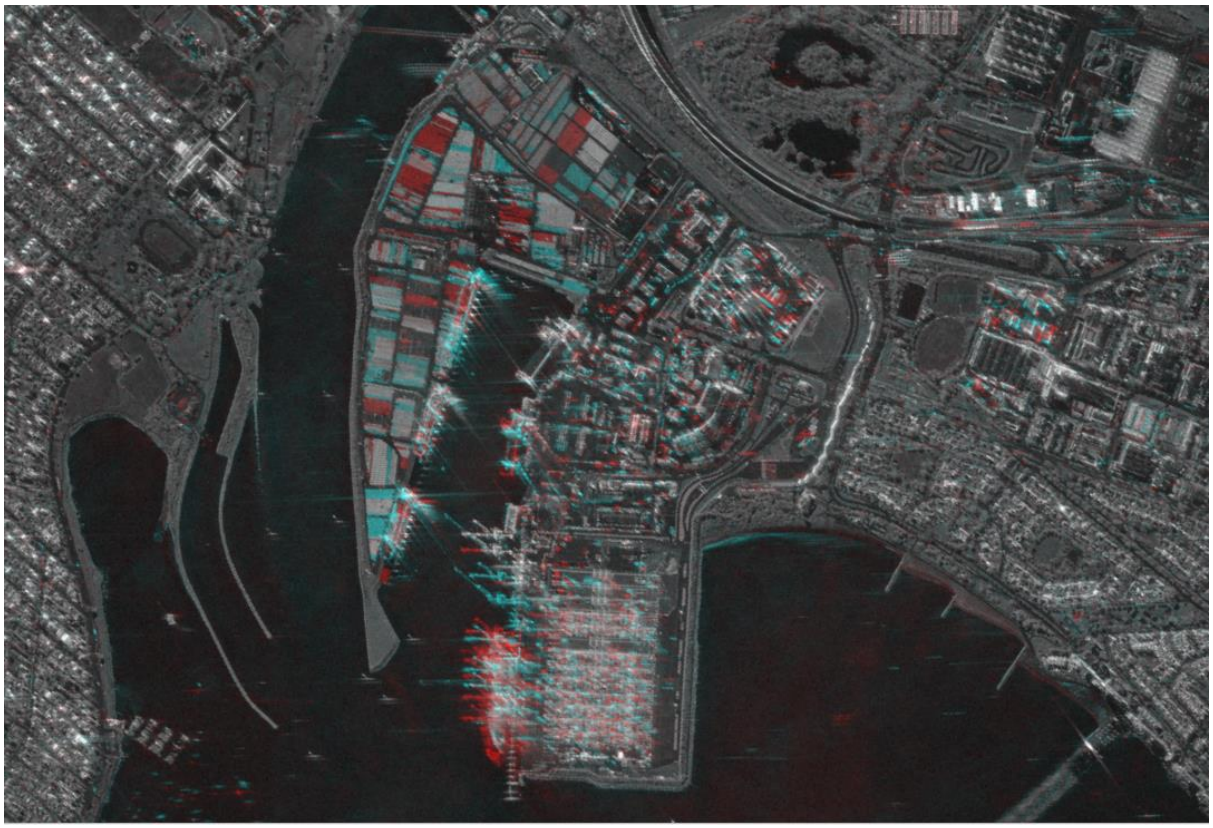

Fig – RGB Composite Displaying Amplitude Change Detection Results ( red is fled, blue is new )

# Results and Discussion:

The figure above displays the Amplitude Change Detection (ACD) image generated from two UMBRA SAR images acquired on October 10, 2023, and October 16, 2023. This ACD image visualizes changes in amplitude between the two dates. Areas of no change are represented in gray. Red areas indicate features present on October 10th but absent on October 16th, while cyan areas represent features present on October 16th but absent on October 10th.

This visualization allows for a clear assessment of amplitude changes occurring within the six-day period.

1. Tracking of ships : Arrival and Departure of Ships

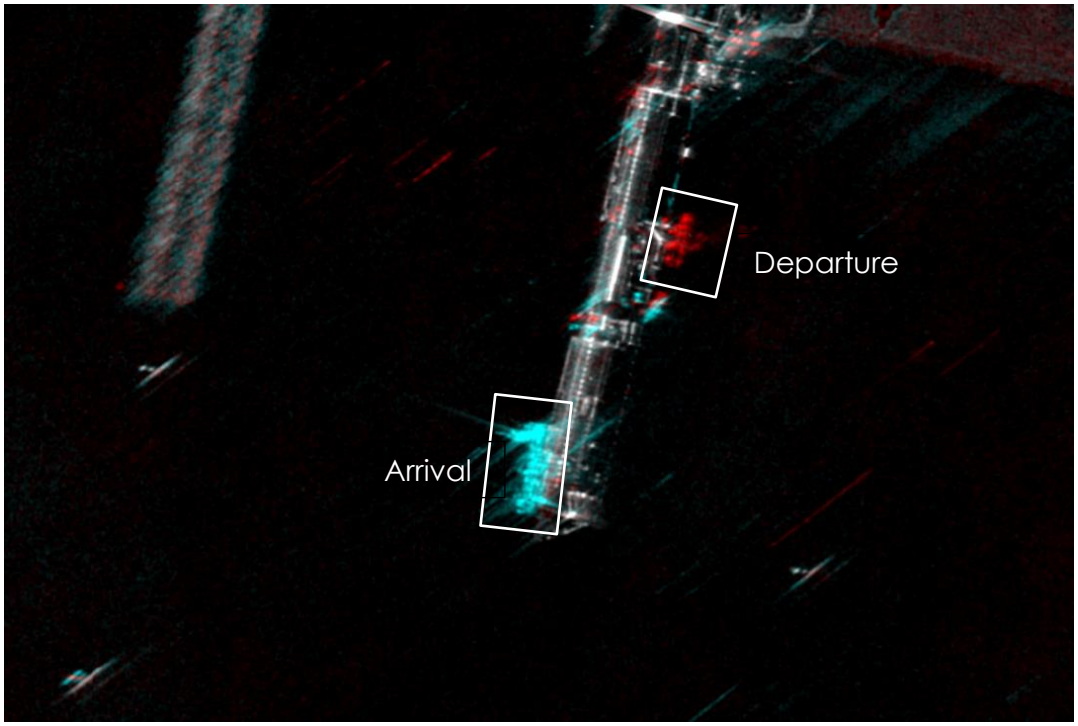


Fig – A Zoomed In portion of Amplitude Change Detection Image

2. Changes in Parking Area :

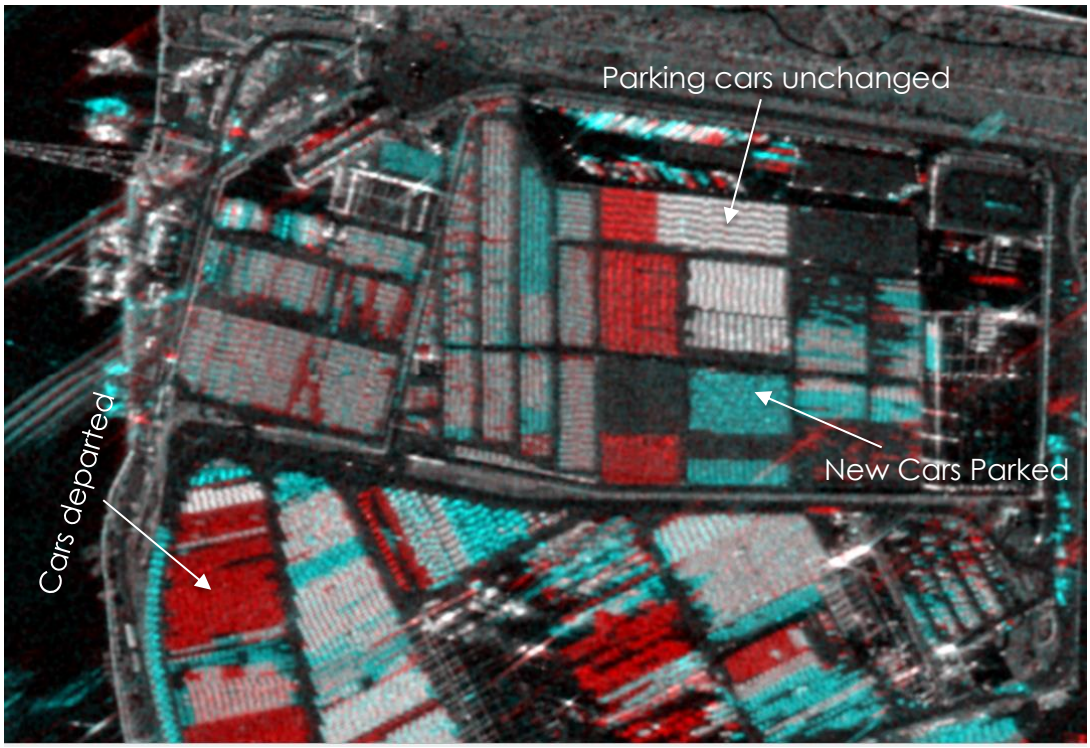


Fig – A Zoomed In portion of Amplitude Change Detection Image

3. Changes in water level : The secondary image reveals newly exposed land, indicating a drop in water level between the two dates.

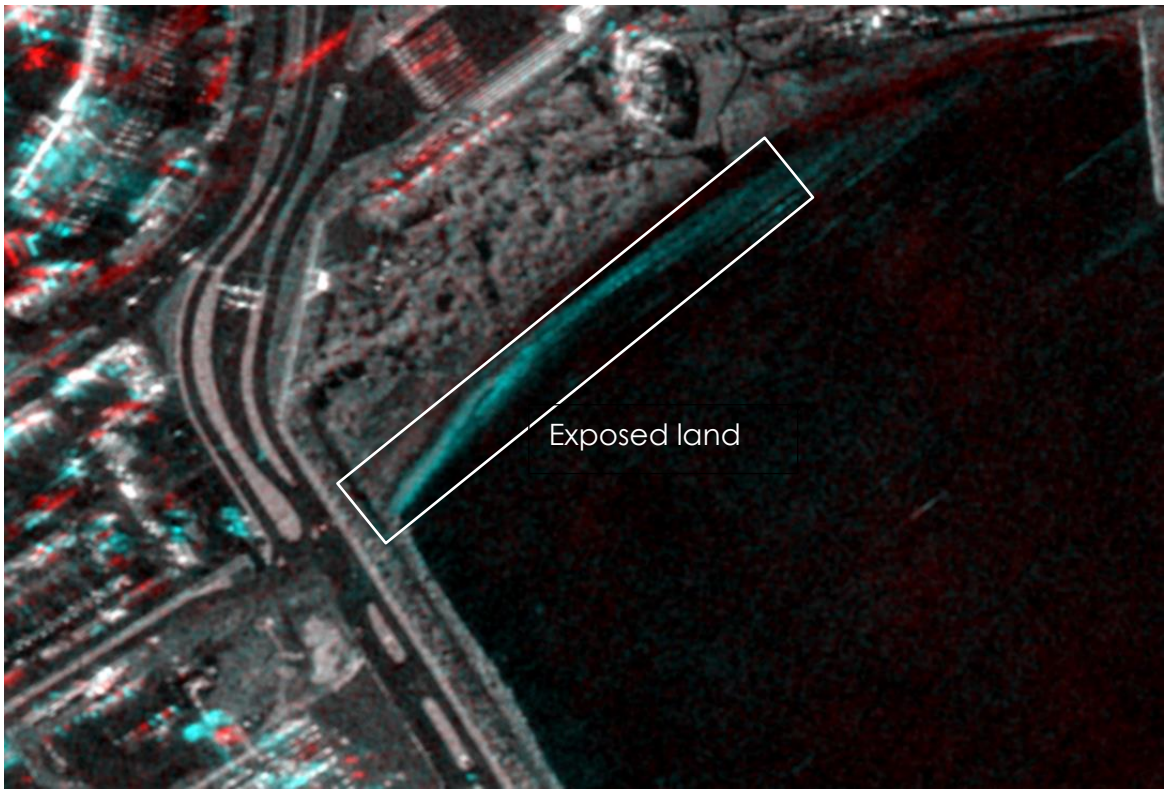


Fig – A Zoomed In portion of Amplitude Change Detection Image

## Areas for future R&D

In dense urban areas, including within ship and port zones, multiple reflections and side lobe artifacts pose challenges for accurate change detection. Further research and development are needed to maximize side lobe removal, potentially through techniques such as side lobe filtering or sub-aperture processing. Improved side lobe suppression will contribute to more accurate and reliable change detection results.

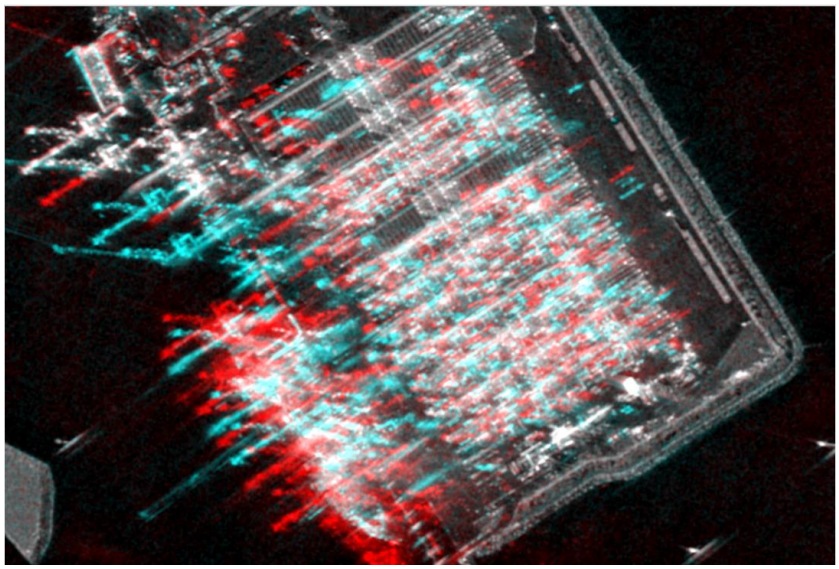

Fig – Close-up of ACD image: Port zone complexity

## Appendix

The following Python script automates the process of parsing the individual JSON metadata files and combining their contents into a single, readily usable CSV file.

```
import json
import os
import pandas as pd

def flatten_json(json_data, prefix='', exclude_keys=None):
    """Flattens a nested JSON object, excluding specified keys."""
    flattened = {}
    for key, value in json_data.items():
        if exclude_keys and key in exclude_keys:  # Check if key should be excluded
            continue

        new_key = prefix + str(key) if prefix else str(key)
        if isinstance(value, dict):
            flattened.update(flatten_json(value, new_key + '_', exclude_keys))
        elif isinstance(value, list):
            for i, item in enumerate(value):
                if isinstance(item, dict):
                    flattened.update(flatten_json(item, new_key + f'_{i}_', exclude_keys))
                else:
                    flattened[new_key + f'_{i}'] = item
        else:
            flattened[new_key] = value
    return flattened


def process_json_files(folder_path, output_file_path, exclude_keys=None):
    """Processes all JSON files in a folder and saves them to a CSV."""
    all_data = []
    for filename in os.listdir(folder_path):
        if filename.endswith(".json"):
            file_path = os.path.join(folder_path, filename)
            try:
                with open(file_path, 'r') as f:
                    data = json.load(f)

                    if 'collects' in data and isinstance(data['collects'], list):
                        for collect in data['collects']:
                            flattened_collect = flatten_json(collect, exclude_keys=exclude_keys)
                            flattened_data = flatten_json(data, exclude_keys=exclude_keys)
                            combined_data = {**flattened_data, **flattened_collect} # Merge
dictionaries.
                            all_data.append(combined_data)
                    elif 'derivedProducts' in data and isinstance(data['derivedProducts'], dict):
```

```
                    for product_type, product_list in data['derivedProducts'].items():
                        if isinstance(product_list, list):
                            for product in product_list:
                                flattened_product = flatten_json(product,
f"derivedProducts_{product_type}_", exclude_keys=exclude_keys)
                                flattened_data = flatten_json(data, exclude_keys=exclude_keys)
                                combined_data = {**flattened_data, **flattened_product}
                                all_data.append(combined_data)

                    else: # Handle JSON files without collects or derivedProducts
                        flattened_data = flatten_json(data, exclude_keys=exclude_keys)
                        all_data.append(flattened_data)

            except json.JSONDecodeError:
                print(f"Error: Invalid JSON format in {filename}")
            except Exception as e:
                print(f"An error occurred while processing {filename}: {e}")

    if all_data:  # Check if any data was collected
        df = pd.DataFrame(all_data)
        df.to_csv(output_file_path, index=False)
        print(f"JSON data from {len(all_data)} files saved to {output_file_path}")
    else:
        print("No valid JSON files found in the folder.")


# Applying the generated function to te input JSON folders
folder_path = r'G:\Umbra\Json\Melbourne'  # Replace with the path to your folder
output_file_path = r'G:\Umbra\Json\Melbourne\Output.csv'  # Replace with desired output
CSV path
exclude_keys = ["apertureReferencePointPolynomial", "footprintPolygonLla"]  # Keys to
exclude

process_json_files(folder_path, output_file_path, exclude_keys)
```