Glacies Indicium

User Manual

Version 2
04/19/2021

Project Sponsors	Team Members	Faculty Mentor
Dr. Heather Lynch	Beck Bohnker	Andrew Abraham
Dr. Mark Salvatore	Joe Carter	
Helen Eifert	Kassandra Coxen	
Brian Szutu	Logan Garrett	
	Zach Spielberger	

Table of Contents

1 Introduction

Visualizing Antarctica is a powerful web-based application that allows for easy visualization and mapping of a large amount of raster images over Antarctica. Some of the key features include:

- A preprocessing module taking multiband geotiff images and regional coordinate data to produce regional mosaics for visualization.
- An interactive map of single band regional mosaics filterable by band
- Easy access to download layer in view

The goal of this manual is so our clients can successfully deploy a local version of the server to ensure that the data looks correct and that an IT team at one of your universities can take it from there to get a live deployment on the internet. Our aim is to make sure that the scientific community surrounding Antarctic geology can work with this large scale of data. This document will assume the reader can use a command prompt and commands are prefaced with a '>>'.

2 Installation

As part of final delivery, our product has a series of prerequisite installations to set up a local environment on a platform of your choice. Over time, however, you may want to move to a new platform or re-install the product with a new source code base or new set of raster images.

2.1 Install Python

To get started, first install Python, which can be found at https://www.python.org/downloads/. Open a command prompt to verify that Python is installed by running the following command:

```
>> py --version
```

```
C:\Users\zach1>python --version
Python 3.9.4
```

Figure 1 Python version command

You should see the Python version reported back to like above.

2.2 Install Anaconda

To install Anaconda for your platform of choice visit https://anaconda.com/products/individual for the direct download.

To open Anaconda Prompt and verify installation:

- Windows: Click Start, search, or select Anaconda Prompt from the menu.
- macOS: Cmd+Space to open Spotlight Search and type "Navigator" to open the program.
- Linux-Ubuntu: Open the Dash by clicking the upper left Ubuntu icon, then type "terminal"

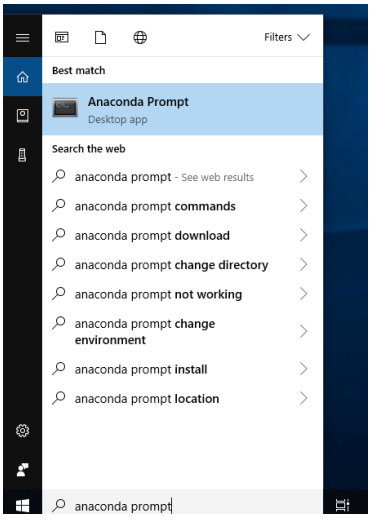

Figure 2 Anaconda Prompt

Enter the command below in the Anaconda terminal. If Anaconda is installed and working, this will display a list of installed packages and their versions as seen below.

```
>> conda list
```

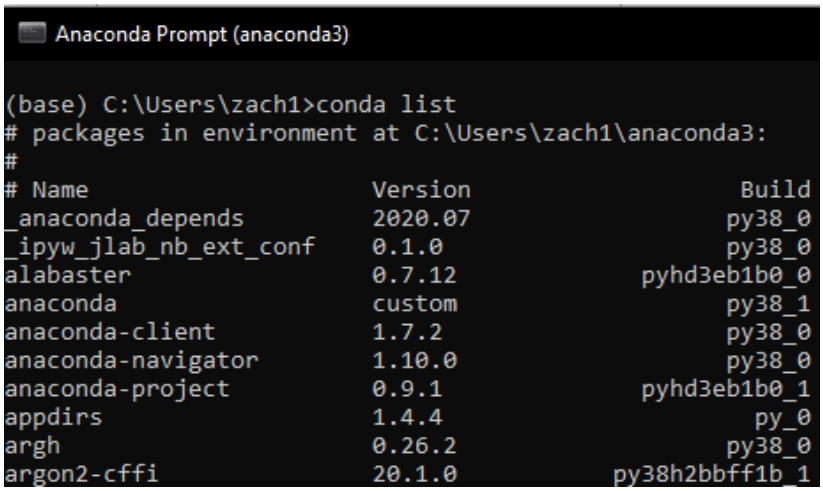

Figure 3 Anaconda Packages List

2.3 Install Git

If Git is not already installed locally, visit https://git-scm.com/downloads to install on your platform. We recommend, if you do not have one already, to also make a Github account so that we can transfer ownership of the current project to one of our clients for future capstone teams' development. Then, to verify Git is installed, run the following command.

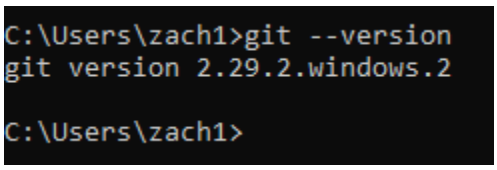

Figure 4 Git Version

```
>> git --version
```

2.4 Install Visualizing Antarctica

Once all the above have been installed properly, the visualization tool can then be installed and set up. To install the application, go to https://github.com/mlgarrett/glacies-indicium and clone the main branch, as it is the most stable version using the following command. Then, navigate to this folder using the Anaconda prompt and open to the glacies-indicium directory.

a. Clone the GitHub repository into local computer.

```
>> git clone https://github.com/mlgarrett/glacies-indicium
```

b. Change into the GitHub Repository and create the Terracotta environment.

```
>> cd glacies-indicium

>> conda env create -f environment.yml
```

If this command finishes without errors, then Terracotta has been successfully installed. Run the below command to activate Terracotta before starting the application.

```
>> conda activate terracotta
```

The Anaconda prompt will reflect the current environment within the first parenthesis as show below. After the Terracotta environment has been activated, the application is now ready to install GDAL and rasterio.

Figure 5 Parentheses Representing Terracotta Environment is active.

2.5 Install GDAL

Within the Anaconda terminal, run:

```
>> conda install -c conda-forge gdal
```

Verify GDAL was installed by running the following command:

```
>> gdalinfo --version
```

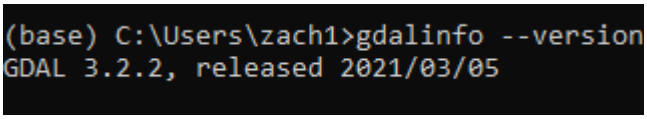

Figure 6 GDAL Confirmation of Version

2.6 Install rasterio

In the same Anaconda prompt from above, run:

```
>> conda install -c conda-forge rasterio
```

With both libraries installed, the application is now ready to run the preprocessing module and run Terracotta to visualize GeoTiff images which are covered in the following sections.

3 Configuration and Daily Operation

3.1 JSON Files Configuration

First, ensure that there is a bandNames.json file located under the terracotta/client/static/data directory. This file is used in the preprocessing script and for displaying the bands in the user interface. This file should also contain an array of JSON objects as shown below. There should also be at least one object in the array.

```
[
  {
    "name": "Blue"
  },
  {
    "name": "Dolerite"
  },
  {
    "name": "Granite"
  },
  {
    "name": "Green"
  },
  {
    "name": "Ice"
  },
  {
    "name": "Red"
  },
  {
    "name": "Shadow"
  },
  {
    "name": "Snow"
  }
]
```


Figure 7 Example 'Dummy' Bands for Current Selection of Source Data

The second required JSON file is the regions.JSON file which is also located under the same directory as the bandNames.json file. This JSON file will be generated during the preprocessing module to represent the hierarchy of regions from the raster images in the user interface. An example of this file is shown below which contains one tier one and one two region as well as six tier three regions.

```
[
    {
        "name": "Central Transantarctic Mountains",
        "id": "fe1e",
        "subregions": [
            {
                "name": "Shackleton Glacier",
                "id": "b641",
                "subregions": [
                    {
                        "name": "Nilsen Peak",
                        "id": "1b13",
                        "subregions": []
                    },
                    {
                        "name": "Mount Speed",
                        "id": "a99e",
                        "subregions": []
                    },
                    {
                        "name": "Mount Wendland",
                        "id": "4447",
                        "subregions": []
                    },
                    {
                        "name": "Taylor Nunatak",
                        "id": "6c14",
                        "subregions": []
                    },
                    {
                        "name": "Mount Greenlee",
                        "id": "076e",
                        "subregions": []
                    },
                    {
                        "name": "Mount Cole",
                        "id": "e164",
                        "subregions": []
                    }
                ]
            }
        ]
    }
]
```


Figure 8 Example of Regional JSON File

3.2 Comma Separated Values Files

There are three CSV files included in the prep_scripts directory that are used to generate the JSON structure and to associate the raster images to the appropriate regions. The file `L4_regions_sparse.csv` contains the coordinates for all of the available tier 4 regions. This is used in the region rekeying portion of the preprocessing script to find the closest region by using the coordinates and the Haversine formula.

The second file, `all_regions.csv` contains every available region within Antarctica and is organized by tier in every row using four columns for each tier. These columns are organized by starting with the largest region and going down to the smallest region. As shown below, the largest tier in row 2 is Northern Victoria Land and the smallest tier is Mt. Newell.

```
1   Tier1,Tier2,Tier3,Tier4
2   Northern Victoria Land,McMurdo Dry Valleys,Asgard Range,Mt. Newall
3   Northern Victoria Land,McMurdo Dry Valleys,Asgard Range,Round Mountain
4   Northern Victoria Land,McMurdo Dry Valleys,Ferrar Glacier,Kukri Hills
5   Northern Victoria Land,McMurdo Dry Valleys,Ferrar Glacier,Briggs Hills
6   Northern Victoria Land,McMurdo Dry Valleys,Ferrar Glacier,Thomas Heights
7   Northern Victoria Land,McMurdo Dry Valleys,Kukri Hills,Mt. Coates
```

Figure 9 Example of csv regional structure

The final csv file, `available_regions.csv` is used to store the regions found during the preprocessing script and uses the same format for rows and columns as the `all_regions.csv` file. After the script finishes creating each region, this file is used to create the regions.JSON file from above.

3.3 Run Preprocessing Script

To run the preprocessing script, navigate to the prep scripts folder in Anaconda prompt and pass in a path to a directory of raster images, as shown below.

```
>> python raster-prep.py -d <input folder path> [OPTIONS]
```

This command will run the preprocessing images on a set of images inside the input folder provided. If the preprocessing script runs successfully, you will see an output like the one below.

```
centerX, centerY: -34903.28594999999, -549924.91806
band extraction...
        ..\terracotta\client\static\mosaics\TaylorNunatak_Blue.tif
        ..\terracotta\client\static\mosaics\TaylorNunatak_Dolerite.tif
        ..\terracotta\client\static\mosaics\TaylorNunatak_Granite.tif
        ..\terracotta\client\static\mosaics\TaylorNunatak_Green.tif
        ..\terracotta\client\static\mosaics\TaylorNunatak_Ice.tif
        ..\terracotta\client\static\mosaics\TaylorNunatak_Red.tif
        ..\terracotta\client\static\mosaics\TaylorNunatak_Shadow.tif
        ..\terracotta\client\static\mosaics\TaylorNunatak_Snow.tif

Raster preprocessing is now complete! 🎉
PS D:\dev\terracotta\prep_scripts>
```

The optional command `-r` or `--remove_image_border` can be appended to the command above which will remove the borders around the raster images. As we discussed in client meetings, the no data

values will mask real data in the raster images unless this is accounted for in the clients processing pipeline. Otherwise use the remove image flag as seen below:

```
>> python raster-prep.py -d <input folder path> -r
```

3.3 Run Terracotta

3.3.1 Optimize Raster Images

Optimizing the raster images will improve the overall performance of the web application. To convert the raster images to cloud optimized GeoTiff format, run the following command which points to an already existing and empty output folder. Please note that for the raster image download to work properly, there must be raster images in the optimized folder.

```
>> terracotta optimize-rasters terracotta\client\static\mosaics\*.tif
-o terracotta\client\static\mosaics\optimized\ [OPTIONS]
```

Additional options can be added to the end of the above command from the list below if needed:

`--overwrite`

- Force overwrite of existing files in output folder.

`--resampling-method <resampling_method>`

- Resampling method for overviews

  Default: average

  Options: average|nearest|bilinear|cubic

`--reproject`

- Reproject raster files to Web Mercator for faster access.

  Default: False

`--in-memory, --no-in-memory`

- Force processing raster in memory/ not in memory

  Default: process in memory if smaller than 120 million pixels

`--compression <compression>`

- Compression algorithm to use

  Default: auto (ZSTD is available, DEFLATE otherwise)

  Options: auto|deflate|lzw|zstd|none

`-q`

- Suppress all output to the terminal
  Default: False

More information about this command and other Terracotta commands can be found at this link, https://terracotta-python.readthedocs.io/en/latest/cli-commands/serve.html.

3.3.1 Start Terracotta Server

To start Terracotta, first open two instances of the Anaconda prompt and navigate to the glacies_indicium directory in each terminal. In one terminal, run the following command to serve the newly created set of optimized raster images in the web browser. If any changes occur to the output data of the processing module be aware that the keys (words listed inside curly braces) should reflect the output naming conventions of the module. These act as keys to match the region hierarchy and bands to specific regional mosaics.

```
>> terracotta serve -r terracotta\client\static\mosaics\optimized\
{region}_{band}.tif
```

3.3.2 Connect to Terracotta Server

In the second terminal, run the following command to connect to Terracotta which will automatically open the application in the browser.

```
>> terracotta connect http://localhost:5000
```

3.3.3 Connect to Terracotta API

Terracotta also provides API endpoints for retrieving data from the raster images. These endpoints can be found below and can be entered in any web browser to view the data. These endpoints will return data in JSON format from the available raster images.

- http://localhost:5000/keys
- http://localhost:5000/datasets
- http://localhost:5000/apidoc

4 Deployment and Maintenance

The source code for this project can be found at: https://github.com/mlgarrett/glacies-indicium. The README.md file also includes an outline of how to download and run Visualizing Antarctica as well. The following is primarily a baseline for an IT department to provide a live deployment of our product for our clients. The following is a bit more technical and our clients are not expected to go through this process.

Deployment of this tool to a live web server can be achieved by taking the following steps:

1. Terracotta Dependencies and Installation
    a. Install Anaconda according to instructions on Anaconda website
    b. Update Anaconda

        ```
        >> conda update -n base -c defaults conda
        ```

    c. Install pip

        ```
        >> conda install pip
        ```

    d. Create Anaconda environment for Gunicorn WSGI instance; activate

```
>> conda create --name gunicorn

>> source activate gunicorn
```

e. Install Terracotta dependencies

```
>> sudo apt install build-essential gdal-bin libgdal-dev

>> pip install cython
```

f. Enter repository directory, install tool

```
>> cd /path/to/code/repo

>> pip install -e .
```

g. Install Gunicorn WSGI

```
>> pip install gunicorn
```

2. Install and verify Nginx web server

```
>> sudo apt install nginx

>> sudo systemctl status nginx
```

3. Install geos suite for raster manipulation

```
>> conda install geos
```

4. Get data and optimize it for Terracotta
   a. Optimize your preprocessed source data

```
>> terracotta optimize-rasters /path/to/source/data/*.tif -o
/path/for/optimized/data/
```

   b. Ingest optimized preprocessed source rasters into a Terracotta SQL database. The filename keying scheme must reflect the naming structure of the source files to ensure desired navigational characteristics

```
>> terracotta ingest
/path/to/optimized/data/{region}_{band}.tif -o
/path/to/database/terracotta.sqlite
```

5. Create system service to manage the Terracotta server Gunicorn process
   a. Create the file `/etc/system/terracotta_server.service`

```
[Unit]
Description=Gunicorn instance to serve Terracotta
After=network.target

[Service]
User=sammy
Group=www-data
WorkingDirectory=/mnt/data
Environment="PATH=/home/ubuntu/anaconda3/envs/gunicorn/bin"
```

```
Environment="TC_DRIVER_PATH=/path/to/database/terracotta.sq
lite"
ExecStart=/home/ubuntu/anaconda3/envs/gunicorn/bin/gunicorn
\
             --workers 3 --bind unix:terracotta.sock -m 007
terracotta.server.app:app

[Install]
WantedBy=multi-user.target
```

b. Start and enable Gunicorn service

```
>> sudo systemctl start terracotta_server
>> sudo systemctl enable terracotta_server
>> sudo systemctl restart terracotta_server
```

6. Configure Nginx to forward web requests to server application
   a. Create file `/etc/nginx/sites-available/terracotta`

```
server {
    listen 80;
    server_name VM_IP;

    location / {
        include proxy_params;
        proxy_pass
http://unix:/path/to/database/terracotta.sock;
    }
}
```

   b. Link Nginx configuration to sites-enabled

```
>> sudo ln -s /etc/nginx/sites-available/terracotta
/etc/nginx/sites-enabled/terracotta
```

7. Restart Nginx and Terracotta Services
   a. Restart Nginx and Terracotta services

```
>> sudo systemctl restart nginx
>> sudo systemctl restart terracotta_server
```

At the end of this procedure, the Terracotta server application has been installed and configured, and Nginx will forward internet requests into the Terracotta server.

5 Troubleshooting

5.1 Preprocessing

If an error occurs during preprocessing, ensure that the mosaics folder in the terracotta/client/static/ directory is empty and does not contain any images. Both GDAL and raster io are also required for the preprocessing module, so it is crucial to have those dependencies installed locally using Anaconda and the above commands. Also, ensure that the JSON file for regions and bands is accurate to the content we listed above.

If the pyproj or emoji module did not install during the Terracotta installation, please run `pip install pyproj` and/or `pip install emoji` in the prep_scripts directory to fix this issue.

5.2 Terracotta

Depending on the speed of the server, it may take up to 30 seconds for Terracotta to fully ingest the data for the first time in the web browser and this will occasionally cause images not to appear. If this occurs and error appears in the window or in the console, the fastest solution is to just refresh the page.

5.3 Performance

Be aware that due to the scope of data available our product was only tested with a few GBs of raster images. For the whole continent of Antarctica preprocessing will take a notable amount of time and in addition ingestion may suffer from the same problem. Luckily display of the raster data should maintain the half second tiling time. If regions become so large they took longer, it is easily implementable to add functionality to the band splitting section of the preprocessing pipeline to downsample high tier regions.

6 Conclusion

This concludes the Visualizing Antarctica user manual. We wish you years of productive use of this product, and we are happy to have been able to help bring this application to life. While we are all moving on to professional careers, we would be happy to answer short questions in the coming months to help you get the product deployed and operating optimally in your organization. With best wishes from your Visualizing Antarctica developers:

Beck Bohnker (rlb462@nau.edu)

Joe Carter (jsc383@nau.edu)

Kassandra Coxen (klc624@nau.edu)

Logan Garret (mlg238@nau.edu)

Zachary Spielberger (zps9@nau.edu)