Mugin Georeferencing Pipeline

A Python package for processing geospatial reference data.

Installation

Automated Installation (Windows)

1. Clone the repository
2. Run `setup_windows.bat`

Linux/Mac

1. Clone the repository
2. Make the script executable: `chmod +x setup_linux.sh`
3. Run `./setup_environment.sh`

Manual Installation using Conda (Recommended)

1. Create and activate conda environment (in miniconda3 or similar)

```
conda nv create -f environment.yml
conda activate georef_env
```

2. Install the package `pip install .`

Get a Digital Elevation Model (DEM)

The georeferencing pipeline needs a DEM (or DTM / DOM) to georeference the images to. If no DEM is provided a default DEM (Earth Oblated Spheroid) is used. The following describes how to get a custom DEM. This is only one possible method go get a DEM for the selected area, other methods to get a custom DEMs do exist also.

1. Find relevant elevation models (e.g. on Geonorge, resp. Kartverket høydedata or similar) and download them (normally *.geotiff files)
2. Load into these geotiff files into QGIS
3. Merge all elevation models into one raster-layer

In QGIS:

- Raster -> Miscallenious
- Merge (Select all rasters to be merged) -> Run

5. Clip raster to the area needed In QGIS:
    - Raster
    - Extract
    - Clip raster by Extent

    - Select layer (assign a no-data value (e.g. -9999.0), save to file, clipping extent)
    - Run
6. Export raster to file as *.geotiff
7. Copy the *.geotiff DEM file into "\DATA\DigElev<DEM_name.tif>"
8. Update the path with the DEM name in the config.ini

Folder Structure

```
├── src
│   ├── objects
│   │   ├── <python object files>
│   ├── tools
│   │   ├── <helper function files>
│   ├── __init__.py
│   ├── main.py
├── CONFIG
│   ├── config.ini
│   ├── parameters.xml
├── DATA
│   ├── DigElev
│   ├── ├── Digital_elevation_model.tif
│   ├── ├── model_temp (autogenerated)
│   ├── ├── model.ply (autogenerated)
│   ├── input
│   ├── output
├── enviroment.yml
├── README.md
├── requirements.txt
├── setup_linux.sh
├── setup_windows.sh
├── setup.py
└── .gitignore
```

Georeference images, workflow

Georeferencing Mugin images with this pipeline shall be done as follows:

> **Note:** Before starting the georeferencing process make sure a DEM is loaded in "\DATA\DigElev<DEM_name.tif>" and the name is updated in the config.ini file

1. Load images into the input folder specified in the config.ini file (normally: "\DATA\input")
2. Load the gpslog file into the same input folder as the images. The filename must start with "gpslog...".
3. Make sure the output folder is specified correctly in the config.ini file (normally "\DATA\input")
4. Check all the other settings in the config.ini file (see below):
5. run "/src/main.py"

Configuration file

Part of the configuration file is autogenerated: - change to appropriate value - don't change (autogenerated)

Config file parameter	Value	Description
missionname	`'string'`	Title for the project
inputfolder	`'path'`	older containing all the images in *.png or *.jpg format and the gpslog.txt file
outputfolder	`'path'`	Georeferenced geotiff files are saved in this folder
sensor	`'sensor name'`	Which imaging sensor was used the drone (This is used to read the relevant parameters from parameters.xml. Sensor names (`'sensor name'`) are: • P1 35mm • H20T Thermal • H20T Zoom 2x • Sony ILX-LR1
output_epsg	`'epsg-code'`	The EPSG code for the Universal Transverse Mercator (UTM) onto which the images shall be georeferenced, see here for more details (e.g. 32623 for WGS-84 / UTM zone 32N)
overwrite_output	`'boolean'`	If `'True'`, files in the output folder will be overwritten automatically (`'True'` or `'False'`)
downscale_factor_imgs	`'int'`	Factor (> 1) by which the output images shall be downscalled (1 or -1 for no downscalling)
mirror_images	`'string'`	Mirror georeferenced images: 3 options: • no mirroring = `'none'` • mirror images horizontally = `'horizontal'` • mirror images vertically = `'vertical'`
delta_north	`'float'`	Tuning parameter, move output images north [m]
delta_east	`'float'`	Tuning parameter, move output images east [m]
pitch_angle	`'float'`	Tuning parameter, rotation of the camera due to mounting of the camera in the UAV and due to trimmed flight conditions of the UAV [°]
roll_angle	`'float'`	Tuning parameter, rotation of the camera due to mounting of the camera in the UAV and due to trimmed flight conditions of the UAV [°]
yaw_angle	`'float'`	Tuning parameter, rotation of the camera due to mounting of the camera in the UAV and due to trimmed flight conditions of the UAV [°]
rotation	`'float'`	Tuning parameter, rotate output images clockwise [°]

Config file parameter	Value	Description
dem_path	'path'	Path to the DEM geotiff file
model_path	'path'	Path to the DEM model (autogenerated)
subsample_factor	'int'	subsample factor for the DEM (choose a large no. (e.g. 1000) for smooth surfaces like ocean, and a smaller number for more rugged surfaces.
epsg_wgs84	'int'	EPSG code (autogenerated, do not change!!!)
dem_epsg	'int'	EPSG code of the DEM (autogenerated, do not change!!!)

Processing flow

```
graph TD
    Main["main()"] --> Init["initialize()"]
    Init                <--> Config[Configuration]
    Init                <--> Params[Parameters]
    Init                --> geoPose[Calculate ECEF
 Position & Attitude]
    geoPose             --> demCheck{DEM
 Available?}
    Config              --> |DEM Path|demCheck
    Params              --> geoPose
    demCheck            --> |No|j2_dem[Generate J2
 Spheroid DEM]
    demCheck            --> |Yes|cust_dem[Load Custom DEM]
    cust_dem            --> mesh3d[Generate 3D Mesh]
    j2_dem              --> mesh3d
    mesh3d              --> rayIntersect[Ray-Mesh
 Intersection]
    rayIntersect        --> |image corner positions|georef[Georectification]
    images[Input Images\n*.png] --> georef
    Config              --> |EPSG Code|georef
    Config              --> |Tuning Values
 File Paths|geoPose
    georef              --> output[Georeferenced
 GeoTIFF]

    subgraph Input
        images[Input Images
 *.png]
    end


    ConfigFiles[Configuration Files]
    code[Processing steps]
    inpOut[Input / Output]

    subgraph Output
        output[Georeferenced\nGeoTIFF]
```

```
    end

classDef config fill:#e6f3ff,stroke:#333,stroke-width:2px
classDef io fill:#ffe6cc,stroke:#333,stroke-width:2px
classDef process fill:#f5f5f5,stroke:#333,stroke-width:2px
class Config,Params,ConfigFiles config
class images,output,inpOut io
class Init,geoPose,j2_dem,cust_dem,mesh3d,rayIntersect,georef,code process
```