Introduction of VIIRS Flood Detection Software

Version 1.0

Sanmei Li Donglian Sun

George Mason University

Email: [slia@gmu.edu](mailto:slia@gmu.edu)

Contents

[Introduction of VIIRS Flood Detection Software 1](#_Toc35016804)

[1. Introduction 4](#_Toc35016805)

[2. Algorithm description 4](#_Toc35016806)

[3. Module Description 5](#_Toc35016807)

[3.1 Module Description 5](#_Toc35016808)

[3.1.1 Module 1: VIIRS Swath Projection 5](#_Toc35016809)

[3.1.2 Module 2: VIIRS Flood Detection 6](#_Toc35016810)

[3.1.3 Module 3: Image Display 7](#_Toc35016811)

[3.1.4 Module 4: Mosaick\_Subset\_Flood\_Process\_NC 7](#_Toc35016812)

[3.1.5 Module 5: VIIRS\_Composition\_NC 7](#_Toc35016813)

[4. Data Input/Output 7](#_Toc35016814)

[4.1 Input/Output of module: VIIRS\_Swath\_Projection 7](#_Toc35016815)

[4.1.1 Input 7](#_Toc35016816)

[4.1.2 Output 8](#_Toc35016817)

[4.2 Input/Output of module: VIIRS\_Flood\_Detection 8](#_Toc35016818)

[4.2.1 Input 8](#_Toc35016819)

[4.2.2 Output 9](#_Toc35016820)

[4.3 Input/Output of module: Mosaick\_Subset\_Flood\_Process\_NC 9](#_Toc35016821)

[4.3.1 Input 9](#_Toc35016822)

[4.3.2 Output 9](#_Toc35016823)

[4.4 Input of module: VIIRS\_Composition\_NC 10](#_Toc35016824)

[4.4.1 Input 10](#_Toc35016825)

[4.4.2 Output 10](#_Toc35016826)

[4.5 Input of module: Image\_Display 10](#_Toc35016827)

[4.5.1 Input 10](#_Toc35016828)

[4.5.1.1 Input of *flood-nc2tif.sh* 10](#_Toc35016829)

[4.5.1.2 Input of *flood-tif2png.sh* 11](#_Toc35016830)

[4.5.1.3 Input of *flood-tif2shape.sh* 11](#_Toc35016831)

[4.5.2 Output 11](#_Toc35016832)

[4.5.2.1 Output of *flood-nc2tif.sh* 11](#_Toc35016833)

[4.5.2.2 Output of *flood-tif2png.sh* 11](#_Toc35016834)

[4.5.2.3 Output of *flood-tif2shape.sh* 11](#_Toc35016835)

[5. Environment requirements 12](#_Toc35016836)

[5.1 Source Codes: 12](#_Toc35016837)

[5.2 System requirements: 12](#_Toc35016838)

[5.3 Running time 13](#_Toc35016839)

[6. Build and run modules 13](#_Toc35016840)

[6.1 Build and run module: VIIRS Swath Projection 13](#_Toc35016841)

[6.1.1 Build 13](#_Toc35016842)

[6.1.2 Run the module 13](#_Toc35016843)

[6.2 Build and run module: VIIRS\_Flood\_Detection 14](#_Toc35016844)

[6.2.1 Build 14](#_Toc35016845)

[6.2.2 Run the module 15](#_Toc35016846)

[6.3 Run module: Mosaick\_Subset\_Flood\_Process\_NC 15](#_Toc35016847)

[6.3.1 Build 15](#_Toc35016848)

[6.3.2 Run the module 15](#_Toc35016849)

[6.4 Run module: Mosaick\_Subset\_Flood\_Process\_NC 16](#_Toc35016850)

[6.4.1 Build 16](#_Toc35016851)

[6.4.2 Run the module 17](#_Toc35016852)

[6.5 Run module: Image Display 18](#_Toc35016853)

[6.5.1 *flood-nc2tif.sh* 18](#_Toc35016854)

[6.5.2 *flood-tif2png.sh* 18](#_Toc35016855)

[6.5.3 *flood-tif2shape.sh* 18](#_Toc35016856)

# Introduction

VIIRS Flood Detection software is designed for global (80S to 80N in latitude) or regional automatic flood detection using Suomi-NPP & NOAA-20 VIIRS data. It consists of five modules: VIIRS\_Swath\_Projection module, VIIRS\_Flood\_Detection module, Mosaick\_Subset\_Flood\_Process\_NC module, VIIRS\_Composition\_NC module and Image\_Display module. With these five modules, the software projects VIIRS Imager bands (SVI01, SVI02, SVI03, SVI05), geometric angles (stored in GITCO), and VIIRS Cloud Mask (JRR-CloudMask) in equidistant cylindrical projection. Then it detects floods in 89-s projected granules through a series of process to generate flood detection result in HDF4 or netCDF4 formats. The based on the detected dataset, it mosaicks the 89-s granules into regional domains and composites the mosaicked domains into daily or 5-day composites. Finall, the software outputs the near real-time or multiple-day composited flood detection results in geotiff, png and shapefile formats.

The software is developed in C/C++ and Python languages, with support from frees software MS2GT0.24 and Gdal, and runs in 64-bit Linux system (recommended).

# Algorithm description

The software utilizes a series of algorithms for flood automations. Figure 1 gives a simple sketch of algorithm flow chart. The details of these algorithms can be found in the following references:

* + SanmeiLi, DonglianSun, Mitchell D.Goldberg & Bill Sjoberg (2015). Object-based automatic terrain shadow removal from SNPP/VIIRS flood maps, International Journal of Remote Sensing, Vol. 36, No. 21, 5504–5522
  + SanmeiLi, DonglianSun, YunyueYu, Ivan Csiszar, Antony Stefanidis & Mitchell D. Goldberg (2012). A New Shortwave Infrared (SWIR) Method for Quantitative Water Fraction Derivation and Evaluation with EOS/MODIS and Landsat/TM data. IEEE Transactions on Geoscience and Remote Sensing, Vol. 51, Issue 3
  + Sanmei Li, Donglian Sun & Yunyue Yu (2013). Automatic cloud-shadow removal from flood/standing water maps using MSG/SEVIRI imagery, International Journal of Remote Sensing, 34:15, 5487-5502
  + Sanmei Li & Donglian Sun, 2013. Development of an integrated high resolution flood product with multi-source data, UMI Dissertations Publishing 2013, ISBN: 9781303635939, <http://search.proquest.com/docview/1492669000>, 2013
  + DonglianSun, YunyueYu, RuiZhang, SanmeiLi, and Mitchel D. Goldberg (2012). Towards Operational Automatic Flood Detection Using EOS/MODIS data. Photogrammetric Engineering & Remote Sensing, 78 (6).



Figure 1 Algorithm flow chart of VIIRS Flood Detection Software

# Module Description

## Module Description

### Module 1: VIIRS Swath Projection

VIIRS Swath Projection Module is to project daytime VIIRS SDR data including SVI01, SVI02, SVI03, SVI05, Solar zenith angle, solar azimuth angle, sensor zenith angle and sensor azimuth angle at 375-m resolution, and VIIRS cloud mask data at 750m data in equidistant cylindrical projection. MS2GT0.24 serves as the main projection tool.

There are two ways for the projection:

* **Project the whole granule:** input GITCO file name and file path, it will automatically search SVI01, SVI02, SVI03, SVI04, SVI05 in the same directory and cloud mask files in the cloud mask directory, and then project the whole granule.
* **Project in subsets according to users’ AOIs:** To project subsets according to users’ AOI, besides the above files, there is an extra txt file to input: User\_AOI\_Definition.txt. Users can add any regions to the txt file before running the software. Table 1 lists the format of User\_AOI\_Definition.txt.

Table 1 Content of User\_AOI\_Definition.txt

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RegionID | Minimal Longitude | Maximal Longitude | Minimal Latitude | Maximal Latitude |
| 1 | -163.0 | -159.0 | 59.0 | 63.0 |
| 2 | -160.0 | -155.0 | 61.0 | 66.0 |
| 3 | -146.0 | -138.0 | 62.0 | 67.0 |
| 4 | -100.0 | -95.0 | 46.0 | 51.0 |

There are several situations that the module doesn’t do any projection:

* Night-time data
* Data with maximal latitude north to 85°N or south to 85°S; or data with average latitude north to 75°N or south to 75°S
* Data outside of the regional geographic range defined in the User\_AOI\_Definition.txt if regional projection is applied

If a granule meets all the above conditions but the granule is with longitude range from 180°W to 180°E, it only projects the majority half hemisphere data in the granule.

### Module 2: VIIRS Flood Detection

This module is to detect flood information using VIIRS projected SDR, cloud mask data and ancillary datasets. The algorithm flow chart and specific inputs/outputs can be seen in Figure 1.

In the following situations, the module doesn’t work:

* 95% of the projected VIIRS SDR data is with fill-values or unreasonable values caused by calibration errors;
* Lack of any necessary data inputs;
* Other system errors such as insufficient memory.

### Module 3: Image Display

This module is developed to output VIIRS flood detection results in geotiff, png and shapefile formats.

### Module 4: Mosaick\_Subset\_Flood\_Process\_NC

This module is used to mosaick VIIRS near real-time flood maps into new maps in the defined domains. Given a defined domain, the module can generate VIIRS flood maps from multiple VIIRS 89-S granules.

### Module 5: VIIRS\_Composition\_NC

This module is used to do multiple-day composition using the mosaicked VIIRS flood datasets or daily composited regional flood datasets, and output multiple-day composite flood products.

# Data Input/Output

### Input/Output of module: VIIRS\_Swath\_Projection

### Input

VIIRS\_Swath\_Projection module requires both real-time VIIRS data input and static ancillary data inputs.

The real-time VIIRS data input includes two parts:

* **VIIRS Imager Bands at 375m resolution (must-have):**
  + SVI01, SVI02, SVI03, SVI05
  + Geometric angels: solar zenith angle, solar azimuth angle, sensor zenith angle and sensor azimuth angle
  + Geolocations files (GITCO)
* **VIIRS EDR Cloud mask at 750m resolution (optional):**
  + JRR-CloudMask\_v2r1,
* **VIIRS CLAVRX cloud mask at 750m resolution (optional):**
  + clavrx\_j01\*.level2.hdf

The real-time datasets should be VIIRS 89-s granules in the naming format:

SVI01\_npp\_d20191217\_t1828141\_e1829382\_b42171\_c20191218174358312388\_nobc\_ops.h5

SVI02\_npp\_d20191217\_t1828141\_e1829382\_b42171\_c20191218174350165150\_nobc\_ops.h5

SVI03\_npp\_d20191217\_t1828141\_e1829382\_b42171\_c20191218174356877471\_nobc\_ops.h5

SVI05\_npp\_d20191217\_t1828141\_e1829382\_b42171\_c20191218174504776480\_nobc\_ops.h5

GITCO\_npp\_d20191217\_t1828141\_e1829382\_b42171\_c20191218161408178449\_nobc\_ops.h5

The VIIRS enterprise cloud mask is in the following naming format:

JRR-CloudMask\_v2r1\_npp\_s201912171828141\_e201912171829382\_c201912171910210.nc

The VIIRS CLAVRX cloud mask is in the following naming format:

clavrx\_j01\_s201912171828141\_e201912171829382\_c201912171910210.nc

If the module is used to project data in regions, then a user-defined text file is required as input. For example, User\_AOI\_Mosaick\_test.txt lists two regions for projection. In this text file, 019 is the region ID, -105 is minimal longitude, -90 is the maximal longitude, 30 is the minimal latitude, and 45 is maximal latitude of the region.

019 -105.0 -90.0 30.0 45.0

020 -90.0 -75.0 30.0 45.0

### Output

The ouputs include the projected VIIRS sdr data and cloud mask data, and a log file: VIIRS\_Swath\_Projection\_log.txt, which stores all the error records during the process:

* **Projected VIIRS sdr data:** hdf5 format with I-band 01, 02, 03 and 05, solar zenith angle, solar azimuth angle, sensor zenith angle and sensor azimuth angle available. For example:

GITCO\_Prj\_SVI\_npp\_d20191217\_t1828141\_e1829382\_b42171\_cspp\_dev\_000.h5

* **Projected VIIRS cloud mask:** hdf5 format. This is an optional input. If unavailable, the module can still go without this input. For example: JRR-CloudMask\_v2r1\_npp\_prj\_s201912171828141\_e201912171829382\_000.h5

### Input/Output of module: VIIRS\_Flood\_Detection

### Input

The inputs of the module VIIRS\_Flood\_Detection include the projected VIIRS SDR and cloud mask data from the module: VIIRS\_Swath\_Projection, which is listed in 4.1.2, and the ancillary data:

**Projected VIIRS sdr and cloud mask data:**

* **Projected VIIRS sdr data:** hdf5 format with I-band 01, 02, 03 and 05, solar zenith angle, solar azimuth angle, sensor zenith angle and sensor azimuth angle available.
* **Projected VIIRS cloud mask:** hdf5 format. This is an optional input. If unavailable, the module can still go without this input.

**Static ancillary data:**

* Global land cover at 1km resolution (raw data)
* Global Digital Elevation Model at 1km resolution (raw data)
* Digital Elevation Model at 375m resolution between 90°N and 60°S (raw data)
* Sun-glint mask (raw data)
* Global land/sea mask at 1km resolution (raw data)
* Global water mask resampled at 375m resolution (raw data)
* Land/sea surface temperature 16-day climatology at 5km resolution (raw data)
* Global Albedo monthly climatology at 5km resolution in visible channel (raw data)
* Pre-trained decision tress and tree attribute files (text files)

### Output

The outputs of this module include a log file: VIIRS\_Flood\_Detection\_Log.txt to store the process errors and the VIIRS flood detected dataset in netCDF format with naming rule:

VIIRS-Flood\_{sat ID}\_s{start time}\_e{end time}\_c{creation time}.nc, e.g.: VIIRS-Flood\_j01\_s202002251909150\_e202002251910395\_c202003102350016.nc

The output includes two datasets: WaterDetection and QualityFlag.

### Input/Output of module: Mosaick\_Subset\_Flood\_Process\_NC

### Input

The inputs of the module Mosaick\_Subset\_Flood\_Process\_NC are the VIIRS flood detection results listed in 4.2.2, and a user-defined text file listing all the regions for mosaicking process.

For example, the text file: User\_AOI\_Mosaick\_NWS.txt lists the 8 NWS domains (Rgion ID min\_longitude max\_longitude min\_latitude max\_latitude):

001 -169.0 -129.0 54.0 72.0

002 -91.0 -66.0 35.0 52.0

003 -106.0 -81.0 37.0 54.0

004 -91.0 -75.0 24.0 36.0

005 -115.0 -90.0 36.0 53.0

006 -115.0 -90.0 23.0 40.0

007 -125.0 -113.0 35.0 52.0

008 -125.0 -113.0 28.0 45.0

The text file: User\_AOI\_Definition\_Global\_VIIRS.txt lists the 136 domains over the global land between 60ºS and 80ºN.

### Output

The outputs of this module are the mosaicked flood dataset in netCDF format and a log file to store all the error records: Mosaick\_Subset\_Flood\_Process\_log.txt. If the domain is a NWS one, then the naming rule is in the following format:

VIIRS-Flood-NWS{domain ID}\_{sat ID}\_s{start time}\_e{end time}\_c{creation time}.nc. e.g.,

VIIRS-Flood\_j01\_NWS019\_s202002251907492\_e202002251913310\_c202003131529589.nc

If the domain is among the 136 global AOIs, then the naming rule is in the following format:

VIIRS-Flood-AOI{domain ID}\_{sat ID}\_s{start time}\_e{end time}\_c{creation time}.nc, e.g.,

VIIRS-Flood\_j01\_AOI019\_s202002251907492\_e202002251913310\_c202003131518369.nc

### Input of module: VIIRS\_Composition\_NC

### Input

The inputs of the module **VIIRS\_Composition\_NC** include the mosaicked VIIRS near real-time flood detection results listed in 4.3.2, a daily composited VIIRS flood detection results generated by this module, and some ancillary data:

* VIIRS near real-time flood data in the eight NWS domains or the 136 AOIs
* VIIRS daily composited flood dataset in the eight NWS domains or the 136 AOIs
* Global land/sea mask at 1km resolution (raw data)
* Global water mask resampled at 375m resolution (raw data)

### Output

The outputs of this module are the daily or 5-day composited VIIRS flood detection data in the 8 NWS domains or the 136 AOIs, and a log file to store all the error records: VIIRS\_Composite\_log.txt.

The naming rule a composited data in a NWS domain is in the following format:

VIIRS-Flood-{n}day-NWS{region ID}\_blend\_s{start time}\_e{end time}\_c{creation time}.nc

The naming rule a composited data among the 136 AOIs is in the following format:

VIIRS-Flood-{n}day-AOI{region ID}\_blend\_s{start time}\_e{end time}\_c{creation time}.nc

{n} could be 1 or 5.

### Input of module: Image\_Display

### Input

### Input of *flood-nc2tif.sh*

The input of the module *flood-nc2tif.sh* is any netCDF file output by the modules: VIIRS\_Flood\_Detection, Mosaick\_Subset\_Flood\_Process\_NC, and VIIRS\_Composition\_NC.

### Input of *flood-tif2png.sh*

The input of the module *flood-tif2png.sh* is any geotiff file output by *flood-nc2tif.sh.*

### Input of *flood-tif2shape.sh*

The input of the module *flood-tif2shape.sh* is any geotiff file output by *flood-nc2tif.sh.*

### Output

### Output of *flood-nc2tif.sh*

The outputs of the module *flood-nc2tif.sh* are the VIIRS flood maps in geotiff format. The naming rules of these geotiff files are the same of the input netCDF files with the suffix “.tif”. The creation date and time in the geotiff files are the ones that the geotiff files are created.

### Output of *flood-tif2png.sh*

The outputs of the module *flood-tif2png.sh* are the VIIRS flood maps in png format. The naming rules of these png files are the same of the input geotiff files with the suffix “.png”. The creation date and time in the png files are the ones that the png files are created.

### Output of *flood-tif2shape.sh*

The outputs of the module *flood-tif2shape.sh* are the zip files with 1 vector file in shapefile format with five types of polygons: 2: 1≤WF≤19; 4: 20≤WF≤39; 6: 40≤WF≤59; 8: 60≤WF≤79; 10: 80≤WF≤100. The naming rules of these zip files are the same of the input geotiff files with the suffix “.shapefiles.zip”. The creation date and time in the zip files are the ones that the zip files are created.

Figure 2 presents the framework of the VIIRS flood detection system.



Figure 2 Framework of the VIIRS flood detection system

# Environment requirements

## Source Codes:

**VIIRS Swath Projection module**: written in C/C++, and python 3.6 or newer versions are required to run MS2GT0.24.

**VIIRS Flood Detection module**: written in C/C++.

**Mosaick\_Subset\_Flood\_Process\_NC:** written in C/C++.

**VIIRS\_Composition\_NC:** written in C/C++.

**Image Display module**: written in scripts to run gdal software.

## System requirements:

The software is recommended to run in Linux 64-bit system with at least 8GB memory.

To compile, build and run the software, the **GNU Compiler Collection** including GCC/GCC C++, and python 3.6 are required. Besides that, MS2GT0.24 version should also be installed before building and running the software.

The installation of MSG2GT0.24 is available in this website:

<https://nsidc.org/data/modis/ms2gt/>

Other libraries required to build and run the software include:

* Szip libraries
* Zlib libraries
* Hdf4 libraries (hdf4 4.2.6 or newer versions)
* Hdf5 libraries (hdf5 1.8.8 or newer versions)
* NetCDF4 libraries (netCDF 4.0 or newer versions)
* Python3.6 or new versions with:
  + Numpy
  + Gdal (These gdal commands must be available in $PATH: gdal\_translate, gdalwarp, gdal\_calc.py, gdal\_polygonize.py), compiled with python bindings, NetCDF4, shapefile, & GeoTIFF support
  + zip (unix command)
  + mktemp (unix command)

## Running time

Running time depends on the region size and flooding situation. For example, it takes about 25 minutes to finish all the process in Alaska (size: 11862×5338) under clear-sky condition.

# Build and run modules

## Build and run module: VIIRS Swath Projection

### Build

Before build this module, make sure MS2GT0.24 is installed, Python paths are set and added in the ${path} correctly, remap.py is executable, szip, zlib, hdf5 and netcdf4.0 libraries are installed.

Source codes are under /src directory and Makefile is under /build directory.

To build the module, open Makefile and modify the szip and hdf5 library and include variable paths.

If everything is set, then in a Linux terminal run the following commands:

*$make clean*

*$make*

Then all the objects are built under /build directory and the executive module *VIIRS\_Swath\_Projection*in created in the root directory.

### Run the module

The parameters to run *VIIRS Swath Projection*include:

* **-i**: [Necessary], VIIRS sdr data path, for example, /home/VIIRS/sdr/
* **-a**: [Necessary], path of users’ AOI definition file path, for example, /home/ VIIRS /prj/. If -n is set as 0, it is an optional parameter. Otherwise, it is a necessary parameter.
* **-u**: file name of users’ AOI definition file, for example, User\_AOI\_Definition.txt. If -n is set as 0, it is an optional parameter. Otherwise, it is a necessary parameter.
* **-g**: [Necessary], VIIRS SDR375-m terrain-corrected geo-location filename
* **-o**: [Necessary], file path for projected SDR data, for example, /home/VIIRS/prj/
* **-l:** [Necessary], log file path, for example, /home/VIIRS/logfile/
* **-n**: [Necessary], switch of projection range between projecting whole granule and projecting users’ subsets: 0 is to project the whole granule, and 1 is to project subsets according to users’ AOIs which are listed in users’ AOI definition txt file.

To run the executive module: *VIIRS\_Swath\_Projection* in subsets with VIIRS SDR data: *GITCO\_j01\_d20200225\_t1909150\_e1910395\_b11768\_c20200226160606186943\_nobc\_ops.h5*, the test script is written as:

*./ VIIRS\_Swath\_Projection -i /home/VIIRS/sdr/ -a /home/ VIIRS /prj/ -u User\_AOI\_Definition.txt -g GITCO\_j01\_d20200225\_t1909150\_e1910395\_b11768\_c20200226160606186943\_nobc\_ops.h5 -o /home/VIIRS/prj/ -n 1*

To run the executive module: *VIIRS\_Swath\_Projection* in enitre granule, the test script is written as:

*./ VIIRS\_Swath\_Projection -i /home/VIIRS/sdr/ -a /home/ VIIRS /prj/ -u User\_AOI\_Definition.txt -g GITCO\_j01\_d20200225\_t1909150\_e1910395\_b11768\_c20200226160606186943\_nobc\_ops.h5 -o /home/VIIRS/prj/ -n 0*

Or:

*./ VIIRS\_Swath\_Projection -i /home/VIIRS/sdr/ -g GITCO\_j01\_d20200225\_t1909150\_e1910395\_b11768\_c20200226160606186943\_nobc\_ops.h5 -o /home/VIIRS/prj/ -n 0*

## Build and run module: VIIRS\_Flood\_Detection

### Build

Before build this module, make sure szip, zlib, netcdf4, hdf4 and hdf5 libraries are installed correclty.

Source codes are under /src directory and Makefile is under /build directory.

To build the module, open Makefile and modify the szip, hdf4 and hdf5 library and include variable paths.

If everything is set, then in a Linux terminal run the following commands:

*$make clean*

*$make*

Then all the objects are built under /build directory and the executive module *VIIRS\_Flood\_Detection* in created in the root directory.

### Run the module

There are five parameters (each one with an index name such as -h):

* **-h: [Necessary]**, VIIRS projected hdf5 file path, for example, /home/VIIRS/prj
* **-a: [Necessary]**, ancillary data file path, for example, /home/VIIRS/assdata
* **-v: [Necessary]**, VIIRS sdr projected 375-m geo-location filename,
* **-o: [Necessary]**, file path of the output flood dataset in hdf4 format
* **-c: [Optional]**, VIIRS projected cloud mask file path, for example, /home/VIIRS/prj
* **-l: [Necessary]**, log file path, for example, /home/VIIRS/logfile
* **-f: [Necessary]**, output data format1: hdf4, 2: netCDF4

To run the executive module: *VIIRS\_Flood\_Detection* with projected data: *GITCO\_Prj\_SVI\_j01\_d20200225\_t1909150\_e1910395\_b11768\_cspp\_dev\_000.h5* and output the results in netCDF4 format, the test script is written as:

*./ VIIRS\_Flood\_Detection -h /home/VIIRS/prj -a /home/VIIRS/assdata -v GITCO\_Prj\_SVI\_j01\_d20200225\_t1909150\_e1910395\_b11768\_cspp\_dev\_000.h5 -o /home/VIIRS/prj/output -c /home/VIIRS/prj -l /home/VIIRS/logfile –f 2*

## Run module: Mosaick\_Subset\_Flood\_Process\_NC

### Build

Before build this module, make sure netcdf4, and hdf5 libraries are installed correclty.

Source codes are under /src directory and makefile is under /build directory.

To build the module, open makefile and modify the paths to the netcdf4 and hdf5 libraries.

If everything is set, then in a Linux terminal run the following commands:

$make clean

$make

Then all the objects are built under /build directory and the executive module Mosaick\_Subset\_Flood\_Process\_NC in created in the root directory.

### Run the module

The parameters to run **Mosaick\_Subset\_Flood\_Process\_NC** include:

* **-h: [Necessary]**, file path of VIIRS near real-time flood dataset files
* **-s: [Necessary]**, sensor name: VIIRS
* **-m: [Necessary]**, satellite name: npp or j01
* **-a:** **[Necessary]**, file path of the ancillary files
* **-k: [Necessary]**, file path of the mosaciked files
* **-f**: **[Necessary]**, filename of user-defined domain text file
* **-d**: **[Necessary]**, dates of the files to mosaick, e.g.: 20190808
* **-l**: **[Necessary]**, file path of the log file

Examples:

To mosaick NOAA-20/VIIRS near real-time flood maps defined in the User\_AOI\_Definition\_WestHemisphere.txt from VIIRS 89-s granules on Feb. 25, 2020:

*./ Mosaick\_Subset\_Flood\_Process\_NC -h /data/VIIRS -s VIIRS -m j01 -a /data/assdata -k /data/VIIRS /mosaick -f User\_AOI\_Definition\_WestHemisphere.txt -d 20200225 -l /data/logfile*

If there are NOAA-20 VIIRS 89-s granules from 19:06 to 19:16 (UTC) in the folder: /data/VIIRS:

VIIRS-Flood\_j01\_s202002251906252\_e202002251907480\_c202003102345015.nc

VIIRS-Flood\_j01\_s202002251907492\_e202002251909137\_c202003102347016.nc

VIIRS-Flood\_j01\_s202002251909150\_e202002251910395\_c202003102350016.nc

VIIRS-Flood\_j01\_s202002251910407\_e202002251912052\_c202003102353052.nc

VIIRS-Flood\_j01\_s202002251912065\_e202002251913310\_c202003102355051.nc

VIIRS-Flood\_j01\_s202002251913322\_e202002251914550\_c202003102359052.nc

VIIRS-Flood\_j01\_s202002251914562\_e202002251916207\_c202003110003008.nc

VIIRS-Flood\_j01\_s202002251916220\_e202002251917465\_c202003110006017.nc

For region 19 defined in the User\_AOI\_Definition\_WestHemisphere.txt, In region 19 (019 -105.0 -90.0 30.0 45.0), there will be four files generated in the folder /data/VIIRS /mosaick:

VIIRS-Flood\_j01\_AOI019\_s202002251907492\_e202002251909137\_c202003111744059.nc

VIIRS-Flood\_j01\_AOI019\_s202002251907492\_e202002251910395\_c202003111744209.nc

VIIRS-Flood\_j01\_AOI019\_s202002251907492\_e202002251912052\_c202003111744366.nc

VIIRS-Flood\_j01\_AOI019\_s202002251907492\_e202002251913310\_c202003111744548.nc

## Run module: Mosaick\_Subset\_Flood\_Process\_NC

### Build

Before build this module, make sure netcdf4, and hdf5 libraries are installed correclty.

Source codes are under /src directory and makefile is under /build directory.

To build the module, open makefile and modify the paths to the netcdf4 and hdf5 libraries.

If everything is set, then in a Linux terminal run the following commands:

$make clean

$make

Then all the objects are built under /build directory and the executive module **VIIRS\_Composition\_NC** in created in the root directory.

### Run the module

The parameters to run **VIIRS\_Composition\_NC** include:

* **-h:** [Necessary], file path of VIIRS near real-time flood dataset files
* **-v:** [Necessary], filename of the VIIRS near real-time flood dataset file with the last date among all the files. For example, if to composite VIIRS granules from Aug. 01 to 05, 2019, the filename must be a VIIRS near real-time file on Aug. 05, 2019.
* **-a:** [Necessary], file path of the ancillary files
* **-o:** [Necessary], file path of the output composited results
* **-l**: [Necessary], file path of the log file
* **-r**: [Necessary], index of region: NWS or AOI
* **-n**: [Necessary], number of composited days, e.g. 1 or 5 days

Examples:

To do daily composition on VIIRS files:

VIIRS-Flood\_j01\_AOI019\_s202002251731064\_e202002251732309\_c202003131516086.nc

VIIRS-Flood\_j01\_AOI019\_s202002251907492\_e202002251913310\_c202003131518369.nc

VIIRS-Flood\_npp\_AOI019\_s202002251816367\_e202002251823425\_c202003131749085.nc

VIIRS-Flood\_npp\_AOI019\_s202002251957369\_e202002252003173\_c202003131751003.nc

The script can be written as:

*./ VIIRS\_Composition\_NC -h /data/VIIRS -v*

*VIIRS-Flood\_j01\_AOI019\_s202002251907492\_e202002251913310\_c202003131518369.nc -a /data/ assdata -o /data/VIIRS -l /data/ logfile –r AOI -n 1*

Then the module will output a 1-day composited dataset in the folder */data/prj*:

VIIRS-Flood-1day\_AOI019\_blend\_s202002251731064\_e202002252003173\_c202003131802398.nc

To do 5-day composition on VIIRS files:

WATER\_VIIRS\_Prj\_SVI\_npp\_d20190225\_t1710\_t1710\_cspp\_dev\_158\_4152\_6524\_01.hdf

WATER\_VIIRS\_Prj\_SVI\_npp\_d20190227\_t1806\_t1811\_cspp\_dev\_158\_4152\_6524\_01.hdf

WATER\_VIIRS\_Prj\_SVI\_npp\_d20190227\_t1948\_t1948\_cspp\_dev\_158\_4152\_6524\_01.hdf

WATER\_VIIRS\_Prj\_SVI\_j01\_d20190227\_t1715\_t1720\_cspp\_dev\_158\_4152\_6524\_01.hdf

WATER\_VIIRS\_Prj\_SVI\_j01\_d20190226\_t1914\_t1920\_cspp\_dev\_158\_4152\_6524\_01.hdf

*The script can be written as if using near real-time mosaicked data:*

*./VIIRS\_Composition\_NC -h /data/VIIRS -v*

*VIIRS-Flood\_j01\_AOI019\_s202002251907492\_e202002251913310\_c202003131518369.nc -a /data/ assdata -o /data/VIIRS -l /data/ logfile –r AOI -n 5*

Or The script can be written as if using daily composited data:

*./VIIRS\_Composition\_NC -h /data/VIIRS -v*

*VIIRS-Flood-1day\_AOI019\_blend\_s202002251731064\_e202002252003173\_c202003131802398.nc -a /data/ assdata -o /data/VIIRS -l /data/ logfile –r AOI -n 5*

Both the two scripts will output a 5-day composited dataset from Feb. 21 to 25, 2020 in the folder */data/VIIRS*:

VIIRS-Flood-5day\_AOI019\_blend\_s202002211748283 \_e202002252003173\_c202003131802398.nc

## Run module: Image Display

There are three scripts in the Image display module: *flood-nc2tif.sh, flood-tif2png.sh* and *flood-tif2shape.sh.*To run the three scripts, the included runtime, suitable for CentOS 7 (and like) systems can be directly used by setting up the paths in the environment:

export PATH="$PWD"/runtimes/ShellB3/bin:"$PATH"

export GDAL\_DATA="$PWD"/runtimes/ShellB3/share/gdal/

In this way, no libraries are required.

Otherwise, users can set up the environment by themselves and the following libraries are required:

* Python 3.6, with:

numpy

gdal, compiled with python bindings, NetCDF4, shapefile, & GeoTIFF support (These gdal commands must be available in $PATH: gdal\_translate, gdalwarp, gdal\_calc.py, gdal\_polygonize.py)

* zip (unix command)
* mktemp (unix command)

### *flood-nc2tif.sh*

This script creates a geotiff file from the input netCDF file. For example, to create a geotiff file from nc file:

VIIRS-Flood\_j01\_s202002251729040\_e202002251731005\_c202003011858011.nc, the script can be written as:

flood-nc2tif.sh VIIRS-Flood\_j01\_s202002251729040\_e202002251731005\_c202003011858011.nc

And it will create a geotiff file:

VIIRS-Flood\_j01\_s202002251729040\_e202002251731005\_c202003011858011.tif

### *flood-tif2png.sh*

This script creates a png file from the input geotiff file. For example, to create a png file from the geotiff file:

VIIRS-Flood\_j01\_s202002251729040\_e202002251731005\_c202003011858011.tif, the script can be written as:

*flood-tif2png.sh* VIIRS-Flood\_j01\_s202002251729040\_e202002251731005\_c202003011858011.tif

And it will create a png file:

VIIRS-Flood\_j01\_s202002251729040\_e202002251731005\_c202003011858011.png

### *flood-tif2shape.sh*

This script creates a vector file in shapefile format from the input geotiff file. For example, to create a shapefile vector file from the geotiff file:

VIIRS-Flood\_j01\_s202002251729040\_e202002251731005\_c202003011858011.tif, the script can be written as:

*flood-tif2shape.sh* VIIRS-Flood\_j01\_s202002251729040\_e202002251731005\_c202003011858011.tif

And it will create a zipped shapefile vector file:

VIIRS-Flood\_j01\_s202002251729040\_e202002251731005\_c202003011858011. shapefiles.zip