



GRASS GIS 7.4: What's new in a nutshell

Markus Neteler, Veronica Andreo, Luca Delucchi, Martin Landa, Moritz Lennert, Vaclav P
GRASS Development Team

grass.osgeo.org

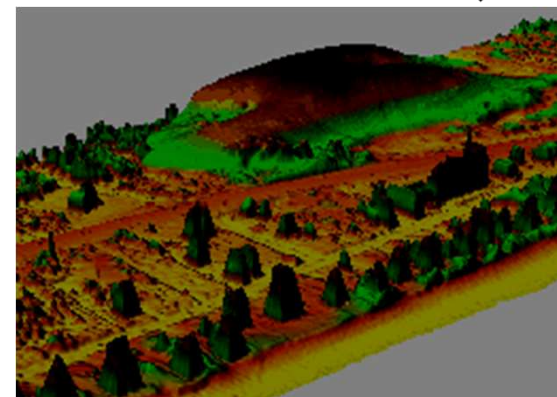
FOSS4G 2018 – Dar es Salaam



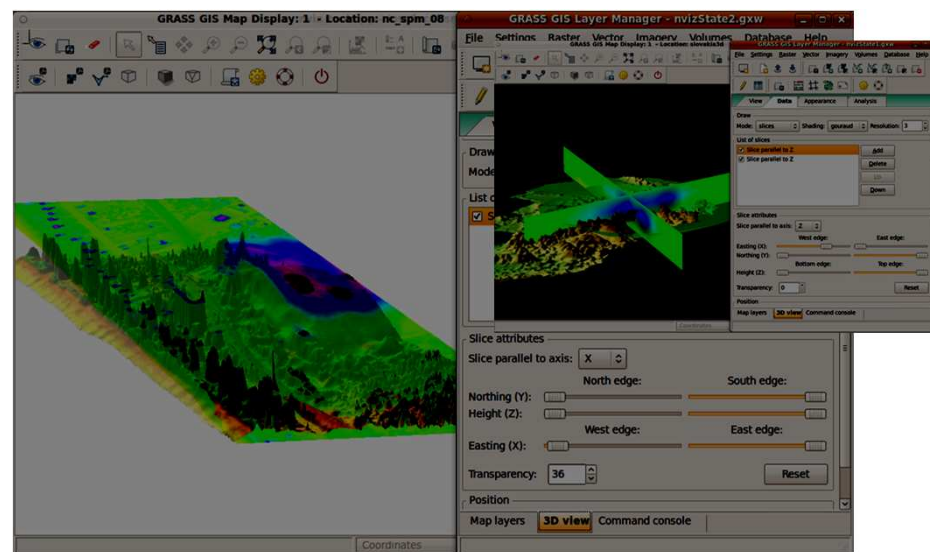


What is GRASS GIS?

- .GRASS GIS is a hybrid, modular GIS software
- .GRASS = Geographic Resources Analysis Support System
- .GNU General Public License - freely available
- .Raster and topological vector data functionality
- .3D raster (voxel) processing
- .Image processing
- .Visualization options
- .Time series analysis
- .Portable software ("all" operating systems)
- .Graphical user interface and command line



Nags Head LiDAR time series: dune moving over time





What is new in GRASS GIS 7.4?

New stable version GRASS GIS 7.4

- Improved usability and graphical user interface
- New compression of internal "no data" file which can be huge
- Support for global data extending beyond -180/+180, -90/+90
- Orthorectification with user interface was newly implemented in GRASS GIS 7
- GUI: New *Download* button for sample data
- ... over 480 improvements since G7.2.0



Data catalog improvements



GRASS GIS 7.4.0svn Layer Manager

File Settings Raster Vector Imagery 3D raster Database Temporal Help

Search:

GRASS locations in /opt/grassdata

- gismentors
 - PERMANENT
 - dibavod
 - ochrana_uzemi
 - osm
 - ruian
 - ruian_praha
 - user1
- subdayprecip-wgs84
 - PERMANENT
 - vector
 - ufa

Layers Console Modu

1. Copy

2. Paste

3. Reprojection

Reprojection

Map layer <ufa> needs to be reprojected.
Please review and modify reprojection parameters:

Maximum segment length:

For more reprojection options, please see v.proj

Close Reproject

Data tab (catalog): Copying of raster and vector maps between

New Orthorectification GUI



Manage Ground Control Points

use	source E	source N	source Z	target E	target N	target Z	Forward error	Backward error
1	3433.76399027	4013.92944039	0.0	635890.539036	5082323.73716	700	103893.989338	1206.797055
2	5663.63017032	3315.20681265	0.0	630698.420898	5083666.03223	750	75667.125929	2593.658967
3	3484.43309002	4965.99756691	0.0	635880.958794	5080131.73067	750	118984.847243	2147.705000
4	3519.34793187	4907.66423358	0.0	635757.957044	5080294.5271	720.3481	115606.753734	2096.784943
5	1980.11435523	4745.01216545	0.0	639373.871778	5080707.57861	500	803733.994864	2206.519653
6	4006.45255474	2026.76399027	0.0	634353.593597	5086780.06445	850	189157.523637	1216.300242
7	-3868.13043706	-4309.67306804	0.0	632330.631784	5081340.00072	700	347670.528834	1469.770170

Source Display

Target Display

GRASS GIS Map Swipe

Graphical Modeller



GRASS GIS 7.4.1svn Layer Manager

File Settings Raster Vector Imagery 3D raster Database Temporal Help

Display 1

stary@ruian

dalnice5km

GRASS GIS Map Display: 1 - gismentors/user1

2D view

GRASS GIS Graphical Modeller - buffer.gxm*

File Settings Model Help

input silnice@osm

(1) v.extract

output/input dalnice

output dalnice

(2) v.buffer

Remove

Display

Properties

Model Items Variables Python editor Command output

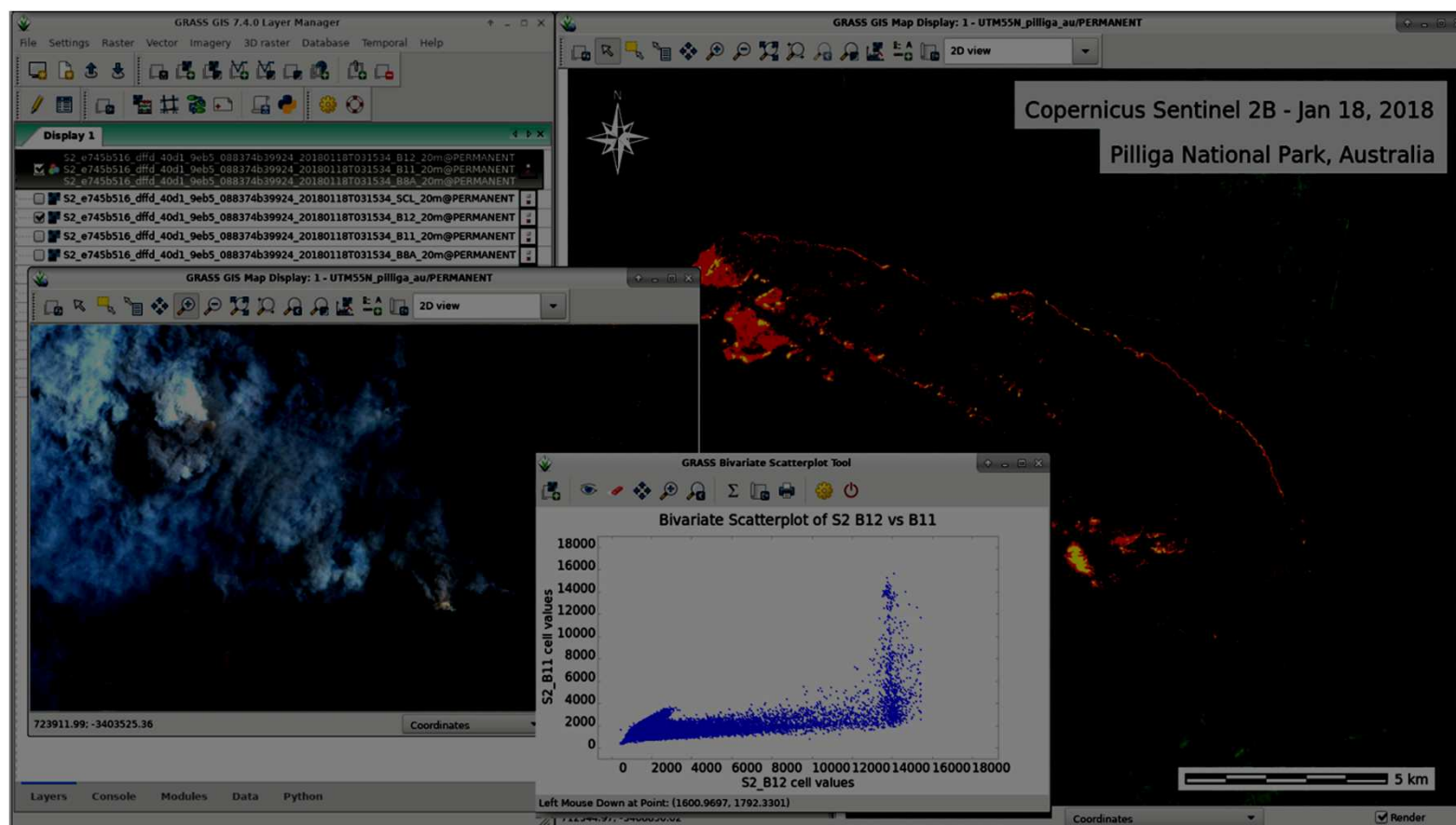
- mark data to be displayed
- print computational time elapsed
- delete intermediate data when computation finished
- export to Python



Copernicus Sentinel-2 processing

New addons: *i.sentinel.download*, *i.sentinel.import*, *i.sentinel.preproc* and *i.se*

Example:
Wildfire in
Australia





Python Editor

Integrated Python editor for rapid prototyping

Example:
Vector buffer

The screenshot shows the GRASS GIS 7.4.1 interface. A 'Simple Python Editor' window is open, displaying the following Python code:

```
#!/usr/bin/env python
import grass.script as gscript

def main():
    streets = "streets"
    buffer = "street_buffer"
    gscript.run_command('v.buffer',
                       input=streets, output=buffer,
                       distance=10)

if __name__ == '__main__':
    main()
```

The background shows a 'Map Display' window with a 2D view of a road network. The roads are highlighted in orange, representing a vector buffer. The interface includes a 'Layer Manager' on the left, a 'Layers' panel at the bottom, and a status bar at the bottom right showing coordinates (639058.18; 236085.12) and a 'Render' button.

Vaclav Petras

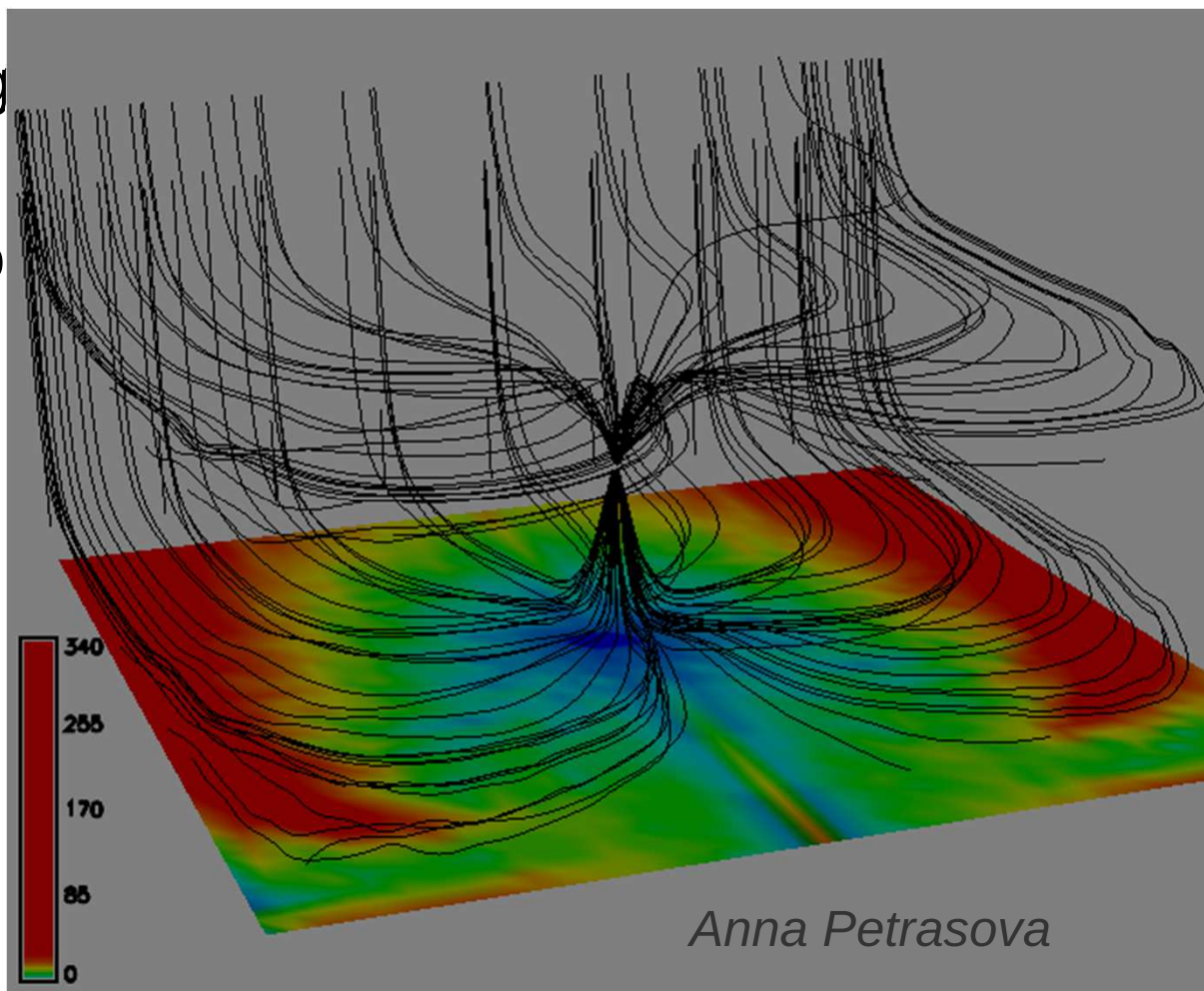


3D raster gradients and flowlines

3D raster (voxel) processing

[r3.flow](#) and [r3.gradient](#) to co

and re





TGRASS: *t.rast.algebra* and *t.rast3d.algebra*: temporal algebra

Compute annual hydro-thermal coefficients (HTC) from daily climate data

$$HTC = \frac{\sum P_{(T>10^{\circ}C)}}{\sum T_{(T>10^{\circ}C)} \cdot \frac{1}{10}}$$

T := daily temperatures,
P := daily precipitation

~ 60 years of daily data, each pixel in time = virtual meteo station

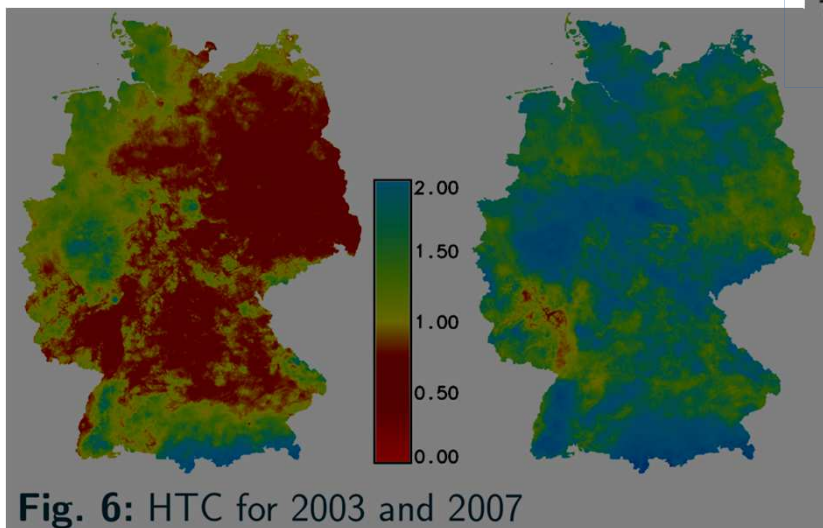


Fig. 6: HTC for 2003 and 2007

Leppelt & Gebbert, EGU 2015

```
t.rast.algebra "HTC = (D {+,contains,1} if(T >= 10, P, 0)) /  
(D {+,contains,1} if(T >= 10, T / 10, 0))"
```

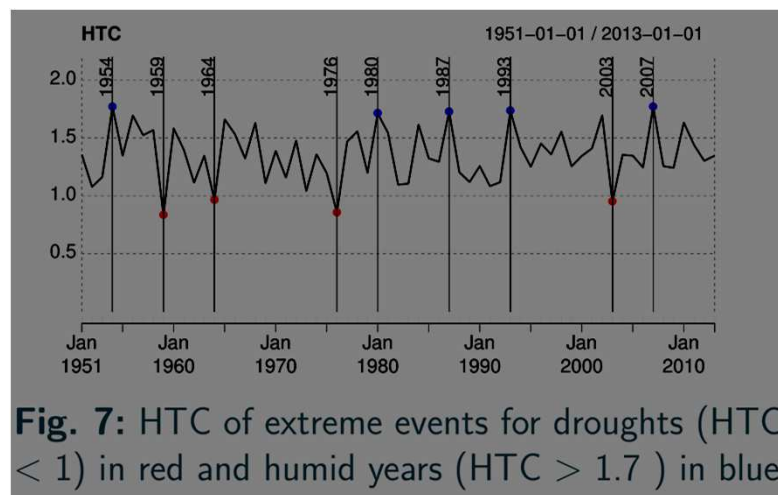


Fig. 7: HTC of extreme events for droughts (HTC < 1) in red and humid years (HTC > 1.7) in blue



GRASS GIS and Python

Using GRASS GIS from “outside” through “grass-session”

pip install grass-session

Now it's easy to use GRASS GIS
as a processing backend in Python!

Combine with GDAL, OTB, ...

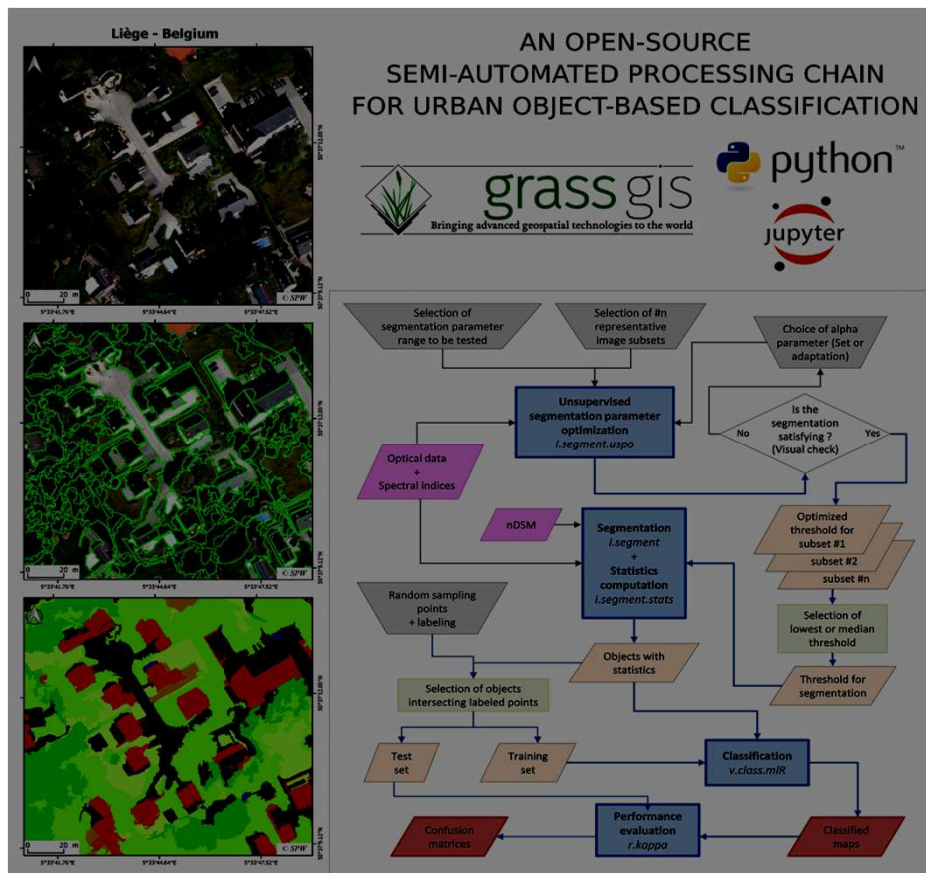
```
#!/usr/bin/env python
# filename: test_session.py

from grass_session import Session
from grass.script import core as gcore

# create a new location from EPSG code (can also be a GeoTIFF or SHP or ... file)
with Session(gisdb="/tmp", location="location",
             create_opts="EPSG:4326"):
    # do something in permanent
    print(gcore.parse_command("g.gisenv", flags="s"))
# {u'GISDBASE': u''/tmp/';",
# u'LOCATION_NAME': u''epsg3035';",
# u'MAPSET': u''PERMANENT';",}

# create a new mapset in an existing location
with Session(gisdb="/tmp", location="location", mapset="test",
             create_opts=""):
    # do something in the test mapset.
    print(gcore.parse_command("g.gisenv", flags="s"))
# {u'GISDBASE': u''/tmp/';",
# u'LOCATION_NAME': u''epsg3035';",
# u'MAPSET': u''test';",}
```

Remote sensing in GRASS GIS: object-based image analysis



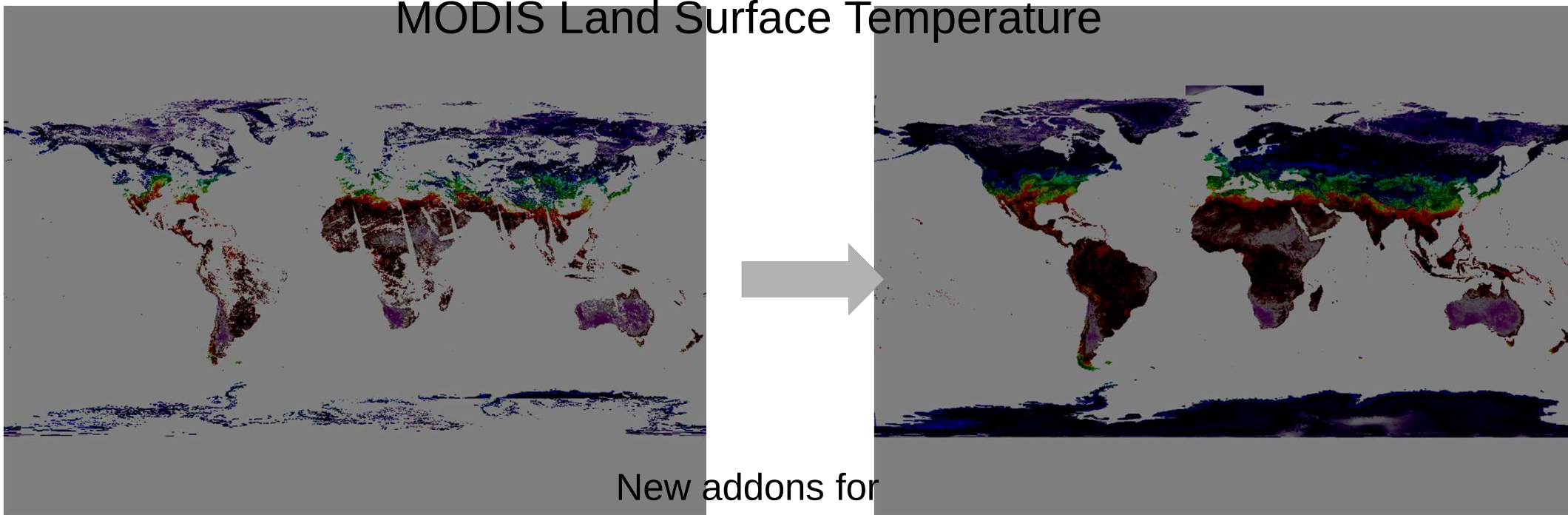
- Complete toolchain from segmentation
- Including
 - unsupervised segmentation parameter optimization
 - high performance object statistics calculation
 - module-level parallelization
- Recently created module for SLIC support

Source : <http://dx.doi.org/10.3390/rs9040358>



High-performance computing

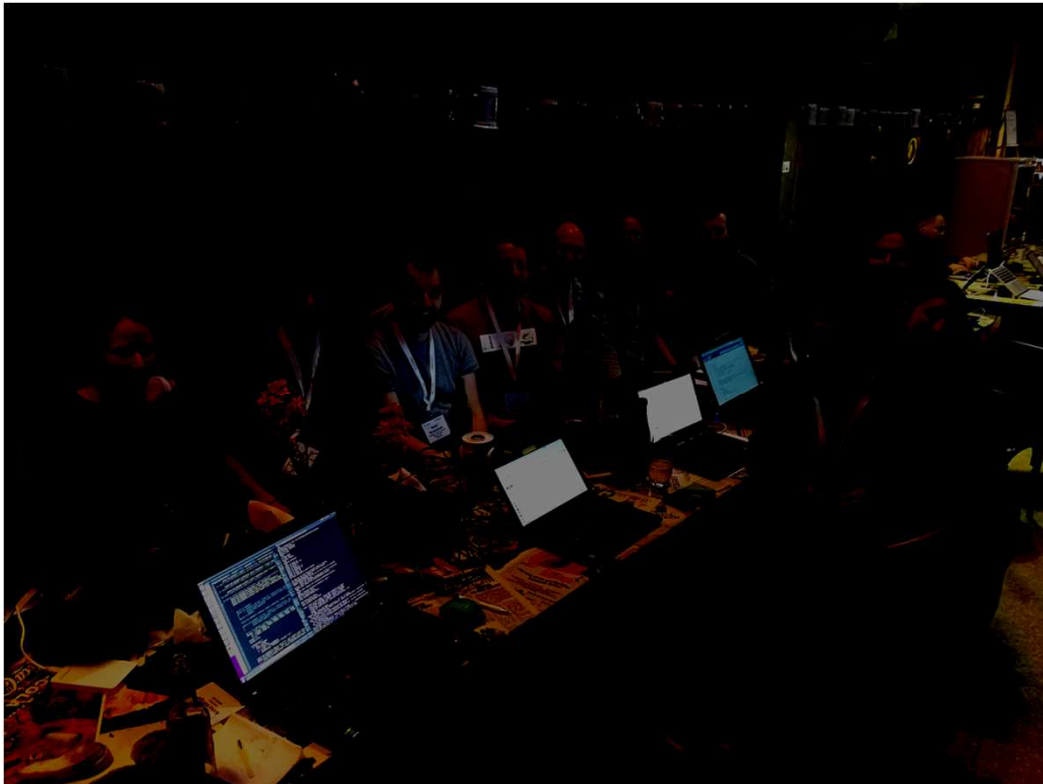
MODIS Land Surface Temperature



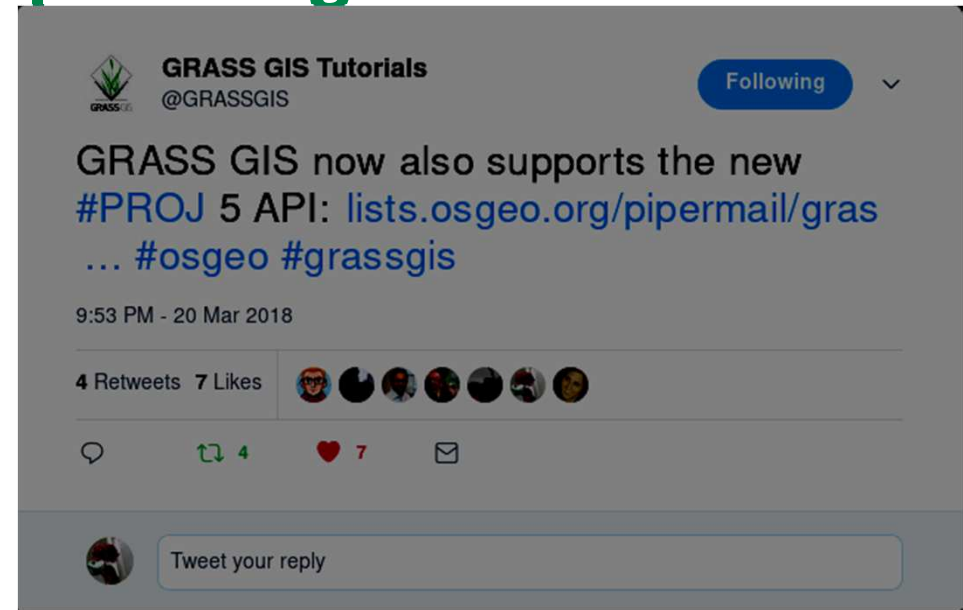
temporal + spatial processing for reconstruction of missing pixels

Data: <https://zenodo.org/record/1135230>

Community activities: Code Sprint 2018 at FOSSGIS Bonn Basecamp – Integration



20 March 2018





Community activities: Google Code-IN for 13-17 year old pre-university students

https://grasswiki.osgeo.org/wiki/GRASS_GCI_Ideas_2017

3.1 **Install** GRASS GIS on your computer and
download North Carolina dataset

3.2 **Compile** GRASS GIS

3.3 Add examples and/or screenshots to different **manual** pages

3.4 Add **test suites** to different modules

3.5 **Designs**

3.5.1 Splash screen for GRASS GIS GUI start-up

3.5.2 T-shirt for 2018 Code Sprint

3.5.3 Banner for location wizard

3.6 **Blog** entry about GRASS GIS

3.7 **Videos**

3.7.1 How to create a location

Around 120 students



Community activities: GSoC 2018

Google Summer of Code 2018

<https://trac.osgeo.org/grass/wiki/GSoC/2018>

OSS-Fuzz - Continuous **Fuzzing** for Open Source Software for GRASS GIS

Implement a series of **image fusion** algorithms in GRASS GIS

Enhance 3D **rendering** capabilities in GRASS GIS

Additional functionality for running GRASS GIS modules in **Jupyter** Notebook

Integration of **PDAL** into GRASS GIS

Benchmarking framework for GRASS GIS

GRASS GIS as a post-processing part of **WebODM**

Additional **GUI** tools for image analysis

Module to create quadtree **tiling**

Tools for generating **unit tests** from examples in the manual

Mapnik rendering engine for GRASS GIS

Generalized GUI code for **Qt-based GUI**

GRASS GIS **3D viewer** NVIZ module independent of the main GUI

Integration of v.profile into **GUI** profiling tool

Add **CMake** build system for GRASS GIS

Add a cloud masking module for **Sentinel** data in GRASS GIS

Full support of **Python 3** in GRASS GIS

Improve GRASS integration in **QGIS 3**

New easy-to-use **CLI** and **API** for GRASS GIS

Thanks for your attention!



GRASS GIS

grass.osgeo.org