

Processing UAV and lidar point clouds in GRASS GIS

XXIII ISPRS Congress, Prague 2016

Vaclav Petras (Vashek)

Anna Petrasova, Justyna Jeziorska, Helena Mitasova

Center for Geospatial Analytics

NC STATE UNIVERSITY

July, 2016

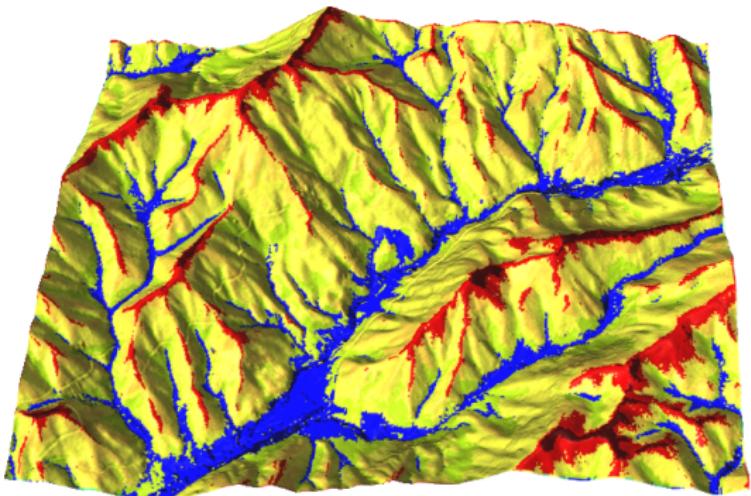


available at

wenzeslaus.github.io/grass-lidar-talks

Providing algorithms to the community

- ▶ new landform recognition approach – geomorphons
- ▶ by Jasiewicz and Stepinski from AMU, Poland and University of Cincinnati, USA
- ▶ not just a paper
Geomorphology, 2013
- ▶ not just a code
at some webpage
- ▶ *r.geomorphon*
module in GRASS GIS addons repository



GRASS GIS

- ▶ all in one
 - ▶ hydrology modeling, image segmentation, point clustering, ...
- ▶ driven by needs of users
 - ▶ direct access to development process
- ▶ from small laptops to supercomputers
 - ▶ Raspberry Pi, Windows, Mac, GNU/Linux, FreeBSD, IBM AIX
- ▶ learn now, use forever
 - ▶ over 30 years of development and interface refinement
- ▶ used by
 - ▶ US Oak Ridge National Laboratory, Edmund Mach Foundation, JRC, ...

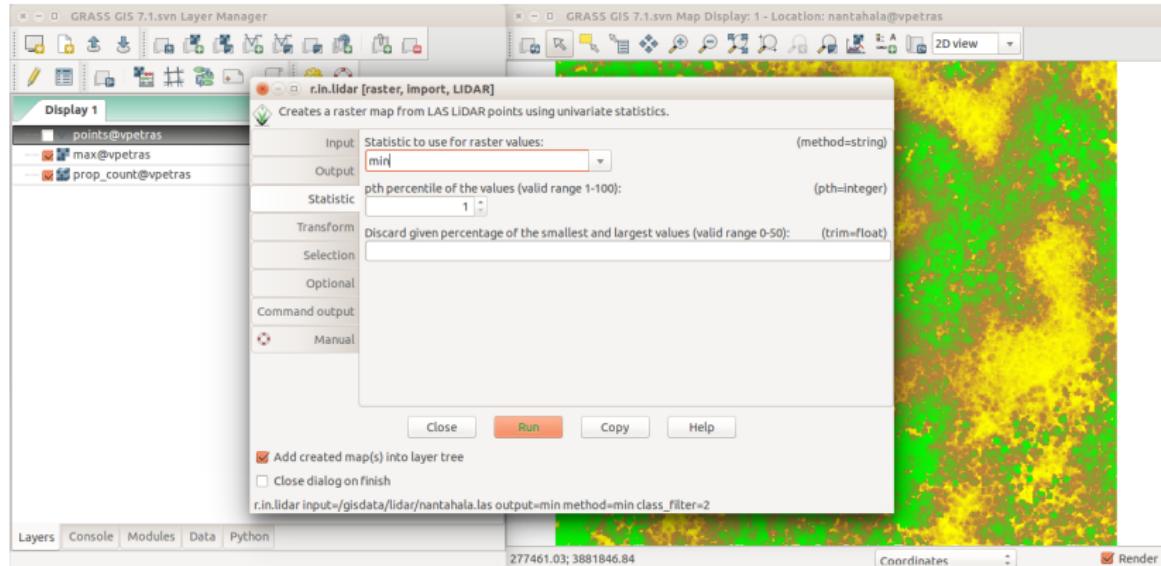


GRASS GIS

latest release 7.0.4

May 1, 2016

GUI



Python and command line interfaces

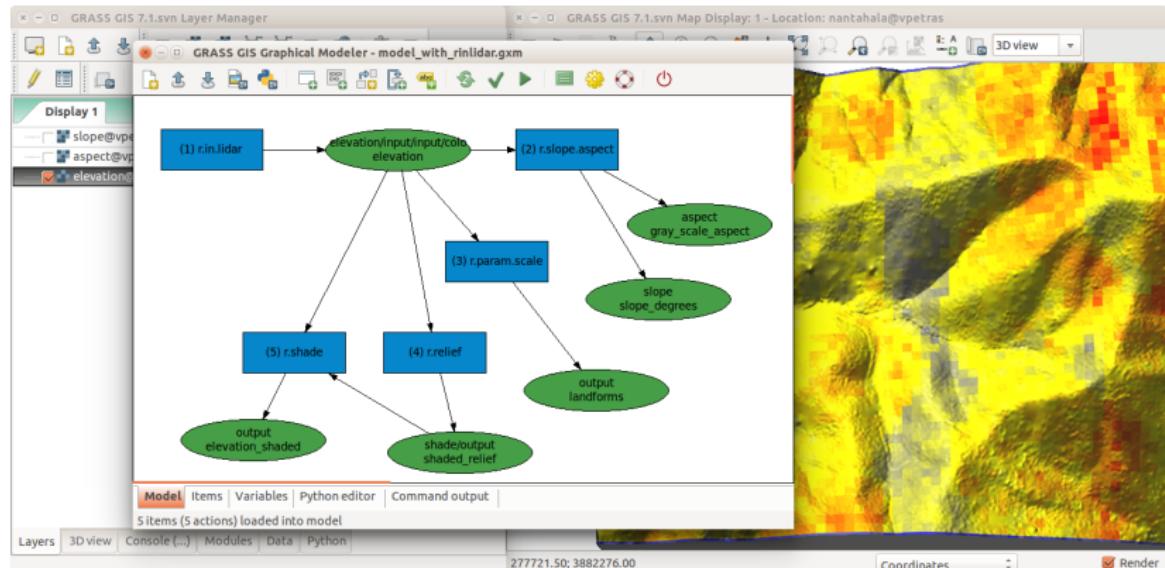
Command Line:

```
r.in.lidar input=points.las \
    output=elevation -e
```

Python:

```
from grass.script import run_command
run_command('r.in.lidar',
            input="points.las",
            output="elevation",
            flags='e')
```

Graphical Modeler



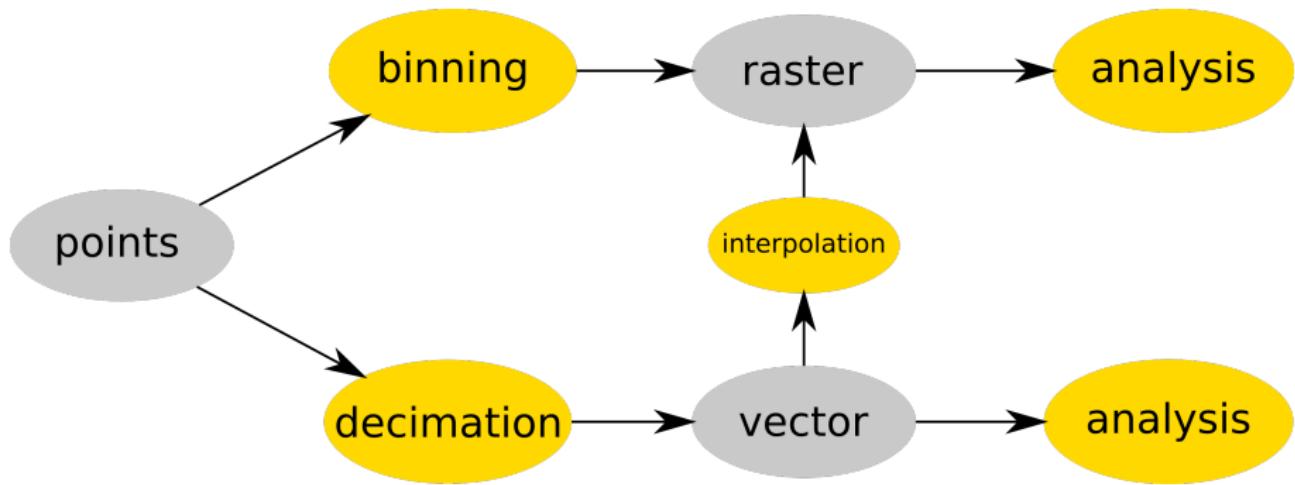
Points

- ▶ collected by lidar
- ▶ generated by Structure from Motion (SfM) from UAV imagery



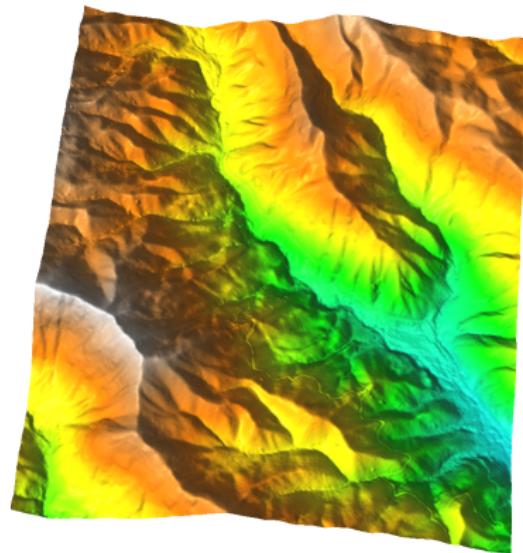
surface interpolated from points and visualized in GRASS GIS

Workflow overview



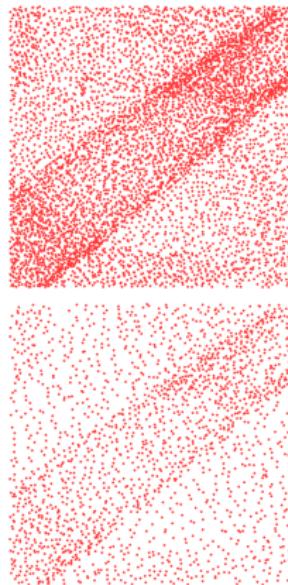
Surface interpolation

- ▶ *v.surf.idw*
 - ▶ Inverse Distance squared Weighting
- ▶ *v.surf.bspline*
 - ▶ Bicubic or bilinear Spline interpolation with Tykhonov regularization
- ▶ *v.surf.rst*
 - ▶ Regularized Spline with Tension
 - ▶ *v.surf.rst.mp* (experimental)
 - ▶ 2 millions of points in 11 minutes



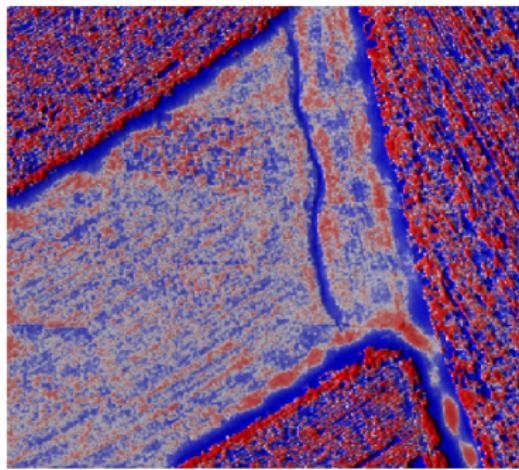
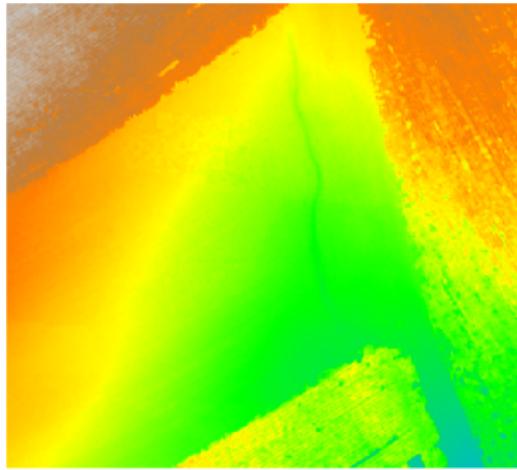
Import and decimation

- ▶ *v.in.lidar*
 - ▶ libLAS
 - ▶ LAS/LAZ to GRASS GIS native vector
 - ▶ data stored in GRASS GIS database
- ▶ interpolation, clustering, ... are costly
- ▶ often more points than we need
- ▶ decimation \approx thinning \approx sampling
 - ▶ count-based decimation (skips points)
 - ▶ grid-based experimental, others needed?



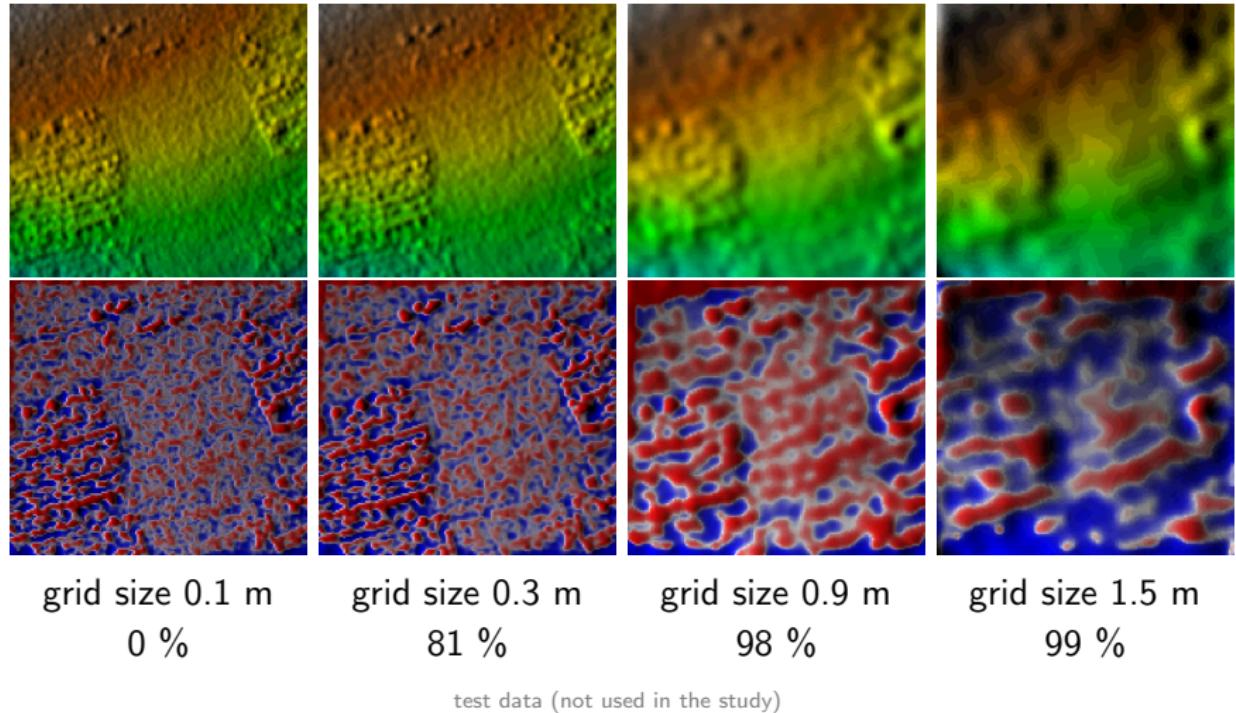
Evaluating level of detail

- ▶ Local relief model (LRM)
- ▶ *r.local.relief* (micro-topography, features other than trend)

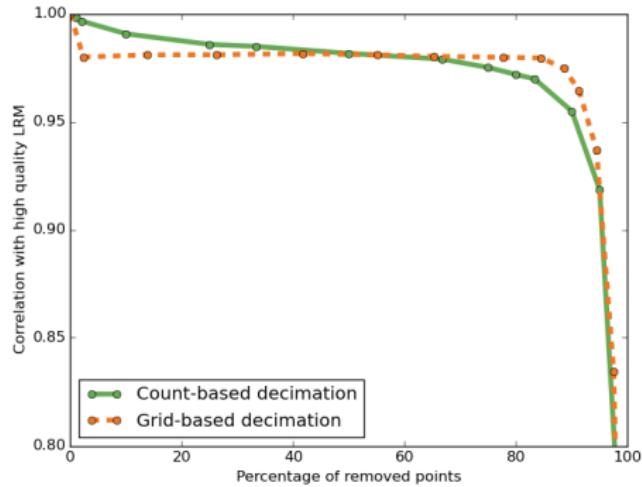


30-60cm wide, 30cm deep, 60m long gully (resolution 30cm)

Influence of grid-based decimation resolution



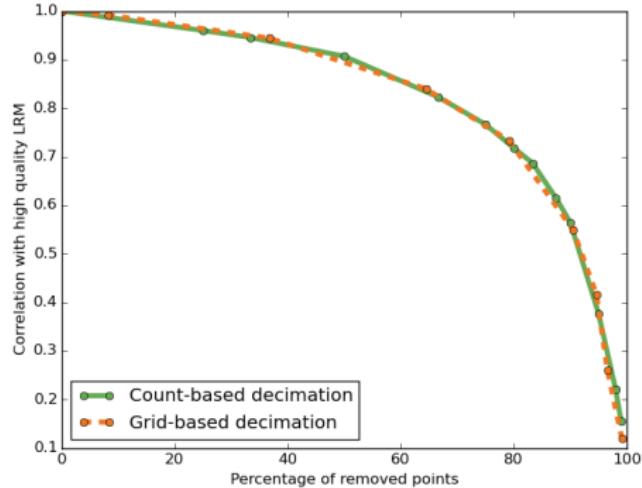
Decimating UAV/SfM point cloud



grid-based decimation may give slightly better results

at resolution 0.5 m for all raster calculations, 72 point per 1 m²

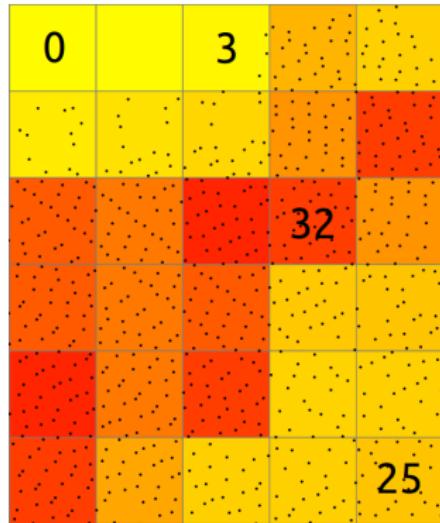
Decimating lidar point cloud



fast count-based decimation as good as more advanced grid-based decimation at resolution 0.5 m for all raster calculations, 1 point per 1 m^2

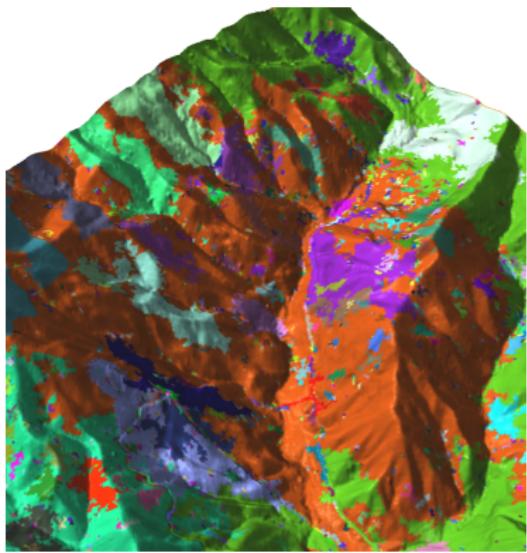
Binning points to raster

- ▶ *r.in.lidar*
- ▶ import and analysis
- ▶ statistics of point counts,
height and intensity
 - ▶ n, min, max, sum
 - ▶ mean, range,
skewness, ...



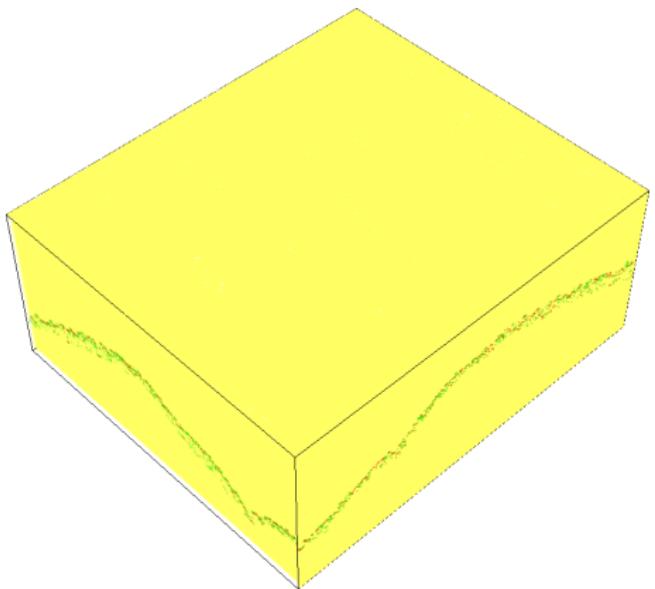
Raster processing

- ▶ many algorithms are raster-based
 - ▶ 163 raster modules
 - ▶ 45 imagery modules
 - ▶ 20 spatio-temporal raster modules
- ▶ example:
 1. count of ground points
 2. count of non-ground points
 3. used as image bands
 4. segmentation using *i.segment*



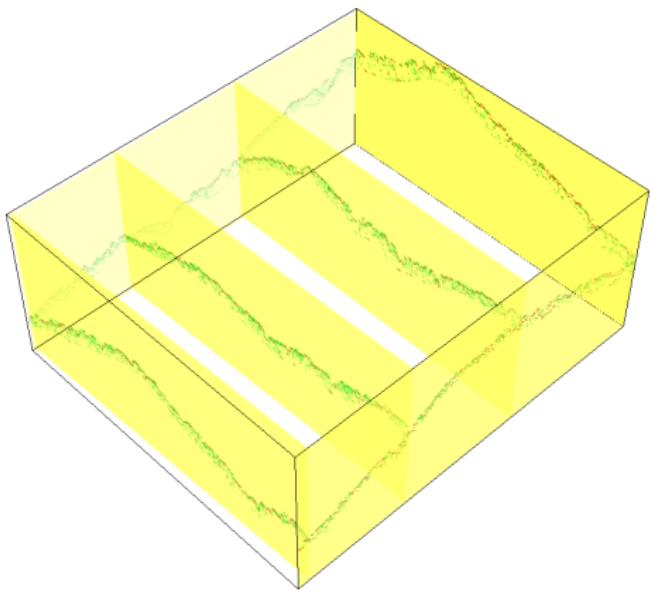
3D raster

- ▶ same principles as in 2D
 - ▶ e.g. 3D raster map algebra
- ▶ challenging to visualize



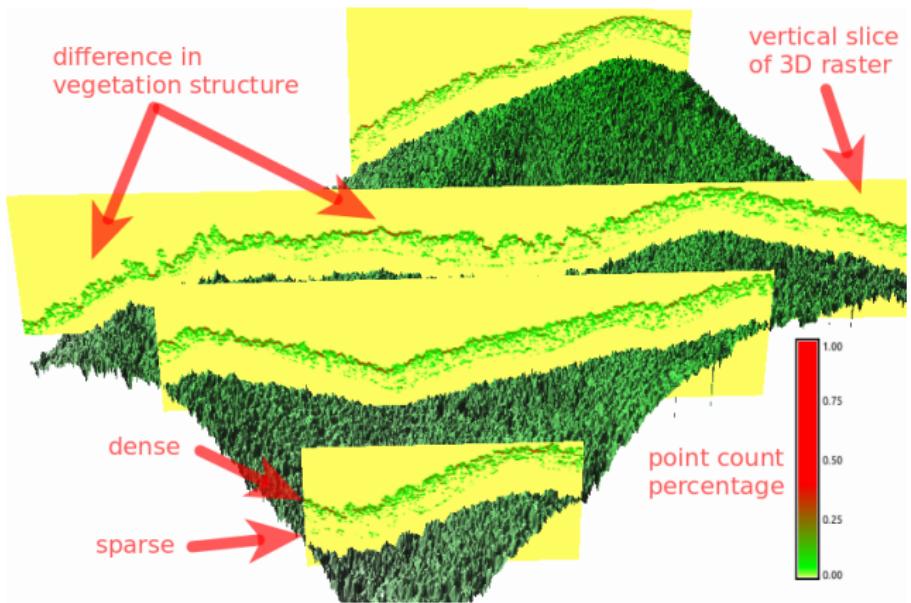
3D raster

- ▶ same principles as in 2D
 - ▶ e.g. 3D raster map algebra
- ▶ challenging to visualize

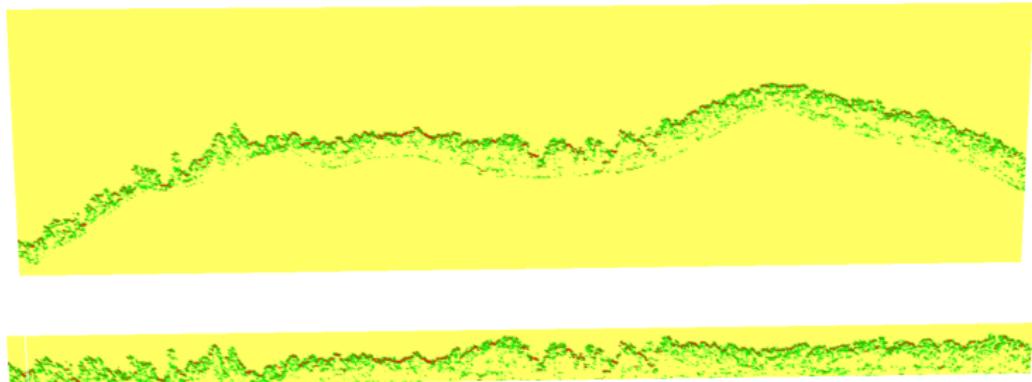


Binning points to 3D raster

- ▶ *r3.in.lidar*
- ▶ count per 3D cell relative to the count per vertical column



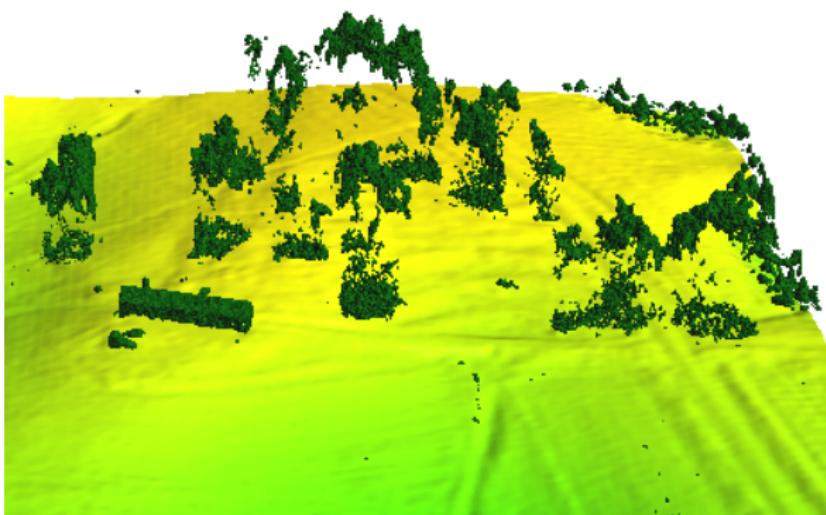
Point heights reduced to surface



- ▶ *r3.in.lidar*, option *base_raster*
- ▶ height reduced by 2D raster values

Ground detection

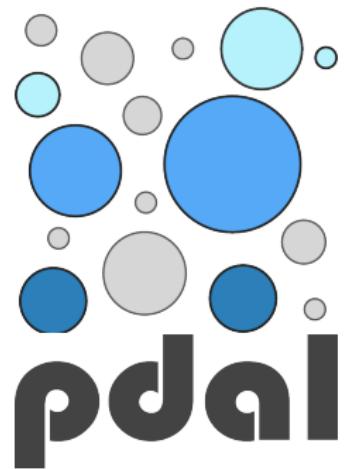
- ▶ *v.lidar.edgedetection*,
v.lidar.growing,
v.lidar.correction
 - ▶ by Brovelli, Cannata, Antolin & Moreno
- ▶ *v.lidar.mcc*
 - ▶ multiscale curvature based classification algorithm
 - ▶ by Blumentrath, according to Evans & Hudak
- ▶ PDAL filters.ground
 - ▶ currently in *v.in.pdal*
 - ▶ progressive morphological filter by Zhang
 - ▶ provided by PCL



Integration with PDAL

PDAL

- ▶ Point Data Abstraction Library
- ▶ format conversions
- ▶ processing, filtering

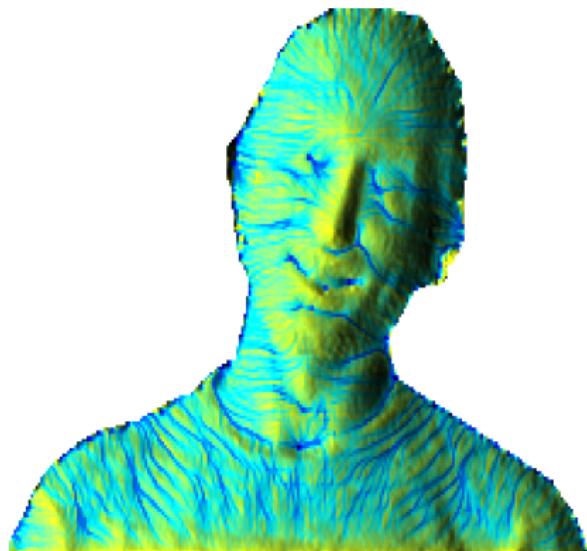


Using other open source projects

r.in.kinect

- ▶ scans using Kinect
- ▶ OpenKinect libfreenect2
- ▶ Point Cloud Library (PCL)
- ▶ GRASS GIS libraries

used in
Tangible Landscape



Summary

- ▶ decimation or *rasterize early* approach for large point clouds
- ▶ 3D rasters
- ▶ PDAL integration



Get GRASS GIS 7.3 development version at
grass.osgeo.org/download

GRASS user mailing list
lists.osgeo.org/listinfo/grass-user

Paper and slides available at
wenzeslaus.github.io/grass-lidar-talks



Acknowledgements

Software

Presented functionality is work done by Vaclav Petras, Markus Metz, and the GRASS development team.

Thanks to users for feedback and testing, especially to Doug Newcomb, Markus Neteler, Laura Belica, and William Hargrove.



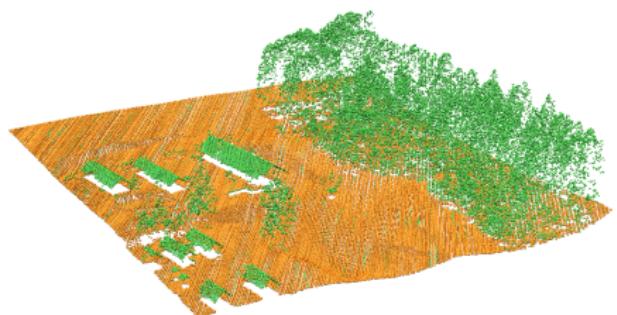
Acknowledgements

Datasets

Lidar and UAV Structure from Motion (SfM) data for GIS595/MEA792: UAV/lidar Data Analytics course

Nantahala NF, NC: Forest Leaf Structure, Terrain and Hydrophysiology. Obtained from OpenTopography.

<http://dx.doi.org/10.5069/G9HT2M76>



Acknowledgements

Presentation software

Slides were created in L^AT_EX using the BEAMER *class*.