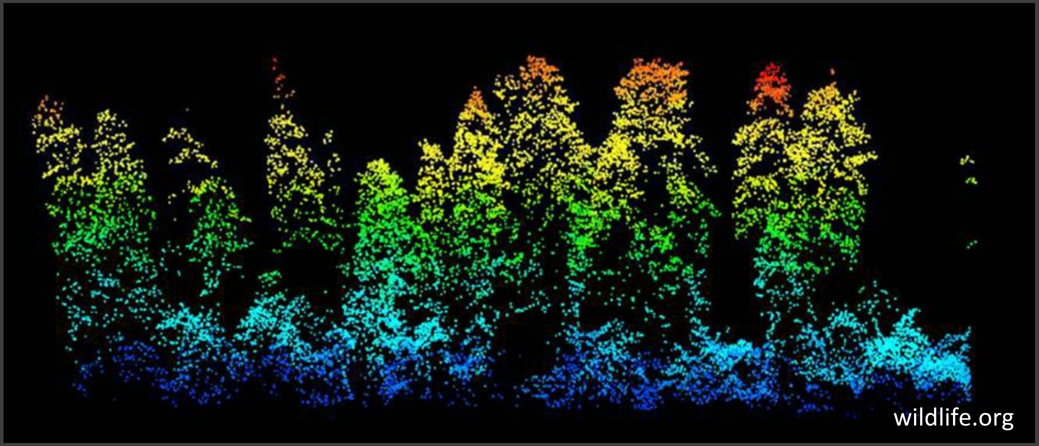
forest3D

David Diaz, Martin Bagaram, Zhehao Li



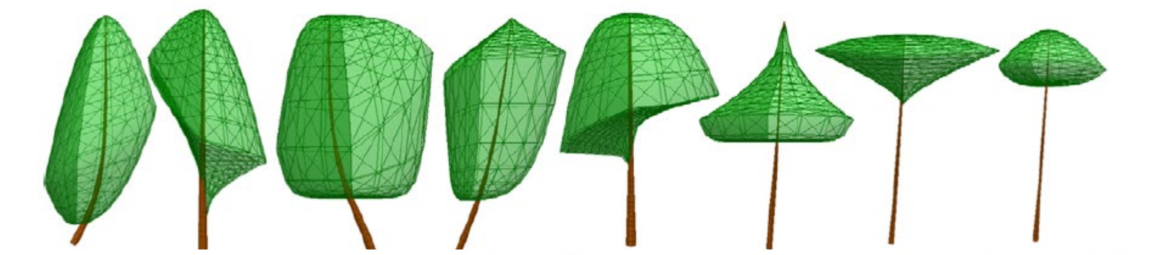


Fig. 5. Examples of asymmetric hulls showing plasticity of this model to represent tree crowns (inspired by Cescatti [10]).

C. Pradal et al. / Graphical Models 71 (2009) 1–21

Tree Data Pre-Processing

functional requirements:

- Read in raw tree list data in text (*.txt, *.csv) and geospatial (*.shp) file formats
- Retain desired attributes from different data sources, reformat to consistent specs
- Identify coordinate reference system, if present; project locations to known coordinate system

Tree Data Pre-Processing

`Pandas, GeoPandas`

- How it works
  - Pandas + geometry data type
- Advantages
  - Flexible,
  - Light weight,
  - Easy to incorporate in a project
- Disadvantages
  - quite slow for large datasets

`ArcPy, ArcGIS, QGIS`

- How it works
  - Full GIS packages / Arcpy and qGIS python library for inline GIS operations
- Advantages
  - Prebuild-ins, GUI, Intuitive
- Disadvantages
  - Expensive (ArcGIS), proprietary,
  - Not so flexible, requires special environment

Point Cloud Pre-Processing

functional requirements:

- Read raw lidar point clouds (*.las, *.laz)
- Generate non-closed surface mesh/manifold from 3D point cloud
- Translate points in mesh to numpy arrays

Point Cloud Pre-Processing

PDAL

- Open-source C++ library with Python extension for point cloud I/O and processing.
- Pipelines with multiple steps can be specified in JSON.
- Some API connections to PCL routines.

python-PCL

- Open-source C++ library (PCL)
- Partial coverage with Python bindings by Univ. Lab
- Point Cloud I/O, processing, state-of-the-art algorithms.

Kazhdan algorithms

- Open-source C++ algorithms for surface reconstruction, Windows binary executables for processing ascii/PLY point clouds with normals

Point Cloud Pre-Processing

PDAL

- 61 (7 core) contribs +
- v1.0 2015, now v1.8 ++
- Python API by core team ++
- Thorough ReadtheDocs.
  Numerous examples. +++
- Focus on pipelines and data translation, Pipelines +

Kazhdan algorithms

- 3 (1 core) contribs
- Peer-reviewed algorithms
- Command line examples, flat files only

python-PCL

- Python: 21 (2 core) contribs
  PCL: 331 (20 core) contribs +++
- PCL v1.0 2011, now v1.9 ++
  Python since 2013.
- Python API by small community team -
- Limited Python docs. -
  Few examples. -
  C++ functions documented. +/-
- Focus on algorithm dev.

3D Visualization

functional requirements:

- Produce interactive 3D plots—scatter and mesh (triangulated surface)—for point clouds and simulated trees
- Handle many thousands to millions of points (webGL under the hood)
- Enable easy user control and updates/callbacks from (ipy)widgets in Jupyter Notebooks

3D Visualization

`Plotly & ipyvolume`

- Open source, free, conda- installable
- Use WebGL for rendering lots of data
- Native support for ipywidgets
- Easily manipulated in Jupyter Notebook
- Good documentation, many examples
- Declarative syntax to build interactive visualizations

3D Visualization

Plotly

- GUI importing and analyzing data into a grid
- More pre-loaded data annotations (e.g., pop-ups on hover)
- More polished toolkit for interation (e.g., changing 3D navigation)
- Company supported, 8 core contributors
- https://plot.ly/python/3d-mesh/

ipyvolume

- Related to Vaex package aimed at exploring large tabular datasets
- Community supported, single core contributor
- https://ipyvolume.readthedocs.io/en/latest/