3D GIS

Laser Scanning

Laser scanning has revolutionized how we collect three-dimensional spatial data, enabling detailed characterization of terrain, vegetation, buildings, and other features at unprecedented scales and resolutions. This module covers the fundamentals of laser scanning technology, how data are collected and processed, the various applications across different disciplines, and practical tools for working with laser scanning data. We'll explore both airborne platforms collecting regional data and terrestrial systems for high-detail local analysis. Understanding laser scanning is essential for modern geospatial work.

Module Topics

1. Point cloud file types and data structures
2. ALS core concepts
3. Noise removal
4. Ground point classification
5. RANSAC
6. Point cloud classification
7. Ground normalization
8. Point cloud filtering
9. RGB and RGBN encoding
10. Rasterization
11. Voxelization
12. TLS core concepts
13. Registration/Coregistration
14. Transformation matrix
15. GeoSLAM
16. CloudCompare
17. lidR

2

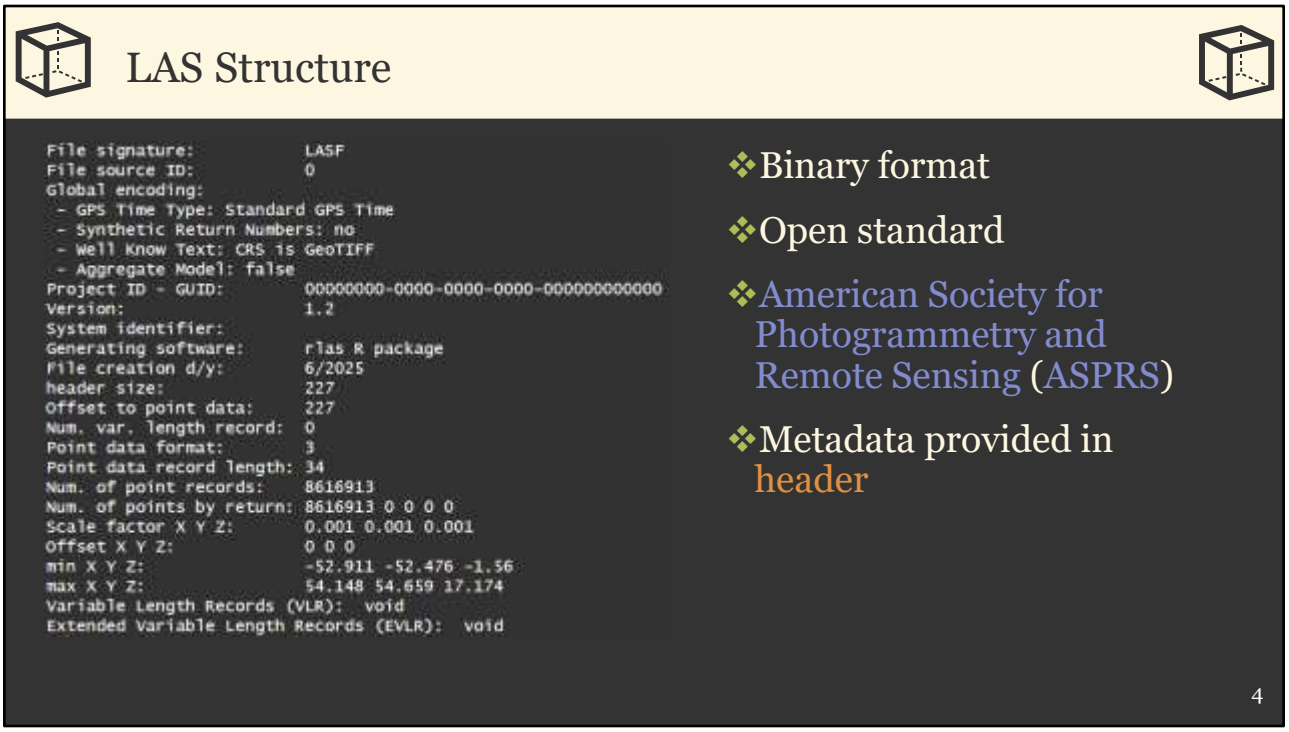
These are the topics covered in this module.

Point Cloud File Formats

- LAS (.las, .laz, .zlas)
- E57 (.es7)
- Text/ASCII (.xyz, .asc)
- PTX (.ptx)

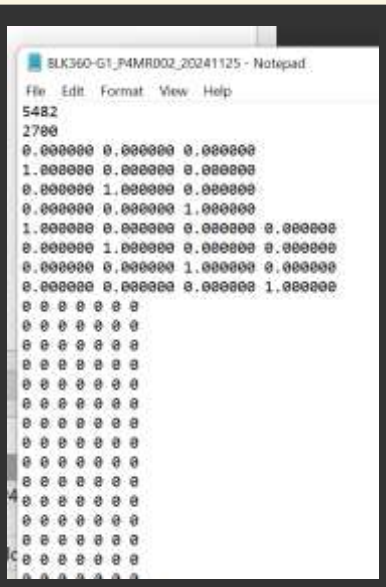
3

Multiple file formats store point cloud data, each with different characteristics. LAS format includes variants .las, .laz (compressed), and .zlas (zstandard compressed). E57 format (.e57) stores point clouds with extended capabilities. Text or ASCII formats store data in .xyz or .asc files. PTX format (.ptx) is used particularly for terrestrial laser scanning data. Each format has advantages depending on storage requirements, compression needs, and software compatibility.



LAS is the most widely used point cloud format. It's a binary format, an open standard developed by the American Society for Photogrammetry and Remote Sensing (ASPRS). LAS files include metadata in the header describing the data characteristics, coordinate system information, and point cloud extents.

PTX Structure

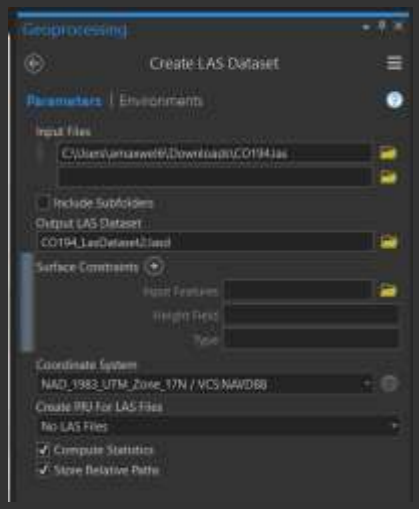


- Text/ASCII
- Row and column count
- Transformation matrix
- Point data (*x*, *y*, *z*, intensity, R, G, B)
- Stores pulses with no associated returns at coordinates [0, 0, 0]

5

PTX format is a text-based format that includes row and column counts for structured point cloud data. A transformation matrix defines the spatial relationship. Point data includes x, y, z coordinates and can include intensity and RGB values. PTX format stores pulses with no associated returns at coordinates [0, 0, 0].

Rasterize LiDAR in ArcGIS Pro



- LAS data must be converted to a LAS Dataset

6

In ArcGIS Pro, point clouds in LAS format must first be converted to LAS Datasets. A LAS Dataset is similar to a mosaic dataset: it is not a new dataset, but a reference to a set of multiple files, in this case multiple LAS files. LAS Datasets can then serve as input to subsequent processing and analysis.

Aerial Laser Scanning (ALS)

7

Aerial Laser Scanning (ALS), also called airborne LiDAR, collects point cloud data from aircraft or helicopters. ALS enables rapid coverage of large geographic areas at moderate to high spatial resolution. ALS is the dominant technology for national-scale elevation data collection.

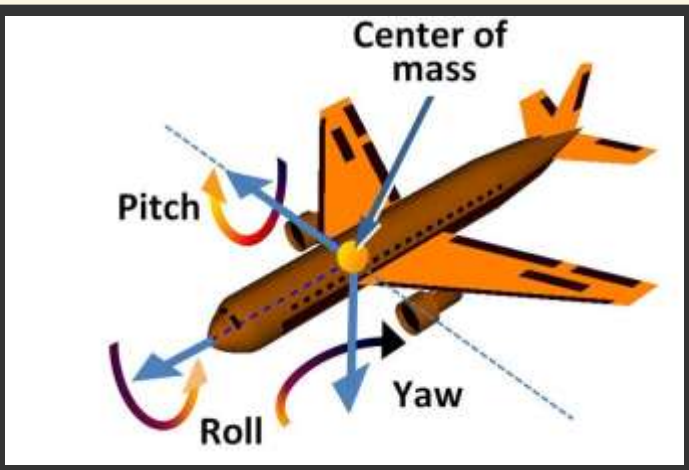
Components

- Laser range finder/scanner = two-way travel time between sensor and surface
- Inertial Measurement Unit (IMU) = angular orientation of the laser with respect to the ground
- GPS = locate platform and reference data relative to a datum

8

ALS systems require three key components. A laser range finder or scanner measures the two-way travel time between the sensor and the surface. An Inertial Measurement Unit (IMU) determines the angular orientation of the laser with respect to the ground. GPS locates the platform and references data relative to a geodetic datum.

IMU



https://commons.wikimedia.org/wiki/File:Roll_Pitch_Yaw.JPG

9

The IMU measures roll, pitch, and yaw: the three rotational components that define the laser's orientation in three-dimensional space. These measurements are critical for converting laser measurements into correctly oriented 3D coordinates.

Travel Time to Distance

$$\text{Velocity} = \frac{Distance}{Time}$$

$$\text{Time} = \frac{Distance}{Velocity}$$

$$\text{Distance} = \text{Velocity X Time}$$

10

The fundamental relationship between velocity, time, and distance allows converting laser travel time to distance. Distance equals velocity multiplied by time. Since light travels at a known constant speed, the two-way travel time can be converted to distance.

Collecting Data (Airborne LiDAR)

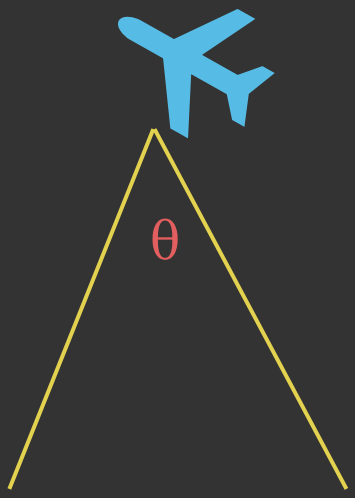
- Ground GPS stations are established, or existing ground stations are used
- GPS, IMU, and laser range finding data are collected onboard the aircraft
- All areas must be covered by flight lines
- Density of points is impacted by
  - Pulse rate
  - Scan frequency
  - Flight height
  - Overlapping coverage

11

Ground data collection begins by establishing GPS stations or using existing ground control points. GPS, IMU, and laser range finding data are collected onboard the aircraft during flight. All areas of interest must be covered by overlapping flight lines. Point density is impacted by pulse rate, scan frequency, flight height, and overlapping coverage between flight lines.

Flight Collection Characteristics

- Pulse Rate = how many pulses released a second (kHz)
- Scan Frequency = how fast sensor scans back-and-forth (Hz)
- Maximum Scan Angle = maximum angle of scan (°)
- Flight Height = height of sensor above the ground (meters or feet)
- Flight Speed = average flight speed (knots)
- Average Post Spacing = average density of returns (points per unit area)



12

Flight collection characteristics determine point cloud density and coverage. Pulse rate is how many laser pulses are released per second. Scan frequency is how fast the sensor scans back-and-forth. Maximum scan angle defines the maximum scanning angle. Flight height is the sensor elevation above ground. Flight speed is the aircraft's average speed. Average post spacing describes the resulting point density.

Post-Processing LiDAR

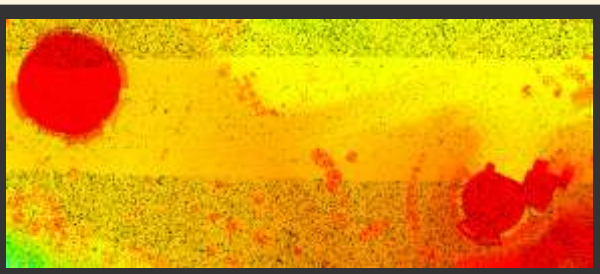
1. GPS data differentially corrected
2. GPS, IMU, and laser range finder data combined to *x*, *y*, *z* points relative to datum
3. Raw data stored in LAS format
4. Points are classified using an algorithm
5. Manual clean up of point classification
6. Quality control and accuracy assessment
7. Creation of derived products

13

Post-processing LiDAR begins with differentially correcting GPS data for improved accuracy. GPS, IMU, and laser range finder data are combined to create x, y, z points relative to the geodetic datum. Data are commonly stored in LAS format.

Points are classified using algorithms to identify ground, vegetation, buildings, and other features. Manual cleanup can be used to refine automated classifications. Quality control and accuracy assessment verify data quality. Derived products, such as DSMs and DTMs, can later be created from the point cloud.

Point Attributes



- Return Number = 1st, 2nd, 3rd, 4th
- Classification = Unassigned, Ground, Low Vegetation, Medium Vegetation, High Vegetation, etc.
- Return Intensity = measure of amount of energy returning to sensor
- RGB values = assign image RGB values to points

14

Point attributes stored with each laser return commonly include return number identifying 1st, 2nd, 3rd, or 4th returns and classification (e.g., unassigned, ground, vegetation, buildings, and others). It is common for only ground and not ground/unassigned to be differentiated. Return intensity measures the amount of energy returning to the sensor. RGB values assign image color to points.

Return Intensity

- Correlates with:
  - Return number
  - Surface material
  - Moisture level

15

Return intensity of the laser pulse is impacted by many factors including the number of returns, the surface material, and the moisture content. The provided image shows a large number of points symbolized using return intensity. Darker shades indicate a lower return intensity while brighter areas indicate more return intensity. These data were collected with an infrared laser.

Grass and low vegetation tend to show a high return intensity since there is no vertical structure to break the pulse into multiple returns and because vegetation tends to reflect NIR radiation. In contrast, paved surfaces tend to absorb NIR radiation and are dark in the image.

Forests tend to show a mixture or texture of high and low return intensities. This is because vegetation tends to reflect NIR radiation. However, due to vertical structure, a single pulse is often broken into multiple returns.

Although not represented in this example, water tends to absorb a large amount of NIR radiation and will show low return intensity or no returns, as it may absorb all the energy.

These data can be useful for data visualization and as input to classification routines. However, it should be noted that return intensity has been shown to

be hard to calibrate. So, it is difficult to use these measurement in a quantitative manner. For example, it would be difficult to convert return intensity to NIR surface reflectance.

LAS Classification Codes

Classification Value	Meaning
0	Never classified
1	Unassigned
2	Ground
3	Low Vegetation
4	Medium Vegetation
5	High Vegetation
6	Building
7	Low Point
8	Reserved
9	Water
10	Rail

Classification Value	Meaning
11	Road Surface
12	Reserved
13	Wire - Guard (Shield)
14	Wire - Conductor (Phase)
15	Transmission Tower
16	Wire-Structure Connector (Insulator)
17	Bridge Deck
18	High Noise
19-63	Reserved
64-255	User Definable

http://desktop.arcgis.com/en/arcmap/10.3/manage-data/las-dataset/lidar-point-classification.htm

16

LAS files have defined, standard classification codes, as described in the provided tables. Generally, ground points are assigned a value of 2. If only ground points are differentiated, all other points will be coded to 1, or unassigned, or labelled as an outlier, generally using code 7 or 18. Codes have been defined for vegetation based on height, buildings, water, rails, roads, wires, transmission towers, and bridge decks. Some codes are reserved for special use, and analysts can use remaining codes to define user-specific classes. This coding scheme uses 8-bit encoding, so 256 values are available.

LAS Class Flags

- Provide a secondary description or classification for LiDAR points

http://desktop.arcgis.com/en/arcmap/10.3/manage-data/las-dataset/lidar-point-classification.htm

Field Name	Description
Synthetic	A point that was created by a method other than lidar collection, such as digitized from a photogrammetric stereo model.
Key-point	A point considered to be a model key-point and should not be withheld in any thinning algorithm.
Withheld	The point should not be included in processing.
Overlap	A point that is within the overlap region of two or more flight lines or swaths (LAS version 1.4 specific).

17

Class flags can also be assigned to specific points. For example, synthetic points are those not created by the LiDAR collection, such as those manually added or obtained using photogrammetric methods. The key-point flag is used to note that a point is important and should not be removed by any thinning process. For example, points near a slope break may be flagged as key-points so that the steep slope is well represented even if the data are thinned. The withheld flag is used to make sure that points are not included in subsequent processing or analysis. For example, outlier points could be flagged this way. Lastly, the overlap flag can be used to indicate points that occur in areas of overlapping flight lines.

Use Cases: Archeology

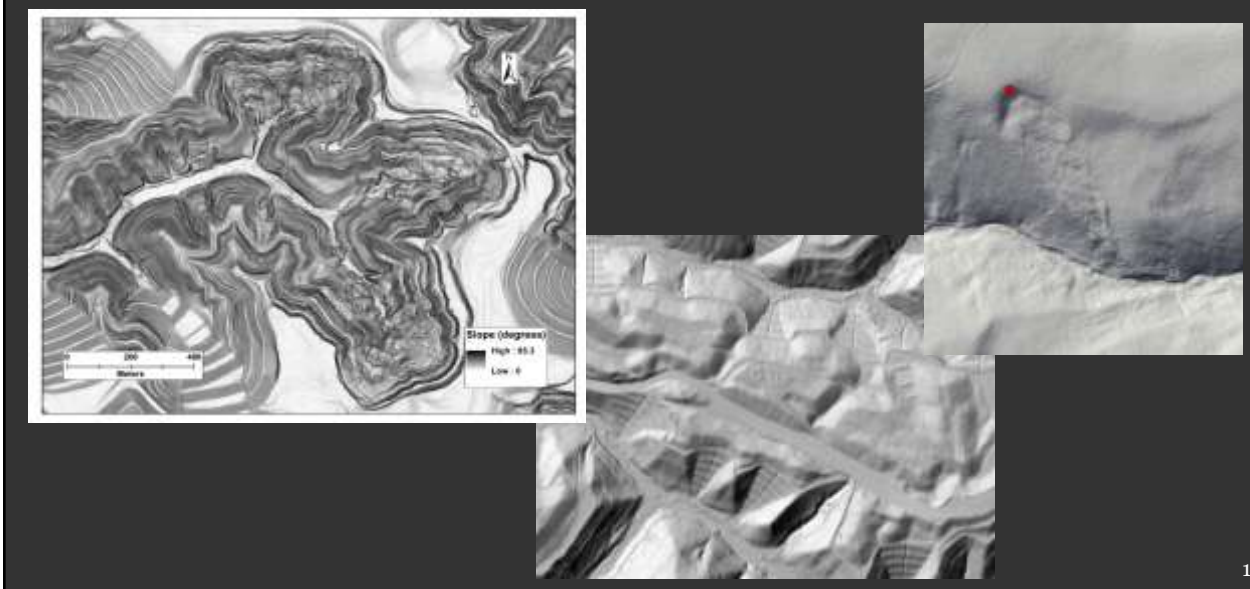
https://youtu.be/YFC5CwZVCEw?si=WXdF3TFwSPDAFw6

Video from TED

18

ALS has remarkable applications in archaeology, revealing hidden structures and landscape features beneath vegetation that would be invisible on the ground. Airborne LiDAR can penetrate forest canopy to reveal buried archaeological features.

Use Cases: Geomorphology



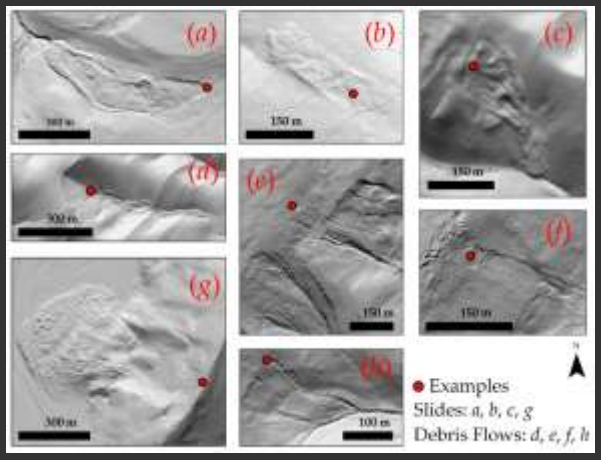
19

Geomorphology benefits greatly from high spatial resolution elevation data. ALS can characterize landforms, identify erosion features, map mass movements, and analyze terrain processes at scales impossible with traditional methods.

Use Cases: Geohazards

https://www.mapwv.gov/flood/map/

Maxwell, A.E.; Sharma, M.; Kite, J.S.; Donaldson, K.A.; Thompson, J.A.; Bell, M.L.; Maynard, S.M. Slope Failure Prediction Using Random Forest Machine Learning and LiDAR in an Eroded Folded Mountain Belt. *Remote Sens.* **2020**, *12*, 486.



20

Geohazard assessment using ALS includes landslide susceptibility mapping, flood modeling, and volcanic hazard assessment. High spatial resolution terrain data enables identifying steep slopes, drainage patterns, and other geomorphic indicators of hazard.

Use Cases: Forestry

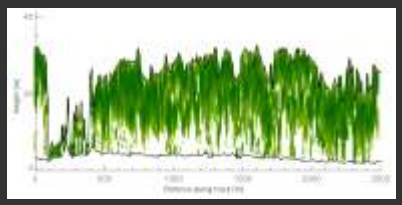
- Forest structure
- Tree crown delineation
- Tree speciation
- Biomass
- Fuel loads

21

Forestry applications include measuring forest structure, delineating individual tree crowns, estimating species composition, calculating biomass, and assessing fuel loads. ALS provides three-dimensional characterization of forest structure.

Use Cases: Global Forest and Carbon Monitoring

- Global Ecosystem Dynamics Investigation
- International Space Station (ISS)
- Carbon sequestration
- Forest height and vertical structure
- Forest canopy characteristics
- Deforestation



ATBD #	Data products	Product leads	Resolution
L1A-2A	1A: Raw waveforms; 2A: Ground elevation, canopy top height, relative height (RH) metrics	Michelle Hofton Bryan Blair	25 m (~82 ft) diameter
L1B	Geolocated waveforms	Scott Luthcke Tim Pennell Taylor Thomas Tarsha Pennington	25 m (~82 ft) diameter
L2B	Canopy Cover Fraction (CCF), CCF profile, Leaf Area Index (LAI), LAI profile	Hao Tang John Armston	25 m (~82 ft) diameter
L3	Gridded Level 2 metrics	Scott Luthcke Terence Sabaka Sandra Preaux	1 km (~0.6 mi) grid
L4A	Footprint level above ground biomass	Jon Kellner Laura Duncanson John Armston	25 m (~82 ft) diameter
L4B	Gridded Above Ground Biomass Density (AGBD)	Sean Healey Paul Patterson	1 km (~0.6 mi) grid
Demonstrative products	Prognostic ecosystem model outputs	George Hurtt	Grid size: Variable
Demonstrative products	Enhanced height/biomass using fusion with TanDEM-X	Lola Fatoyinbo Seung-Kuk Lee	Grid size: Variable
Demonstrative products	Enhanced height/biomass and biomass change using fusion with Landsat	Matt Hansen Chengquan Huang	Grid size: Variable
Demonstrative products	Biodiversity/habitat model outputs	Scott Goetz Patrick Jantz Pat Burns	Grid size: Variable

https://gedi.umd.edu/

22

The Global Ecosystem Dynamics Investigation (GEDI) on the International Space Station uses spaceborne lidar to measure forest height and vertical structure, understand canopy characteristics, track deforestation, and monitor carbon sequestration across the globe.

Data Processing

23

Raw point cloud data requires processing to be useful. Processing steps include removing noise, classifying ground points, normalizing heights, creating raster products, and extracting features.

Noise Removal

- Statistical Outlier Removal
  - Calculate mean distance to *k*-nearest neighbors
  - Assume Gaussian distribution
  - Remove points beyond a defined threshold (based on standard deviations from mean)
- Radius Outlier Removal
  - For each point, count neighbors within a fixed radius
  - Remove points with fewer than a threshold number of neighbors
- Voxel Grid Filter
  - Divide into voxel grid
  - Replace points with voxel centers

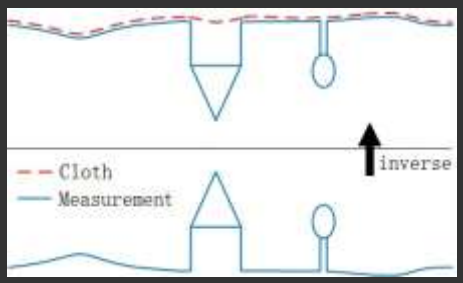
24

Noise removal identifies and removes erroneous points. Statistical outlier removal calculates mean distance to k-nearest neighbors, assumes Gaussian distribution, and removes points beyond defined thresholds. Radius outlier removal counts neighbors within a fixed radius and removes isolated points. Voxel grid filtering divides space into voxels and replaces multiple points with voxel centers.

Ground Classification: Cloth Simulation Filter (CSF)

- Simulate cloth draped over an inverted point cloud
- Cloth surface approximates terrain
- Classify points falling on cloth as ground points
- Parameters
  - Class Threshold: Maximum distance to simulated cloth to classify a point into the ground class
  - Cloth Resolution: Distance between coordinates defining the cloth
  - Rigidness: how rapidly terrain surface can vary
  - Iterations: number of processing iterations
  - Time Step: time step when simulating the cloth under gravity

Zhang, W., Qi, J., Wan, P., Wang, H., Xie, D., Wang, X. and Yan, G., 2016. An easy-to-use airborne LiDAR data filtering method based on cloth simulation. *Remote sensing*, *8*(6), p.501.



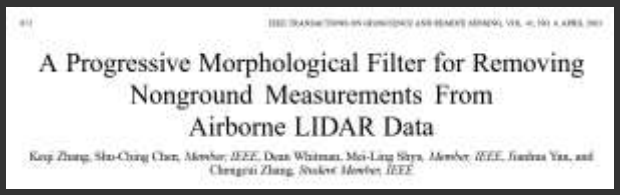
25

The Cloth Simulation Filter (CSF) simulates a cloth draped over an inverted point cloud. The cloth surface approximates terrain while elevated points represent vegetation or buildings. Points on the cloth are classified as ground. Parameters include class threshold for ground classification distance, cloth resolution, rigidness controlling terrain variability, iterations, and time step.

Ground Classification: Progressive Morphological Filter (PMF)

- Iteratively apply filter at increasing window sizes
- Remove small objects first followed by larger objects
- Remove high features → restore general shape of the surface
- Parameters
  - Initial window size: detect small features
  - Max window size: detect larger features
  - Slope threshold: control terrain sensitivity
  - Elevation threshold: separation tolerance

Zhang, K., Chen, S.C., Whitman, D., Shyu, M.L., Yan, J. and Zhang, C., 2003. A progressive morphological filter for removing nonground measurements from airborne LIDAR data. *IEEE transactions on geoscience and remote sensing*, *41*(4), pp.872-882.


A Progressive Morphological Filter for Removing Nonground Measurements From Airborne LIDAR Data

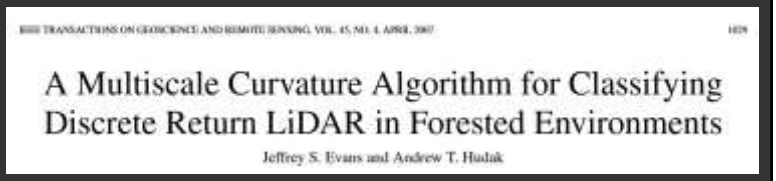
26

The Progressive Morphological Filter (PMF) iteratively applies filtering at increasing window sizes. Small objects are removed first, followed by larger objects. This approach removes high features while restoring the general surface shape. Parameters control initial and maximum window sizes, slope sensitivity, and elevation thresholds.

Ground Classification: Multiscale Curvature Classification

- Based on calculation of surface curvature at multiple scales
- Parameters
  - Scale
  - Curvature threshold

Evans, J.S. and Hudak, A.T., 2007. A multiscale curvature algorithm for classifying discrete return LiDAR in forested environments. *IEEE Transactions on Geoscience and Remote Sensing*, *45*(4), pp.1029-1038.



A Multiscale Curvature Algorithm for Classifying Discrete Return LiDAR in Forested Environments

Jeffrey S. Evans and Andrew T. Hudak

27

Multiscale Curvature Classification calculates surface curvature at multiple scales. Points are classified based on curvature analysis. This method works particularly well for terrain with varying slopes.

RANSAC

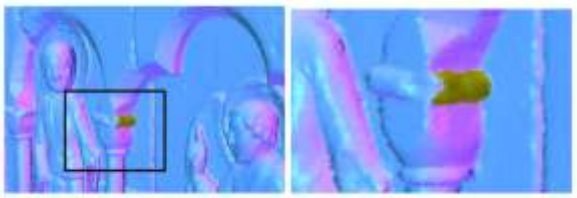


Figure 2: *A small cylinder that has been detected by our method. The shape consists of 1066 points and was detected among 341,587 points. That corresponds to a relative size of* 1/3000.

Schnabel, R., Wahl, R. and Klein, R., 2007, June. Efficient RANSAC for point-cloud shape detection. In *Computer graphics forum* (Vol. 26, No. 2, pp. 214-226). Oxford, UK: Blackwell Publishing Ltd.

- Random Sampling and Consensus
- Use do detect shapes within point clouds: plane, sphere, cylinder, cone, torus
- Based on random sampling and fitting equations representing different geometric features

28

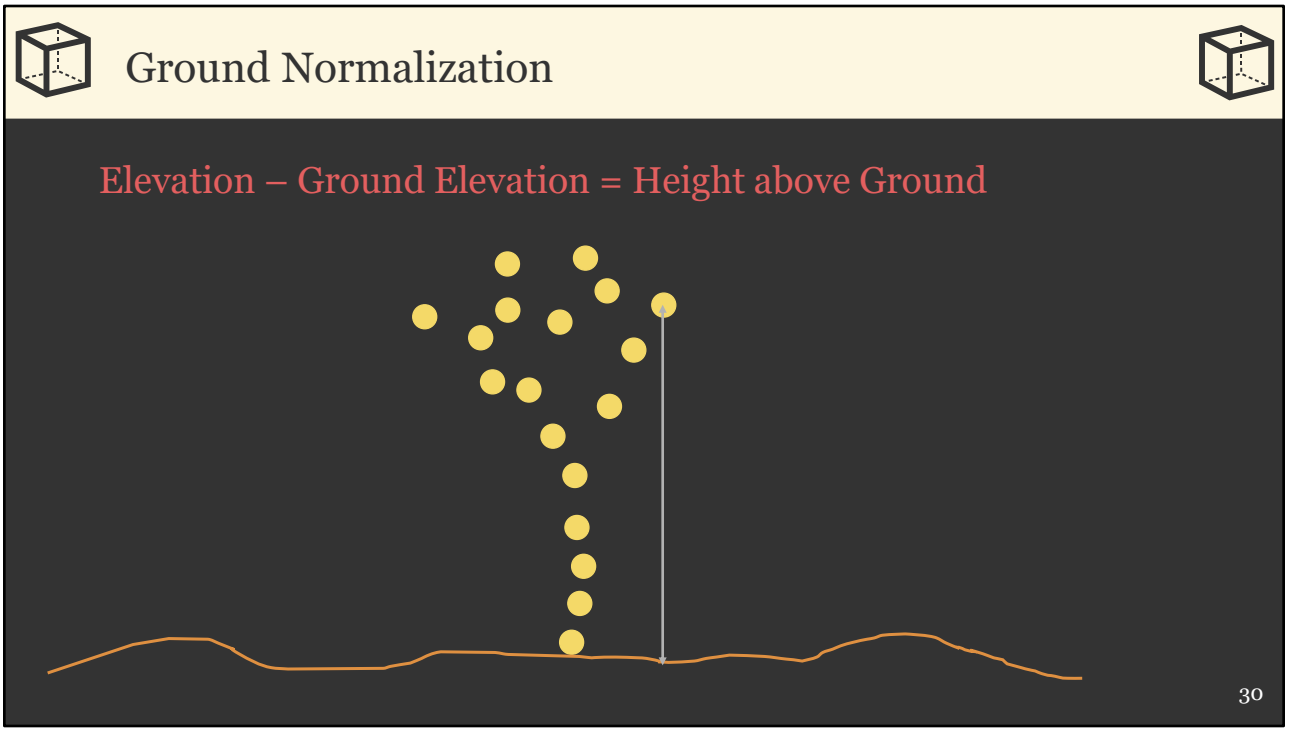
RANSAC (Random Sampling and Consensus) detects geometric shapes within point clouds. It identifies planes, spheres, cylinders, cones, and tori by random sampling and fitting equations. RANSAC is often effective for finding man-made structures.

General Classification

1. Manual
2. Shape-Fitting
3. Deep Learning-Based Semantic Segmentation

29

Point cloud classification methods include manual interpretation, shape-fitting approaches like RANSAC, and deep learning-based semantic segmentation. Each approach has strengths for different data types and classification goals.



Ground normalization estimates height above ground for each point by subtracting ground elevation from absolute elevation. This normalization removes topographic variation to emphasize features like buildings and vegetation.

Filtering

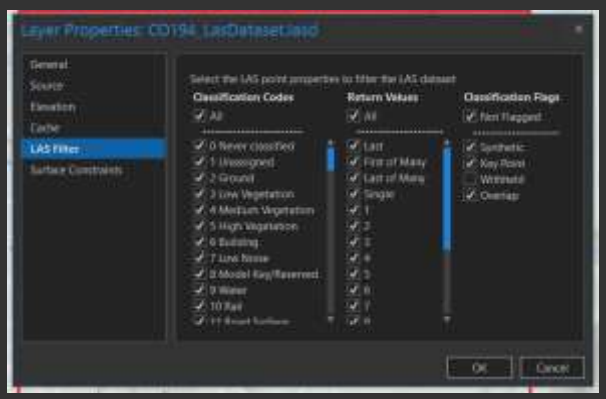
1. Region of Interest (RoI)
2. Elevation ranges
3. Height above ground
4. Classification
5. Return number
6. Flight path
7. Collection
8. Scan angle
9. GPS time
10. Area of overlap
11. Return intensity
12. RGB or RGBN values

31

Filtering operations select subsets of point clouds. Filters can be based on a region of interest, elevation ranges, height above ground, classification, return number, flight path, collection date, scan angle, GPS time, overlap areas, return intensity, or RGB values.

Filter Prior to Rasterization

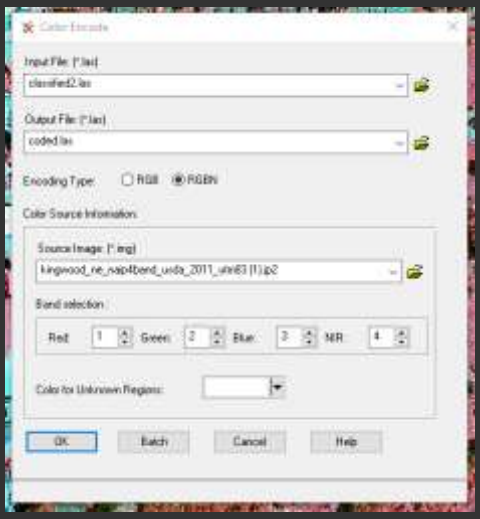
- Data must be filtered to obtain the desired product
- Can filter by class, return number, and/or classification flags



Creating useful products requires filtering data to obtain desired components. Filter by classification to isolate ground or vegetation. Filter by return number to use first or last returns. Apply classification flags to exclude withheld points or points in areas of overlapping flight lines.

RGB/RGBN Encoding

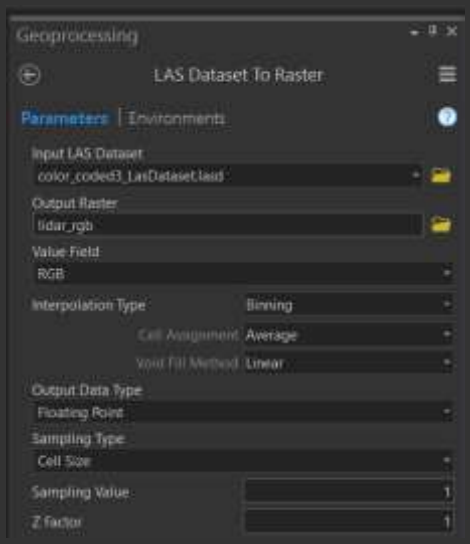
- Can encode points with values from an image
- RGB or RGBN
- Can symbolize using the codes



RGB values can encode image information onto points. RGB or RGBN (with near-infrared) values assign image colors to points, enabling visualization and analysis of spectral characteristics.

Encoded RGB to Raster

- Can convert RGB values stored on points to a multiband raster



34

RGB values stored on points can be converted to a multiband raster image. This creates image products directly from the point cloud data.

Surface Examples

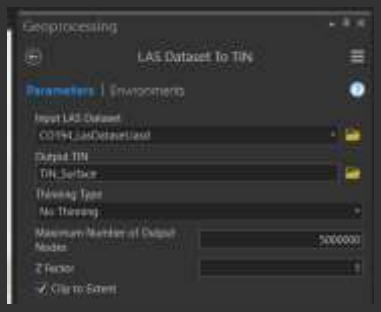
Intensity DTM DSM

35

Common derived products from ALS include intensity raster grids showing return energy, DTMs (digital terrain models) showing ground elevation, and DSMs (digital surface models) showing elevation of surface features.

Rasterize LiDAR in ArcGIS Pro

- Create raster surface using LAS Dataset to Raster Tool
- Can Convert to TIN using LAS Dataset to TIN Tool



36

There are a couple of rasterization options in ArcGIS Pro. The LAS Dataset to Raster tool allows for a raster grid to be generated from a point cloud using some predefined methods. Alternatively, point clouds can be converted to a TIN surface. TIN surfaces can be rasterized after they are created.

Rasterize LiDAR in ArcGIS Pro

- Interpolation Methods
  - Binning = cell is assigned a statistic from the points that fall inside of it: average, minimum, maximum, IDW, nearest
  - Triangulation = calculate a cell based by rasterizing a TIN generated from the point cloud
- Void filling
  - Linear = linear interpolation across voids
  - Natural Neighbor = uses natural neighbor interpolation
  - Simple = averages the values from data cells immediately surrounding a NoData cell
  - None = voids are not filled

37

The LAS Dataset to Raster Tool in ArcGIS Pro provides two interpolation methods: binning and triangulation. Binning calculates a statistic from the filtered points that occur within each cell extent to assign a value to each cell. This measure can be the average, minimum, or maximum. IDW will generate an inverse-distance weighted average where returns near the center of the cell are weighted higher than those toward the margins. Nearest will simply use the return nearest to the center of the cell and ignore all other returns. The second method, triangulation, involves generating a TIN surface from the point cloud and then rasterizing the triangular facets.

If cells do not have any returns that occur within them, then voids will exist in the raster datasets. A couple options are available for filling these voids. First, linear interpolation can be used to fill the voids by linearly interpolating raster cell values on the margins of the gap. In contrast, nearest neighbor only considers the nearest cell value while simple will average all cells that touch or immediately surround the gap.

If you do not want to fill the voids, then they can be assigned a NoData value.

When I create a DTM, I tend to use the binning method, the average of the points within the cell, linear void filling, and filter to just use the ground classified points. To create a DSM, I tend to use the same settings other than

replacing the average cell value with the maximum and using a first returns or single returns filter.

To interpolate a raster grid from intensity data, I tend to using the binning method, the average of the point intensity values within the cell, and no void filling. This is because voids may occur as a result of full absorption of the laser pulse. For example, it is common for all NIR energy to be absorbed by water. So, there are likely to be gaps over rivers and ponds. I then recode these cells to 0, based on the assumption of full absorption. To rasterize the intensity of the first returns only, you will want to apply a filter so that only single returns or first of many returns are included.

Voxel Generation

- Similar to 2D raster
- Statistic from points occurring in voxel
- Simple occurrence (filled or not filled)
- Can interpolate gaps

38

Voxel generation creates three-dimensional raster grids. Each voxel contains a statistic from points within it or a simple filled/not-filled indicator. Gaps can be interpolated between voxels.

Terrestrial Laser Scanning

39

Terrestrial Laser Scanning (TLS) collects high-detail point cloud data from ground-based stationary or mobile platforms. TLS provides millimeter to centimeter accuracy over shorter ranges than ALS.

Types of Sensors

- Time-of-Flight (ToF)
  - Measures two-way travel time of laser
  - Uses speed of light to get two-way distance

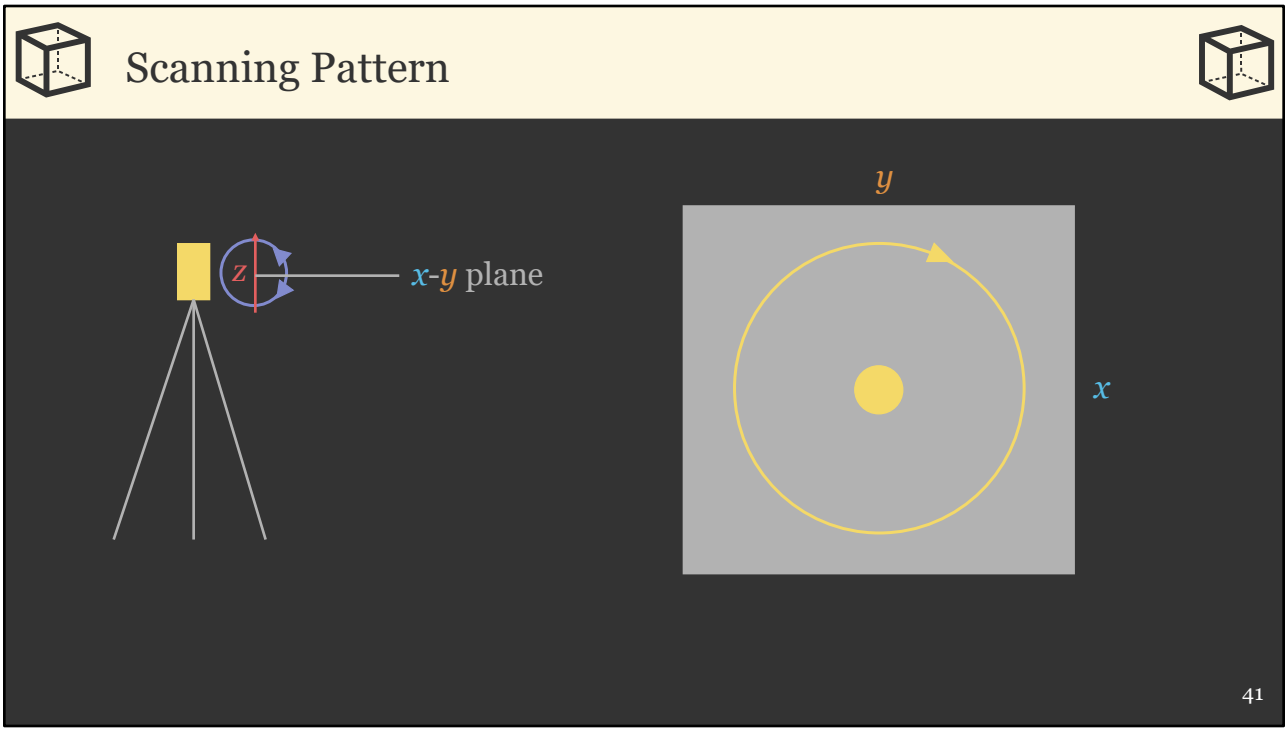
$$d = \frac{c \times t}{2}$$

- Phase Shift
  - Emit continuous, modulated laser wave
  - Compare phase shift of returning signal
  - Distance inferred from phase difference

$$d = \frac{\Delta\phi}{4\pi} \times \lambda$$

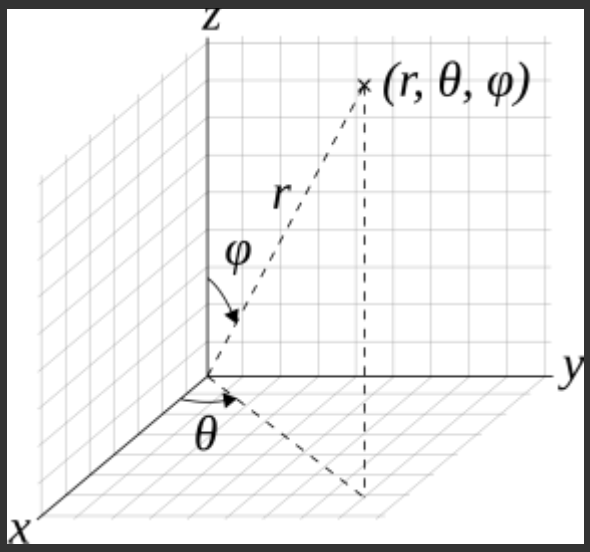
40

TLS sensors use two primary measurement methods. Time-of-Flight measures the two-way travel time of laser pulses and uses the speed of light to calculate distance. Phase Shift emits continuous modulated laser waves and infers distance from phase differences in returning signals.



Scanning patterns differ between sensor types. A scanning pattern may sweep through the x-y plane while maintaining constant z, creating a fan-like field of view.

Coordinate Systems



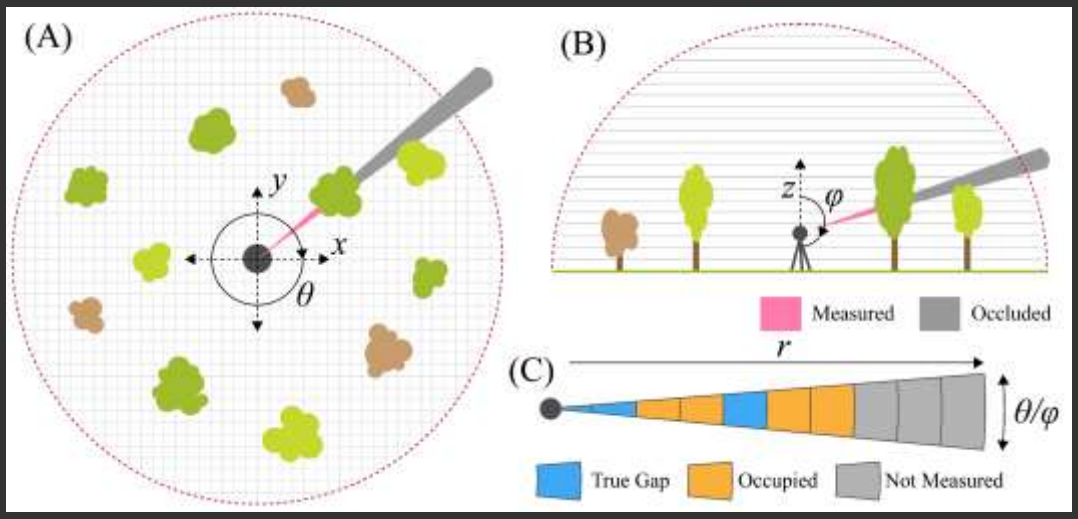
- Spherical
  - θ = angle of rotation around *x*-*y* plane
  - φ = angle relative to *z* axis
  - *r* = distance from the origin
- Cartesian
  - *x* = distance along x-axis
  - *y* = distance along y-axis
  - *z* = distance along z=axis

https://commons.wikimedia.org/wiki/File:3D_Spherical_2.svg

42

Coordinate systems can be expressed in spherical coordinates (theta angle, phi angle, radius distance) or Cartesian coordinates (x, y, z). Converting between coordinate systems is common in processing.

Occlusion



43

Occlusion occurs where line-of-sight is blocked between the scanner and points. Objects or terrain can block laser signals, creating gaps in coverage. Multiple scan positions are typically needed to achieve complete coverage.

RGB Encoding

- Camera data collected alongside laser data
- Based on spatial co-occurrence

44

RGB encoding uses camera data collected alongside laser data. Colors from photographs are assigned to points based on spatial co-occurrence between laser points and image pixels.

Registration

- Registration = align with coordinate reference system (CRS)
- Co-registration = align multiple scans

Castorena, J., Dickman, L.T., Killebrew, A.J., Gattiker, J.R., Linn, R. and Loudermilk, E.L., 2025. ForestAlign: Automatic forest structure-based alignment for multi-view TLS and ALS point clouds. *Science of Remote Sensing*, *11*, p.100194.

45

Registration aligns data with a coordinate reference system (CRS). Co-registration aligns multiple scans from different positions. Proper registration is essential for combining data from multiple scan positions.

Transformation Matrix

$$T = \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_x \\ R_{21} & R_{22} & R_{23} & T_y \\ R_{31} & R_{32} & R_{33} & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Top-left 3x3 = rotation
- Last column = translation
- Can apply a series of transforms
- Supports translation, rotation, and scaling

$$p' = pT^T \text{ (For row vector)}$$

46

A transformation matrix is a mathematical tool that defines how to move, rotate, and scale objects in three-dimensional space. In the context of point clouds, a transformation matrix provides instructions for repositioning every point in a dataset to align with a specific coordinate reference system or to integrate multiple point clouds collected from different positions.

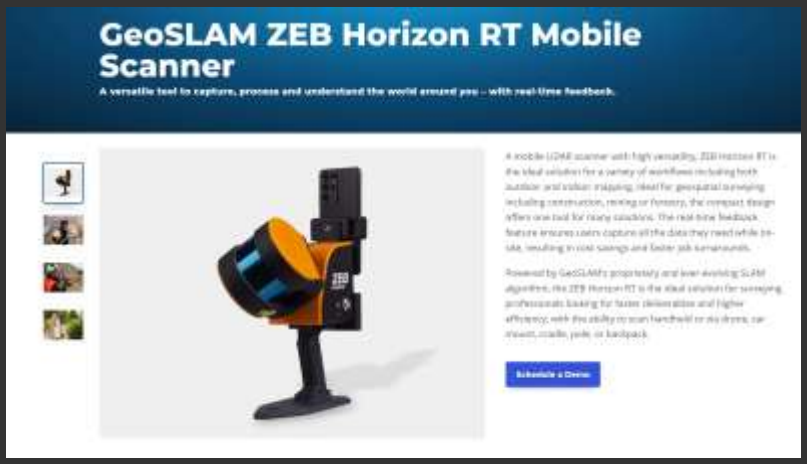
Mathematically, a 4×4 transformation matrix in homogeneous coordinates encodes these operations. The top-left 3×3 elements (R11-R33) define rotation, controlling how the point cloud is oriented in space. The right column (Tx, Ty, Tz) defines translation, specifying how far the entire point cloud moves along the X, Y, and Z axes. The bottom row remains constant and enables homogeneous coordinate mathematics, which simplifies complex geometric transformations through standard linear algebra operations.

Many point cloud formats, including PTX files, store transformation matrices as part of their file structure. These matrices convert the raw coordinates stored in the file into corrected real-world coordinates (for example, transforming scanner-relative coordinates (where the scanner is positioned at [0, 0, 0]) into geographic coordinates within a specific coordinate reference system). When integrating multiple scans, transformation matrices can be applied sequentially: a first transformation might reference all points to a scanner's position, while a second transformation aligns that scan with data

from a different scanner position. This sequential approach enables complex repositioning through a series of simpler, manageable transformations.

Mobile Scanners

https://www.navvis.com/vlx-3



https://www.faro.com/en/Products/Hardware/GeoSLAM-ZEB-Horizon-RT

47

Mobile scanners mounted on vehicles, backpacks, or handheld devices enable TLS data collection in motion. Mobile scanners produce registered point clouds while moving through space.

GeoSLAM Technology

- Geospatial Simultaneous Localization and Mapping
- Does not require GPS (works outdoors and indoors)
- Uses laser scanner
- Attempts to match features from different scan positions
- Use algorithm to estimate position (x, y, z) and orientation (roll, pitch, and yaw)
- Produce registered point cloud in real time
- Close loop to help reduce error
- Generally, less accurate that static TLS (currently)

48

GeoSLAM (Geospatial Simultaneous Localization and Mapping) technology works without GPS, functioning both outdoors and indoors. It uses laser scanners to match features from different positions, estimating position and orientation. The algorithm produces registered point clouds in real time by closing loops to reduce error accumulation.

CloudCompare

49

CloudCompare is a powerful open-source tool for processing and visualizing point clouds. It provides comprehensive capabilities for point cloud analysis, registration, and conversion.

CloudCompare

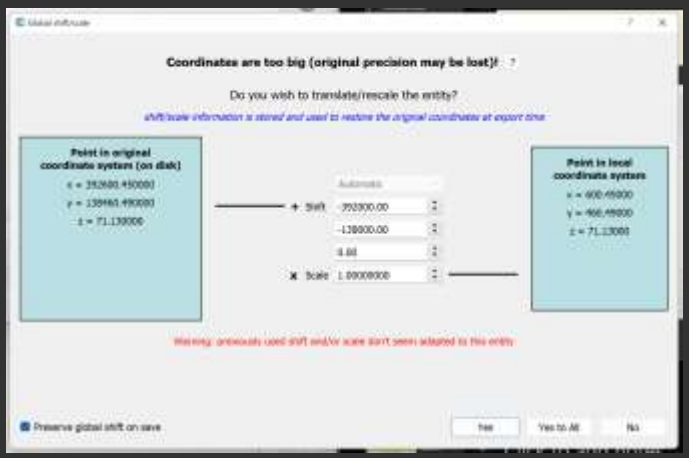
- Free and open-source
- Project began in 2003
- Originally developed by Daniel Girardeau-Montaut as part of PhD dissertation
- Written in C++
- GPL license

50

CloudCompare is free and open-source. The project began in 2003 as a PhD dissertation by Daniel Girardeau-Montaut. It's written in C++ under the GPL license, making it accessible to researchers and professionals.

Loading Data

- Can open a variety of file formats: .las, .e57, .ptx, etc.

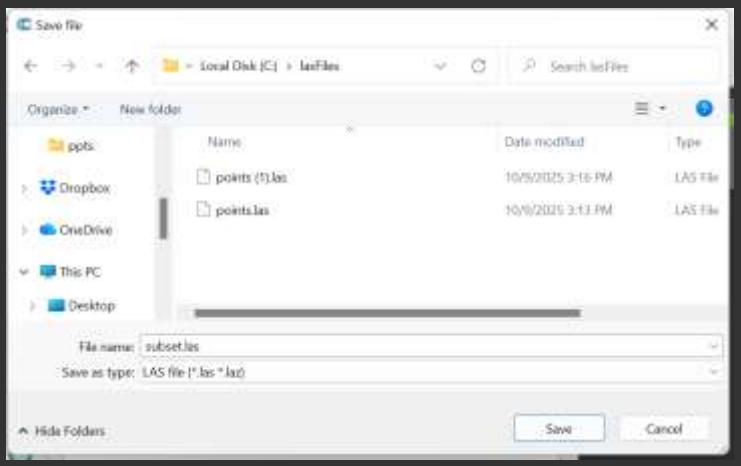


51

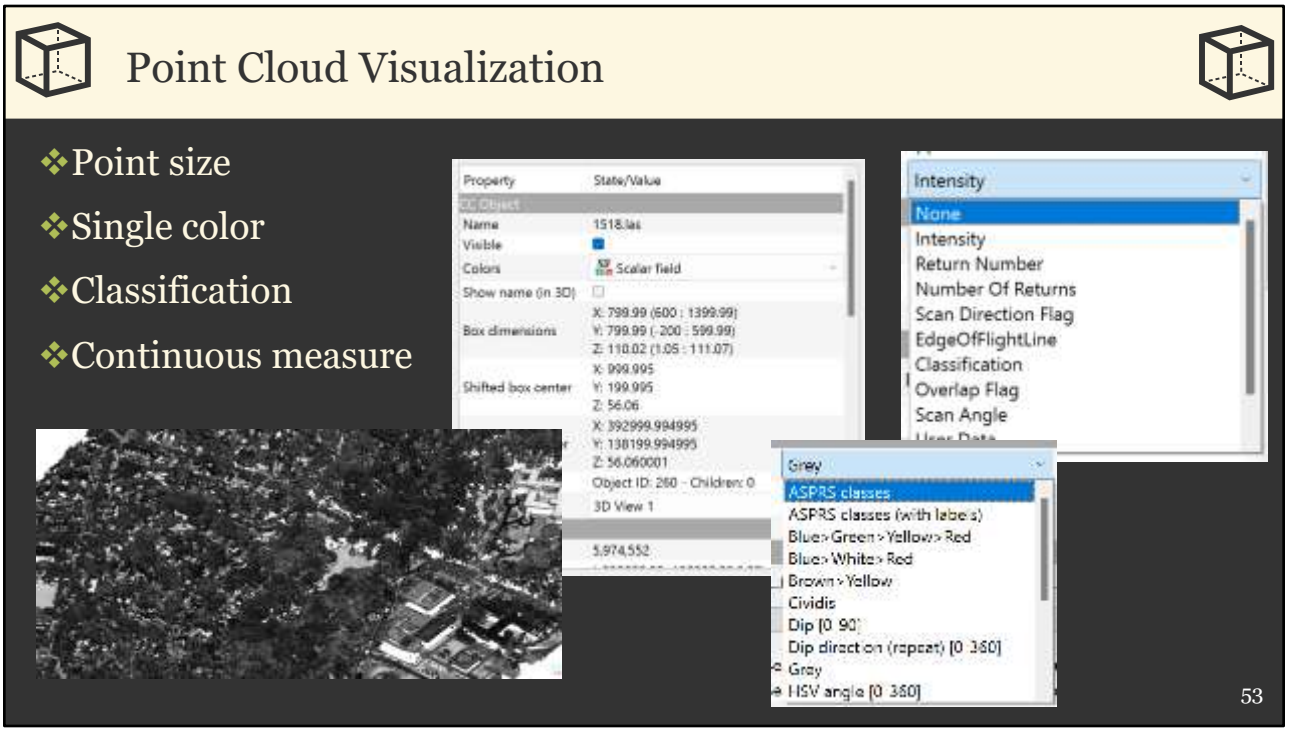
CloudCompare can open multiple file formats including .las, .e57, .ptx, and others. Broad format support makes CloudCompare compatible with data from most point cloud sources.

Save Point Cloud

- File → Save

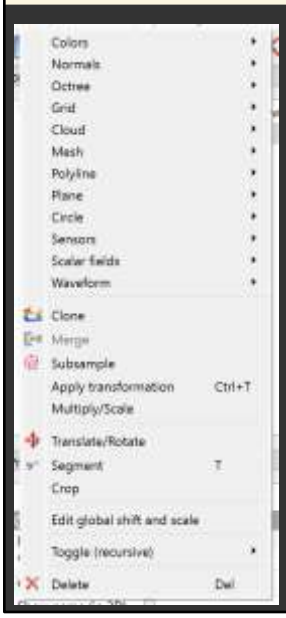


Point clouds are saved in CloudCompare using the File > Save menu. Multiple export formats are available.



Point cloud visualization options include rendering by point size, single color, classification values, or continuous measures. Different visualization approaches reveal different aspects of the data.

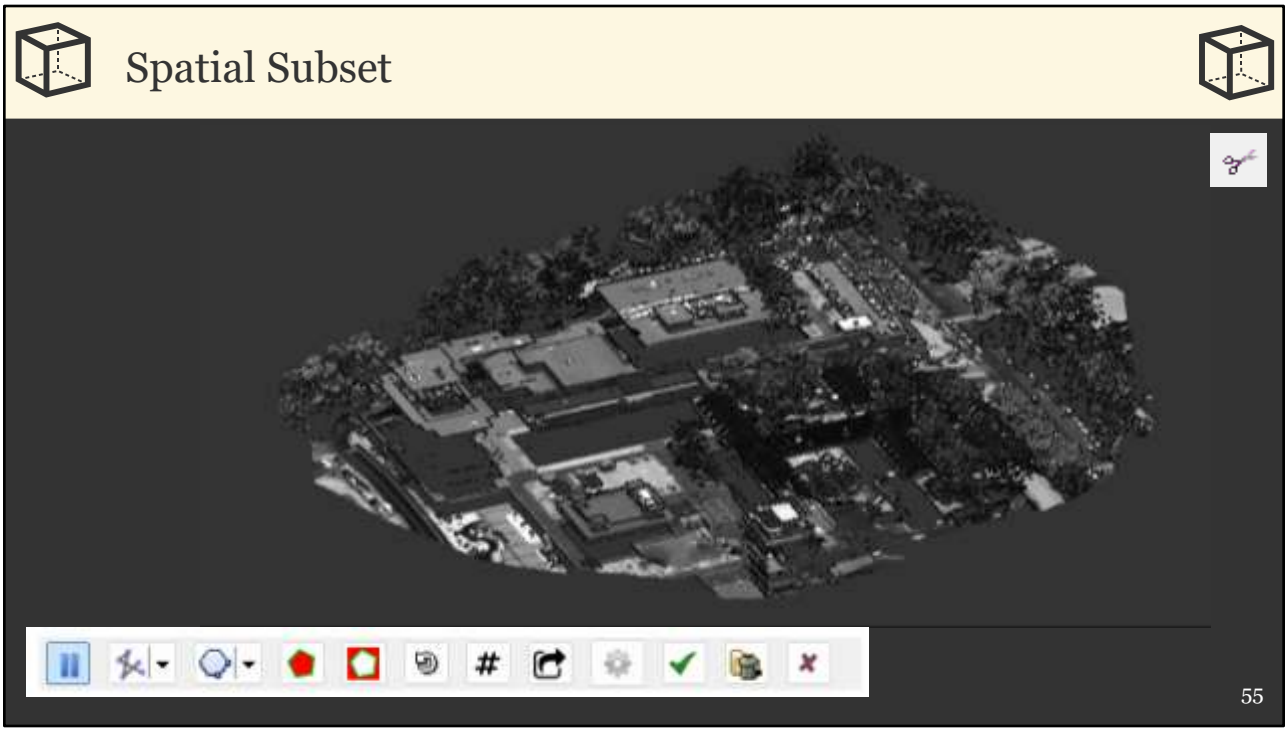
General Processing Tasks



- Calculate normals
- Calculate Octree
- Statistical outlier removal
- Noise removal
- Change global shift and scale
- Convert to mesh
- Subsample or thin data
- Scale
- Segment
- Crop
- Translate/rotate
- Clone scan
- Merge scans
- Apply transformation matrix

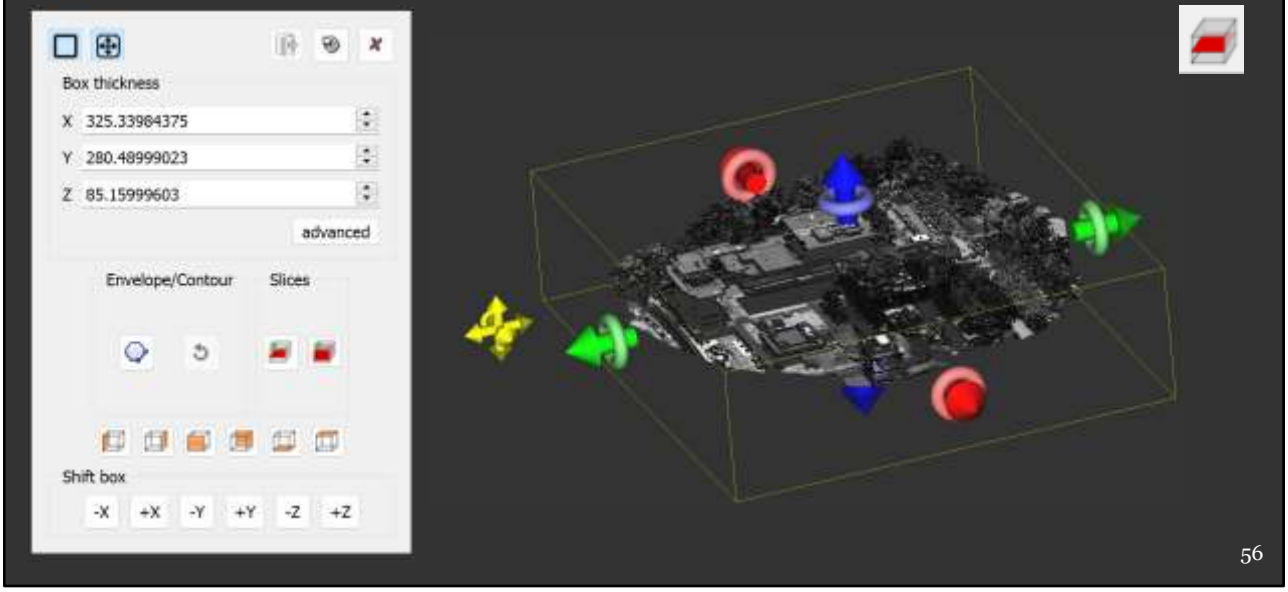
54

General processing tasks in CloudCompare include calculating normals for surface orientation, computing octrees for spatial indexing, removing statistical outliers, filtering noise, adjusting global shift and scale, converting to mesh representations, subsampling data, scaling, segmenting, cropping, translating and rotating, cloning scans, merging multiple scans, and applying transformation matrices.

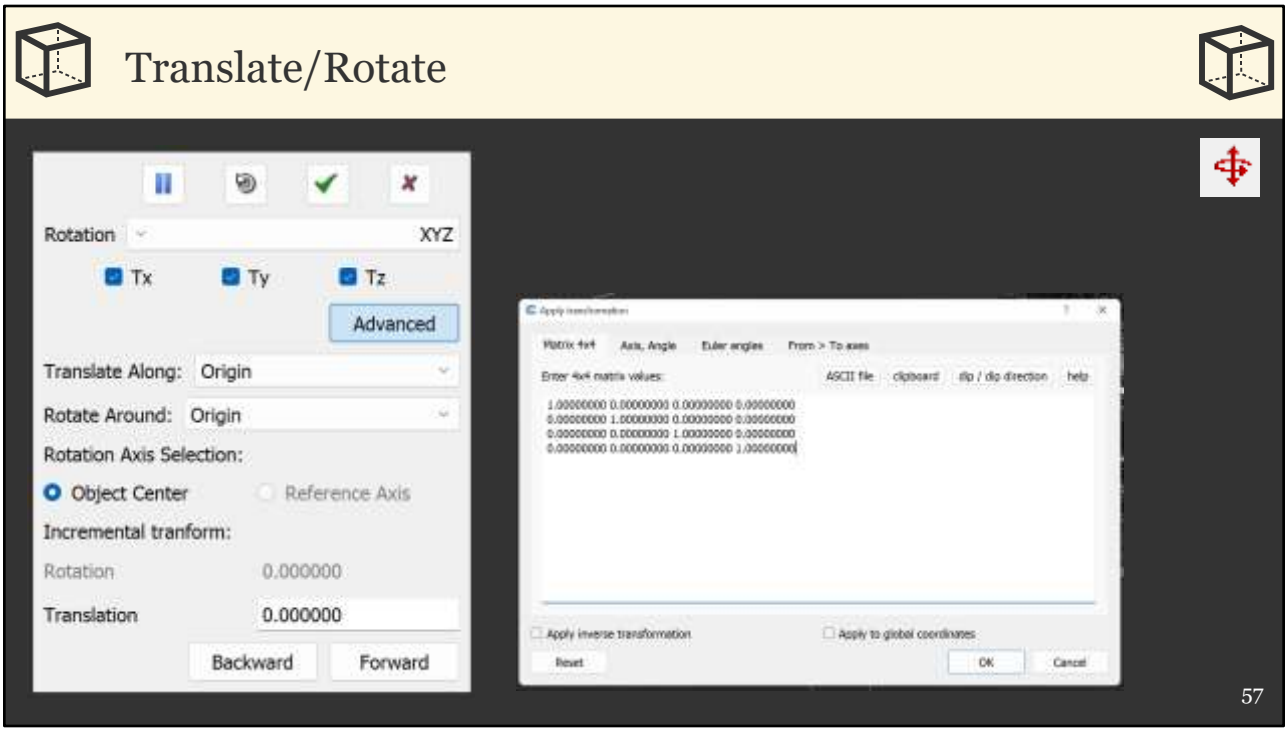


Spatial subsetting selects portions of point clouds based on spatial criteria or a drawn extent. This reduces data size for focused analysis.

Cross Section

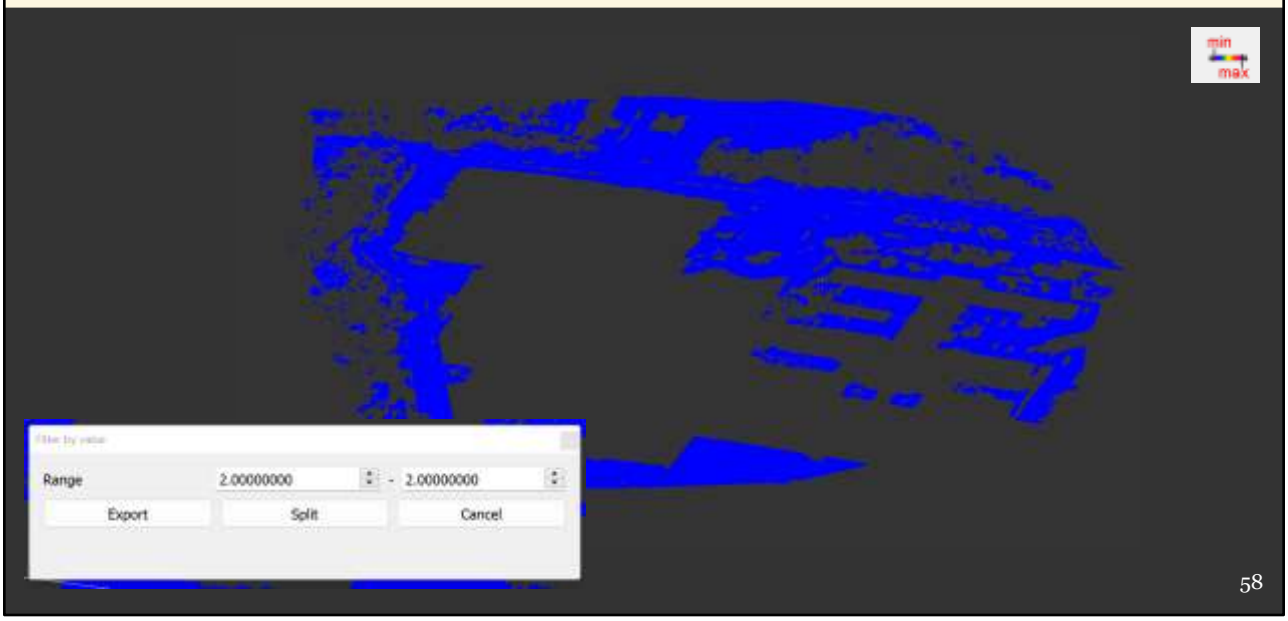


Cross-sectioning allows you to crop data relative to different axes or a bounding box.



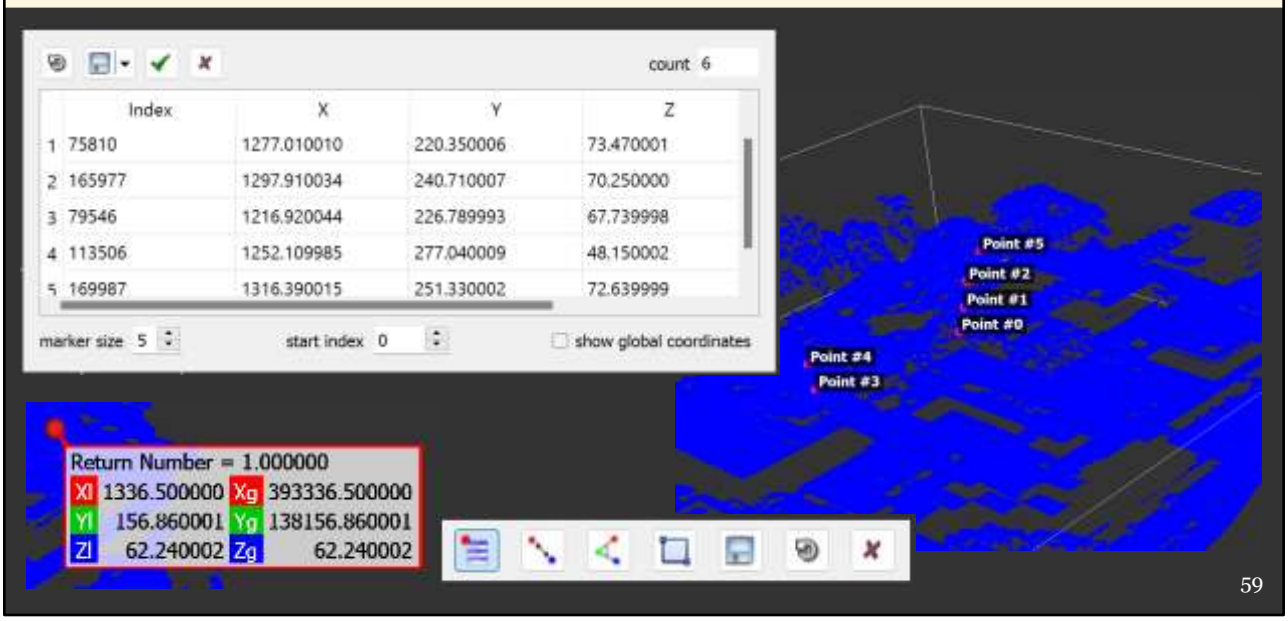
Translation and rotation transforms reposition point clouds in space. These operations are essential for aligning multiple scans.

Filter by Value or Class



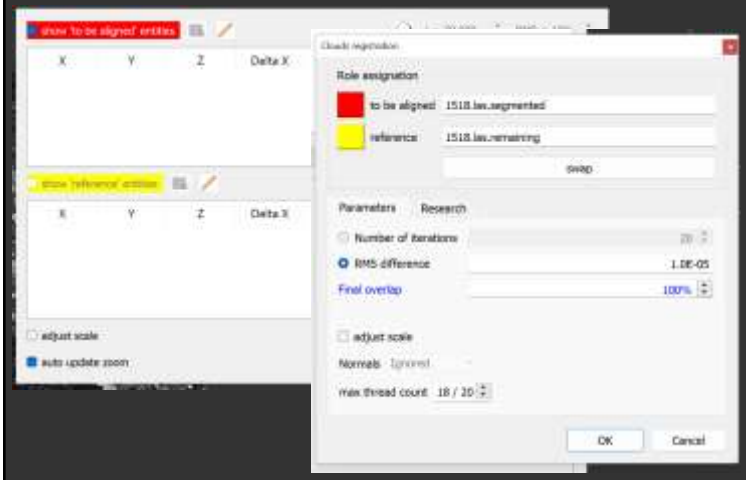
Filtering by value or class removes points meeting specified criteria. This is crucial for isolating specific data types.

Point Picker



The point picker tool selects individual points for examination and analysis. This enables detailed point-level inspection.

Co-registration



- Similar to georeferencing
- Pick points to align
- Generates a transformation matrix to align one scan with another
- Fine registration with ICP (iterative closest point)

Besl, P.J. and McKay, N.D., 1992, April. Method for registration of 3-D shapes. In *Sensor fusion IV: control paradigms and data structures* (Vol. 1611, pp. 586-606). Spie.

60

Co-registration aligns multiple scans from different positions. Similar to georeferencing, you pick corresponding points between scans. The process generates a transformation matrix. Fine registration uses Iterative Closest Point (ICP) for precise alignment.

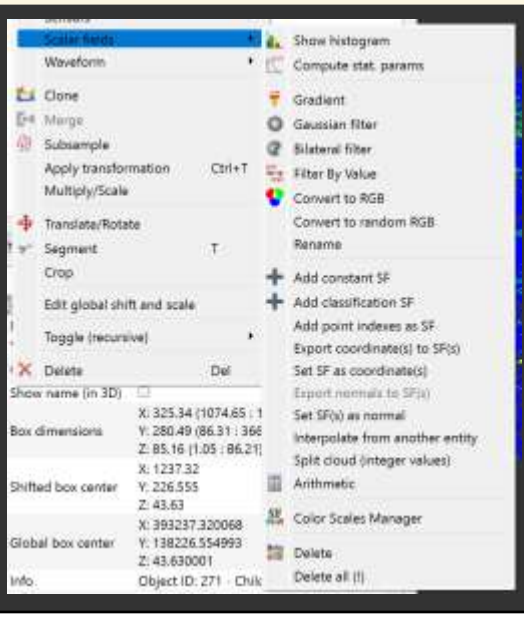
Rasterization

- Define cell size (step)
- Set active value
- Statistic to calculate
- Gap-filling method

61

Rasterization creates 2D grids from point clouds. You define cell size, select an active value to rasterize, choose a statistic to calculate, and select a gap-filling method.

Manual Classification



- Select/subset points to reclassify
- Edit → Scalar fields → Add constant SF
- Merge subsets as needed

62

Manual classification reclassifies points by selecting subsets and editing scalar fields. Selected subsets can be reclassified as needed and merged.

Some Implemented Algorithms

- Random Sampling and Consensus (RANSAC): shape detection
- Cloth Simulation Filter (CSF): ground classification
- Iterative Closest Point (ICP): fine co-registration
- Poisson Surface Reconstruction: point cloud → mesh
- 3DMASC: point cloud classification
- CANUPO: point cloud classification
- TreeIso: isolate individual trees

63

CloudCompare implements numerous algorithms for point cloud processing. RANSAC detects geometric shapes. CSF classifies ground points. ICP performs fine co-registration. Poisson surface reconstruction converts points to mesh. 3DMASC and CANUPO provide point classification. TreeIso isolates individual trees.

lidR

64

lidR is a comprehensive R package for processing airborne LiDAR data. It provides tools for reading, filtering, classifying, and analyzing point cloud data within the R statistical environment.

lidR

65

The lidR package offers integrated workflows for complete LiDAR analysis in R. It's particularly powerful for statistical analysis and integration with other geospatial R packages.

Reading and Filtering

```
lasAll <-
readLAS("gslrData/chpt4/data/points.laz",
                  select = "xyzicr",
                  filter = "-drop_withheld
-keep_xy 644077 4191186  646579 4193195")

lasGrd <-
readLAS("gslrData/chpt4/data/points.laz",
                  select = "xyzicr",
                  filter = "-drop_withheld
-keep_class 2 -keep_xy 644077 4191186
646579 4193195")
```

Filter	Use
filter_poi()	Use query
filter_first()	First returns
filter_firstlast()	First and last returns
filter_firstofmany()	First of many returns
filter_ground()	Ground classified
filter_last()	Last returns
filter_nth()	Based on return number
filter_single()	Single returns
filter_duplicates()	Find duplicates

Function	Description
clip_roi()	Clip based on vector geometry (e.g., sf object)
clip_rectangle()	Clip based on rectangular extent defined by x-left, y-bottom, x-right, and y-top coordinates
clip_circle()	Clip based on circular extent defined by x- and y-center coordinates and a radius
clip_transect()	Clip along a transect between two points and using a given width

66

Reading and filtering LAS files in lidR is straightforward. The readLAS function reads data and applies filters. The filter parameter allows selecting subsets by attributes, classification, spatial extent, or return type.

Reading and Filtering

Attribute	Description
x	X coordinate
y	Y coordinate
z	Z coordinate (elevation)
i	Return intendity
t	GPS time
a	Scan angle
n	Number of returns associated with pulse
r	Return number
c	Classification
s	Synthetic data flag
k	Keypoint flag
w	Withheld flag
o	Overlap flag
p	Point source ID
e	Edge of flight line flag
d	Direction of scan flag

```
lasAll <-
readLAS("gslrData/chpt4/data/points.laz"
,
                  select = "xyzicr",
                  filter = "-
drop_withheld -keep_xy 644077 4191186
646579 4193195")

lasGrd <-
readLAS("gslrData/chpt4/data/points.laz"
,
                  select = "xyzicr",
                  filter = "-
drop_withheld -keep_class 2 -keep_xy
644077 4191186  646579 4193195")
```

67

Attribute codes in lidR represent point cloud characteristics. x, y, z coordinates, intensity, GPS time, scan angle, return metrics, classification, and various flags are accessible.

Reading and Filtering

- `readLAS()`
- `readALSLAS()`
- `readTLSLAS()`
- `readUAVLAS()`

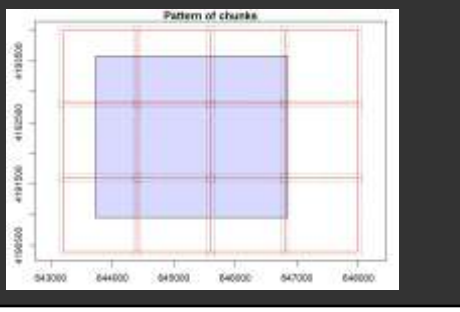
68

Specialized reading functions accommodate different data sources: readLAS() for general data, readALSLAS() for airborne data, readTLSLAS() for terrestrial data, readUAVLAS() for unmanned aerial vehicle data.

LAS Catalog

- Reference and work with multiple LAS files at once
- Processing engine for larger datasets
- Processing completed using tiles or chunks
- Process in memory or save temp files to disk
- User-defined processing options

```
lasCat <-
readLAScatalog("gslrData/chpt4/data/")
```



69

A LAS Catalog references multiple LAS files simultaneously. This enables processing larger datasets by managing tiles or chunks. Processing can occur in memory or using saved temporary files.

Structure in R

- S4 LAS object
- @head stores information and metadata about the file
- @data stores point data and associated attributes

X	Y	Z	Intensity	ReturnNumber	Classification
646430.9	4193103	2137.37	24939	1	2
646430.8	4193102	2137.31	20864	1	2
646430.8	4193101	2137.40	18582	1	2
646430.8	4193099	2137.33	27384	1	2
646430.7	4193098	2137.34	26895	1	2
646430.7	4193097	2137.66	24287	1	2

70

The S4 LAS object structure in R includes a head slot storing metadata and a data slot containing point data with associated attributes.

Performing Checks

```
Checking the data
 - Checking coordinates... ✓
 - Checking coordinate type... ✓
 - Checking coordinate range... ✓
 - Checking coordinate quantization... ✓
 - Checking attribute type... ✓
 - Checking ReturnNumber validity... ✓
 - Checking NumberOfReturns validity... ✓
 - Checking ReturnNumber vs. NumberOfReturns... ✓
 - Checking NIR validity... ✓
 - Checking absence of NAs... ✓
 - Checking duplicated points...
   ⚠ 4716 points are duplicated and share XYZ coordinates with ot
 - Checking degenerated ground points...
   ⚠ There were 859 degenerated ground points. Some X Y Z coordin
   ⚠ There were 111 degenerated ground points. Some X Y coordinat
 - Checking attribute population... ✓
 - Checking gpstime incoherence skipped
 - Checking flag attributes... ✓
 - Checking user data attribute... skipped
Checking the header
 - Checking header completeness... ✓
 - Checking scale factor validity... ✓
 - Checking point data format ID validity... ✓
 - Checking extra bytes attributes validity... ✓
 - Checking the bounding box validity... ✓
 - Checking coordinate reference system... ✓
Checking header vs data adequacy
 - Checking attributes vs. point format... ✓
 - Checking header bbox vs. actual content... ✓
 - Checking header number of points vs. actual content... ✓
 - Checking header return number vs. actual content... ✓
Checking coordinate reference system...
 - Checking if the CRS was understood by R... ✓
Checking preprocessing already done
 - Checking ground classification... yes
 - Checking normalization... no
 - Checking negative outliers... ✓
 - Checking Flightline classification... skipped
Checking compression
 - Checking attribute compression... no
```

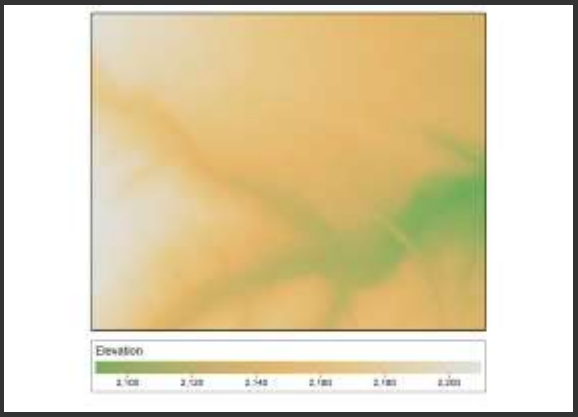
```
las_check(lasAll)

lasAll <- lasAll |> filter_duplicates()
lasGrd <- lasGrd |> filter_duplicates()
```

71

Data quality checks identify issues. The las_check function validates data structure. Duplicates can be removed using the filter_duplicates function.

Ground Classification and Create DTM

```
dtm <- rasterize_terrain(lasGrd,
                         res=5,
                         algorithm=knnidw(k = 10L,
                                          p = 2))
```



- Progressive Morphological Filter (PMF) (`pmk()`)
- Cloth Simulation Function (CSF) (`csf()`)
- Multiscale Curvature Classification (MMC) (`mcc()`)
- Progressive TIN Densification (PTD) (`ptd()`)

72

Ground classification and DTM creation uses the rasterize_terrain function with selected algorithms. Multiple algorithms are available: Progressive Morphological Filter (PMF), Cloth Simulation Function (CSF), Multiscale Curvature Classification (MCC), and Progressive TIN Densification (PTD).

Terrain Derivatives (Land Surface Parameters)


```
slp <- terrain(dtm,
               v ="slope",
               unit="radians")
asp <- terrain(dtm,
               v ="aspect",
               unit="radians")

hs <- shade(slp,
            asp,
            angle=45,
            direction=315)
```

73

Terrain derivatives can be calculated once a DTM is generated.

Canopy Height Model (CHM)

```
dsm <- rasterize_canopy(lasAll,
                        res = 5,
                        pitfree(thresholds = c(0, 10, 20),
                                max_edge = c(0, 1.5)))

fill.na <- function(x, i=5) { if (is.na(x)[i]) { return(mean(x, na.rm = TRUE)) }
else { return(x[i]) }}
w <- matrix(1, 3, 3)

dsm <- terra::focal(dsm,
                    w,
                    fun = fill.na)
dsm <- terra::focal(dsm,
                    w,
                    fun = mean,
                    na.rm = TRUE)


ndsm <- dsm - dtm
```

74

Canopy height models (CHM) represent vegetation height above ground. The rasterize_canopy function is used to estimate canopy height models.

Canopy Height Model (CHM)

```
chm <- rasterize_canopy(las, res = 0.5,
algorithm = dsmtin())

chm <- rasterize_canopy(las, res = 0.5,
pitfree(thresholds = c(0, 10, 20), max_edge
= c(0, 1.5)))

chm <- rasterize_canopy(las, res = 0.5,
spikefree(1.5))
```

- Triangulation (`dsmtin()`)
- Pit-Free Algorithm (`pitfree()`)
- Spike-Free Algorithm (`spikefree()`)

75

Multiple algorithms are available to estimate CHMs including triangulation, pit-free, and spike-free.

Height Normalization

```
lasAllN <- normalize_height(lasAll,
knnidw())
```

- *k*-Nearest Neighbor (*k*NN) inverse distance weighting (IDW)
- Can use a DTM or work from ground classified points

76

Height normalization estimates height above ground for all points. Methods include *k*-nearest neighbor (*k*NN) and inverse distance weighting (IDW). You can normalize from a raster-based DTM or ground classified points.

Point Cloud Summarization

- `cloud_metrics()`: entire point cloud
- `plot_metrics()`: within RoI or polygon extent
- `crown_metrics()`: for each differentiated tree
- `grid_metrics()`: within pixels
- `voxel_metrics()`: within voxels
- `point_metrics()`: for each point using points in local neighborhood

PlotID	zmax	zmean	zsd	zskew	zkurt	zentropy	pzabovezmean	pzabove2	zq5
1.00	8.22	2.14	1.75	0.87	3.65	0.77	48.35	51.65	0.15
2.00	7.53	2.94	1.66	0.12	2.54	0.89	49.60	71.54	0.18
3.00	7.03	3.05	1.43	−0.09	2.74	0.84	51.14	78.04	0.24
4.00	8.57	3.97	1.70	−0.38	2.87	0.86	53.67	87.78	0.23
5.00	34.03	3.44	1.82	0.83	13.00	0.54	49.72	81.46	0.17
6.00	8.51	3.32	1.67	−0.01	2.61	0.86	49.89	78.78	0.20
7.00	6.89	2.40	1.14	0.00	3.23	0.78	51.01	65.46	0.19
8.00	7.90	3.25	1.78	−0.19	2.20	0.90	53.15	75.44	0.18

77

Point cloud summarization calculates metrics at different scales. cloud_metrics summarizes entire point clouds, plot_metrics summarizes within regions of interest, crown_metrics summarize individual trees, grid_metrics summarize within pixels, voxel_metrics summarize within voxels, and point_metrics summarize neighborhoods around points.

Point Cloud Summarization

- Can summarize *x*-coordinates, *y*-coordinates, *z*-coordinates, intensity, RGB values, ext.
- Example metrics
  - Mean
  - Median
  - Range
  - Minimum
  - Maximum
  - Interquartile Range
  - Standard Deviation
  - Entropy
  - Percentiles
  - Quantiles
  - L-moments
  - Fractal dimensions
  - Custom metrics

78

Metrics summarize x, y, z coordinates, intensity, RGB values, and more. Computed statistics include mean, median, range, minimum, maximum, interquartile range, standard deviation, entropy, percentiles, quantiles, L-moments, fractal dimensions, and custom metrics.

Tree Detection and Segmentation

- Tree detection (`locate_trees()`)
  - Detect tree locations as points
  - Local maximum filter with fixed window size (`lmf()`)
  - Local maximum filter with variable window size (`lmf()`)
- Tree segmentation (`segment trees_trees()`)
  - Dalponte and Coomes (2016) algorithm (`dalponte2016()`)
  - Watershed (`watershed()`)
  - Li et al. (2012) algorithm (`li2012()`)
  - Silva et al. (2016) algorithm (`silva2016()`)

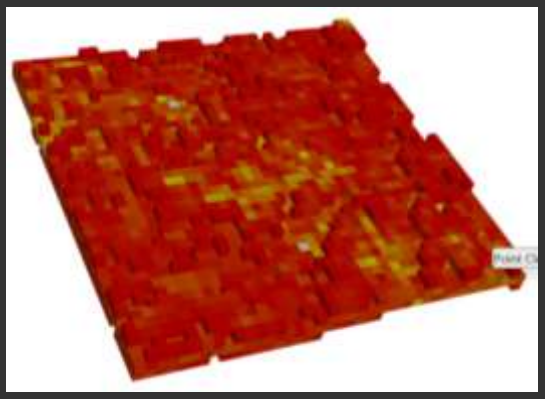
https://r-lidar.github.io/lidRbook/itd.html

79

Tree detection and segmentation identifies individual trees in point clouds. locate_trees detects tree locations. Local maximum filter approaches identify canopy peaks. Multiple segmentation algorithms including Dalponte, watershed, Li, and Silva algorithms provide different segmentation characteristics.

Convert to Voxel

1. Compute voxel metrics with lidR
2. Split the voxel table by z level
3. Rasterize each z slice to a 2D layer
4. Stack layers into a SpatRaster
5. assign the z/depth coordinate
6. export with `writeCDF()`



80

Voxel conversion creates volumetric representations. lidR computes voxel metrics, splits by depth level, rasterizes each z-slice to 2D layers, and stacks layers into a 3D raster. NetCDF format exports are supported.

Write Output to Disk

- `writeLAS()`
- `writeRaster()`
- `write.csv()`

81

Output products are written to disk using writeLAS for point clouds, writeRaster for raster products, and write.csv for tabular data.

This is the end of this lecture module.

Please return to the West Virginia View Webpage for additional content.

82

Thanks! Hope you found this useful.