3D Analyst – Working with Terrain Datasets

Lindsay Weitz
Content

• Overview
• Terrain dataset implementation
• Terrain dataset analysis
• Demonstration
• Suggestions
• Resources
Terrain Dataset

- A Terrain is a multi-resolution surface created from measurements stored in feature classes

- Typical applications:
  - Topographic mapping
  - Bathymetric mapping

- Typical data sources:
  - Photogrammetric data
  - Lidar
  - Sonar
Motivating Forces

• Scalability
  - Large collections of mass point data (e.g. LIDAR) have been a problem

• Data integration
  - Need surface to live with source data

• Data management
  - Database tools
  - Editing/update
  - Multi-user
Limitations to Overcome

- TINs have an effective limit of 20 million points
  - Based on 2GB per process limit of Win32
  - It’s recommended not to go over 3-5 million
- Updating TINs relative to edits of source measurement data is difficult
  - They are disconnected
  - Easiest thing to do is rebuild from scratch
- TINs only support Workstation Arc/Info projections
- Rasters are derivative
  - Difficult to update without rebuilding from source data
Need for Maintaining Topographic Baseline

- Many organizations are charged with keeping accurate and up to date topographic/bathymetric surfaces
  - Construction projects/permitting
  - Hydrologic/hydraulic modeling
  - Navigation
- Terrains offer database oriented solution for managing source data from which these surfaces are derived
What are Terrain Datasets?

- Terrain datasets live inside feature datasets within the geodatabase.
- Identify which feature classes participate and how they contribute.
- Rules specify how features are used to define a surface.
Multi-Resolution Surface Model

- Points and Breaklines
- Terrain Pyramids
  - Thinned Point Set
  - Full Resolution

Multi-resolution terrain dataset (TIN structure)
Implementation – Levels of Detail

- TIN surface generated on-the-fly for given area of interest and level of detail
- Supports point, multipoint, polyline, and polygon based features
- Seamless
- Fast
- Scalable
Implementation - Tiling

- Spatial coherence and *tiling* (point clustering)
- Z tolerance and *vertical indexing*
- Measurement update and *dirty-areas*
- Localized processing
Implementation - Tiling

- Data is structured, internally, into tiles
- Spatially coherent parts
- Each tile contains a manageable amount of data
- Facilitates processing large amounts of data
Tile System Definition

• Defined by nominal point spacing and coordinate system
  - Point spacing controls tile size
  - Coordinate system defines origin and extent

• Terrain maintains properties that define tile system
  - Tile boundaries are not stored
  - Mathematically derived on-demand
Preventing/Reducing Tile Artifacts

• Problem associated with generic tile based processing
  - Interpolation neighborhoods are incomplete around tile boundaries
  - Artifacts when merging results of interpolation for multiple tiles

• Terrains address this issue automatically
  - Overlapped tiles provide a solution
  - Since neighborhoods are well defined around neat line boundaries tile derivatives merge seamlessly
Preventing/Reducing Tile Artifacts

- Systems that only use overlapped tiles can still have problem with incomplete or empty tiles
  - Occur over water bodies, obscured areas
- Terrain handles these problematic tiles automatically by identifying and flagging them as *composite* tiles
  - Include references to nearest points in surrounding tiles
  - Complete surface definition for area represented by tile
Composite Tiles

Tile in center is void of samples but references those in neighboring tiles. Triangulation of those points covers the tile.
Vector Based Pyramids

- Similar to raster pyramids in concept, but comprised of source measurements

- Point thinning
  - Heavy thinning for coarse levels
  - Lighter thinning for more detailed levels
  - No thinning at full resolution

- User defined scale threshold associated with each level

- For analysis as well as display

- Two pyramiding techniques: Z Tolerance or Window Size
Cumulative Pyramid Levels

Base set of points comprise coarest pyramid level.

Additional points.

The combination of point sets define the next higher detailed pyramid level.
Z Tolerance Pyramid

- TIN based decimation
- Generalized surface, for each pyramid level, within user defined vertical accuracy of full resolution surface
- Appropriate for bare earth data
- Should not be used with 1st return lidar surfaces (i.e., buildings and vegetation)
# Z Tolerance Pyramid

The distance between the lower resolution surfaces and the full resolution surface will not exceed a given Z-tolerance.

<table>
<thead>
<tr>
<th>Points</th>
<th>Levels of Detail</th>
<th>Z-Tolerance</th>
<th>Scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1:1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2500</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>5</td>
<td>10,000</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>10</td>
<td>50,000</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>20</td>
<td>100,000</td>
</tr>
</tbody>
</table>
Window Size Pyramid

- Simple binning or block filter
- Space partitioned into squares and one or two points selected for each square
- Selection criteria:
  - Min z, max z, min and max z, closest to mean z
- Effective for all data types
- Should be used with 1st return lidar
## Window Size Pyramid

<table>
<thead>
<tr>
<th>Level</th>
<th>Window Size</th>
<th>Scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1:1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2500</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>5,000</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>10,000</td>
</tr>
</tbody>
</table>
Pyramid Levels

- The multiple levels of a terrain pyramid as defined by a series of scale thresholds and associated vertical tolerances.

- Points per pyramid level for a terrain of the Neuse River Basin in North Carolina are included to show relationship between point count and vertical tolerance.

<table>
<thead>
<tr>
<th>Scale Threshold</th>
<th>Vertical Tolerance (feet)</th>
<th>Points Per Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:1,000</td>
<td>0.0</td>
<td>477,032,641</td>
</tr>
<tr>
<td>1:5,000</td>
<td>1.0</td>
<td>58,625,869</td>
</tr>
<tr>
<td>1:10,000</td>
<td>2.5</td>
<td>13,300,911</td>
</tr>
<tr>
<td>1:20,000</td>
<td>5.0</td>
<td>3,791,397</td>
</tr>
<tr>
<td>1:50,000</td>
<td>10.0</td>
<td>1,042,804</td>
</tr>
<tr>
<td>1:100,000</td>
<td>25.0</td>
<td>263,615</td>
</tr>
</tbody>
</table>
Point Clustering

- One database row per point is too expensive
- Instead, points belonging to the same tile and pyramid level are grouped into *multipoints*
- A multipoint is stored as an individual shape occupying one database row
- Reads and writes become more efficient
- Storage cost is reduced
- Only measurements are stored, TINs built on-the-fly
Point Clustering

- Many points are combined into a shape called a multipoint that is stored using one database row.
Input Data Formats - LAS

• LAS files are industry standard binary format for lidar

• Loaded using LAS to Multipoint tool

• Benefits
  - Avoids pitfall associated with ASCII format points
  - Extent, point count, and spatial reference in header

• Drawbacks
  - Built in metadata is lacking in some areas
    - Can’t always tell how ‘raw’ the data is
    - Classification codes are not described
Input Data Formats - ASCII

- **XYZ, XYZI**
  - 3D points
  - Loaded using ASCII3DToFeatureClass GP tool

- **GENERATE**
  - 3D points, lines, polygons
  - Loaded using ASCII3DToFeatureClass GP tool
Handling Lidar (LAS) Attributes

• Per point attributes (e.g. return number, class code) optionally stored in BLOBs
• A separate BLOB field is used for each attribute
• Array of values with one-to-one correspondence with a set of grouped points is stored with points in same database row
Editing

• Updates accomplished through edits to source measurements
  - Coarse grained area operators to append, remove, replace mass points
  - Standard/custom edit tools (e.g. ArcEditor) used to modify polylines, polygons, spot heights
  - Terrain rebuild based on dirty-areas

• Support for versioning in SDE
Terrain Wizard
Terrain Dataset Layer
Interactive Surface Analysis

- Interactive surface tools

3D Analyst toolbar in ArcMap
Geoprocessing Analysis

- Geoprocessing with Terrain Datasets
  - Terrain Management toolset
    - Creation
    - Modification
  - Data conversion toolset
    - Data loading
    - Surface conversions
  - Terrain and TIN Surface toolset
    - Analysis conducted directly on terrains
Analysis Capabilities for Terrain Datasets

• QA/QC lidar data
• DEM / DSM creation
• Slope
• Aspect
• Contours
• Surface differencing
• Intensity image generation
• Estimating Forest Canopy
• Data area delineation
• Thinning / reducing noise
• Spot interpolation
• Profiling
Working with Terrain Datasets

Lindsay Weitz
Terrain Dataset Workflow

Data Conversion
- LIDAR post-processed data
- SONAR post-processed data
- Photogrammetric data
- Proprietary or de-facto standard formats

Product Generation
- ArcGIS Terrain Dataset

Surface Integration
- Pyramid TIN surfaces

Contours, Points, Breaklines, DEM, TIN
Common Analysis: Creating Raster DEMs and DSMs

- **Digital Elevation Model**: Bare earth surface made using only ground hits.
- **Digital Surface Model**: Includes ground, trees, and buildings made using first returns.
Mapping and Visualization - ArcMap

- Displayed as a TIN
- Symbology same as TIN
- Resolution changes depending on zoom level
Mapping and Visualization – ArcGlobe / ArcScene

- Terrain datasets can be displayed as either elevation or draped layers in ArcGlobe
- Terrain datasets are not directly supported in ArcScene
Converting TINs to Terrain datasets

• First, look to see if the source feature data used to make a TIN is available and use it to make a terrain.

• Only if the source feature data is not available:
  - Decompose a TIN to features with GP tools
  - Make the terrain from the features
A terrain dataset is a multiresolution, TIN-based surface built from measurements stored as features in a geodatabase. They're typically made from lidar, sonar, and photogrammetric sources. Terrains reside in the geodatabase, inside feature datasets with the features used to construct them.

Terrains have participating feature classes and rules, similar to topologies. Common feature classes that act as data sources for terrains include the following:

- Multipoint feature classes of 3D mass points created from a data source such as lidar or sonar
- 3D point and line feature classes created on photogrammetric workstations using stereo imagery
- Study area boundaries used to define the bounds of the terrain dataset

The terrain dataset's rules control how features are used to define a surface. For example, a feature class containing edge of pavement lines for roads could participate with the rule that its features be used as hard breaks. This will have the desired effect of creating linear discontinuities in the surface.

Rules also indicate how a feature class participates through a range of scales. Edges of pavement features...
Known Limits – Personal Geodatabase

- Not storage efficient
- Limited 2GB capacity
- Significant performance drop before capacity reached
- Not recommended for terrains over 20 million points
Known Limits – File Geodatabase

- ‘In-line’ geometry storage model makes terrain rebuilds relatively expensive.
- SDE supports ‘out-of-line’ storage and is more efficient in some regards because of this.
Known Limits – No Geographic Coordinates

- Terrain dataset use Delaunay triangulation
  - Method is valid only when data is projected
- Tools on user interface will prevent creation of terrains in feature datasets that use Angular Coordinate Systems
Best Practices

• LAS Over ASCII
• Use File or SDE GDB (Personal - 2GB Limit)
• Consider file or enterprise geodatabase for large datasets (> 1-2 billion points)
• Terrain dataset must be stored in a feature dataset
• Use projected coordinates
• Use Consistent Units (x, y, and z) and contiguous datasets
• Breakline enforcement
• Use ArcGIS for lidar derived rasters

Workflow to serve elevation:

1. Import 7 County LiDAR raster
2. Perform necessary geospatial processing in ArcGIS
3. Publish the processed data to an ArcGIS Server
4. Use the ArcGIS Server to serve the elevation data
Performance and Size Estimates

- **Import:**
  - 800 million LAS points per hour

- **Terrain pyramid build:**
  - 80 million points per hour using z-tolerance filter
  - 400 million points per hour using window size filter

- **Storage:**
  - 150 million points (geometry only) = 1GB
  - Terrain pyramid will be roughly same size as source multipoint feature class so total storage can double
    - Can use option to embed points to recover space

Timed using HP xw4400 Core2 Duo 2.67 GHz PC
Reads/writes using same drive
File Geodatabase
Steps to evaluate UC sessions

- My UC Homepage > “Evaluate Sessions”

- Choose session from planner
  OR

- Search for session

www.esri.com/ucsurveyseessions
• Thank you for attending
• Have fun at UC2012
• Open for Questions

• Please fill out the evaluation:
  
  www.esri.com/ucsessionssurveys

  Offering ID: 950