Hands-on practice: LIDAR data quality analysis and fine-georeferencing

Christian Briese
cb@ipf.tuwien.ac.at

1 Institute of Photogrammetry and Remote Sensing
Vienna University of Technology

2 Ludwig Boltzmann Institute for Archaeological Prospection and Virtual Archaeology, Vienna

ADDRESSSS training course, 19-28 August 2010, Balaton Limnological Research Institute, Hungary
ALS data acquisition

- **Configuration**
  - Laser Scanner (LS) \((v, \chi, r)\)
  - Inertial Measurement Unit (IMU) \((\omega, \varphi, \kappa)\)
  - Global Positioning System (GPS) \((X_0, Y_0, Z_0)\)

- **Synchronisation**
  - Time Stamp \((t)\)
What are the characteristics of ALS data?

- **Point density**
  → ALS-points are scattered irregular on the ground; distribution depends on flying height, flight movements, etc.

- **Random errors**
  Are caused by measurement noise.

- **Systematic errors**
  Are caused by errors of the calibration of the sensors, and errors of the relative and absolute orientation of the strips.
**Systematic errors**

- Two types:
  - Absolute: discrepancies at ground check features
  - Relative: discrepancies between adjacent, overlapping laserscanner strips

- Errors of the ALS data directly influence the quality of the derived products (DTM)

- Reasons:
  - IMU misalignment
  - GPS initialization
  - Calibration error, …

- Possible solution:
  - absolute / relative improvement of orientation using strip adjustment
    → improved transformation parameters

Strip differences documenting errors of relative orientation

2 strips  
Overlap  
Differences: original:
Examples

- Improvement of transformation parameters using strip adjustment

Strip difference of original data:

Strip difference after strip adjustment:
ALS quality documentation

- **point density** (per strip and for the aggregation of all strips)
- **measurement noise** → accuracy of points (sigma-dtm)
- **relative orientation** → strip differences

The following programs are used
- **OPALS** (Orientation and Processing of ALS Data) – scientific processing software
- **SCOP++**
- **SCOP.GVE**
OPALS - Orientation and Processing of Airborne Laser Scanning data

OPALS stands for Orientation and Processing of Airborne Laser Scanning data. It is a modular program system consisting of small components (modules) grouped together thematically in terms of packages. The aim of OPALS is to provide a complete processing chain for processing airborne laser scanning data (waveform decomposition, georeferencing, quality control, structure line extraction, point cloud classification, DTM generation and several fields of application like forestry, hydrology/hydraulic engineering, city modelling and power lines).

The manual is divided into three parts, each of which is sub-divided into several sections.

User Documentation

- Section Installation discusses how to download and install OPALS
- Section Getting Started gives a 15 minute introduction on how to use OPALS
- Section Software Concept describes the basic concept of OPALS in detail
- Section Workflow Management shows how to combine OPALS modules using scripts
- Section Supported Formats overviews the supported vector and raster file formats
- Section FAQ answers frequently asked questions concerning OPALS
- Section Bibliography contains a list of OPALS related articles

Reference Documentation

- Section Module Reference contains a list of all OPALS modules and a detailed description of each module
- Section OPALS Datamanager describes the ALS data administration concept in detail
- Section Parameters / Configuration Files / Parameter Mapping explains parameter categories and types, and how to specify respective values
- Section Logging / error handling contains details about the way OPALS logs information and handles errors
- Section Filters explains the detailed syntax used to filter vector data
- Section OPALS Format Definition shows how to operate with generic user-defined vector formats
- Section Using Python Bindings describes how to embed OPALS modules in a Python programming/scripting environment
- Section Using C++ Bindings deals with embedding OPALS modules in a C++ programming environment
- Section C++ API Reference contains the detailed OPALS module class documentation (public functions, parameters, etc.)
- Section Third Party Software lists all the external libraries and programs used within OPALS
- Section Glossary contains a list of a keywords and acronyms together with a description of their meaning

OPALS Packages

- Package opalsPreprocess:” Signal analysis and point cloud derivation
- Package opalsQuality: Quality control and documentation
- Package opalsGeoref:” ALS strip adjustment
- Package opalsGeomorph:” Terrain feature extraction (breaklines, lines, etc.)
- Package opalsClassify:” 3D classifi cation of ALS point cloud
- Package opalsSurface:” Surface interpolation (DTM/DSM) and visualisation
- Package opalsHydro:” Hydrology/Hydraulics applications
- Package opalsForests:” Forestry applications
- Package opalsCity:” Building and city modelling

*” package not yet available **” package only available partially
OPALS Processing

- Running OPALS modules:
  open a Command Prompt (e.g. Start → Run → cmd)
or Start → All programs → Accessories → Command Prompt
  - change to your project directory
  - start the program with the appropriate input parameters
  - e.g.
    C:\opalsCell -i input.odm -cellSize 5 -feature pdens -oFormat Gtiff
    (one such call may cover several lines on the screen)
  - Several calls can be put in a so-called batch-file (.bat).

- Help on OPALS:
  C:\Program Files\OPALS\doc\opalsManual.html
Before any OPALS module can work with the ALS-data, that data needs to be imported and stored in a suitable format (ODM = OPALS Data Manager). This is done by **opalsImport**

**Example 1:**

```bash
opalsImport -inFile G101ALL.bxyz
```

→ Imports the points on file G101ALL.bxyz and generates G101ALL.odm.dat and G101ALL.odm.idx. This file pair is later referenced by G101ALL.odm

**Example 2:**

```bash
opalsImport -inFile G101ALL.bxyz -inFile G102ALL.bxyz -inFile G102ALL.bxyz -outFile ALL.ODM
```

→ Imports the points on the files G101ALL.bxyz, G102ALL.bxyz and G103ALL.bxyz and generates ALL.odm.dat and ALL.odm.idx. This file pair is later referenced by ALL.odm
opalsCell

program to derive one representative z-value per raster cell from all original points inside the cell. The parameter –feature defines this representative value.

Important parameters:
--inFile: Input file
--feature:
  * min: lowest attribute value
  * max: highest attribute value
  * nmin: n-th lowest attribute value
  * nmax: n-th highest attribute value
  * mean: mean (average) of all attribute values
  * median: median of all attribute values
  * rms: root mean square of all attribute values
  * pdens: point density of all (valid) cell points
  * pcount: point count of all (valid) cell points
--cellSize: grid width of output (s in figure right)
--outFile: (optional)

e.g.
opalCell -inFile L:\TOM_UE\part1\group1\G105ALL.ODM -cellSize 5 -feature pdens
→ Creates file G105ALL_pdens.tif

Note: The tif-files created by many OPALS modules contain float-values and not 8bit. Thus viewing these tif-files in e.g. IrfanView makes not much sense.
program to create a digital elevation model from a given point set by using either snap grid, nearest neighbour, moving average or moving planes interpolation.

Important parameters:
--inFile: Input file
--interpolation:
  * Moving planes: For each grid cell n nearest ALS points (-neighbours) are queried and a best fitting tilted plane (minimizing the vertical distances) is estimated. The height of the resulting plane at the grid point (x,y) position is mapped to the grid cell
--neighbours: Number of nearest neighbours used for grid point interpolation
--searchRadius: Maximum search radius for point selection (smax in figure right). Only points within smax are considered for the interpolation of a single grid post. If the search area contains too few points for successful interpolation, the respective grid post is marked as ‘nodata’.
--feature:
  * sigma: sigma z of grid post interpolation adjustment
  * density: point density estimate (moving average/planes only)
  * excentricity: distance between grid point - center of gravity of data points (epsilon in figure right)
  * slope: steepest slope in %
  * exposition: slope aspect = azimuth of steepest slope line
--gridSize: grid width of output
--outFile: (optional)

```
e.g.
opalsGrid -inFile L:\TOM_Ue\part1\group1\G105ALL.ODM -gridSize 1 -feature sigma -feature excentricity
         -interpolation movingPlane -searchRadius 2.1 -neighbours 9
-> Creates files G105ALL.tif, G105ALL_sigma.tif and G105ALL_excen.tif
```
opalsDiff

program to create the difference between two digital elevation models as:
\texttt{Inputfile1 \textasciitilde minus Inputfile2}

Important parameters:
--\texttt{inFile: Inputfile1,Inputfile2}
--\texttt{outFile: (optional)}

e.g.
\texttt{opalsDiff -inFile G105ALL.tif,G106ALL.tif}

→ Creates file \texttt{diff_G105ALL_G106ALL.tif}
Color Coding of Strip differences

Unmasked strip difference: with vegetation

Masked strip difference: without vegetation
Exercise Data – Schönbrunn 2004

- Location: Schönbrunn, Vienna
- Acquisition date: 30.08.2004
- Scanner: Riegl LMS-Q560 Fullwave Scanner
- Flight lines: 11 strips, 1 Punkt/m², strip overlap >60%,

2 Folders:
- SB2004.R0 - raw data
- SB2004 - fine georef
4 strips are selected
Workflow: Hands-on opals

View data, e.g. By SCOP.GVE

Processing steps:

- Import
  - opalsImport
- Pointdensity
  - opalsCell, opalsZzcolor
- DSM
  - opalsGrid, opalsZcolor, opalsShade
- Mask
  - opalsAlgebra
- Difference model
  - opalsDiff
- Repeat the processing steps with the fine-georeferenced data (see SB2004)

see → run_all.bat
Results: Point density
Result: Strip differences