Open Source Point Cloud Processing
Software APIs and tools for manipulating ASPRS LAS data

- Started in 2007, first release in 2008
  - 20+ releases since 2007
- BSD license
OPEN SOURCE

• Community-driven
• Public source repository
• Public bug tracker
• Public mailing list
• Liberal license
SCOPE

• Feature-rich
• Shoulders (GDAL, LASzip, Boost)
• Multi-platform (Win32/64, OS X, Linux)
• Multi-language (C++ - C/Python/.NET)
AUDIENCE

• Software developers
• System integrators
  • Processing pipelines - horizontal scaling
SOFTWARE

- Cadcorp SIS Desktop
- Myriax Eonfusion
- LASERDATA LIS
- LizardTech LiDAR Compressor
- SAGA GIS
- ERDAS LPS/eATE
- Safe FME
- TopoDOT Point Cloud Processing Tool Suite

http://trac.liblas.org/wiki/WhoUsesLibLAS
PROJECT ACTIVITY

- 11 commiters
- 119 mail list subscribers
- 4-12 IRC members #liblas - irc.freenode.net
- 60 bug tracking members
- 200+ bugs filed
SPONSORSHIP

- Iowa Department of Natural Resources
- LizardTech
- US Army Corps Cold Regions Research and Engineering Laboratory
FEATURES

• Filtering
• Transformation
• Reprojection (GDAL)
• Indexing and Tiling
• Compression (LASzip)
FORMAT SUPPORT

• ASPRS LAS 1.0, 1.1, 1.2, 1.3 (points only)

• LASzip - compressed LAS (read/write)

• TerraSolid .bin (read)

• Oracle Point Cloud (write)

• ASCII (read/write)
FILTERING
FILTERING

$ las2las --input in.las \
    --output out.las \
    --drop-intensity ">=1000" \
    --keep-scan-angle "<=15" \
    --keep-classes 2
COLOR ASSIGNMENT

$ las2las -i input.las \ 
   --color-source image.img \ 
   --output output.las \ 
   --file-format 1.2 \ 
   --point-format 3 \ 
   --color-source-scale 256 \ 
   --color-source-bands 3 1 2
REPROJECTION

$ las2las --input epsg26915.las \
--output wgs84.las \
--scale 0.000001 \ 0.000001 \ 0.01 \
--a_srs EPSG:26915 \ 
--t_srs EPSG:4326
VERTICAL TRANSFORM

$ las2las --input srs.las \n  --t_srs EPSG:26915+5703 \n  --output navd88.las
ADDING VLRs

$ las2las --input in.las \
--output out.las \
--add-vlr CUSTOM_VLR \
42 "A VLR description"\"myfile.vlr"
>>> from liblas import file
>>> f = file.File('file.las', mode='r')
>>> for p in f:
...     print 'X, Y, Z: ', p.x, p.y, p.z
C++

```cpp
ifstream ifs;
ifs.open("input.las", ios::in | ios::binary);

liblas::Reader reader(ifs);

liblas::Header const& header = reader.GetHeader();

while (reader.ReadNextPoint())
{
    liblas::Point const& p = reader.GetPoint();

    cout << p.GetX() << ", " << p.GetY() << ", " 
        << p.GetZ() << "\n";
}
```
CHIPPER

$ lasblock in.las --capacity 10000
INDEXING

- Octree with optional z-binning
- VLR serialization
- Iterator-style access
- Frustum queries in the future?
• Arithmetic encoder by Martin Isenburg
• LGPL license - http://laszip.org
• Standard LAS header with a VLR
• 4:1 - 16:1
COMPRESSION

• Intended for wire and archival purposes

• Currently sequential in nature

• Header is still LAS -- uses a VLR to inform the layout and compression options
COMPRESSION

- Transparently enabled by libLAS
- Windows binaries available from OSGeo4W
THE ULTIMATE POINT
CLOUD FORMAT
THE ULTIMATE POINT
CLOUD FORMAT

...WILL NEVER EXIST
LIBPC

- Feature creep of libLAS
- Variable schema
- Format drivers
- Impedance mismatch
GDAL

- Format drivers
- Strive for performance, aim for flexibility
- 125 raster formats (5/3/2011)
- Industry-wide use
COMMON GDAL ABSTRACTIONS

- Pixel
- Band
- Block/Stripe

- Dataset
- Metadata
- Coordinate Reference
POINT CLOUD COMMONALITY

- Irregularly-spaced points
- $X, Y, Z$ or $r, \theta, \phi$
- Spatially blocked or scan-line oriented storage
POINT CLOUD COMMONALITY

- Schema
- Coordinate reference
- Metadata
- Block/Chunk
STAGES

- Schema
- Bounds
- Coordinate System
- Metadata
- Iteration
PIPELINE

Oracle Writer
PIPELINE

LAS Reader → Crop ← Rescale → Oracle Writer
Pipeline

- LAS Reader
- Crop
- Rescale
- Schema
- Oracle Writer
- Iter
DRIVERS

• LAS/LAZ (fixed schema)
• Oracle Point Cloud (generic XML-driven schema)
• BAG (fixed schema)
• LizardTech MG4 (read-only, fixed schema)
DRIVERS

• Bring out y’er formats!
COME HELP US :) 

- http://liblas.org/community.html
libLAS - LAS 1.0/1.1/1.2 ASPRS LiDAR data translation toolset

libLAS is a C/C++ library for reading and writing the very common LAS LiDAR format. The ASPRS LAS format is a sequential binary format used to store data from LiDAR sensors and by LiDAR processing software for data interchange and archival. See Features for more details on what libLAS can provide a LiDAR software developer. See Getting Started with libLAS for how to get started using the library.

libLAS’ initial development was supported in 2007-2008 by the IGSB of the Iowa DNR for use in its statewide LiDAR project. Ongoing support for libLAS is provided by a number of organizations including the U.S. Army Cold Regions Research and Engineering Laboratory.

libLAS builds upon by Martin isenburg and Jonathan Shewchuk of LLNL/UC Berkeley in their LASTools project to do a number of things. First, the libLAS focuses almost completely on providing an easy-to-program-with library for software developers wishing to implement the LAS specification in their own software. Second, libLAS exists to provide a truly open source library (see License for terms) – LASTools has no explicit licensing terms. Third, libLAS exists to provide advanced functionality and concentrate almost solely on the specification – not LiDAR data processing in general. libLAS a building block for developers to use to implement their own LiDAR data processing when working with ASPRS LAS data.

For more information, this document provides a comparison and description of the relationship of libLAS to LASTools.

See also: http://trec.liblas.org contains the previous incarnation of the libLAS website. If you can’t find something here, it should still be there.