

Current Topics in Open Source Lidar

Michael P. Gerlek
Flaxen Geo Consulting
mpg@flaxen.com

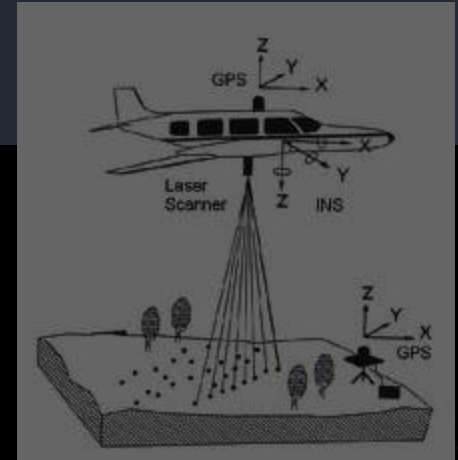
13 June 2011

Agenda

- What is lidar?
- The role of open source
- Four libraries to know about
- .NET interoperability

Lidar is...

- Laser beams!
 - X,Y,Z position
 - plus multiple returns, intensities, ...
 - “point cloud” concept
- Platforms
 - aerial, fixed, mobile
- What’s it good for
 - terrain modeling (DEMs)
 - building extraction
 - power line corridor mapping
 - ...



Lidar is **not**...

- Not vector features
 - don't use shapefile POINT objects
- Not raster / pixels
 - lidar is X,Y,Z (not just X,Y)
 - plus other attributes
 - not on a regular grid
 - sometimes you don't want the interpolated DEM

(data sets aren't small, either: *bazillions* of points)

The Role of Open Source

- Addresses the interoperability problem
 - too many file formats
 - and some questionable implementations
 - role model: GDAL
- Helps commercial vendors
 - library support often too low-level
 - gives a level playing field
 - allows for focus on core competencies

Some lidar libraries

- libLAS
- PDAL
- laszip
- libE57

The libLAS Project

- LAS standard
 - run by ASPRS
 - for aerial point lidar collection
- Library features
 - read/write support for the LAS standard
 - and some other formats (ASCII, Oracle, ...)
 - spatial indexing
- Command-line tools
 - format conversion
 - filtering, cropping, merging/splitting,

www.liblas.org
Howard Butler
BSD license

Status

- Very LAS-centric
 - core data structures make some assumptions
 - fixed set of fields in a point record
 - assumes Cartesian coordinates – no polar
- Performance
 - not what we need
- Version 1.6 released
 - stable, well-tested and well-used
 - used commercially
 - no significant new features planned
 - but maintenance expected indefinitely

Example

```
liblas::ReaderFactory f;  
liblas::Reader reader = f.CreateWithStream ifs);  
  
liblas::Header const& header = reader.GetHeader();  
int64 numPoints = header.GetPointRecordsCount();  
  
while (reader.ReadNextPoint())  
{  
    liblas::Point const& p = reader.GetPoint();  
    float x = p.GetX();  
    float y = p.GetY();  
    float z = p.GetZ();  
}
```

one point at a time

The PDAL Project

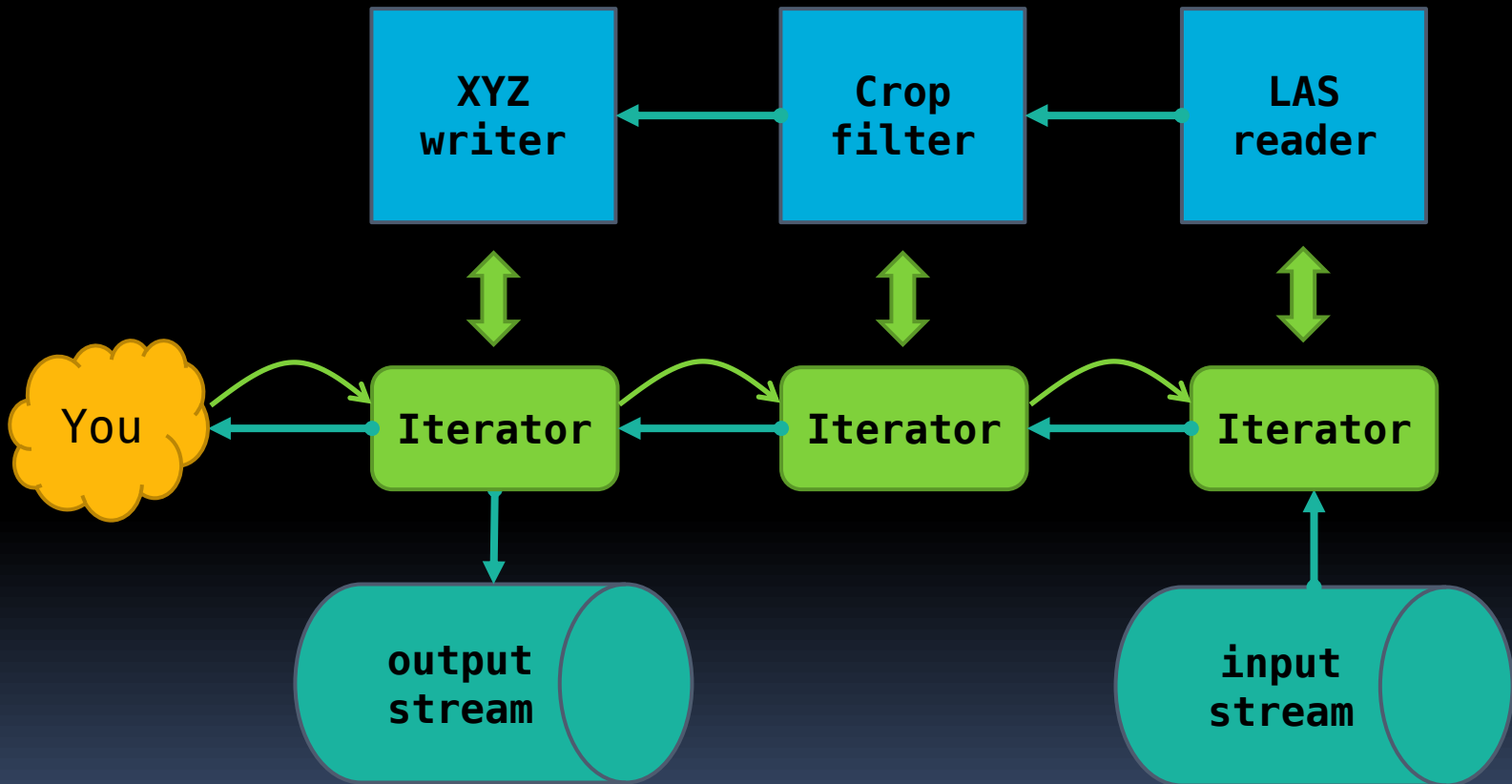
- “libLAS version 2”
 - many point cloud formats
 - not just LAS
 - many more workflows
 - pipelined design: readers, writers, filters
- No API compatibility
 - speed
 - extensibility
 - parallelism

www.pointcloud.org
Howard Butler, mpg
BSD license

Status

- Not at 1.0 yet
 - core data structures pretty stable
 - but still working on filters, tools, formats
 - performance looks good
- Beta plans
 - aiming for FOSS4G in September
 - *depending on funding, natch*

Architecture



Example

```
LasReader reader(filename);  
Schema& schema = reader.getSchema();  
SchemaLayout layout(schema);  
PointBuffer data(layout, 1024);  
SequentialIterator* iter =  
    reader.createSequentialIterator();  
uint32 numRead = iter->read(data);
```

customizable layout

parallel accessors

The laszip project

- LAS is not compressed
 - fixed size per point record
 - (fast/indexed access)
- compress the point record
 - maintain the basic LAS file format
 - “aware” of LAS point field semantics
 - lossless – 10-20% of original size
 - fast, too

www.laszip.org
Martin Isenburg
LGPL license

Status

- Used by libLAS and PDAL
- Stable
 - tested, commercially used
- New features underway
 - support for random seeks
 - fast access, for viewing

The libE57 Project

www.libe57.org
ASTM
BSD license

- E57: new format
 - v1.0 issued earlier this year
 - addresses shortcomings of LAS
 - designed for more general/arbitrary point clouds
 - polar coords, metadata, 2D images, ...
- libE57
 - reference implementation of the standard

Status

- Still in beta
- Two APIs
 - “foundation” and “simple”
- Will be supported as a PDAL format

Example

```
e57::Reader eReader("file.e57");

e57::CompressedVectorReader dataReader =
    eReader.SetupData3DPointsData(scanIndex,
        nSize,           // size of each of the buffers
        xData, yData, zData); // buffers with the x,y,z data

while ((size = dataReader.read()) > 0)
{
    for(unsigned long i = 0; i < size; i++)
    {
        Point p(xData[i],yData[i],zData[i]);
    }
}
```

.NET?

- These are all C/C++ libraries
- But with .NET bindings:
 - libLAS: stable
 - PDAL: in development
 - libE57: possible future work
 - laszip: not planned *(likely not needed)*

SWIG is the key

SWIG?

- C/C++ tribe dominates open source geo
 - .NET support is always an add-on
- SWIG tool
 - generates C# stubs from C/C++ headers
 - very widely used for generating Python bindings
 - *The easy stuff is easy, and the hard stuff is hard.*

SWIG Rules

Be swig-friendly

- avoid heavy /complex template usage
- avoid multiple inheritance
- you don't need the entire API
- automated builds & tests
 - *to catch "drift" in the C/C++ APIs*

Summary

- Lidar is a new, 3rd kind of data
- Open source community providing needed infrastructure
 - formats, compression, speed, interoperability
- .NET follows along behind
 - usually demand- (or ability-) driven
- Users and Contributors always welcome!

Thank you.

Questions?

Have you hugged an open source developer today?