To appear in the *International Journal of Remote Sensing*
Vol. 00, No. 00, Month 20XX, 1–16

REVIEW ARTICLE

Airborne Laser Scanning Data Storage and Indexing: State of the Art Review

ANH-VU VO[†], DEBRA F. LAEFER[†,‡,*] and MICHELA BERTOLOTTO[§]

[†] *Urban Modelling Group, School of Civil Engineering, University College Dublin, Ireland*;
[‡] *Earth Institute, University College Dublin, Ireland*;
[§] *School of Computer Science, University College Dublin, Ireland*

(*01 March 2016*)

While the rapid growth in the availability and quality of airborne laser scanning data offers unprecedented information, it challenges the existing data management and dissemination solutions. Data management has become a bottleneck to effective laser scanning data exploration. This paper documents the state-of-the-art techniques for aerial laser scanning data storage and indexing during the post-processing stage. A brief history of laser scanning technology is presented with the intention to offer an overview about the evolution of the technology from the data point of view inclusive of full waveform data management. The key strategies for data handling, including data modelling and indexing are extensively discussed.

Keywords: LiDAR; airborne laser scanning; spatial indexing; spatial database; spatial data storage

1. The genesis and advancement of aerial LiDAR or laser scanning technology

The acronym LiDAR (Light Detection and Ranging) was coined by Middleton and Spilhaus (1953) to indicate a technique that used light pulses generated from electric sparks to measure distances to the cloud ceiling. Seven years later the first successful laser was built by Maiman (1960). Despite its origin, the term LiDAR today implies a surveying technique that uses a laser to actively sample the surrounding visible surfaces in various directions. Laser scanners often combine range and angular measurements to deliver geometric representations of the measured surfaces in the format of three-dimensional (3D) discrete sampling points in a Euclidean coordinate system, often called point clouds. When laser scanning is conducted from a moving platform such as an aircraft or a motor vehicle, Global Positioning System navigation devices (GPS) and Inertia Measurement Units (IMU) are required to position the platform. This enables point cloud geo-referencing and is particularly common for airborne/aerial laser scanning (ALS).

As early as 1965, airborne laser rangers were used to measure the earth surface elevation (Shepherd 1965). The laser ranging principles at that time have remained

*Corresponding Author: School of Civil Engineering; U3D Printing Hub; Earth Institute; University College Dublin, Newstead, Belfield, Dublin 4, Ireland, Tel: +353 1 716 3226

Email addresses: anh-vu.vo@ucdconnect.ie (Anh-Vu Vo), debra.laefer@ucd.ie (Debra F. Laefer), michela.bertolotto@ucd.ie (Michela Bertolotto)

essentially unchanged today. Ranges are measured based on either time-of-flight of spontaneous light pulses or phase differences of continuous modulated optical waves. In the 1960s measurement was only possible along a single line per flight track (i.e. the laser profiler), as opposed to capturing an entire swath as is true with today's ALS. Notably, the maturity of the scanning mechanism, GPS, and IMU technologies did not enable the first commercial airborne scanner until 1993 (i.e. Optech's ALTM-1020). Since the mid-1990s, the technology has gained rapid advancement and become commercially widespread (Flood 2001) (Figure 1).

Since its genesis in 1960s, there have been progressive improvements in accuracy, as well as the introduction of waveform digitization (1976) and Multiple Point in Air technology (2007) and the rapid improvement of pulse repetition frequency. Conventionally, raw ALS data are processed post-flight by the ALS acquisition companies. The process of extracting discrete points from electronic signals to form the point clouds is proprietary and inaccessible to the data users. As an alternative, full waveform (FWF) digitization provides the entire backscatters in the form of a time series that can be recorded and delivered to the end users. This gives users greater control over the data origin and richer information about the scanned scenes. The idea of retaining the entire return signal dates back to 1976, when Mamon et al. (1978) integrated a waveform digitizer atop a laser profiler to capture forest foliage structures. In 1999, the first operational full waveform ALS system (Laser Vegetation Imaging Sensor) was built by NASA (Bretar and Chauve 2008). The first commercial FWF system was introduced by Riegl in 2004: the LiteMapper-5600 (Hug, Ullrich, and Grimm 2004). FWF LiDAR was originally used for forestry to improve the capture of complex canopy structures. More recently, selective efforts have been made to employ FWF laser scanning in urban settings for automatic land use categorization (Mallet et al. 2011) and vertical object detection near airports (Parrish 2007). From the data point of view, FWF digitization offers pulse and wave data types in addition to the points, as in the case of discrete point acquisition (Bunting et al. 2013). A pulse is often represented by a ray or a line segment indicating the location and orientation of the emitted/returned laser beam. Other non-geometric data such as wavelength and GPS time may complement the geometry. A waveform is a time series of either the outgoing or incoming laser signal. Points, pulses, and waveforms of a scan are all interrelated since they represent different views of the same measurement. Thus, there is a demand for integrating them within a data management system.

The next milestone was in 2007 with the launch of commercial Multiple Pulse in Air (MPiA) systems. Traditional laser scanners can handle only one pulse at a time. Thus, the pulse must return before the next can be emitted. This constrains the pulse emission speed by the laser pulse travel time, which is proportional to the flying attitude. MPiA circumvents the constraint by allowing emission of a second pulse prior to receipt of the previous one. Hence MPiA technology can shorten acquisition time, achieve higher density at a given flight height, or enable an increased flight attitude to avoid air turbulence, while retaining the sampling density (Roth and Thompson 2008).

Another important ALS development has been the rapid increase in pulse repetition frequency (PRF). Figure 2 portrays changes in ALS PRF through 2014, which is no longer linear. To the authors' best knowledge, the record of 1 MHz set by the Leica Dragon Eye dual head scanner remains the fasted PRF. Notably the Leica Dragon Eye combines two scanning heads to double its PRF. Besides the increase in PRF, there has been a continuous improvement in measurement accuracies. For example, the vertical accuracy of Optech scanners increased from

H/1000 (ALTM-1020) in 1993 to H/5500 (ALTM-3100EA Gemini) in 2006 (Shan and Toth 2008) to H/16,666 in 2010 (Leica 2010); where H is the flying attitude.

In addition to the above-mentioned advances, other ALS developments include various improved scanning mechanisms, multispectral LiDAR, and Flash LiDAR (i.e. Geiger mode). An interesting example of the improved scanning mechanisms is the oblique scanning capacities of the Leica AHAB Dragon Eye, which shoots each target at several angles of incidence ranging from 8° to 20° to increase data coverage on vertical surfaces and to alleviate shadows. Multispectral LiDAR, as suggested by its name, scans objects using optical waves of multiple wavelengths. Multi-wavelength LiDAR was investigated as early as 2002 (Songxin and Narayanan 2002), with the initial motivation to capture distinctively varying surface types (e.g. sea beds versus dry surfaces). Specifically, a near infrared laser is suitable for most ground-based objects, while green light is necessary to penetrate water surfaces. More recent systems such as the Optech's Titan can operate at more than 2 wavelengths (i.e. 532, 1064 and 1550 nm). Multiple wavelength LiDAR also improves the likelihood of capturing more information about scanned objects [e.g. gaps in tree canopies or foliage reflectance characteristics for improved species classification (Hovi 2015)]. Another notable development is the invention of flash LiDAR, which substantially extended the possibilities in airborne 3D laser mapping technology by widening simultaneous data capture opportunities (Albota et al. 2002). Namely, traditional aerial laser scanners orient a spot-like laser pulse using a mechanical scanning mechanism such as a rotating mirror or prism to capture the cross swath and rely on the platform movement to provide coverage along the flying direction. In contrast, a flash LiDAR sensor transmits and receives an array of thousands of pixels with every pulse, which captures the range variation within a large extent, at a better speed, while avoiding mechanical scanning problems (Albota et al. 2002; Marino and Davis 2005).

Two notable trends are recognized by observing these advances in ALS data acquisition from a data management point of view. First is the rapid increase in point cloud size as a function of acquisition capabilities. MPiA, FWF, and multi-wavelength LiDAR data, as well as the ever-increasing PRF, have all contributed to the amount of data that can be produced per square metre and per minute of flight time. Notably demands for ALS data storage and management have also been driven by the increasing availability and usage of the data, which in turn has further driven the demand for denser data sets (see Section 2). The second trend is the emergence of new types of data and the demand to integrate disparate data types. As in the case of FWF ALS, pulses (i.e. linear segments in 3D space) and waveforms (i.e. time series) need to be stored and integrated with conventional point cloud data. Similarly, there is a demand for connecting data scanned at different wavelengths in the case of multi-wavelength LiDAR.

2. Growth in scale and resolution of ALS data

The increasing popularity of LiDAR, particularly for large-scale projects, has also been making more data management demands. For example at least six countries have already completed country-wide ALS projects (i.e. Denmark, England, Finland, the Netherlands, Slovenia, and Switzerland), with many more in progress (e.g. Italy, Poland, Spain, and Sweden) and countries like Japan nearly complete (GSI 2016).

One of the pioneers in national ALS surveying is the Netherlands, where 50%

of the land lies below sea level. In 1996, for the purpose of dyke monitoring, the Netherlands launched its first country-wide airborne laser scanning campaign named AHN1 Actueel Hoogtebestand Nederland. This was completed in 2003, with most of the country mapped at densities of 0.06-1 points/m^2 (Swar 2010). In the period 2007-2012, the second national scan (AHN2) was completed, with an increased point density of 6-10 points/m^2 resulting in 640 billion laser points. The third Netherlands-wide scan, AHN3, began in 2015 and is scheduled to be completed in 2019 (on a repeating 6 year cycle). Similar repeated surveys are also being done in Austria and Northern Italy (Rieg et al. 2014).

In the United States, while a national ALS initiative is still in discussion, many states have completed coverage (e.g. Connecticut, Indiana, Iowa, Louisiana, North Carolina, Ohio and Pennsylvania) for purposes including flood plain mapping and seismic fault studies. Extensive historical datasets throughout the US are maintained at three national centres: (1) the San Diego Super Computer Center (SDSC) at the University of California San Diego (Krishnan et al. 2011), (2) the United States Geological Survey (Stoker et al. 2006), and (3) the National Oceanic and Atmospheric Administration (NOAA Office for Coastal Management 2015). While there is no published report about the total amount of American ALS data, the volume maintained at SDSC was 46 billion points in 2011 (Krishnan et al. 2011) and has already grown to 835 billion points covering 84,286 km^2 at the time of writing. The increasing data density is shown in Table 1, with 70 pts/m^2 for a single flight strip and 335 pts/m^2 for multiple strips. Flying multiple times over a short time span is becoming common practice in urban areas to achieve comprehensive coverage of built environments through extensive flight strip overlap and multiple flight strip orientations (Hinks, Carr, and Laefer 2009).

The explosive growth in ALS data challenges many existing data management solutions and has been attracting significant research efforts(e.g. Mosa et al. 2012; Elseberg, Borrmann, and Nüchter 2013; Ramsey 2013; van Oosterom et al. 2015; Martinez-rubi et al. 2015). In the following sections, the state-of-the-art of data handling techniques to tackle these challenges is presented. The review is separated into two parts, each for one of the key components of an ALS data management system: data modelling and data indexing. It subsequently is a discussion on managing full waveform ALS data.

3. ALS data modelling

An ALS dataset often exists in different formats during its life cycle (Shan and Toth 2008). During the pre-processing stage, where raw sensor data are registered, geocoded, and corrected, ALS data are often treated as a set of files, each of them containing a group of spatially and/or temporally coherent points. Hosting laser points in a data management system (DBMS) at this stage has no clear benefits and only serves to decelerate many data manipulation operations. However, in the post-processing and data distribution stages, a DBMS for laser point clouds can offer significant advantages as outlined by Shan and Toth (2008); Schön, Laefer Debra, and Bertolotto (2009); van Oosterom et al. (2015). When being managed within a DBMS, laser scanning points can be accessed via standardized, declarative languages (e.g. SQL) without knowledge about the data's internal structure. Multiple programs can access the same dataset without requiring excessive data transformation. In addition, there is a built-in transaction management capability, as well as many functions supporting I/O optimization (e.g. caching), data query-

ing, and administration. Thus, a database solution can be ideal for distributing large LiDAR datasets in a network environment. During the post-processing stage, additional attributes are often computed for the points (e.g. normals), and derived models (e.g. digital surface, digital terrain, finite element) can be generated, which may need to be kept alongside the original points. Furthermore, when the data integration is needed, using a DBMS is often preferable (van Oosterom et al. 2015). The remaining portion of the paper focuses on ALS data in the post-processing stage, where the data are ready for visualisation or for further processing to extract application specific information.

3.1. *File-based environments*

Within a file-based environment, there are three common methods for encoding a laser scanning point cloud: plain text, marked up language, and binary. While plain text (e.g. PTX format by Leica, PCD format by PointCloudLibrary) is the simplest, human readable, and flexible (in terms of defining point attributes), it is the least efficient in terms of storage. Namely, one ASCII character (i.e. 8 bits) is needed for every digit representing the point coordinates and attributes. A point record is usually stored as a line in a file where point coordinates and attributes are separated by a special character such as a space or a comma. Metadata (e.g. coordinate system, number of points, label of point attributes) are sometimes recorded at the beginning of a file. Depending on the number of digits needed for each point record, a plain text file of a point cloud can be significantly larger than a binary file storing the same information. Thus, reading and writing point clouds in a text format requires more input/output (I/O) than its binary counterpart.

When a point cloud is stored in a binary format, the number of bits allocated for each attribute can be tailored to fit its respective requirements. Thus, the usage of binary resources can be minimized. Reading a binary file is more complex and requires an agreement between encoders and decoders usually via a file format specification. The most widely used file format for exchanging ALS data is the LAS format introduced by the American Society of Photogrammetry and Remote Sensing (ASPRS 2011). This file format offers a number of pre-defined, fixed structures for representing point records from which the user can select. For example, a typical laser point with a GPS time stamp can be represented using LAS's record format 1. Storage of this takes 28 bytes per point based on the point's 10 attributes and coordinates. In a LAS file, metadata can be stored in the file header, in variable length records at the beginning of a file, or in extended variable length records at the file's end.

One advantageous feature in the point coordinate encoding in a LAS file is the use of scaled integers (i.e. 4 bytes per number), instead of the straightforward double precision floating point arrangement (i.e. 8 bytes per number). In addition to reducing the required binary space, this approach ensures uniform numeric accuracy throughout each numeric dimension, which well suits the characteristics of ALS data (Isenburg 2013). Another efficient method for compacting the data without loss of precision is to replace the points' coordinates by their order in a spatial filling curve such as a Hilbert curve (Elseberg, Borrmann, and Nüchter 2013). This method is feasible when the spatial extent of the cloud is not too large when compared to the required resolution. While this approach can reduce the storage requirements, the method incurs extra computational costs for converting the coordinates and the spatial codes.

An alternative approach for point cloud storage is the marked up language [E57

format (Huber 2011) defined within ASTM E2807 (ASTM Standard Committee 2011)]. That format is capable of storing general 3D measurements alongside 2D imagery data. At the top level, an E75 file is structured into a hierarchy in an XML format. At the lowest level within the hierarchy, the main portions of the data (i.e. the 3D point clouds) are stored in a binary format, which can be logically partitioned into lines or blocks. Notably the E57 format offers flexible definition of point attributes, while still keeping a minimal number of bits dedicated for the point data attributes. Nevertheless, similar to the LAS format, the E57 format was designed to store sensor information and not intended to be used as a working format. Thus, affiliation of the derived attributes into point data is not straightforward. Furthermore, storing models derived from LiDAR point clouds is not natively supported.

When LiDAR data are hosted as a collection of files, they are usually organized into rectangular tiles. Data are often disseminated to users by tile selection. In this way, if a user is interested in a region crossing multiple tiles, they need to download all the tiles, merge the data and crop the segment of interest on their local computer. Even though the approach leaves the tedious work to data user, it seems to be the most common way of managing and disseminating ALS data as seen in USGS's CLICK (Stoker et al. 2006), the national ALS data dissemination services of Denmark, Finland, Poland, and Switzerland, just to name a few. An alternative to the pure tile-based approach is area-of-interest (AOI) selection. An example of successful AOI implementation on a file-based system is OpenTopography (Krishnan et al. 2011; Nandigam, Baru, and Crosby 2010). A dedicated server at the back end of the system is used for hosting all LiDAR data in LAS format where metadata, including the spatial extents of each of the files, are handled within an IBM DB2 database. A similar approach was utilized for the unofficial AHN2 data dissemination platform called MATAHN by Peters and Ledoux (2014). The system stores the point data as a collection of LAS files and uses Postgres/PostGIS to manage the spatial extents of the files. Notably, the LAS file manipulation in OpenTopography is facilitated by libLAS (Butler et al. 2016), while Lastools (Rapidlasso 2014) is used to support MATAHN.

3.2. *Database environments*

ALS data can be managed using a spatial database management system (SDMS) as an alternative to the above file-based approach. The major SDMSs supporting point cloud hosting include Oracle and PostgreSQL/PostGIS. MonetDB (Martinezrubi et al. 2015) is also currently being developed to handle point clouds. However that database's support for laser scanning data is currently restricted to rectilinear range searches and, therefore, is not discussed further herein.

The two most common approaches for hosting LiDAR data within Oracle and PostgreSQL DBMSs are flat tables and blocks. With the flat table method, one tuple is allocated for each point record including the point's attributes. A point cloud is logically represented in the form of a single table, which may be partitioned into multiple physical segments. This approach is simple and flexible since inserting/removing point records and modifying point attributes are both straightforward activities. The main disadvantage, however, is that this approach cannot cope well with large datasets due to its poor scalability on ordinary computers (van Oosterom et al. 2015).

In contrast, both Oracle and PostgreSQL [with Ramsey's extension (Ramsey 2013)] offer a blocking solution as the native support for point clouds with the

names of SDO_PC and PCPATCH, respectively. The two implementations are relatively similar as they both partition a given point dataset into spatially consistent blocks. Each block is considered as a unit to be stored in a tuple of the database. Since there are significantly fewer items to be handled when compared to the case of storing one point per tuple, the approach copes better with larger datasets (Ramsey 2013; van Oosterom et al. 2015). In Oracle's SDO_PC data type, a point block, stored as a binary large object (BLOB), is composed of a various number of fixed size point records. A point record consists of the point's id, its coordinates, and a maximum of 8 numeric attributes each of which occupies 8 bytes. This encoding approach is generic and requires use of more binary bits. A point record having 10 attributes [such as the LAS's point record format 1 (ASPRS 2011)] would consume 112 bytes (i.e. 4 times the LAS's consumption), if represented with this approach. Unlike Oracle's fixed-size field approach, PCPATCH by Ramsey (2013), which allocates a minimum number of bits for each attribute, is similar to the LAS format approach. In PCPATCH, the structure of a point record is stored as an XML document. The concept of scale and offset are similar to what are used in the LAS file format and are also utilized to reduce the required byte space and achieve uniform accuracy over the numeric dimensions.

The third approach, for storing point cloud in a DBMS, is to use the DBMSs' point geometry type (i.e. SDO_GEOMETRY with GTYPE of 3001 in Oracle, Point type in PostGIS). While this solution is theoretically feasible, and built-in spatial indices can be used directly, it is being used to a lesser extent due to its poor performance (Mosa et al. 2012; Ramsey 2013). A similar finding was reported by the OpenTopography team (Nandigam, Baru, and Crosby 2010), while investigating the possibility of using IBM DB2 Spatial Extender to manage ALS data. That study revealed that storing one spatial point per tuple led to excessive disk consumption (i.e. 420 bytes/point, approximately 4 times the flat table approach). To the authors' knowledge, the only exception that reported a successful use of this approach for datasets of billions of points is that by Lewis, Mc Elhinney, and McCarthy (2012).

Having reviewed various methods for encoding laser scanning point data records in both the file-based and the database environments, the following can be considered as the state-of-the-art techniques for laser point data representation in both environments:

- Use of the exact number of bits for each point attribute;
- Storage of point coordinates as scaled integers to reduce byte space and ensure uniform accuracy;
- Modelling a point cloud as a collection of small, spatially coherent blocks of points, instead of the manipulation of each individual point;
- Use of an XML document to store the description of the structure of the point record so that the data can have a flexible structure, while still being self-explanatory;
- Use of XML to represent the hierarchy of different datasets within a file.

4. Data indexing

In addition to storing point data, a working LiDAR data management system should provide rapid access to portions of the dataset satisfying certain constraints, which may contain spatial elements and/or non-spatial attributes (e.g. classification, return number, and intensity). Data retrieval is usually facilitated via in-

dexing structures that aim to accelerate access by reduced data scanning. Spatial indexes are often used to support spatial queries, while non-spatial queries require non-spatial indexes. A dataset can be simultaneously indexed using multiple structures. The following review focuses mainly on spatial indexing of point clouds. Major spatial indexing approaches include grid-based indexing, tree like indexing, and combinations of multiple indexes.

4.1. *Grid-based (tile-based) indexing*

Traditionally, ALS point data has been spatially partitioned into multiple 2D rectangular tiles of manageable size. This is the simplest spatial index of a point cloud and is referred to as regular, grid-based indexing. The approach is commonly used in both file-based and database environments (Rieg et al. 2014). While the method is straightforward to implement and can be adequate for simple operations (e.g. point-in-2D-polygon query), efficiency is low due to brute-force scanning. Furthermore, the splitting may introduce discontinuities at tiles boundary, and is problematic due to working across multiple tiles (e.g. clip a segment spanning more than a tile, search for neighbour points located in other tiles). Finally, more sophisticated functionalities such as neighbour searching are not supported.

4.2. *Tree like indexing*

More advanced indexing strategies are often implemented to support laser scanning data access. They commonly include the r-tree (Guttman 1984), the quadtree (Finkel and Bentley 1974) and its 3D sibling the octree, and the kd-tree (Bentley 1975). Each index type has multiple variations. The r*-tree (Beckmann et al. 1990) and Hilbert r-tree (Kamel and Faloutsos 1994) are two examples of dozens of different existing implementations based on r-tree indexing. MX quadtree, PR quadtree, and bucket PR quadtree are amongst variants of the quadtree indexing type (Samet 2006). The bucket PR kd-tree (Orenstein 1982) can be seen as a variant of the kd-tree. While the indexing types vary in implementation, their underlying principles are highly similar. The indexing mechanisms aim to create spatially organized structures of the data using spatial hierarchical partitioning. A data searching algorithm should browse as few branches of the hierarchy as possible to obtain the addresses of the data objects to be returned. All of the mentioned structures are capable of facilitating the two most common spatial queries on point clouds: polygon clipping and nearest neighbour searching. Their respective efficiency mostly depends on the number of nodes in a tree that need to be visited during searching and the required computations to determine the spatial relations between geometries. The searching algorithm, data distribution, index construction parameter (e.g. bucket capacity, splitting condition), and searching conditions are amongst many factors influencing the efficiency of a given indexing structure. This makes comparison of different indexing structures a complicated task, which is beyond the scope of this paper but has been considered selectively by others (e.g. Kothuri, Ravada, and Abugov 2002; Mosa et al. 2012).

The r-tree indexing is an extension of the widely used, standard b-tree (i.e. balanced tree) (Bayer and McCreight 1972) in a higher dimensional space. The r-tree is well known for its balanced characteristic (i.e. all leaves are at the same depth). An r-tree is adaptable for non-uniform data and can be constructed without a known bounding box. Its two major disadvantages are the high level of node overlapping and its relatively expensive construction cost. At regions where multiple

nodes are overlapped, a query resolver must visit all the nodes. Thus, processing a query can be slow. The r*-tree indexing by Beckmann et al. (1990) improved the original r-trees by Guttman (1984) by introducing a more sophisticated [and more expensive] strategy for building the tree to reduce node overlapping. Leaf nodes in an r*-tree are disjoint but overlapping may occur at the higher level in the hierarchy. Thus, the issue is partially solved. Both Oracle Spatial and PostgreSQL have an r*-tree as the built-in indexing scheme.

The quadtree index originated from the generalization of the famous binary tree from 1D problems to 2D spaces in order to solve data queries involving 2 composite keys (Finkel and Bentley 1974). When being used to index point cloud data, a quadtree is usually defined as a tree structure created by recursively tessellating the data based on the data points (i.e. tree-based quadtree) or their embedded space (i.e. trie-based quadtree) into 4 axis-aligned quadrants. The octree indexing is the natural extension of the quadtree into a 3D space, which iteratively splits data into 8 octants. The use of quadtrees and octrees in LiDAR related data management systems is widespread and appears in Lastools the LiDAR toolkit managing and manipulating data in LAS format by Rapidlasso (2014) and extensions to Oracle DBMS (e.g. Mosa et al. 2012; Hoefsloot 2012). While not implemented within a data management system, quad/octrees are also frequently selected as the accessing structure in various research (e.g. Girardeau-Montaut et al. 2005; Wenzel et al. 2011; Elseberg, Borrmann, and Nüchter 2013; Richter, Discher, and Jurgen 2015). The quad/octrees are extensively used in many visualization applications, as the trees themselves are well-shaped structures representing multiple levels of detail (Yang and Huang 2014; Richter and Döllner 2014).

Within the Oracle DBMS, there have been a number of attempts to use a quad/octree as a spatial index. Prior to version 10g, Oracle supported two spatial indexing types for its queries on geometric objects: r-tree and quadtree. However, from version 10g onward the native quadtree indexing was deprecated due to a belief that it (1) demonstrated poorer performance, (2) was more difficult to tune, and (3) offered less support for neighbour search than the r-tree (Oracle 2003). Similar results were reported by Kothuri, Ravada, and Abugov (2002). Notably these comparisons were drawn from tests on 2D polygonal data, but not large point datasets like those encountered in LiDAR data. While investigating point cloud indexing within Oracle Spatial, Mosa et al. (2012) implemented an octree indexing (i.e. 3D sibling of quadtrees) into the Oracle DBMS via the Oracle's extensible framework to facilitate 3D clipping operations. The implemented octree index significantly outperformed Oracle's r-tree implementation for point data stored as SDO_GEOMETRY (GTYPE=3001), with one point per tuple. Mosa et al. (2012) suggested that the octree's superior performance attributed to its spatial disjoint arrangement of the nodes in contrast to the r-tree node overlaps; further investigation of this is presented in Schön et al. (2013). Despite Oracle's deprecation of the quadtree, the indexing structure appears in later studies on LiDAR hosting (Hoefsloot 2012; van Oosterom et al. 2015).

A third common index structure is thehe kd-tree, which is a special case of binary partitioning in multi-dimensional space (Bentley 1975). At each node, the spatial domain is split into two halves along one of the three coordinate dimensions (x, y, and z). Similar to the quad/octree indexing, kd-trees can be either tree-based or trie-based. The branching factor of the tree is fixed to 2 (i.e. binary). Therefore, kd-trees are often narrower and deeper than quadtrees and octrees, which have branching factors of 4 and 8 respectively. The performance characteristics of a kd-tree strongly relates to the point insertion order and can be easily unbalanced

depending on this order. Unlike the nodes in a PR quad/octree or r-tree, which represent the bounding box of a group of points, every node of a kd-tree is associated with one data point. Thus, an ordinary kd-tree can be larger than the point data itself. The bucket kd-tree (Samet 2006) improves the kd-tree indexing by allowing a "bucket" of points to be stored on a single node so that the index's size can be reduced. While rarely used as a permanent access structure for data storage (probably because of size requirements), kd-indexing is frequently used to facilitate neighbour searching [(Otepka, Mandlburger, and Karel 2012; Hua et al. 2008; Vo, Truong-hong, and Laefer 2015)]. In fact, most point cloud processing software has a kd-tree as a built-in indexing structure [e.g. PointCloudLibrary(Rusu and Cousins 2011), CloudCompare (Girardeau-Montaut et al. 2005)]. While all of the named indexing structures are able to support neighbour searches, the kd-tree is usually considered as the most efficient (Otepka, Mandlburger, and Karel 2012).

4.3. *Other accessing structures*

While tiling and tree-like structures are very common in LiDAR point data indexing, other accessing strategies exist. Examples include the multi-star structure by Ledoux (2010) and the hashing data structure by Han et al. (2012). Ledoux (2010) proposed that laser point collections such as those derived from laser scanning can be stored in a DBMS as a triangulated irregular network (TIN). Both geometry and topology of the TIN can be represented by explicitly storing the point coordinates together with a star structure for each point. A star structure of a point is, in fact, a set of IDs of vertices of the TIN directly connected to the given point. Even though the structure is redundant as triangles are stored more than once, it is capable of supporting spatial queries on point clouds by simply browsing the TIN. Another notable indexing strategy for point clouds is the hashing virtual grid by Han et al. (2012). In this approach, a point cloud is overlain by a virtual 3D grid structure (i.e. voxel space). As part of this, every cell in the grid maintains a list of pointers to the enclosed points, which is the gateway to the point data. Each cell is associated with a distinctive hashtag providing O(1) access to its pointer.

4.4. *Integrating multiple indexes*

Multiple indexing structures can be combined to facilitate point data. For example, within an in-house simplified relational DBMS, Otepka, Mandlburger, and Karel (2012) incorporated the global regular tiling (2D) and local kd-trees (3D) to support spatial queries including neighbour searches. The in-memory kd-trees were created on-the-fly for selective tiles relevant to the query being processed. Even though this method circumvents storing the excessive indexes on disk and avoids complex updates of the indexes during the point insertion, the indexes cannot be reused. Thus, processing spatial queries incurs overheads for re-creating the local indexes. In this approach, even though the kd-trees were separated (which may cause discontinuity problems), the issue was not explicitly discussed in the paper. Another example of integrating multiple indices was introduced by Vo, Truong-hong, and Laefer (2015) as part of an approach for road detection from ALS data. In that work, hybrid indexing implemented a Hilbert-coded tiling at a global level with kd-trees at a local level. The discontinuity issue was avoided by providing buffers around the tiles. Unlike (Otepka, Mandlburger, and Karel 2012), the kd-trees in Vo, Truong-hong, and Laefer (2015) were reusable by serialization and storage in a database. Another combination of a global octree and local kd-trees

was presented by Hua et al. (2008) where kd-trees were created at run time. That system is restricted to point rendering but does not support generic data demands. An example of a combination of a quadtree at the global level with local 3D r-trees or kd-trees was presented by Yang and Huang (2014) for managing and visualizing integrated aerial and terrestrial point clouds. Notably the r-trees at a local level were constructed based on a bottom-up grouping of facets fitting to the points within a quadrant. The outperformance of the hybrid tree over Oracle's r-tree with SDO_PC was documented therein.

5. Management of full waveform data

While commercial full waveform aerial laser scanners have been around for almost a decade (see Section 1), integrating FWF into a LiDAR management system still appears as a topic of future work in most of the literature (e.g. Krishnan et al. 2011; Otepka, Mandlburger, and Karel 2012; Rieg et al. 2014). The very few examples of efforts in this area include the development of the ASPRS's LAS format version 1.3 (ASPRS 2011), the PulseWaves file format currently being developed by Rapidlasso (Isenburg 2014), and the Sorted Pulse Data library (Bunting et al. 2013). All are file formats whose latest work cover the FWF data indexing issue. To the best of the authors' knowledge, there is no established database solution for storing and managing full-waveform LiDAR data.

In revision 1.3 of the LAS format specification, there are two options for storing full wave data together with the affiliated conventional discrete laser points. In the first, a data packet storing raw waveforms can be attached to the extended variable length record of a LAS file (ASPRS 2011). In the second, an auxiliary file can be used for storage of waveform data. With that, a point is linked to its waveform via a locator stored as a point attribute. While the waveform can be stored, the file format was not designed initially for the waveform concept and, thus, has certain limitations. For example, pulses are not explicitly represented, only a selected portion of the waveform is stored, and information about the outgoing waveform is absent. PulseWaves, currently being developed at Rapidlasso (Isenburg 2014), is an open source LAS compatible file format designed to address LAS's lack of full FWF support. PulseWaves stores the pulses and waves as separated files, which are supposed to complement point data stored in the LAS format.

Sorted Pulse Data (SPD) is another effort to host FWF data in a file-based environment (Bunting et al. 2013). The data format was built upon the generic HDF5 file format, which was originally created for storing Earth observation data of NASA. All three components of a FWF dataset [pulses, points and waveforms (outgoing and incoming)] are explicitly represented in the SPD format. Pulses and points are represented as dedicated data types, while waveforms are stored as arrays of integers. Pulses in SDP are spatially indexed using a grid-based mechanism where all pulses within a grid are allocated a continuous segment in the pulse list. The order of points and waveforms follows the pulses with which they are associated.

6. Concluding remarks

Aerial laser scanning technology development and rapid adoption are causing major challenges to existing data handling solutions. In addition, new technical advances in the field, including the full waveform and multi-wavelength data, are producing novel data types that may need integration with the conventional point data. The

increasing data burden, the diverse data types, and the demand of data integration are the current major challenges for the ALS data management. Between the challenges of data modelling and data indexing, data modelling is arguably more straightforward. Optimization of byte space decoding point data, aggregation of point data into chunks to reducing the number items to be processed, and use of XML to implement self-defined data all seem promising steps towards a more unified solution. In contrast, there is both a notable absence of a systematic study on the performance differences between indexing types and the factors controlling performance, along with no performance prediction tool. Finally, management of full waveform remains a generally unexplored topic. While there are a few file formats developed for hosting FWF data, other issues such as spatial indexing and data querying of FWF data have barely been investigated. Additionally, at the time of this writing, there is no known database system that supports FWF data.

Acknowledgments

The authors thank Mr. Seán William Morrish for his input on the list of national and federal LiDAR datasets, Dr. Hiroshi Masaharu for the informative discussion about ALS usage in Japan, and Mr. Christopher Crosby for up-to-date information about the laser data at San Diego Supercomputing Center.

Funding

This work was funded by European Research Council grant ERC-2012-StG 20111012 “RE-TURN - Rethinking Tunnelling in Urban Neighbourhoods” Project 307836.

References

Albota, Marius a, Brian F Aull, Daniel G Fouche, Richard M Heinrichs, David G Kocher, Richard M Marino, James G Mooney, et al. 2002. “Three-dimensional imaging laser radars with Geiger-mode avalanche photodiode arrays.” *Lincoln laboratory journal* 13 (2): 351–370.

ASPRS. 2011. “LAS specification version 1.4 - R13.” .

ASTM Standard Committee. 2011. “Standard specification for 3D imaging data exchange.” .

Bayer, R, and E McCreight. 1972. “Organization and maintenance of large ordered indexes.” *Acta Informatica* 1: 173–189.

Beckmann, Norbert, Hans-Peter Kriegel, Ralf Schneider, and Bernhard Seeger. 1990. “The R*-tree: an efficient and robust access method for points and rectangles.” *ACM SIGMOD Record* 19 (2): 322–331.

Bentley, Jon Louis. 1975. “Multidimensional binary search trees used for associative searching.” *Communications of the ACM* 18 (9): 509–517.

Bretar, F, and A Chauve. 2008. “Managing full waveform LIDAR data: A challenging task for the forthcoming years.” In *Proceedings of XXI ISPRS Congress,* Vol. 7415–420. Beijing, China.

Bunting, Peter, John Armston, Richard M. Lucas, and Daniel Clewley. 2013. “Sorted pulse data (SPD) library. Part I: A generic file format for LiDAR data from pulsed laser systems in terrestrial environments.” *Computers & Geosciences* 56: 197–206.

Butler, H., M. Loskot, P. Vachon, M. Vales, and F. Warmerdam. 2016. "libLAS: ASPRS LiDAR data translation toolset." `http://www.liblas.org/`.

Elseberg, Jan, Dorit Borrmann, and Andreas Nüchter. 2013. "One billion points in the cloud an octree for efficient processing of 3D laser scans." *ISPRS Journal of Photogrammetry and Remote Sensing* 76: 76–88.

Finkel, R. a., and J. L. Bentley. 1974. "Quad trees a data structure for retrieval on composite keys." *Acta Informatica* 4 (1): 1–9.

Flood, Martin. 2001. "Laser altimetry: from science to commercial LiDAR mapping." *Photogrammetric Engineering & Remote Sensing* 67 (11): 1209–1211, 1213–1217.

Girardeau-Montaut, D., M. Roux, R. Marc, and G. Thibault. 2005. "Change detection on points cloud data acquired with a ground laser scanner." *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 36(3): W19.

GSI. 2016. "Geographical Survey Institute Map Service." `http://maps.gsi.go.jp/`.

Guttman, Atonin. 1984. "R-trees: a dynamic index structure for spatial searching." In *ACM SIGMOD,* 47–57. Boston: Association for Computing Machinery.

Han, Soohee, Sangmin Kim, Jae Hoon Jung, Changjae Kim, Kiyun Yu, and Joon Heo. 2012. "Development of a hashing-based data structure for the fast retrieval of 3D terrestrial laser scanned data." *Computers & Geosciences* 39: 1–10.

Hinks, Tommy, Hamish Carr, and Debra F. Laefer. 2009. "Flight optimization algorithms for aerial LiDAR capture for urban infrastructure model generation." *Journal of Computing in Civil Engineering* 23 (6): 330–339.

Hoefsloot, Wijga. 2012. "Point clouds in a database - data management within an engineering company." .

Höfle, B, and M Hollaus. 2010. "Urban vegetation detection using high density full-waveform airborne lidar data-combination of object-based image and point cloud analysis." *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 38 (7B): 281–286.

Hovi, Aarne. 2015. "Towards an enhanced understanding of airborne LiDAR measurements of forest vegetation." Doctoral dissertation. University of Helsinki.

Hua, L I U, Huang Zhengdong, Zhan Qingming, and L I N Peng. 2008. "A database approach to very large LiDAR data management." *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 37 (B1): 463–468.

Huber, Daniel. 2011. "The ASTM E57 File Format for 3D Imaging Data Exchange." In *SPIE 7864 Three dimensional imaging, interaction, and measurement, 78640A,* International Society for Optics and Photonics.

Hug, C, A Ullrich, and A Grimm. 2004. "Litemapper-5600 a waveform-digitizing LiDAR terrain and vegetation mapping system." *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences* XXXVI, PAR: 24–29.

Isenburg, Martin. 2013. "LASzip: lossless compression of LiDAR data." *Photogrammetric Engineering & Remote Sensing* 79 (2): 209–217.

Isenburg, Martin. 2014. "PulseWaves: An Open, Vendor-neutral, Stand-alone, LAS- compatible Full Waveform LiDAR Standard." `http://rapidlasso.com/pulsewaves/`.

Kamel, Ibrahim, and Christos Faloutsos. 1994. "Hilbert R-tree: an improved using fractals." In *Proceedings of Very Large Data Endowment,* 500–509.

Kothuri, RKV, S Ravada, and Daniel Abugov. 2002. "Quadtree and R-tree indexes in oracle spatial: a comparison using GIS data." In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data,* 546–557.

Krishnan, Sriram, Christopher Crosby, Viswanath Nandigam, Minh Phan, Charles Cowart, Chaitanya Baru, and Ramon Arrowsmith. 2011. "OpenTopography: a services oriented architecture for community access to LiDAR topography." In *Proceedings of the 2nd International Conference on Computing for Geospatial Research & Applications - COM.Geo '11,* 1–8. New York, New York, USA: ACM Press.

Laefer, Debra F., Carol O'Sullivan, Hamish Carr, and Linh Truong-Hong. 2014. "Aerial laser scanning (ALS) data collected over an area of around 1 square km in Dublin city in 2007." `http://digital.ucd.ie/view/ucdlib:30462`.

Ledoux, Hugo. 2010. "Storage and analysis of massive TINS in a DBMS." In *Management*

of massive point cloud data: wet and dry, Vol. 4945–51.

Leica. 2010. *Leica AHAB DragonEye Dual Head Oblique LiDAR System.* Tech. rep.. Heerbrugg, Switzerland: Leica Geosystems.

Lewis, Paul, Conor P. Mc Elhinney, and T McCarthy. 2012. "LiDAR data management pipeline; from spatial database population to web-application visualization." In *Proceedings of the 3rd International Conference on Computing for Geospatial Research and Applications,* 16. ACM.

Maiman, T. H. 1960. "Stimulated Optical Radiation in Ruby." *Nature* 187 (4736): 493–494.

Mallet, Clément, Frédéric Bretar, Michel Roux, Uwe Soergel, and Christian Heipke. 2011. "Relevance assessment of full-waveform lidar data for urban area classification." *ISPRS Journal of Photogrammetry and Remote Sensing* 66 (6): S71–S84.

Mamon, G, D G Youmans, Z G Sztankay, and C E Mongan. 1978. "Pulsed GaAs laser terrain profiler." *Applied Optics* 17 (6): 868.

Marino, Richard M., and William R. Davis. 2005. "Jigsaw: a foliage-penetrating 3D imaging laser radar system." *Lincoln Laboratory Journal* 15 (1): 23–36.

Martinez-rubi, Oscar, Peter Van Oosterom, Romulo Gonc, Theo Tijssen, and Milena Ivanova. 2015. "Benchmarking and improving point cloud data management in MonetDB." *SIGSPATIAL Special - Big Spatial Data* 6 (2): 11–18.

Middleton, Wek, and AF Spilhaus. 1953. "The measurement of atmosferic humidity." *Meteorological Instruments. Toronto: University of Toronto* 105–111.

Mosa, Abu Saleh Mohammad, Bianca Schön, Michela Bertolotto, and Debra F. Laefer. 2012. "Evaluating the benefits of octree-based indexing for LiDAR data." *Photogrammetric Engineering & Remote Sensing* 78 (9): 927–934.

Moser, Gabriele. 2015. "2015 IEEE GRSS data fusion contest: extremely high resolution." March: 40–41.

Nandigam, Viswanath, Chaitan Baru, and Christopher Crosby. 2010. "Database Design for High-Resolution LIDAR Topography Data." In *Scientific and Statistical Database Management,* 151–159. Springer Berlin Heidelberg.

NOAA Office for Coastal Management. 2015. "Digital Coast - Data Access Viewer." `https://coast.noaa.gov/dataviewer`.

Oracle. 2003. *Oracle Spatial - Quadtree Indexing 10g Release 1 (10.1).* Tech. Rep. December.

Orenstein, Jack. 1982. "Multidimensional tries used for associative searching." *Information Processing Letters* 14 (4): 150–157.

Otepka, J, G Mandlburger, and W Karel. 2012. "The OPALS data manager - efficient data management for processing large airborne laser." In *Proceedings of the ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences,* Vol. I-3153–159. Melbourne, Australia.

Parrish, C.E. 2007. "Vertical object extraction from full-waveform LiDAR data using a 3D wavelet-based approach." Doctor of philosophy. University of Wisconsin-Madison.

Peters, Ravi, and Hugo Ledoux. 2014. "MATAHN - A seamless AHN2 download service." Amersfoort. `http://3dsm.bk.tudelft.nl/matahn`.

Pietrzyk, P. J., and R. C. Lindenbergh. 2014. "Detection of harvested trees in forests from repeated high density airborne laser scanning." *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* II-5 (June): 275–280.

Rahman, M Z A, W H W Kadir, A W Rasib, A Ariffin, K A Razak, Real Estate, and Johor Baharu. 2013. "Integration of high density airborne LiDAR and high spatial resolution image for land cover classification." In *Geoscience and Remote Sensing Symposium (IGARSS),* 2927–2930. IEEE.

Ramsey, Paul. 2013. "LiDAR in PostgreSQL with PointCloud." In *FOSS4G,* Nottingham: OSGeo's Global Conference for Open Source Geospatial Software.

Rapidlasso. 2014. "LAStools: converting, filtering, viewing, gridding, and compressing LiDAR data." `http://rapidlasso.com/lastools/`.

Razak, Khamarrul Azahari, Alexander Bucksch, Menno Straatsma, Cees J. Van Westen, Rabieahtul Abu Bakar, and Steven M. de Jong. 2013. "High density airborne lidar estimation of disrupted trees induced by landslides." In *2013 IEEE International Geoscience*

and Remote Sensing Symposium - IGARSS, No. 3. 2617–2620. IEEE. jul.

Richter, Rico, Sören Discher, and Döllne Jurgen. 2015. "Out-of-core visualization of classified 3D point clouds." In *3D Geoinformation Science: The Selected Papers of the 3D GeoInfo 2014,* 227–242. Springer International Publishing.

Richter, Rico, and Jürgen Döllner. 2014. "Concepts and techniques for integration, analysis and visualization of massive 3D point clouds." *Computers, Environment and Urban Systems* 45: 114–124.

Rieg, Lorenzo, Volker Wichmann, Martin Rutzinger, Rudolf Sailer, Thomas Geist, and Johann Stötter. 2014. "Data infrastructure for multitemporal airborne LiDAR point cloud analysis Examples from physical geography in high mountain environments." *Computers, Environment and Urban Systems* 45: 137–146.

Roth, Rb, and J. Thompson. 2008. "Practical application of multiple pulse in air (MPiA) Lidar in large-area surveys." *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XXVII: 183–188.

Rusu, Radu Bogdan, and Steve Cousins. 2011. "3D is here: Point Cloud Library (PCL)." In *2011 IEEE International Conference on Robotics and Automation,* 1–4. IEEE. may.

Samet, H. 2006. *Foundations of multidimensional and metric data structures*. 1st ed. San Francisco, CA, USA: Morgan Kaufmann.

Schön, Bianca, Morrish Sean Laefer Debra, and Michela Bertolotto. 2009. "Three-Dimensional Spatial Information Systems – State of the Art Review." *Recent Patents on Computer Science* 2: 21–31.

Schön, Bianca, Abu Saleh Mohammad Mosa, Debra F. Laefer, and Michela Bertolotto. 2013. "Octree-based indexing for 3D pointclouds within an Oracle Spatial DBMS." *Computers & Geosciences* 51: 430–438.

Shan, Jie, and Charles K. Toth. 2008. *Topographic laser ranging and scanning: principles and processing*. CRC Press.

Shepherd, E. C. 1965. "Laser to watch height." *New Scientist* 1: 33.

Songxin, Tan, and R.M. Narayanan. 2002. "A multiwavelength airborne polarimetric lidar for vegetation remote sensing: instrumentation and preliminary test results." In *IEEE International Geoscience and Remote Sensing Symposium,* Vol. 52675–2677. IEEE.

Stoker, JM, SK Greenlee, DB Gesch, and JC Menig. 2006. "CLICK: The new USGS center for lidar information coordination and knowledge." *Photogrammetric engineering and remote sensing* 72 (6): 613–616.

Swar, L.M.TH. 2010. "How the Up-to-date Height Model of The Netherlands (AHN) became a massive point data cloud." 1–18.

van Oosterom, Peter, Oscar Martinez-Rubi, Milena Ivanova, Mike Horhammer, Daniel Geringer, Siva Ravada, Theo Tijssen, Martin Kodde, and Romulo Gonçalves. 2015. "Massive point cloud data management: Design, implementation and execution of a point cloud benchmark." *Computers & Graphics* 49: 92–125.

Vo, Anh-Vu, Linh Truong-hong, and Debra F Laefer. 2015. "Aerial Laser Scanning and Imagery Data Fusion for Road Detection in City Scale." In *Geoscience and Remote Sensing Symposium (IGARSS), 2015 IEEE International,* 4177–4180.

Wenzel, K, M Rothermel, D Fritsch, and N Haala. 2011. "An out-of-core octree for massive point cloud processing." In *IQmulus 1st workshop on processing large geospatial data,* 53. Cardiff, UK.

Yang, Jiansi, and Xianfeng Huang. 2014. "A hybrid spatial index for massive point cloud data management and visualization." *Transactions in GIS* 18 (129): 97–108.

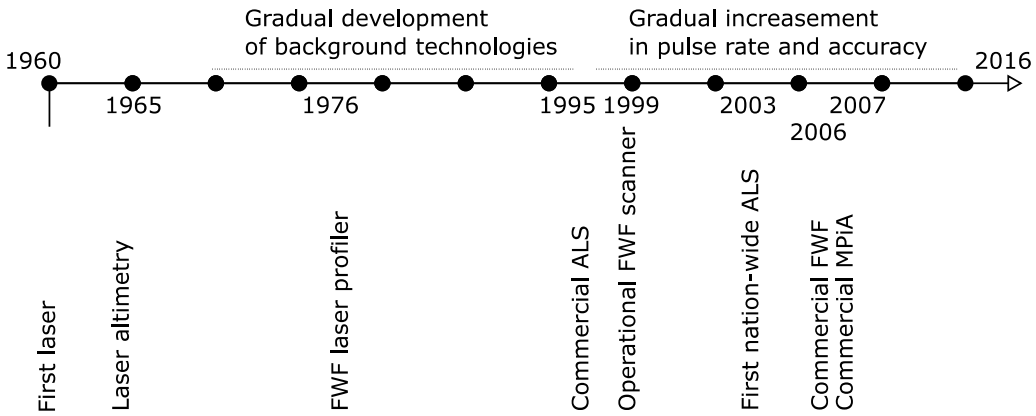

Figure 1. A brief history of the laser scanning technology advancement

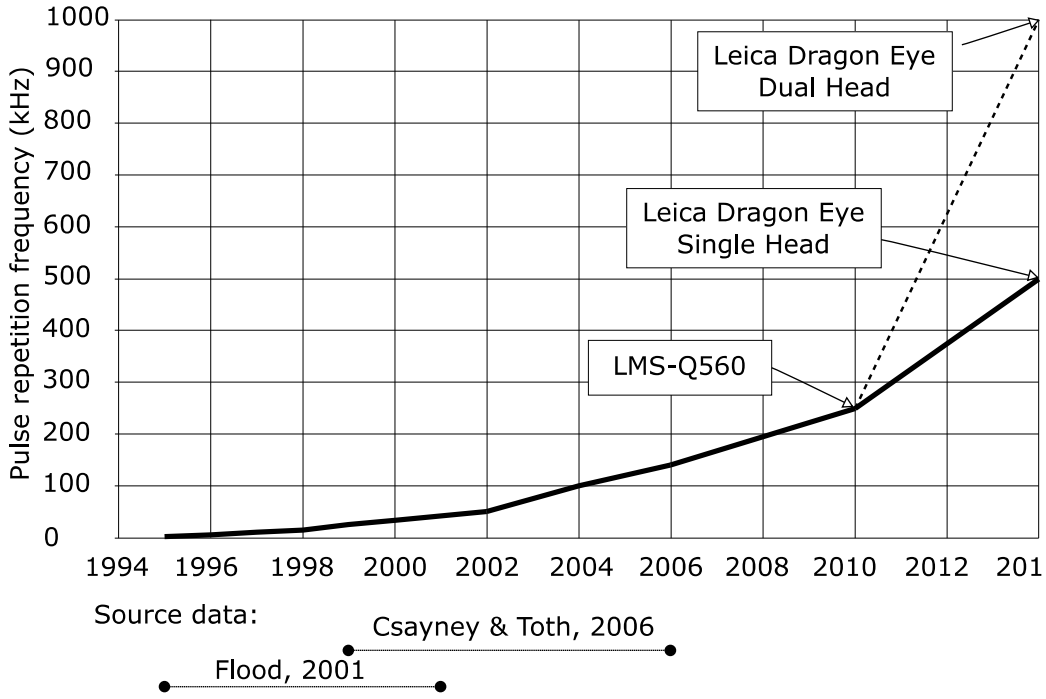

Figure 2. Increase of ALS pulse repetition frequency

Table 1. Inventory of dense ALS datasets; where the bracketed numbers indicate density accumulated from multiple flight strips and the underscores (i.e. _) indicate no published data availability

Location	Year of acquisition	Density *(points/m^2)*	Scanner	Size *(million points)*	Flying attitude *(m)*	Cited in
Vienna, Austria	2006/2007 (FWF)	50	Riegl LMS-Q560	500	_	(Höfle and Hollaus 2010)
Duursche Waarden, Netherlands	2007	70	FLI-MAP 400	_	_	(Rahman et al. 2013)
Dublin, Ireland	2007	(225)	FLI-MAP 2	225	400	(Laefer et al. 2014)
Rotterdam, Netherlands	2008, 2012 (FWF)	(50-200)	_	_	_	(Pietrzyk and Lindenbergh 2014)
Alpes de Haute Provence, France	2009	(170)	Riegl VQ-480i	214	300	(Razak et al. 2013)
Zeebruges, Belgium	2011	65	_	177	300	(Moser 2015)
Dublin, Ireland	2015 (FWF)	50(350)	TopEye S/N 443	1150	300	_