2010

Mobile Mapping System LiDAR Data Framework

P. Lewis
National University of Ireland, Maynooth

C. McElhinney
National University of Ireland, Maynooth

Bianca Schoen-Phelan
Technological University Dublin, bianca.schoenphelan@tudublin.ie

See next page for additional authors

Recommended Citation

Lewis, P., McElhinney, C.. Schoen-Phelan, B., McCarthy, T. (2010). Mobile Mapping System LiDAR Data Framework. *International Archives of the Photogrammetry, Remote Sensing and Spatial Sciences Informattion*, 38, pp.135-138. doi:10.21427/D78D16

MOBILE MAPPING SYSTEM LIDAR DATA FRAMEWORK

P. Lewis [a, *], C. McElhinney [a], B. Schön [a], T. McCarthy [a]

[a] National Centre for Geocomputation, National University of Ireland Maynooth, Maynooth, Co. Kildare, Ireland
(paul.lewis, bianca.schoen, tim.mccarthy)@nuim.ie - conormce@cs.nuim.ie

Commission IV, WG IV/8

KEY WORDS: Spatial Data Framework, Geo-Referenced Lidar, Mobile Mapping Systems.

ABSTRACT:

Mobile Mapping Systems (MMSs) for infrastructural monitoring and mapping are becoming more prevalent as the availability and affordability of solutions that generate high accuracy geospatial data has matured. However, no existent methodology or system exists where all the LiDAR, video, navigation, infrared and multispectral data sources, collected from this mobile platform, are integrated into a single, comprehensive data management solution. Based on empirical experience there is a need for an MMS-data management framework where these types of data can be dynamically accessed and integrated to enable different projects with varying objectives to dynamically access different MMS-data for, in one example, use in feature extraction algorithms. In this paper we introduce the LiDAR aspect of this work towards a MMS-data framework. With large volumes of LiDAR to be stored we have opted for a spatially enabled database (SDB) management solution, specifically PostgreSQL with PostGIS extensions. We detail our approach to storing and querying the LiDAR data in the SDB and provide preliminary results on query times and data returns.

1. INTRODUCTION

Infrastructural mapping and monitoring has become an integral part of the academic, commercial and governmental sphere where detailed knowledge of the built environment is easily accessible. To this end Mobile Mapping Systems (MMS) play an important role in generating these environment-model data sources. They are particularly suited to the road-network infrastructural management case, as multiple environmental modelling sensors can be integrated, transported and calibrated on a single collection platform. Typically, high accuracy near 3D geospatial data can be recorded from which detailed, bespoke and comparative analysis can be performed in order to monitor, plan and understand a road-networks status and/or requirements. For this paper an MMS van has been commissioned, which is equipped with a Global Positioning System (GPS), an Inertial Navigation Sensor (INS), six progressive scan cameras, a Light Detection and Ranging (LiDAR) unit, a multispectral camera and a thermal-imaging camera, which provides approximately 40 gigabytes of data per hour.

Both the storage and post-survey processing of these data present a number of computing challenges because of the high volumes of detailed geospatial information being output. However, it is the storage and accessing of these data that is particularly problematic as no existing integrated framework solution can exploit not only vast data sets such as LiDAR, but also the broader spectrum of spatial information that is being collected, for example video. Such a framework should be able to generate a meaningful, visually-rendered appreciation of the stored geospatial data, which also considers the available levels of detail; for example accuracy, resolution and time; that can ease the task of optimised data retrieval for more detailed analysis uses. Even using optimised industry standard software platforms, such as Terrasolid, it can be enormously time consuming to acquire data from a single source under even the most basic requirements. For example, a typical user may require LiDAR, navigation and imagery data for a particular geographical area, yet the process of isolating these data for congruent segmentation from such a massive data store is extremely challenging.

There exists a significant desire to store vast 3D spatial data in a database management system (DBMS) (Schön et al. 2007; Nandigam, Baru, and Crosby 2010), as DBMSs offer transaction guarantees and multi-user, random access of potentially very large datasets, in addition to advanced features, such as backup and restore capabilities. However, the typical work flow with regards to LiDAR often does not provide the user with the actual raw LiDAR data. Instead, users have to decide on a format for the data that they wish to perform certain analysis on, for example a Digital Elevation Model (DEM). For the MMS context in particular, there typically exist two vast 3D point data sets: one is the navigation points that describe the GPS track of the MMS throughout the survey, and the other being the actual LiDAR survey point cloud. Preserving this information has the potential to empower several queries, where the collection of navigation points can be employed in order to describe the actual LiDAR data set. Consequently, users are currently prohibited from exploiting the full range of opportunities that typical MMS surveys offer. As a result Spatial DBMSs (SDBMSs) appear particularly attractive in this context.

However, with regards to the storage of LiDAR data, while DBMSs have been used in this context, (Schön et al. 2007; Nandigam, Baru, and Crosby 2010; Sharma, Parikh, and Clark 2006; Rottensteiner, Jansa, and Sensing 1999), no significant solution currently exists to support this approach over and above existing LAS file format solutions. In Nandigam et al. (2010) it is suggested that alternative support that includes LAS file formats should form an integral part of their implementations where they only store metadata related to the point data in the DBMS, while the actual data remains stored across several files. An example of a popular SDBMS is PostGIS (2001), which is an implementation of the OCG standard and provides a spatial extender to PostgreSQL. PostGIS enjoys widespread support and substantial integration with GIS software, such as Mapserver, Geotools, FDO and many more. However, the advantages of a system like PostGIS remain relatively unexploited with regards to MMS surveys.

* Corresponding author.

This paper discusses the LiDAR data management aspect of a broader MMS spatial data handling framework, as illustrated in figure 1. This work outlines how PostGIS can be used in order to store these high volume 3D spatial data sets, which subsequently forms the core data storage and management solution in the proposed MMS framework. However, access and retrieval of the high resolution raw LiDAR data from the database is controlled through a GIS constrained modelling and aggregation approach similar to that implemented for MMS video data (Lewis, Fotheringham, and Winstanley 2010; Lewis, Winstanley, and Fotheringham 2009). The objective of this approach is to isolate these data based on use case scenarios, where the output can be used for either visualisation requirements or computationally intensive feature extraction operations in a more efficient manner.

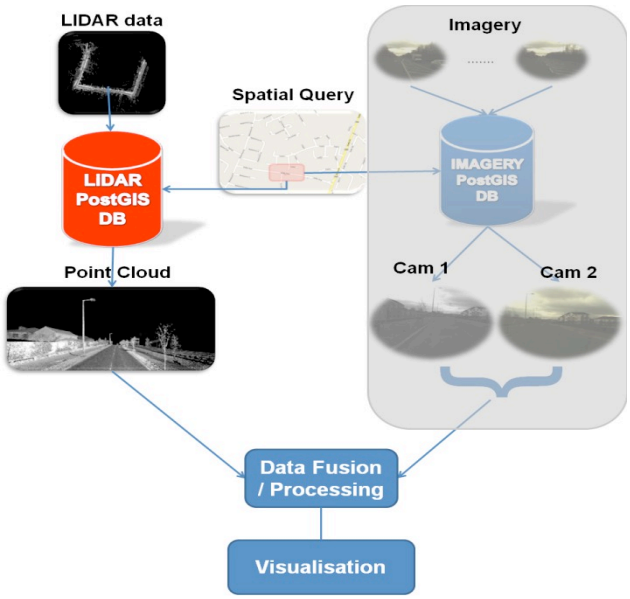

Figure 1. Overview MMS geospatial data handling framework model. Highlighted is the LiDAR PostGIS database solution discussed in this paper.

2. LIDAR DATA-FRAMEWORK

Empirical experience with MMS geospatial data, in particular LiDAR data, suggests that the primary obstacles in the processing of these data is their considerable size and the inability to easily constrain the data based on its spatial attributes or its point attributes. Leading on from this is the extraction and preparation difficulties when using these data for bespoke requirements. For example, an algorithm has recently been developed for the detection of road edges from terrestrial LiDAR (McElhinney et al. 2010), yet this operation is being constrained by the survey-processing methodology that prevails in industry standard software suites being used. These suites provide no context for spatial optimisation across numerous surveys, where data segmentation for road-edge detection can be easily implemented based on where the interest area is rather than which survey it belongs too. Therefore, numerous runs of this algorithm have to be performed on separate surveys. Alternatively, difficult data-assimilation processes have to be followed to generate the single source data set within these software solutions.

However, approaching this problem with a spatial-constraint perspective, it is possible to optimise the LiDAR data being output. This can be achieved through procedures that leverage the power of a platform such as PostGIS and its numerous, integrated, spatial API's. The geo-referenced raw LiDAR is stored in a database where optimised spatial indexes can be generated in order to facilitate efficient querying of the data set. Consequently, optimally located LiDAR data can be output in a user requirements spatial context where use cases, such as the road detection algorithm, can operate on a reduced target data set.

Drawing on the spatial variables collected from an MMS survey, shown in table 1, a selective LiDAR data segmentation example can be shown. Given the known calibration information for the MMS platform a spatial extent query can be performed using the low resolution navigation data to segment the high resolution LiDAR for the road detection algorithm. In this case a bounding box can be constructed where, in the altitude plane, it defines 3D space that is below the GPS track, in the orthogonal plane to the traversal direction, it is extended to a adjustable distance likely to cover beyond the road edge and, in the traversal plane, can extend to an adjustable distance along the GPS track. Based on this bounding box a 3D spatial query can isolate from the larger LiDAR data store all points contained within, thus reducing the amount of points that need to be processed by the road-edge detection algorithm.

Using this selective example it has been shown that the road-edge detection algorithm can produce results more efficiently, however, and importantly, it can be applied across multiple different surveys much easier.

Navigation data	LiDAR data
Ranges from 10Hz – 250Hz	Ranges from 50kHz – 400kHz
GPS Time	GPS Time
Latitude	Latitude
Longitude	Longitude
Altitude	Altitude
Roll	Pulse Width
Pitch	Echo Number
Yaw	Reflectance
	Intensity RGB Value
	Waveform data (multiple values per point)

Table 1. GPS and LiDAR data variables collected during a MMS survey.

3. LIDAR DATA STORAGE AND ACCESS

As is highlighted in figure 1, and described for the video context in Lewis et al. (2010); the LiDAR-data management aspect of the framework is integrated into a PostgreSQL database. Accessing these data is achieved through spatial SQL queries that are dynamically generated and constrained based on a user interaction operation on the navigation data. This operation is in the form of a controllable point or polygon spatial query, either 2D or 3D, that is generated from a planar GIS-mapping operation. The mapping space is displayed in a standard GIS format where the LiDAR's associated navigation data has been displayed to help inform the user of where each survey was completed. In figure 2 a desktop access interface is shown, while in figure 3 a Internet browser version is shown. Both interfaces use the left pane to show the navigation data for all surveys in the database that are relevant to the user defined viewable extent. The intention with this approach to is facilitate the user with an appreciable view of all the survey's in the database relevant to an interest area, not just a single survey approach as highlighted earlier. Through these interfaces the user can interact with the navigation-data map space using standard GIS tools and spatial operations.

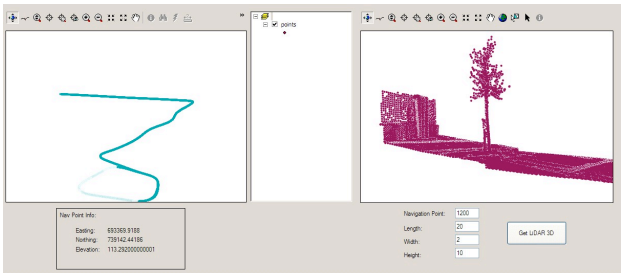

Figure 2. Desktop Interface for dynamic access, visualisation and segmentation of MMS LiDAR data. This implementation is developed in C# using the ESRI ArcGIS SDK platform.

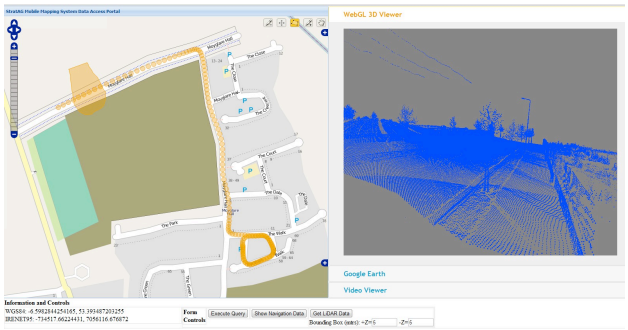

Figure 3. Internet Browser implementation utilising OpenLayers in the query pane to interact with the navigation track data, while a WebGL 3D point cloud viewer in the right pane facilitates viewing and interaction with the segmented LiDAR query results.

In both of these implementations either point or polygon geometries can be created to provide the geographical context for the LiDAR spatial query. From this the user can chose to constrain their LiDAR data retrieval operation in either 2D or 3D space; in the 2D context no altitude (Z parameter) is constrained. These spatial operations return a 3D LiDAR point cloud for visualisation in a 3D viewer. From this further segmentation can be performed to refine the LiDAR returns or these results can be passed to the next system requirement, as determined by the user. As an example figure 4 shows how the LiDAR returns can be constrained visually or programmatically to only return a 3D point cloud which is optimal to a road side feature extraction algorithm.

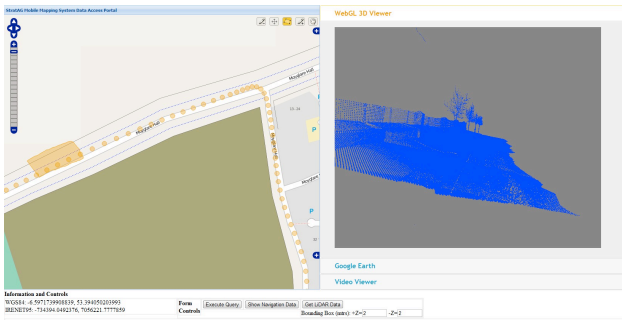

Figure 4. Example illustration of a segmentation process for a road side feature extraction algorithm. In this case it is a visual segmentation, however it can be easily implemented programmatically for automated operations over large geographical spaces.

For a 2D segmentation query a 3D LiDAR data is returned, this is achieved using a spatially contains operation on the X and Y parameters only. In 3D operations a volume is constructed for the query where all LiDAR points that fall into this volume in X, Y and Z space are returned. Table 2 lists the basic operations that can be performed in the GIS-mapping interface where 2D and 3D query space can be generated. In a programmatic implementation all these operations can be performed automatically where the queries are informed from the navigation data.

Interface Spatial Query Object	Drawn Point Extra Parameters	Drawn Polygon Extra Parameters
2D Buffer	Circular Distance	Not Applicable
2D Box	(X,Y) Constraint	Not Applicable
3D Box	(X,Y,Z) Constraint	(+/-Z) Constraint
Irregular 3D Space	3D Box	Freehand Polygon, (+/-Z) Constraint

Table 2. Example set of spatial operation functionality that is used in either the visualisation pane for the navigation data or programmatically in automated routines.

4. PRELIMINARY RESULTS

In this section we highlight preliminary results where average timings from a number of parameter constrained spatial queries have been completed. Each query selected a random navigation point and performed a spatial query on the associated LiDAR point cloud. The reported access times are the result of averaging 20 queries. This timing analysis procedure included the selection of the navigation point, preparation of the spatial query object, i.e. a table 2 object; the spatial SQL statement, running the query on the database and transferring the resulting LiDAR 3D point data to the calling process.

Our test system is a Dell Precision T7500 Desktop PC with 6GB RAM, 64bit OS and 2.27GHz Intel(R) Xeon(R) processor. Moving our data framework solution to a dedicated server would lead to a noticeable decrease in query execution time. It has also been demonstrated by Nandigam et al. (2010) that the configuration of the hardware in such a system can lead to performance increases of greater than 4 times. This should be considered when interpreting the times in the following tables.

In table 3 the results are highlighted for a 2D Box spatial object query where the width parameter has been incrementally increased from a 0.1 metre to 40 metre constraint. The length parameter is maintained at 20 metres and represents a length along the road coincident to the navigation track and can be defined at any point forward, back or around the query space navigation point.

Dimensions (m) – (length x width)	Time – average (s)	Points – average
20x0.1	0.3994	2464
20x0.3	1.1987	9797
20x1	3.3944	34055
20x5	13.4179	151159
20x20	46.2630	536020
20x40	98.5061	1150566

Table 3. Timing results for 2D Box spatial query on the LiDAR database.

In table 4 the results are highlighted for a 3D Box spatial object query where the width parameter has been incrementally increased from a 0.1 metre to 40 metre constraint and the height parameter is fixed at 1 metre from the road surface directly below the van. This 1 meter height constraint is an offset from the navigation data altitude adjusted by its calibrated height above the road surface. Effectively, this 3D box spatially filters

the LiDAR for a road extraction algorithm where all points returned are within 0.5 of a meter, plus or minus, from the traversal surface.

Dimensions (m) – (length x width x height)	Time – average (s)	Points – average
20x0.1x1	0.3938	1481
20x0.3x1	1.0433	5921
20x1x1	2.6971	20677
20x5x1	9.9390	93170
20x20x1	36.2919	360936
20x40x1	82.4819	833445

Table 4. Timing results for 3D Box spatial query on the LiDAR database.

A plot of query time versus the varied width dimension for both 2D and 3D is displayed in figure 5. There are two important characteristics noticeable from this plot. First, there is a near linear relationship between the box objects dimension and the query time and secondly, the 3D query although more complex has a quicker execution time in all cases. We intend to demonstrate that by constraining the spatial query using a points attributes, such as amplitude and pulse width, we can reduce the time taken to return an optimised LiDAR point cloud containing only the required points for processing.

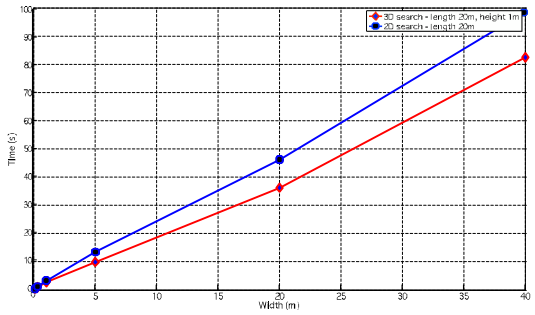

Figure 5. Graph plot for analyses of the timing results from table 1 and 2.

5. CONCLUSIONS AND FUTURE WORK

In this paper we have presented the initial stages of our development work towards a LiDAR-data management solution. Large volumes of point cloud data can be segmented geographically and/or based on non-spatial constraints. These could include user-defined constraints for the selection of specific point cloud attributes or algorithm constraints were data has to be segmented in a specific way. Ultimately these possibilities produce spatially optimised units targeted to a feature extraction algorithms requirements. While this work does not comprehensively define a complete DBMS LiDAR data handling solution it does provide the initial building blocks such that it has encouraged us to continue to pursue this research direction in both software development and hardware acquisition. Based on our requirements for a LiDAR-data spatial segmentation solution we have shown how a PostgreSQL database solution with PostGIS spatial extensions can be an efficient and effective platform for this work.

Taking this work forward a number of issues remain to be tested. Firstly the point cloud I/O has not been fully analysed where we still have significant data intake issues; a solution to this is currently being implemented with initial testing showing large increases in performance. As has been shown, 3D queries are faster than the comparative 2D queries and as the framework matures and more bespoke LiDAR feature extraction algorithms are added, we intend to show how further constraints on both 2D and 3D spatial queries can improve data segmentation and access times.

ACKNOWLEDGEMENTS

Research presented in this paper was funded by a Strategic Research Cluster grant (07/SRC/I1168) by Science Foundation Ireland under the National Development Plan and by the ERA-NET SR01 projects. The authors gratefully acknowledge this support.

REFERENCES

Lewis, Paul, A. Stewart Fotheringham, and Adam Winstanley. 2010. Spatial Video and GIS. *International Journal for Geographical Information Science* In Press.

Lewis, Paul, Adam Winstanley, and A. Stewart Fotheringham. 2009. Using ViewCones to Model Terrestrial Spatial Video. In *3D Geo-Information 2009*,Philippe De Maeyer, Tijs Neutens, and Marijke De Ryck, 149-158. Gent, Belgium: Gent University. http://www.3dgeoinfo.org/.

McElhinney, Conor P, Pankaj Kumar, Conor Cahalane, and Timothy Mccarthy. 2010. Initial results from European Road Safety Inspection (EURSI) mobile mapping project. In *ISPRS Commission V Technical Symposium*, 2007:. ISPRS.

Nandigam, V, C Baru, and C Crosby. 2010. Database Design for High-Resolution LIDAR Topography Data. *Scientific and Statistical Database* 6187/2010: 151-159. doi:10.1007/978-3-642-13818-8_12. http://www.springerlink.com/index/X5Q937840983UN76.pdf.

PostGIS. 2001. PostgreSQL, PostGIS Spatial Extension. *Internet Source*. Refractions Research. http://postgis.refractions.net/.

Rottensteiner, F, J Jansa. 2002. Automatic extraction of buildings from LIDAR data and aerial images. In: *International Archives of Photogrammetry, Remote Sensing* XXXIV/4, pp. 569-574.

Schön, Bianca, Michela Bertolotto, Debra F. Laefer, and Seán W Morrish. 2007. Storage, Manipulation and Visualization Of Lidar Data. In *International Society Of Photogrammetry and Remote Sensing*. Trento, Italy: 3D-ARCH 2009. http://www.isprs.org/proceedings/XXXVIII/5-W1/pdf/schoen_etal.pdf.

Sharma, N., J. Parikh, and M. Clark. 2006. A Lidar Collaboratory Data Management System. *2006 IEEE International Symposium on Geoscience and Remote Sensing*: 817-820. doi:10.1109/IGARSS.2006.209. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4241356.