EVALUATING VOXEL ENABLED SCALABLE INTERSECTION OF LARGE POINT CLOUDS

Jinhu Wang[a,b,*], Roderik Lindenbergh[a] and Massimo Menenti[a]

[a] Dept. of Geoscience and Remote Sensing, Delft University of Technology
Building 23, Stevinweg 1, Post Box 5048, 2628CN Delft, The Netherlands
(jinhu.wang, r.c.lindenbergh, m.menenti)@tudelft.nl
[b] Key Laboratory of Quantitative Remote Sensing Information Technology
Academy of Opto-Electronics, Chinese Academy of Sciences
No. 9 Deng Zhuang South Road, HaiDian District, 100094 Beijing, China

Commission V/5

KEY WORDS: Voxel, Laser Scanning, Point Clouds, Intersection Area

ABSTRACT:

Laser scanning has become a well established surveying solution for obtaining 3D geo-spatial information on objects and environment. Nowadays scanners acquire up to millions of points per second which makes point cloud huge. Laser scanning is widely applied from airborne, carborne and stable platforms, resulting in point clouds obtained at different attitudes and with different extents. Working with such different large point clouds makes the determination of their overlapping area necessary but often time consuming. In this paper, a scalable point cloud intersection determination method is presented based on voxels. The method takes two overlapping point clouds as input. It consecutively resamples the input point clouds according to a preset voxel cell size. For all non-empty cells the center of gravity of the points in contains is computed. Consecutively for those centers it is checked if they are in a voxel cell of the other point cloud. The same process is repeated after interchanging the role of the two point clouds. The quality of the results is evaluated by the distance to the pints from the other data set. Also computation time and quality of the results are compared for different voxel cell sizes. The results are demonstrated on determining he intersection between an airborne and carborne laser point clouds and show that the proposed method takes 0.10%, 0.15%, 1.26% and 14.35% of computation time compared the the classic method when using cell sizes of of 10, 8, 5 and 3 meters respectively.

1 INTRODUCTION

Light Detection And Ranging (LiDAR) mapping techniques enable to quickly acquire geometric information in 3D (Vosselman and Maas, 2010). Compared to airborne photogrammetry, laser scanning has the advantages of high speed and high quality acquisition of points (Baltsavias, 1999a, Rönnholm et al., 2007). In recent years there is a rapid development of different platforms for laser scanning, such as Airborne Laser Scanning (ALS), Terrestrial Laser Scanning (TLS) and Mobile Laser Scanning (MLS). Laser scanning has already been applied to situations, such as tunnel inspection and monitoring (Gosliga et al., 2006) and road engineering (Wang et al., 2013, Jaakkola et al., 2008, Kumar et al., 2013, Pu et al., 2011), vegetation and land cover classification (Bater et al., 2011, Antonarakis et al., 2008, Yunfei et al., 2008, Puttonen et al., 2010), and object extraction and reconstruction (Olsen et al., 2010, Brasington et al., 2012). However, due to the scanning mechanism of the different systems, the obtained point clouds have different properties (Vosselman and Maas, 2010), such as non-uniform point density and occlusion effects (Wehr and Lohr, 1999, Forest and Salvi, 2002). To fully sample a complicated scan, it may be required to scan its objects in multiple runs, and from multiple platforms.

Since different platforms have their own advantageous perspective, multiple platforms are now combined to obtain complete coverage of objects (Zhou and Vosselman, 2012, Baltsavias, 1999b, Holopainen et al., 2013, Sithole and Vosselman, 2004). Once point clouds from multiple sources are used, the determination of area of intersection becomes a compulsory operation. 2D and 3D intersection have been studied and discussed in the fields of computational geometry, computer graphics and robotics. In (Shamos and Hoey, 1976), optimal algorithms were developed to determine the intersection of geometric objects in 2D. In (Dobkin and Kirkpatrick, 1983) low complexity methods for suitably preprocessed polyhedral intersections were distinguished based on convex. In (Mount, 1992), by constructing an enveloping triangulation called a scaffold two preprocessed simple polygons are detected whether they intersect one another. And in (Wang, 1996) the intersection curves of two parametric surfaces are presented based on the concept of normal projection. In (Manocha and Canny, 1991, Grandine and Klein, 1997) new approaches were designed by determining numerical solution of differential algebraic equations. In 3D, approaches were also developed for intersection detection. In (Chazelle, 1989, Dobkin and Kirkpatrick, 1985), an intersection algorithm based on the checking of the relationship between vertices and planes was presented. In (Pan et al., 2012), intersection of objects was determined by detecting objects' bounding boxes, and then objects collision checking and proximity computations were performed. Those literature did give the concept of object intersection determination, but the objects all had a primitive geometry shape and the number of objects is limited. The latest scanners obtain up to millions of high precision points (Vosselman and Maas, 2010, Hyyppä, 2011), and the data sets generated by those laser scanning systems are huge. To determine intersections either it would be needed to convert the discrete point clouds to standard geometry objects or using classic point based methods, which make the procedure computational expensive.

Currently there is commercial software available for point cloud processing, but most of them, such as Leica Cyclone (Leica Geosystems, 2015) and Faro Scene (Faro Scene, 2015), don't support au-

[*] Corresponding author

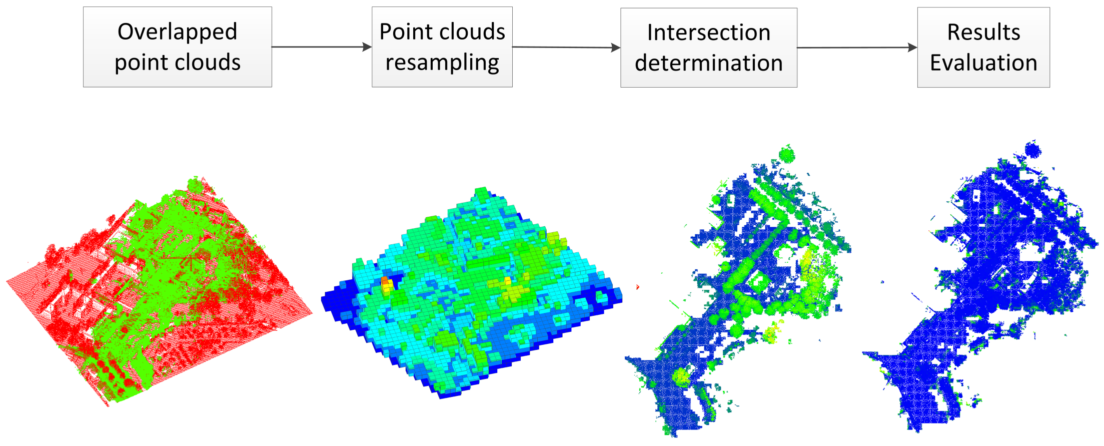

Figure 1: Overall methodology of the algorithm

tomatic intersection determination (Varela-González et al., 2013). Some open source softwares such as CloudCompare (CloudCompare, 2015) and LasTools (Isenburg, 2015), have segmentation tools accessible from a Graphical User Interface (GUI) but don't have data based intersection tools. Also the classic intersection determination based on point clouds neighborhood searching is computational expensive and unable to deal with huge point clouds. Therefore in this paper a novel point cloud intersection method is proposed that systematically visits only part of the points in the input point clouds.

This study presents a method for automatic and scalable area of intersection determination from two huge point cloud data. The results are evaluated by computing the minimum distance between the two overlapping point clouds at different scales. Also, the quality and efficiency of this method are compared to the classic point based intersection method.

2 METHODOLOGY

Intersection problems are fundamental to many aspects of geometric computing. In 2D, the intersection of objects is defined as the set of points or area shared by both objects (Shamos and Hoey, 1976, Grandine and Klein, 1997). While in 3D, the intersection of two objects is the volume that shared by them (Dobkin and Kirkpatrick, 1985, Dobkin and Kirkpatrick, 1990). However, those definitions are from objects that have a definite boundary. For raw point clouds acquired by laser scanning, there are no clear boundaries. So the proposed method is an approximation method. As shown in Figure 1, the overall methodology presented in this paper consists of four steps. (1) Two overlapping point clouds are imported as input. In the figure, the red and green points are the two point clouds respectively. (2) The point clouds are re-sampled as voxel cells and the center of gravity of the points in each cell is computed. (3) The geometric relation between two voxels is checked and the area of intersection is determined. (4) The quality of the results is evaluated by computing the point based distances between the two point cloud data sets restricted to their determined intersection.

2.1 Point clouds resampling

The imported point clouds are re-sampled as voxel cells. As a $Pixel$ is a 2D representation, a $voxel$ is a cube with a predefined edge length in 3D. This edge length defines the resolution of the voxel resampling procedure. Figure 2 is an illustration of a voxel cell in 3D, where P_{max} is the upper right back point and P_{min} is the lower left front point of the cell.

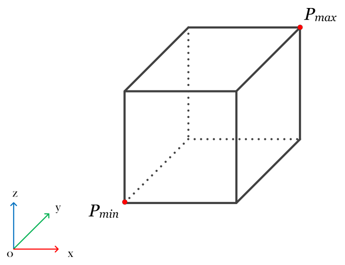

Figure 2: Voxel cell in 3D

The voxelization step is shown in Figure 3. The coordinate system used here is right-hand Cartesian. Firstly the lower left front point and upper right back point are obtained from the input point cloud data. With a predefined voxel cell size, the bounding box of the point cloud is divided into cubic cells. Next the points from the imported point cloud are assigned to the cells they fall in.

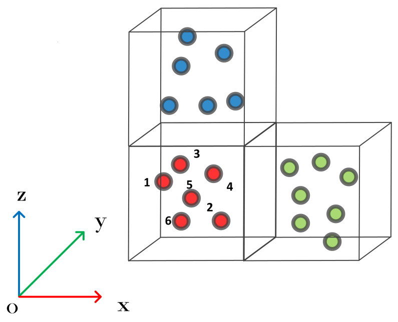

Figure 3: Point cloud resampling using voxel cells.

The data structure designed for a voxel cell includes three entities: (i) the 3D index of the cell; (ii) a container storing the indices of the points that are within the geometric space of this voxel cell; and (iii), coordinates of the 3D center of gravity of all the points stored in the point indices container.

2.2 Intersected area determination

We call one of the imported point clouds $Source$ and the other $Target$. After the data sets are resampled as voxel cells and the center of gravity of the points in all cells is computed, the area of intersection is determined at the voxel cell level. Figure 4 illustrates this strategy. The green rectangles represent the voxel cells, which are generated from $Source$ where the blue dots are the center of gravity of the voxel cells from $Target$. Once the center point of a cell from $Target$ is detected within a voxel cell from $Source$, this cell is then regarded as a cell that belongs to the intersection. The green shaded cells together denote the area of intersection.

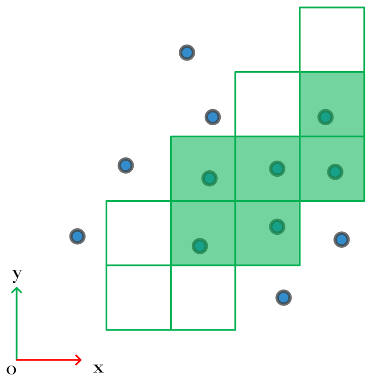

Figure 4: Determination of the area of intersection of two overlapping point clouds

2.3 Point based intersected area determination

In Figure 5, blue points are from $Source$ and green points are from $Target$. Point based intersection starts with a predefined distance threshold R, as depicted by the red dishes in the figure. Then all the points are traversed and neighborhood points are searched. All points within this radius are regarded as overlapping points. Finally the region of intersection of the two point clouds is acquired, as denoted by the yellow area in Figure 5.

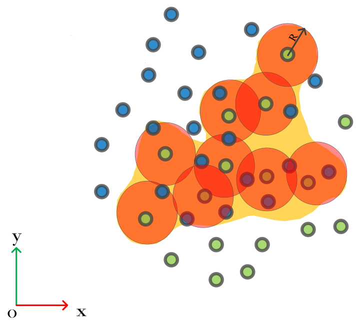

Figure 5: Point based intersection area determination.

2.4 Result evaluation

The evaluation of the results consists of three parts. First the method presented in this paper is implemented and applied at different scales, which means the point cloud data are resampled by voxel cells of different cell sizes. Secondly, point based intersection determination is carried out with different radii on the same test point clouds. Finally, the results of the two different methods are evaluated and compared with respect to their distances.

3 RESULTS AND VALIDATION

3.1 Test data sets

The test data consists of two point clouds from different platforms. One is from airborne laser scanning (ALS) and the other from mobile laser scanning (MLS). General information of the two point clouds is given in Table 3.1. Both of the two point clouds sample part of the campus of Delft University of Technology (TUD).

	MLS	ALS
Nr. of points	27,627,526	1,508,534
Point density	1883 pts/m^2	20 pts/m^2
Scanner	Fugro Drive-Map	Unknown
Attributes	intensity+RGB+xyz	intensity+xyz

Table 1: Information on multiple platform point clouds

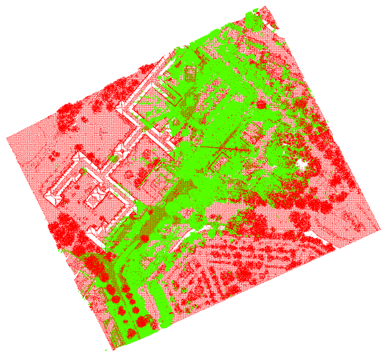

Figure 6: Two overlapping multiple platform point clouds.

A top view of the two point clouds is shown in Figure 6. The point cloud in red is the ALS point cloud from Actueel Hoogtebestand Nederland (AHN-2) (van der Sande et al., 2010), and the other in green is the MLS point cloud obtained by Drive-Map (Fugro, 2015). the overlap of the two pint clouds is considerable and contains several roads and buildings of the TUD campus.

3.2 Scalable testing of the method

Corresponding to the intersection results for four different voxel cell sizes, cloud to cloud distances are computed between the $Source$ and $Target$ point clouds. The distances between two point clouds represents how far of each point in the $Source$, the closest point in the $Target$ point cloud is. We employ this as one aspect of intersection quality evaluation. The bigger the distance, the coarser the intersection area determination, while the smaller, the better the results. As can be seen from Figure 7, the smaller the voxel cell size used for computing, the smaller the distances between the two intersected point clouds. This denotes that if one wants better results, a smaller voxel cell size shall be used.

The next step is to quantify the distance computation results. To do this, statistical analysis is conducted on the results of the four different voxel cell sizes. Suppose the distances are random. Since the values of distances are all non-negative, this variable is a typical Log-Normal distribution (Beaulieu and Xie, 2004,

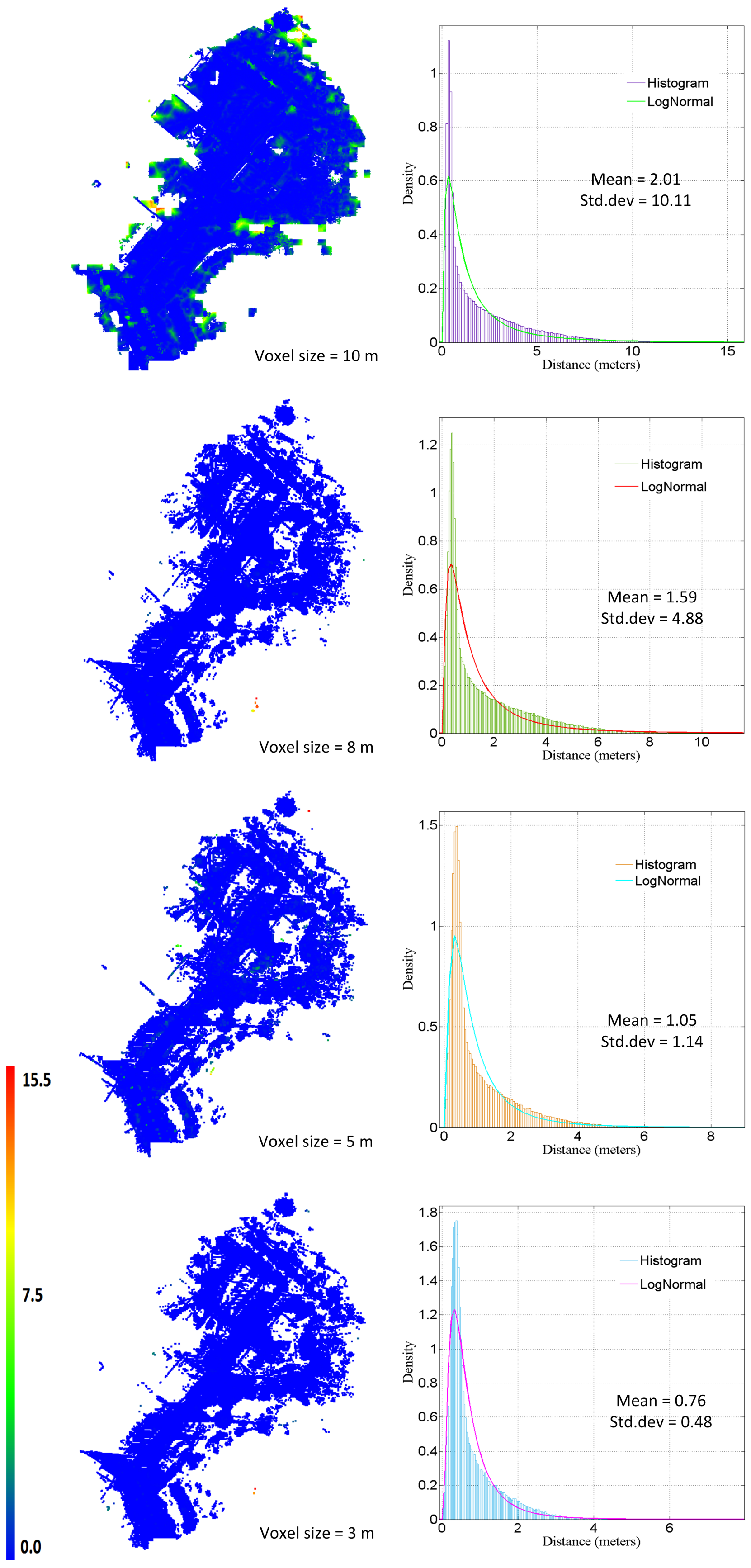

Figure 7: Cloud to cloud distances and statistical analysis from four different voxel cell sizes.

Beaulieu and Rajwani, 2004). The probability density function is given in Equation 1. Here μ is the mean and σ is the standard deviation of $ln(x)$, also named $location\ parameter$ and $scale\ parameter$ respectively.

$$f(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}x\sigma}e^{-\frac{(ln(x)-\mu)^2}{2\sigma^2}}(x>0) \tag{1}$$

The statistical analysis of the distances computed from the four intersection determination results are shown in Figure 7. It can be noted from the figure that the quality of the point cloud intersection varies upon the voxel cell size used. The parameters of the four results are shown in Table 3.2. While the voxel cell size changes from big to small, the average distances of the results are getting small as well. As learnt from the precessing with those different voxel cell sizes, the mean of the distances are around 20% of the voxel cell size. Also the mean and standard deviation of the Log-Normal distribution follow the same trend.

Cell size (m)	Mean	Std.dev
10	2.01	10.11
8	1.59	4.88
5	1.05	1.14
3	0.76	0.48

Table 2: Mean and Std.dev. of the distances

3.3 Comparison with point based intersection results

The point-wise intersection method first needs a predefined cloud to cloud distance as a criterion that defines the area of intersection. The intersection is defined as the points that are closer than the 1.0 meter distance threshold from each other. To evaluate the influence of point numbers to the processing, the original point clouds are down-sampled to different scales. For each scale, the computation time of the two methods are compared. In this study, the processing time of different voxel cell sizes are also compared.

Both algorithms are implemented in C and run on a Dell desktop computer, with a Intel E5-1620 processor and a memory of 16GB. During the tests, the original point clouds are down-sampled at different percentages. As described in Table 3.3, the original point clouds are uniformly down-sampled to three scales.

Scenario	MLS pts number (%)	ALS pts number(%)
1	110,254(7.31%)	71,065(6.86%)
2	345,860(22.9%)	261,280(25.21%)
3	885,084(58.67%)	723,068(69.77%)
4	1,508,534(100%)	1,036,417(100%)

Table 3: Scenarios of down-sampled points

For the $Source$ point cloud 7.31%, 22.9% and 58.67% of the original points are kept and for the $Target$ point cloud 6.86%, 25.21% and 69.77% of the points are kept for comparison. Based on those scenarios, the two algorithms are compared.

The computational time of voxel based method is related to the complexity of the algorithm. Suppose the bounding box of the point cloud is D and the voxel cell size is d, then the number of voxel cell is $N = (\frac{D}{d})^3$. For each cell, its maximum and minimum boundary points are checked whether this cell contains the center point or not. So their are at least 32 operations for each cell. Then the total cost is at least $N_t = 2 \times (\frac{D}{d})^3$. For an imported point cloud, its spatial extent is constant and then the computational time will only related to the voxel cell sizes. Letting d_i and d_j are two different voxel cell sizes that applied to the same point cloud, the ratio of their total computational costs is $R_{ij} = \frac{N_i}{N_j} = (\frac{d_j}{d_i})^3$, where d_i and d_j are the voxel cell sizes. The smaller the voxel cell size, the larger the number of occupied cells and the longer the computational time. However, as the smaller the voxel cell size, there will also more empty cells. This letting the increase ratio of the non-empty cells be smaller than $(\frac{d_j}{d_i})^3$. So the upper bound of computational expenses related to different cell size is $(\frac{d_j}{d_i})^3$. The computational time of the testing data sets is illustrated in Figure 8. For the presented voxel based methods, as denoted by the bottom horizontal label and the blue line, the computation time of voxel based method on different sizes. For the point based method, the more of the points number, the longer of the computation time it takes, as illustrated by the red line in Figure 8.

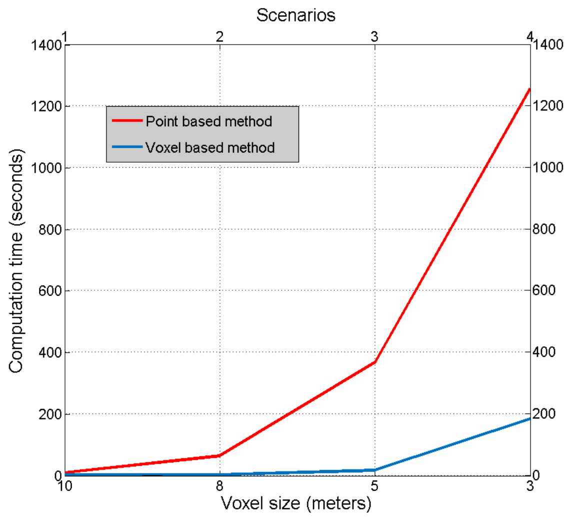

Figure 8: Computation time for different voxel cell sizes and down-sampling scenarios.

In Figure 9 the computation time of the two methods is illustrated. The red line is the computation time of the point based intersection determination. For each intersection the distances between overlapping points in the intersection were calculated. In Figure 10 the mean and standard deviation of the distances are shown. It can be noticed that there is only small variation in the mean and standard deviation of the four pairs of testing scenarios when using the same distance threshold. From the first pair to the fourth pair, the computation time varies from 8.2 seconds to 1255.8 seconds, as shown in Figure 8.

For voxel based method, focusing on the original point clouds, the computation time of the presented method is 0.10%, 0.15%, 1.26% and 14.35% of the time needed by point based methods for voxel cell sizes of 10, 8, 5 and 3 meters respectively.

For the voxel based methods, Figure 9 also shows that computation time is robust to the number of points of the two clouds. However, the voxel cell size, which is also interpreted as scale of detail, effects the computation time and quality of results. If the voxel cell size is smaller, which also means the results get more accurate, the time expense will be more.

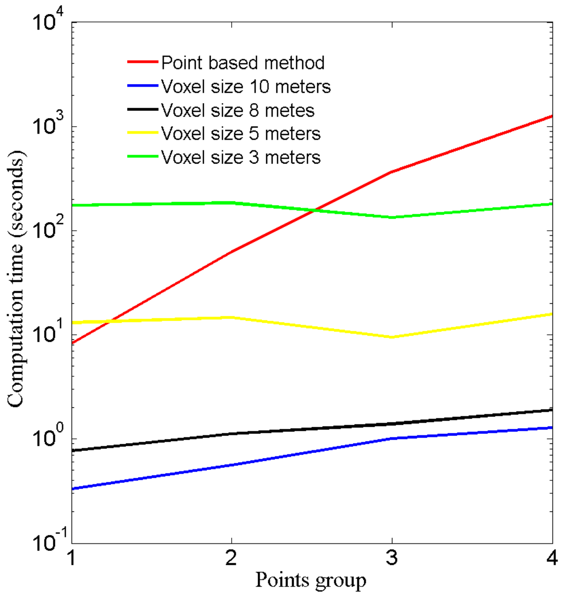

Figure 9: Computation time of the two methods

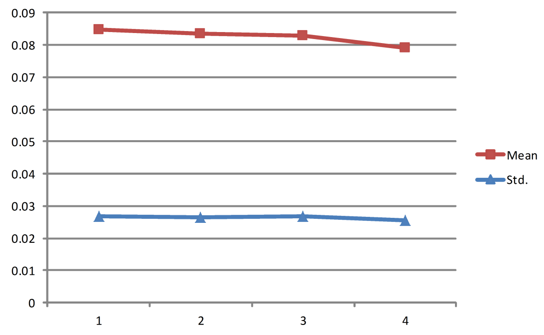

Figure 10: Mean and std.dev. of point based method

4 CONCLUSION AND RECOMMENDATION

This paper first introduces a voxel based scalable 3D point cloud intersection method. Then with the method the intersection of two overlapping point clouds is determined for different voxel cell sizes. The evaluation of the results is conducted by computing the distribution of the distances of points in the area of intersection. Parameters for regressing the distances with a $Log-Normal$ distribution shows that results gets more and more accurate when the voxel cell size gets smaller and smaller as expected. However, the computational expense will also get more an more. Afterwards the method is compared with a traditional point based method. The processing times are compared for different scenarios of down-sampled point cloud pairs.

Experimental results show that to meet the same quality of results, voxel based methods take much less time than point based method. And smaller voxel cell sizes will generate more accurate results, however, computational expenses will also get higher. Experimental results show that the quality of tradition point based method is related to the distance threshold and is generally better than that of the proposed method. However, the point based method is highly dependent on the number of points. While the voxel based method has more flexible choices on quality and computation time. Results on quality and computation time are balanced by the voxel cell size.

During processing and studying the presented methods, we also learned some experiences. Firstly, to more efficiently determine high quality area of intersection using this method, a relatively larger voxel cell size can be applied in the first run to get coarse results. Then based on these results, finer results can be obtained with a smaller voxel cell size. Secondly, the voxel cell resampling is based on uniform space sub-division and locations having no points are also sampled as empty cells. This introduces redundant memory space. As a solution, an octree data structure can be implemented to improve memory efficiency.

5 ACKNOWLEDGMENTS

The authors would like to thanks the Chinese Scholarship Council for the support of this research. Also thanks to Martin Kodde from Fugro for providing the MLS point cloud data sets. This research is partly funded by the IQmulus project (FP7-ICT-2011-318787).

REFERENCES

Antonarakis, A., Richards, K. and Brasington, J., 2008. Object-based land cover classification using airborne lidar. Remote Sensing of Environment 112(6), pp. 2988 – 2998.

Baltsavias, E. P., 1999a. A comparison between photogrammetry and laser scanning. ISPRS Journal of Photogrammetry and Remote Sensing 54(2-3), pp. 83–94.

Baltsavias, E. P., 1999b. Airborne laser scanning: Existing systems and firms and other resources. ISPRS Journal of Photogrammetry and Remote Sensing 54(2-3), pp. 164–198.

Bater, C. W., Wulder, M. A., Coops, N. C., Nelson, R. F., Hilker, T. and Næ sset, E., 2011. Stability of sample-based scanning-LiDAR-derived vegetation metrics for forest monitoring. IEEE Transactions on Geoscience and Remote Sensing 49(6), pp. 2385–2392.

Beaulieu, N. C. and Rajwani, F., 2004. Highly accurate simple closed-form approximations to lognormal sum distributions and densities. IEEE Communications Letters 8(12), pp. 709–711.

Beaulieu, N. C. and Xie, Q., 2004. An optimal lognormal approximation to lognormal sum distributions. IEEE Transactions on Vehicular Technology 53(2), pp. 479–489.

Brasington, J., Vericat, D. and Rychkov, I., 2012. Modeling river bed morphology, roughness, and surface sedimentology using high resolution terrestrial laser scanning. Water Resources Research 48(11), pp. W11519.

Chazelle, B., 1989. An optimal algorithm for intersecting three-dimensional convex polyhedra. In: Foundations of Computer Science, 1989., 30th Annual Symposium on, Research Triangle Park, NC, pp. 586–591.

CloudCompare, C., 2015. Cloudcompaer software. `http://www.danielgm.net/cc/`, accessed April 13, 2015.

Dobkin, D. P. and Kirkpatrick, D. G., 1983. Fast detection of polyhedral intersection. Theoretical Computer Science 27(3), pp. 241 – 253. Special Issue Ninth International Colloquium on Automata, Languages and Programming (ICALP) Aarhus, Summer 1982.

Dobkin, D. P. and Kirkpatrick, D. G., 1985. A linear algorithm for determining the separation of convex polyhedra. Journal of Algorithms 6(3), pp. 381 – 392.

Dobkin, D. P. and Kirkpatrick, D. G., 1990. Determining the separation of preprocessed polyhedra - a unified approach. In: M. Paterson (ed.), Automata, Languages and Programming, Lecture Notes in Computer Science, Vol. 443, Springer Berlin Heidelberg, pp. 400–413.

Faro Scene, F., 2015. Faro scene software. `http://www.faro.com/en-us/products/faro-software/scene/overview`, accessed April 13, 2015.

Forest, J. and Salvi, J., 2002. A review of laser scanning three-dimensional digitisers. In: Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on, Vol. 1, pp. 73–78 vol.1.

Fugro, F., 2015. Fugro drive-map laser scanning system. `http://www.drive-map.eu/`, accessed April 13.

Gosliga, R., Lindenbergh, R. and Pfeifer, N., 2006. Deformation analysis of a bored tunnel by means of terrestrial laser scanning. In: Proceedings of the ISPRS Commission V Symposium on Image Engineering and Vision Metrology, Vol. XXXVI, Par, pp. 167–172.

Grandine, T. A. and Klein, F. W., 1997. A new approach to the surface intersection problem. Computer Aided Geometric Design 14(2), pp. 111 – 134.

Holopainen, M., Kankare, V., Vastaranta, M., Liang, X., Lin, Y., Vaaja, M., Yu, X., Hyyppä, J., Hyyppä, H., Kaartinen, H., Kukko, A., Tanhuanpää, T. and Alho, P., 2013. Tree mapping using airborne, terrestrial and mobile laser scanning - A case study in a heterogeneous urban forest. Urban Forestry and Urban Greening 12, pp. 546–553.

Hyyppä, J., 2011. State of the art in laser scanning. In: D. Fritsch (ed.), Photogrammetric week ' 11, Universität Stuttgart, Stuttgart, Germany, pp. 203–216.

Isenburg, M., 2015. Lastools software. `http://www.cs.unc.edu/~isenburg/lastools/`, accessed April 13, 2015.

Jaakkola, A., Hyyppä, J., Hyyppä, H. and Kukko, A., 2008. Retrieval algorithms for road surface modelling using laser-based mobile mapping. Sensors 8(9), pp. 5238.

Kumar, P., McElhinney, C. P., Lewis, P. and McCarthy, T., 2013. An automated algorithm for extracting road edges from terrestrial mobile LiDAR data. ISPRS Journal of Photogrammetry and Remote Sensing 85(0), pp. 44–55.

Leica Geosystems, L., 2015. Leica cyclone software. `http://hds.leica-geosystems.com/en/Leica-Cyclone_6515.htm`, accessed April 13,.

Manocha, D. and Canny, J. F., 1991. A new approach for surface intersection. International Journal of Computational Geometry and Applications 01(4), pp. 491–516.

Mount, D., 1992. Intersection detection and separators for simple polygons. In: Proceedings of the eighth annual symposium on Computational geometry, SCG '92, ACM, New York, USA, pp. 303 – 311.

Olsen, M. J., Kuester, F., Chang, B. J. and Hutchinson, T. C., 2010. Terrestrial Laser Scanning-Based Structural Damage Assessment. Journal of Computing in Civil Engineering 24(3), pp. 264–272.

Pan, J., Chitta, S. and Manocha, D., 2012. FCL: A general purpose library for collision and proximity queries. In: Proceedings - IEEE International Conference on Robotics and Automation, Univ. of North Carolina, IEEE, Saint Paul, MN, pp. 3859–3866.

Pu, S., Rutzinger, M., Vosselman, G. and Elberink, S. O., 2011. Recognizing basic structures from mobile laser scanning data for road inventory studies. Journal of Photogrammetry and Remote Sensing 66(6, Supplement), pp. S28 – S39. Advances in LiDAR Data Processing and Applications.

Puttonen, E., Suomalainen, J., Hakala, T., Räikkönen, E., Kaartinen, H., Kaasalainen, S. and Litkey, P., 2010. Tree species classification from fused active hyperspectral reflectance and LIDAR measurements. Forest Ecology and Management 260(10), pp. 1843–1852.

Rönnholm, P., Honkavaara, E., Litkey, P., Hyyppä, H. and Hyyppä, J., 2007. Integration of laser scanning and photogrammetry. In: ISPRS Workshop on Laser Scanning, Vol. XXXV-Inumber 1, Espoo, Finland, pp. 355–362.

Shamos, M. I. and Hoey, D., 1976. Geometric intersection problems. In: Foundations of Computer Science, 1976., 17th Annual Symposium on, IEEE, Houston, TX, USA, pp. 208–215.

Sithole, G. and Vosselman, G., 2004. Experimental comparison of filter algorithms for bare-Earth extraction from airborne laser scanning point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 59(1-2), pp. 85–101.

van der Sande, C., Soudarissanane, S. and Khoshelham, K., 2010. Assessment of relative accuracy of AHN-2 laser scanning data using planar features. Sensors (Basel, Switzerland) 10(9), pp. 8198–8214.

Varela-González, M., González-Jorge, H., Riveiro, B. and Arias, P., 2013. Performance testing of LiDAR exploitation software. Computers and Geosciences 54(0), pp. 122–129.

Vosselman, G. and Maas, H., 2010. Airborne and Terrestrial Laser Scanning. Whittles Publishing.

Wang, J., González-Jorge, H., Lindenbergh, R., Arias-Sánchez, P. and Menenti, M., 2013. Automatic estimation of excavation volume from laser mobile mapping data for mountain road widening. Remote Sensing 5(9), pp. 4629–4651.

Wang, Y., 1996. Intersection of offsets of parametric surfaces. Computer Aided Geometric Design 13(5), pp. 453–465.

Wehr, A. and Lohr, U., 1999. Airborne laser scanningan introduction and overview. ISPRS Journal of Photogrammetry and Remote Sensing 54(2-3), pp. 68 – 82.

Yunfei, B., Guopingb, L. and Chunxiang, C., 2008. Classification of lidar point cloud and generation of dtm from lidar height and intensity data in forested area. Beijing, pp. 313–318.

Zhou, L. and Vosselman, G., 2012. Mapping curbstones in airborne and mobile laser scanning data. International Journal of Applied Earth Observation and Geoinformation 18(0), pp. 293–304.