www.cadanda.com
Computer-Aided Design AND APPLICATIONS™

A Rendering Method of Laser Scanned Point Clouds of Large Scale Environments by Adaptive Graphic Primitive Selection

Hiroaki Date[1], Takashi Maeno[2] and Satoshi Kanai[3]

[1]Hokkaido University, hdate@ssi.ist.hokudai.ac.jp
[2]Hokkaio University, t_maeno@sdm.ssi.ist.hokudai.ac.jp
[3]Hokkaido University, kanai@ssi.ist.hokudai.ac.jp

ABSTRACT

In this paper, a rendering method of the laser scanned point clouds of large scale environments is proposed for supporting an easy and intuitive understanding of the scanned environments. In this method, an adaptive primitives selection model and hierarchical point representation are used in the rendering of the scanned environment. Local geometry of the objects are estimated by principal component analysis, and the graphic primitives for points are adaptively created for effective rendering. View-dependent LOD using point hierarchy and an adaptive primitives selection model are also achieved for efficient rendering. Some rendering results for point clouds acquired from different scanning systems are shown and compared with other methods.

Keywords: laser scanning, point clouds, rendering, graphic primitives, LOD.

1. INTRODUCTION

Several types of long-range laser scanners allow us to easily acquire the point clouds of several large scale environments such as manufacturing plants, roads, buildings, bridges, urban areas and cities. Currently, terrestrial laser scanning (TLS), mobile laser scanning (MLS) and airborne laser scanning system (ALS) are often used in several fields, and scanned point clouds are used in wide applications such as measurement, mapping, modeling, forensic investigation, and several simulations.

Rendering the scanned point clouds is one of the basic operations in most applications. Rendering results of the point clouds provide much information of the scanned environment to the users. However, as shown in Fig. 1, it is often difficult to understand the scanned environments or objects by seeing the point rendering results caused by the gaps between the points. Splatting [9,10],[12] and surface generation techniques [1,2] can be used for improving the quality of rendering results of the point clouds data. However, they do not work well for the point clouds of large scale environments, because the point clouds have extremely non-uniform point densities, non-uniform spatial distributions, and they represent various kinds of objects with different scales and shape complexities.

In this paper, a rendering method of the laser scanned point clouds of large scale environments is proposed for supporting an easy and intuitive understanding of the scanned environments. The input of this method is the point clouds with colors, that is, each point has RGB values. The method is based on the adaptive selection of the graphic primitives, octree-based hierarchical points representation, and view-dependent LOD for realizing effective and efficient rendering of the laser scanned point clouds.

A basic idea of our method for effective rendering is to selectively use graphic primitives suitable for the object shape. For example, power lines should be rendered by lines, facades of buildings and road surfaces should be represented by planar primitives, and complicated volumetric objects such as trees should be rendered by polygons or meshes. In our method, principal component analysis (PCA) is used to estimate the local shape of point clouds, and points are classified into three types. According to the classification results, graphic primitives for rendering are adaptively generated.

Another basic idea of our method is to perform LOD rendering using several types of graphic primitives for efficient and effective rendering. Efficient rendering is important because the laser scanned data often includes millions to billions of

Computer-Aided Design & Applications, 11(6), 2014, 685–693, http://dx.doi.org/10.1080/16864360.2014.914386

Taylor & Francis

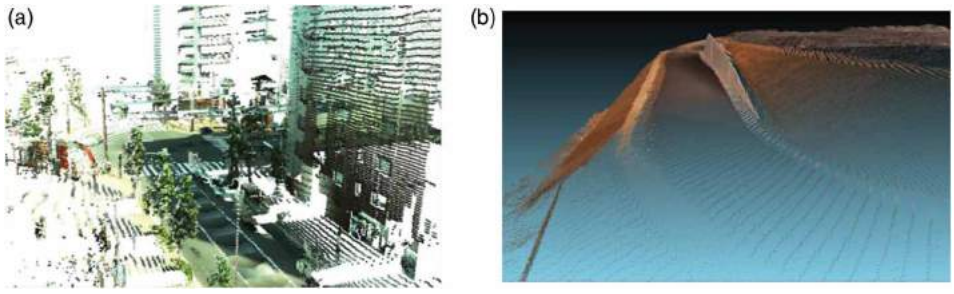

Fig. 1: Point clouds acquired by laser scanning of large scale environments: (a) Point clouds of urban environments from MMS, and (b) Point clouds of a construction site from TLS.

points. View-dependent LOD is often used in graphic applications [5,6] for efficient rendering of the complex scenes. In the rendering results based on view-dependent LOD, points, line segments, quadrilateral splats, and triangles appear simultaneously.

2. RELATED WORKS

Triangular meshes are often generated from scanned point clouds, and they are used in the rendering of scanned data. There are many methods on mesh surface reconstruction from point clouds [1,2]. Using existing methods, mesh generation may succeed for uniform and high density point clouds. However, the challenges for robust generation of correct surfaces still remain. Especially, it is difficult to generate correct surfaces from the point clouds of large scale environments, that have extremely non-uniform point density and include objects with different classes of shapes. Fig. 2(a) is an example of mesh generation from the point clouds of large scale environments. The mesh is created by free software (Mesh LAB [7]). In the result, incorrect surfaces between power lines appear. In our method, the local geometry of the objects points are estimated from the points, and meshes are generated only for complex volumetric objects such as trees. For linear and planar objects, line segments and plane splats are used for their representation.

Point-based rendering is useful to render the objects based on points [9,10],[12]. In the method, graphic primitives are defined at each point individually and used in rendering. For example, splatting defines finite disks or ellipses in object space and renders them in image-space by projecting them onto a screen. Some filters such as EWA (Elliptical Weighted Average) are used for blending, which provide natural rendering results without aliasing. Gap (hole)-free rendering is also achieved by controlling the parameters of the splats. However, there is a limit for representing several types of geometries using similar splats. Moreover, most techniques require normal at each point, but correct normal estimation from the point clouds of large scale environments is difficult. Fig. 2(b) is an example of splatting using ellipsoids. Many undesired splats can be seen on the wired objects and the portion where correct normal estimation is difficult. In our method, quadrilateral splats are used for representing planar objects, on which the normals are stably estimated. Nakagawa [8] developed an excellent point-based rendering method for LiDAR point cloud data. In the method, points are first projected onto the panorama space, such as spheres, taking into account the occlusions from the viewpoints, and the rendering images are created by interpolating the gaps between the points. The method is useful for getting rendering results of point clouds without gaps, but incorrect interpolation sometimes occurs after the large translations of the viewpoint. In our method, the pre-defined rendering model in the object space is used, therefore the quality of the results does not depend on the viewpoints.

LOD techniques have been used in several applications, not only rendering, but also CAD/CAM/CAE and graphic applications [4–6],[10,11]. They achieve efficient applications of the geometric models by appropriately reducing the number of elements of the models. Wand et al. [11] proposed a method for efficient rendering and editing of massive point

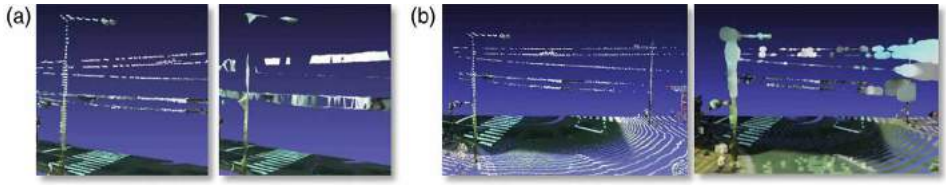

Fig. 2: Mesh generation and splatting for the point clouds of urban areas: (a) Point clouds and meshes, and (b) Point clouds and splats.

clouds using octree. Realtime editing for billions of points are realized. Our LOD method is based on Wand's methods, and the adaptive primitive selection model and its LOD are integrated into the octree-based LOD.

3. POINT CLOUD RENDERING USING ADAPTIVE PRIMITIVE SELECTION MODEL AND LOD

3.1. Overview of Our Method

Fig. 3 shows the overview of our method. In our method, two models are generated for rendering; An adaptive primitive selection model is used for rendering closer views, and a point hierarchy model is used in rendering distant views.

The adaptive primitive selection (APS) model is used for realizing effective rendering of the point clouds of the laser-scanned environments. The model is created by point classification, segmentation and adaptive graphic primitives creation (Fig. 3(a), A1, A2). The model consists of three types of primitives: straight line segments, quadrilateral splats, and triangular meshes. The line segments are used for representing thin linear objects such as power lines. Planar surfaces such as roads and walls of buildings are represented by the quadrilateral splats. Other complex volumetric objects such as trees and cars are represented by the triangular meshes. By using adaptively selected graphic primitives, the gaps between the points are filled in the object space, and the objects are appropriately represented in the rendering results.

Efficient rendering is also realized using LOD techniques. In our rendering, original or down sampled points are used in distant views of the scanned environments. Therefore, the point hierarchy is created by using the octree and random point sampling (Fig. 3(a), A3). For the LOD in the close views, simplified versions of the APS model are also created (Fig. 3(a), A2). In the rendering phase, view-dependent LOD, according to the distance from the viewpoint, is performed using an octree associated with point hierarchy and APS models (Fig. 3, A4). As shown in Fig. 3(b), down sampled points, original points, simplified APS models and original APS models are switched in the rendering phase according to the distance from the viewpoints.

3.2. Adaptive Primitives Selection Model Creation

3.2.1. *Point classification and segmentation using PCA and region growing*

For selecting graphic primitives suitable for object shape at each point, the local shape (point distributions) of point clouds are first recognized by principal component analysis (PCA). In this process,

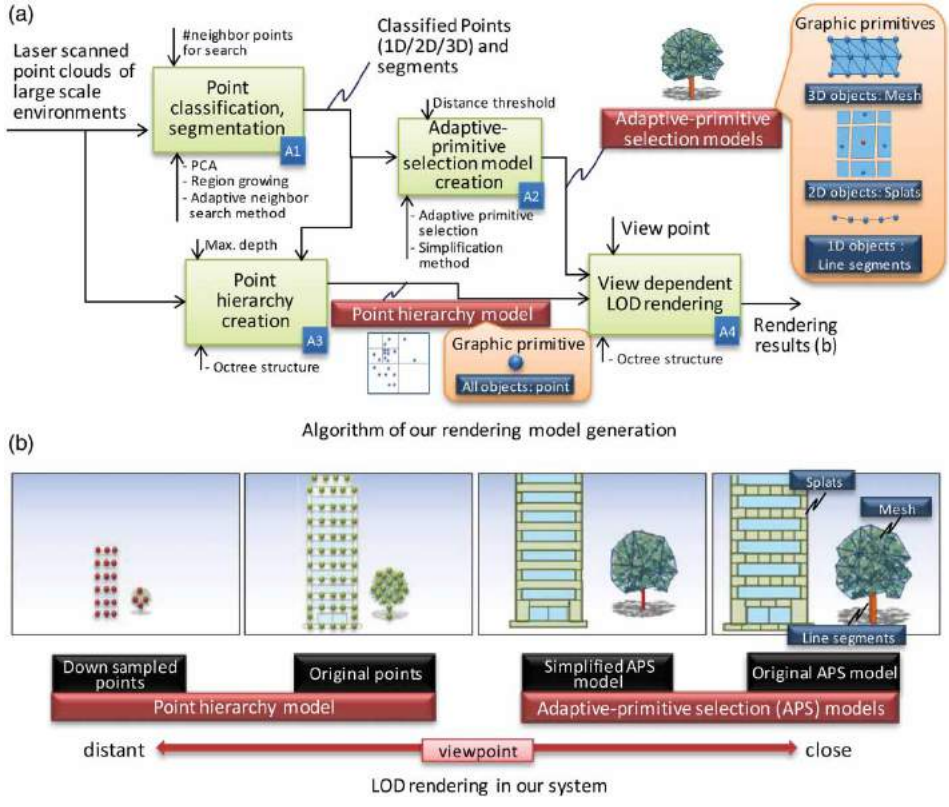

Fig. 3: The proposed rendering method of point clouds data of large scale environments.

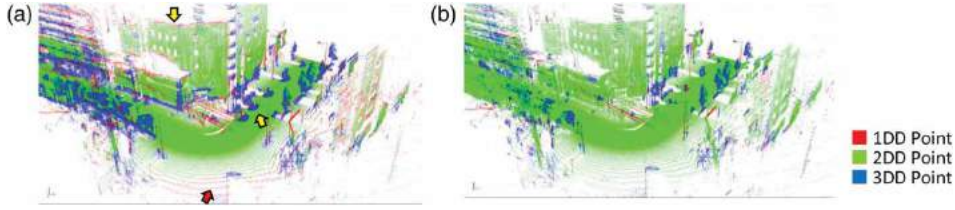

Fig. 4: Point classification: (a) Results of PCA, and (b) Results of region growing after PCA.

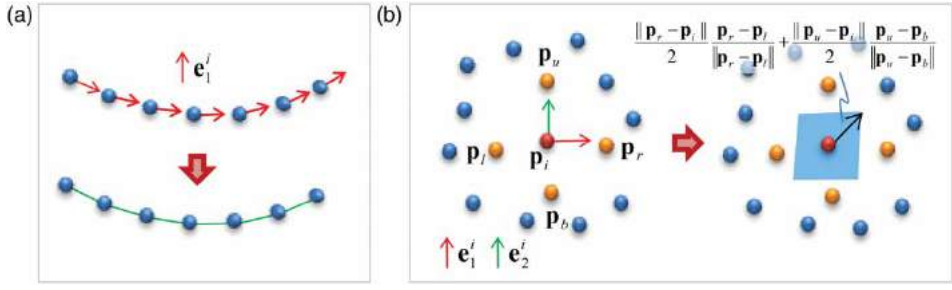

Fig. 5: Rendering primitive generation for 1DD and 2DD points: (a) Line segments for 1DD points, and (b) Quadrilateral splat for 2DD points.

variance-covariance matrix for neighbor points of each point i is created, and eigen analysis is then performed. As a result, three eigen values $\lambda_1^i, \lambda_2^i, \lambda_3^i (\lambda_1^i \geq \lambda_2^i \geq \lambda_3^i)$ and corresponding eigen vectors $\mathbf{e}_1^i, \mathbf{e}_2^i, \mathbf{e}_3^i$, are obtained. Because the magnitude of each eigen value is related to the variances of neighbor points of the point i along the corresponding eigen vector, if the distribution of the neighbor points of point i is liner (1D), λ_1^i becomes larger than the others ($\lambda_1^i \gg \lambda_2^i \approx \lambda_3^i$). If the distribution is planar (2D), λ_1^i and λ_2^i become larger than the other ($\lambda_1^i \approx \lambda_2^i \gg \lambda_3^i$). Therefore, the dimension of local point distribution can be evaluated using the dimensionality feature $d_i = \arg_{d\in\{1,2,3\}} \max(s_d^i)$ [3], where $s_1^i = \lambda_1^i - \lambda_2^i$, $s_2^i = \lambda_2^i - \lambda_3^i$, $s_3^i = \alpha\lambda_3^i$, α is a coefficient to recognize 3D distributions exaggeratingly, and $\alpha = 10$ was set experimentally. According to the d_i, each point i is classified into either 1DD (1 dimension distribution) point ($d_i = 1$), 2DD point ($d_i = 2$), and 3DD point ($d_i = 3$). For each 2DD point, $\mathbf{e}_3^i$ is stored as normal $\mathbf{n}_i$ of point i.

Fig. 4(a) shows a result of the point classification of the MLS point cloud. Most of the points are classified correctly, but misclassification can be seen near the boundary of the objects as indicated by yellow arrows in Fig. 4(a). Scan lines clearly appear around regions far from the scanner, and in such regions, points on a plane are classified as 1DD points as indicated by a red arrow in Fig. 4(a). To modify these misclassifications, reclassification by region growing is performed in our method. In this process, a point i which has a maximum s_2^i is selected as a seed of a region. Then, a plane is defined by the position $\mathbf{p}_i$ and normal $\mathbf{n}_i$ of the seed point i. The region grows from the seed by adding neighbor points into the region, which are lying on the plane. The points in the resulting regions are reclassified as 2DD points. Fig. 4(b) is the result of the reclassification of points using region growing. It can see that the points are classified more accurately.

After the point classification, segments are created. For 2DD points, points in the same region after region growing construct a segment. For 1DD and 3DD points, points with the same classification results are gathered by Euclidean distance-based clustering, and segments, each of which consist of the points with the same classification results, are created.

3.2.2. *Primitives generation*

For segments of 1DD points, sequences of the straight line segments are generated as the graphic primitives. As shown in Fig. 5(a), the sequences are created by connecting two neighboring points along the $\mathbf{e}_1^i$ obtained by the PCA. Triangular meshes are generated for segments of 3DD points. In our implementation, a ball pivoting algorithm [2] is used. A quadrilateral splat is created for each 2DD point so that it fills the gap between the neighbor points. For defining corner points of the splat, four neighbor points on a tangent plane are first found by searching for the closest points along $\mathbf{e}_1^i$ and $\mathbf{e}_2^i$. Then, as shown in Fig. 5(b), normalized difference vectors between neighbor points scaled by the distance between the point i and neighbor points are calculated and used for defining corner points of the splat.

3.2.3. *LOD model generation*

For LOD rendering, a simplified model consisting of smaller numbers of primitives is created. The sequence of line segments for 1DD points are

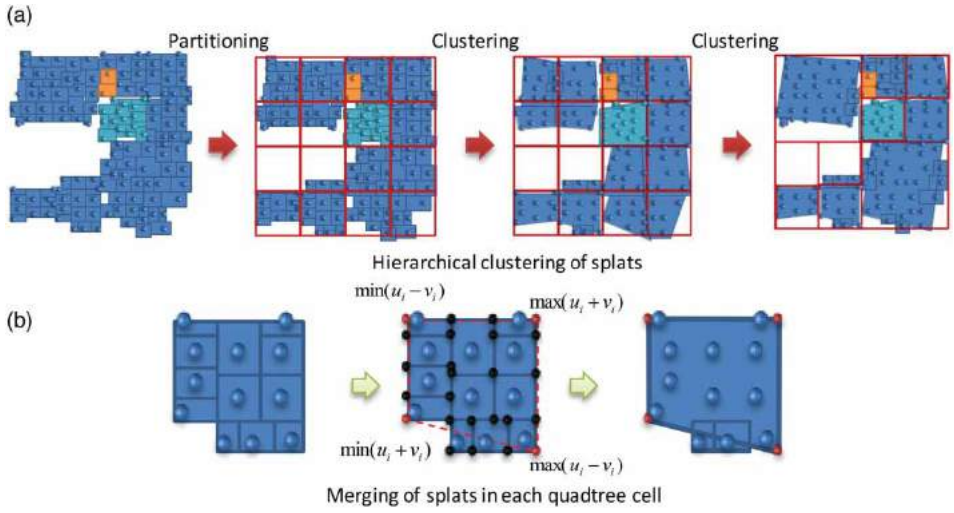

Fig. 6: LOD generation for quadrilateral splats.

simplified by merging two neighboring points. The triangular mesh for 3DD points is simplified by a traditional mesh simplification algorithm. In our implementation, vertex clustering [6] based on quadric error metric [4] is used.

A hierarchical clustering-like approach is used for simplifying a set of quadrilateral splats for 2DD points. As shown in Fig. 6(a), the 2DD points on a plane are first projected onto the plane. Then, a quadtree is created on the plane and the points are partitioned. Finally, coarse splats are created from leaf cells in the tree.

A coarse splat in a cell is created following procedure: points in the cell are first clustered if the point density in the cell is larger than a threshold τ_d and differences between the maximum and minimum RGB values of the points in the cell are smaller than a threshold τ_c. For clustered points, four corner points of the coarse splat are obtained as $\arg_i \min(u_i + v_i)$, $\arg_i \min(u_i - v_i)$, $\arg_i \max(u_i + v_i)$, $\arg_i \max(u_i - v_i)$, where u_i and v_i are the coordinates of the points of splats on a 2D plane, as shown in Fig. 6(b). Thus, the coarse splat is included in the convex hull of the clustered points. Finally, splats which the new coarse splat includes completely are removed. By iterating this process from leaf to root of the quadtree, LOD representation of splats are obtained.

3.2.4. *Adaptive neighbors search radius determination*

In the PCA, region growing, and splat generation, a neighbor search is required. kd-tree is often used for an efficient neighbors search and we also use it in our implementation. However, the numbers of reported points for a constant search range differ by locations in the point clouds with extremely non-uniform point densities. This causes inefficient data processing because the computational time of the search and the following processing often depends on the number of reported points. Fig. 7(a) shows a histogram of the numbers of reported neighbor points from a constant-range search for the point cloud acquired by MLS. The histogram shows that the numbers of reported points using a constant search range quite differ by location.

To solve this problem, we determine the search range adaptively by location according to the point density. First, a uniform grid covering the given point clouds is created, and the number of points n_k in the cell k is calculated. Then, search radius r_k for points in the cell k to obtain the N points is determined by Eq. (3.1),

$$\begin{cases} r_k = \frac{l}{\sqrt{\pi}}\sqrt{\frac{N}{n_k}}, & \left(\frac{l}{\sqrt{\pi}}\sqrt{\frac{N}{n_k}} < r_{\max}\right) \\ r_k = r_{\max} \quad , & \left(\frac{l}{\sqrt{\pi}}\sqrt{\frac{N}{n_k}} \geq r_{\max}\right) \end{cases} \tag{3.1}$$

where, l is the side length of the cell, $r_{\max}$ upper limit of search radius. Eq. (1) is derived from an assumption that the local point distribution is nearly planar and uniform in the cell.

r_k becomes quite large for some cells which include few points only near its boundary. To solve this problem, shifted uniform grid is created, and another r_k is calculated using the new cell. At each point, smaller r_k is used for the neighbor search. Fig. 7(b) shows a histogram of reported neighbor points using adaptive search range r_k.

3.3. Point Hierarchy Generation

For the distant view of the scanned environments, a hierarchy of the point clouds is created by adaptive space subdivision using the octree and down sampling in each octree cell similar to the method described in [11]. The cell of the root node is an axis-aligned bounding cube of the given point clouds. The

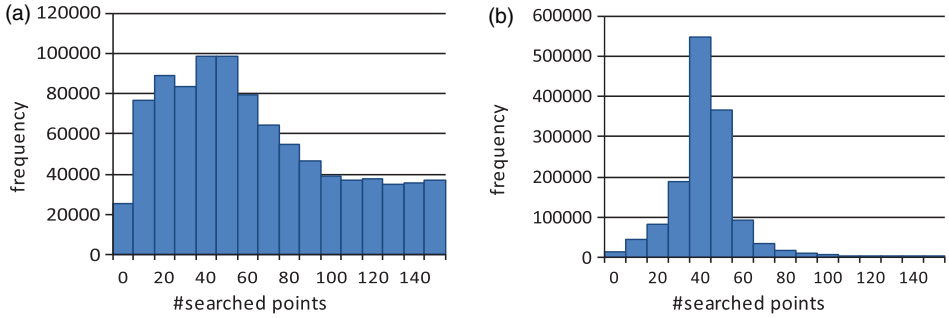

Fig. 7: Histograms of the numbers of searched neighbor points: (a) Fixed search radius (0.5 m), and (b) Adaptive search radius (The number of target points is 50).

cell is uniformly subdivided and sub cells of the child nodes are created. In each cell, down sampled points are stored. The uniform grid is generated in the cell, and a down sampled point is randomly selected from the points in each cell of the grid. The subdivision and down sampled points creation are repeated until the number of points in the cell becomes less than a certain number.

3.4. View Dependent LOD

Using the octree, view-dependent LOD is performed. To switch the LOD of adaptive primitives selection (APS) models using the octree, each segment in the APS model is associated with the cells which include the points of the segment.

When the viewpoint is far from the scene, down sampled points stored in the octree nodes are used in rendering. To determine the octree nodes for rendering, a depth-first search is done during rendering. The nodes satisfying the condition $s/d_c < \delta_1$ are found by the search, and down sampled points in the inner nodes or original points in the leaf nodes are used in rendering. Where s is the side length of the grid cell for down sampling, d_c is the distance from viewpoint to barycenter of the points of each octree node c, and δ_1 is a threshold.

As the viewpoint is moved closer and closer to the point clouds, the d_c becomes smaller. If the d_c of leaf node c used in rendering becomes smaller than a given threshold δ_2, the segments of the coarse APS model associated with the node c are rendered instead of points. Similarly, segments of the original APS model are rendered for closer nodes. As a result, original points, down sampled points and the segments with line segments, splats, and meshes with different resolutions appear in the resulting rendered scene simultaneously.

4. RESULTS AND EVALUATIONS

Our method was applied to some point clouds acquired by different scanning systems. Information of point clouds, the numbers of graphic primitives used in adaptive primitives selection models, and computation times are summarized in Tab. 1. The method was implemented on a PC with Intel Core i7 2.93GHz, 8GB RAM, and GeForce GTX 470 graphics board using OpenGL for rendering.

Fig. 8 shows the rendering results of the point clouds acquired by different laser scanning systems. From left to right, point clouds of urban area from a MLS system, a construction site from a TLS system, and urban area from other MLS system are shown. Upper figures are the rendering results using only points, and the bottoms are the results using our method. Using our method, the gaps between the points are filled and an intuitive and easy understanding of the scanned environments is realized.

Fig. 9 shows the comparison of the rendering results from different methods. In the results of splatting shown in Fig. 9(a), unnatural splats can be seen at the region where the normal estimation is difficult, edges are not represented appropriately, and the

		Adaptive primitive selection model			
Point clouds	#points	#line segments	#splats	#triangles in mesh	Processing time of model generation [sec]
Fig. 8(a)	1.6M	15.3K	1.28M	134.4K	385
Fig. 8(b)	1.4M	1.1K	1.3M	53.7K	179
Fig. 8(c)	1.0M	9.6K	859.0K	83.4K	328

Tab. 1: Point clouds and APS models.

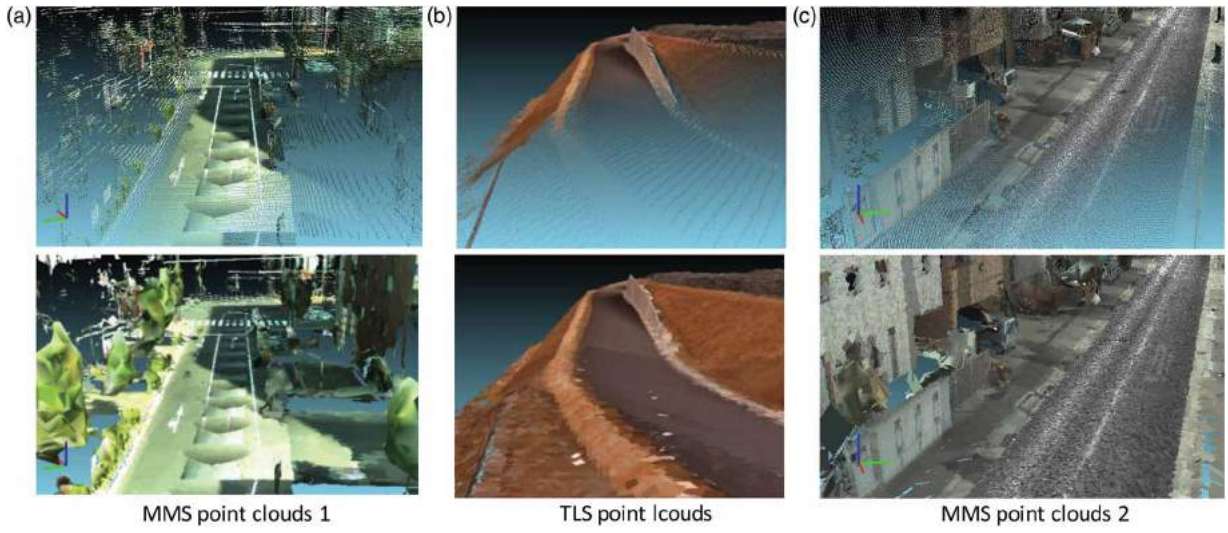

Fig. 8: Rendering results of point clouds acquired by different scanning systems (top: rendering results using points, bottom: results of our method).

Fig. 9: Rendering results using different methods: (a) Splatting, (b) Mesh, and (c) Our method.

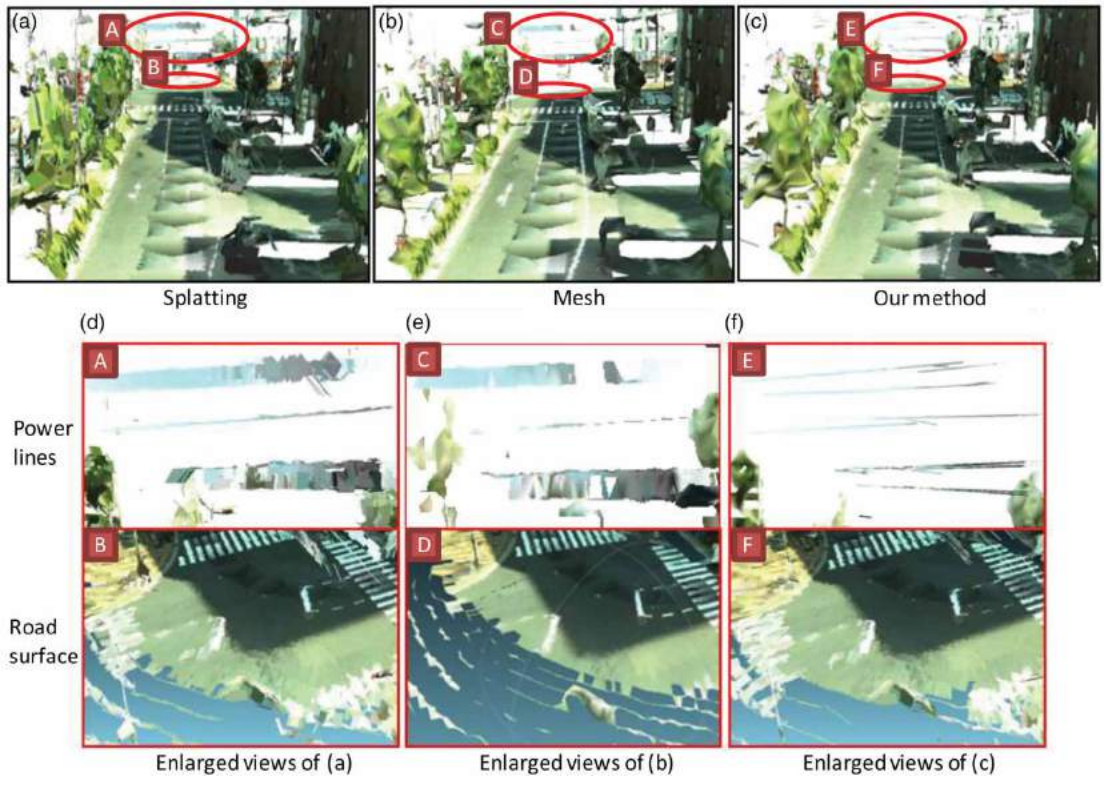

Fig. 10: Comparisons of the rendering results.

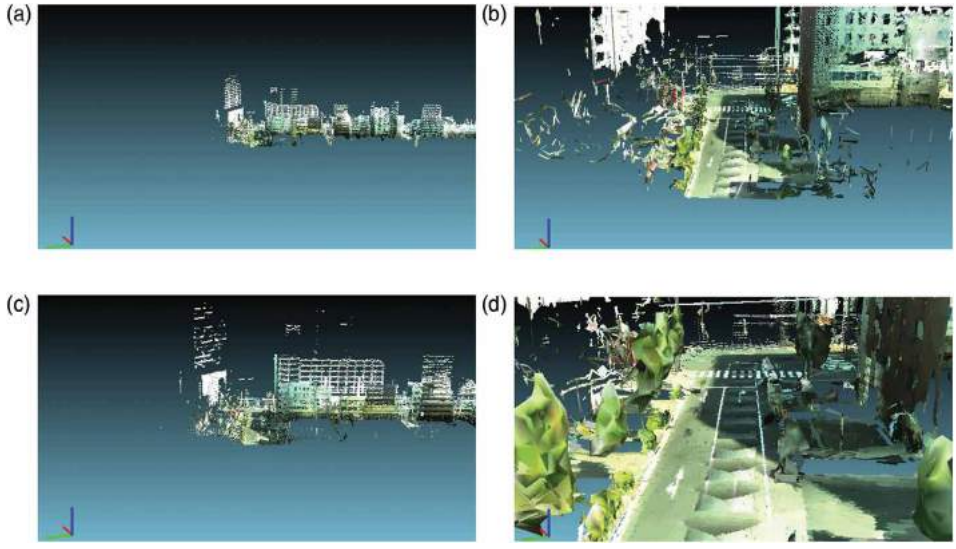

Fig. 11: Rendering results of MMS point clouds at different viewpoints.

gaps are still remaining on the facade of the building. Fig. 9(b) shows the rendering result using triangular meshes. Meshes are appropriately generated in high point density regions, however the mesh are not appropriately generated at the coarse region. Also, the gaps between the closer power lines are filled incorrectly. Fig. 9(c) shows our rendering results. Our splat generation method provides the rendering results without the gaps between points, and surfaces are created at coarse point regions with the help of the point classification by the PCA and splat generation by evaluating the distances to the neighbor points. The problems of Fig. 9(a) and Fig. 9(b) are solved, however, some undesired splats and line segments appeared. One of the reasons is the incorrect point classification and inconsideration of the boundary of the shape. More robust point classification is required for reducing such primitives.

Fig. 10 shows other comparison results. Fig. 10(d)-(f) are the enlarged views of Fig. 10(a)-(c). In the results of splatting and mesh shown in Fig. 10(d) and (e), undesired splats and meshes connecting with two electric wires are generated. In our results shown in Fig. 10(f), such primitives are not generated, because the line segments are used for rendering each line according to the results of point classification by evaluating local point sets. However, in the Fig. 10(f), some unnatural short line segments can be seen. A robust segmentation method is also required for improving the accuracy of primitive generation and getting more natural views.

Data	#points	#line segments	#splats	#triangles in mesh
Fig. 11(a)	126.3K	0	0	0
Fig. 11(b)	638.7K	1.2K	24.3K	1.9K
Fig. 11(c)	1,080.8K	2.3K	324.3K	24.2K
Fig. 11(d)	857.8K	3.5K	443.7K	42.1K

Tab. 2: The numbers of rendered graphic primitives.

Fig. 11 shows rendering results of a point cloud data at the different viewpoints. In distant views, the down-sampled points are rendered. As

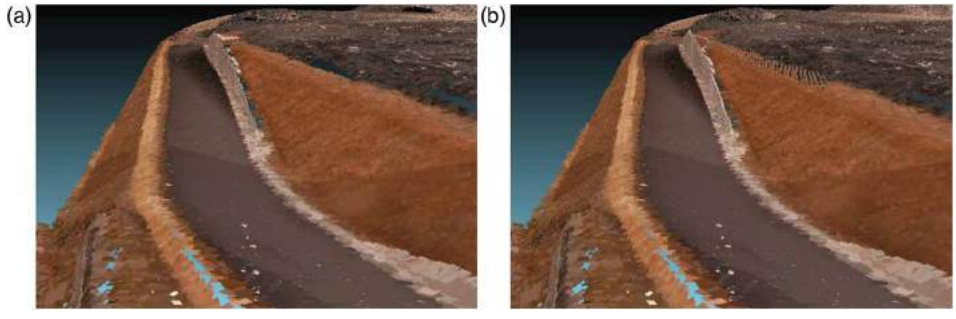

Fig. 12: Original adaptive primitive selection model and LOD model: (a) Original APS model, and (b) View-dependent LOD model.

the viewpoint becomes closer to the scene, original points, simplified and original APS model are gradually used in rendering. The numbers of graphic primitives are shown in Tab. 2. FPS of original APS model and all points were 1.6 and 9.0 respectively. Using LOD method, FPS were from 3 for the closest view to 60 for the distant view. Keeping FPS by limiting the maximum number of graphic primitives may be useful for maintaining the FPS. Fig. 12 shows rendering results of original APS model and LOD model for same data and viewpoints. The numbers of splats, line segments, and triangles are 1.3M, 1.2K, and 53.7K in the original model shown in Fig. 12(a), and 505.6K(38%), 159(13%), 11.3K(21%) in the model from view-dependent LOD shown in Fig. 12(b). Similar rendering result to the original one are obtained from simplified LOD model.

Processing times for rendering model generation are summarized in Tab. 1. For constant search radius (0.5 m), the processing times of PCA, region growing and splat generations were 72s, 160s, and 633s respectively (averages of the ones for three data shown in Fig. 8). On the other hand, by using adaptive search range determination method ($N = 50$) described in section 3.2.4, the processing times became 29s, 25s and 26s respectively, and processing times of processes with neighbor search were reduced to 10% in average.

5. CONCLUSIONS

In this paper, a rendering method of the point clouds for supporting easier and intuitive understanding of the laser scanned environments was proposed. In our method, an adaptive primitive selection model is created according to the results of point classification by PCA and region growing. The model consists of line segments for linear objects, splats for planar objects and mesh for the others. A simplified version of the model is also created and used in LOD for a closer view of the scanned environments. For distant views, point hierarchy based on the octree is used in LOD rendering. Some rendering results for point clouds acquired by different scanning systems are shown and it was confirmed that our method provides better rendering results for easier and intuitive understanding of the scanned environments compared with splatting, mesh based rendering and only point rendering. Future work includes blending the splats and aligning the splats depending on the shape and color boundaries for getting more natural views.

ACKNOWLEDGEMENTS

Point cloud data were provided from The Japan Society for Precision Engineering, Technical Committee for 3D scanning, recognition and modeling of large scale environments, TOPCON Corporation, Koishi Corporation, and Asia Air Survey Co. Ltd.

REFERENCES

[1] Amenta, N.; Bern, M.; Kamvysselis, M.: A New Voronoi-Based Surface Reconstruction Algorithm, Proc. SIGGRAPH '98, 1998, 415–421.

[2] Bernardini, F.; Mittleman, J.; Rushmeier, H.; Silva, C.; Taubin, G.: The Ball-Pivoting Algorithm for Surface Reconstruction, IEEE Transactions on Visualization and Computer Graphics, 5(4), 1999, 349–359.

[3] Demantke, J.; Mallet, C.; David, N., Vallet, B.: Dimensionality Based Scale Selection in 3D LiDAR Point Clouds, Proc. ISPRS Workshop Laser Scanning 2011, 2011.

[4] Garland, M.; Heckbert, P.S.: Surface Simplification Using Quadric Error Metrics, Proc. SIGGRAPH97, 1997, 209–216.

[5] Hoppe, H.: View-dependent refinement of progressive meshes, Proc. *SIGGRAPH'97*, 1997, 189–198.

[6] Luebke, D.; Reddy, M.; Cohen, J.D.; Varshney, A.: Level of Detail for 3D Graphics, Morgan Kaufmann, 2002.

[7] MeshLab., http://meshlab.sourceforge.net/.

[8] Nakagawa, M.: LiDAR VR Generation with Point-based Rendering, Proc. 28th Urban Data Management Symposium, 2011, 223–230.

[9] Pfister, H.; Zwicker, M. Baar, J. V.; Gross, M.: Surfels: Surface Elements as Rendering Primitives, Proc. SIGGRAPH 2000, 2000, 335–342.

[10] Rusinkiewicz, S.; Levoy, M.: QSplat: A Multiresolution Point Rendering System for Large Meshes, SIGGRAPH 2000, 2000, 343–352.

[11] Wand, M.; Berner, A.; Bokeloh, M. ; Fleck, A.; Hoffmann, M. ; Jenke, P.; Maier, B.; Staneker, D.; Schilling, A.: Interactive Editing of Large Point Clouds, Eurographics Symposium on Point-Based Graphics, 2007, 37–46.

[12] Zwicker, M.; Pfister, H.; Baar, J. V.; Gross, M.: Surface Splatting, Proc. SIGGRAPH 2001, 2001, 343–352.