An Integrated Processing Method for Multiple Large-scale Point-Clouds Captured from Different Viewpoints

Yousuke Kawauchi[1], Shin Usuki[2], Kenjiro T. Miura[3], Hiroshi Masuda[4] and Ichiro Tanaka[5]

[1]Shizuoka University, f0030015@ipc.shizuoka.ac.jp
[2]Shizuoka University, dsusuki@ipc.shizuoka.ac.jp
[3]Shizuoka University, tmkmiur@ipc.shizuoka.ac.jp
[4]University of Tokyo, masuda@sys.t.u-tokyo.ac.jp
[5]Tokyo Denki University, tanaka@cck.dendai.ac.jp

ABSTRACT

3D models of industrial plants are useful for simulating maintenance and renovation. We aim to reconstruct 3D models of components from large-scale point-clouds captured for industrial plants. When an engineering facility is measured by a laser scanner, the location of the scanner is restricted in most cases. We have such problems that the components of the engineering facility can be scanned only from one side and they might be partially occluded by other components. In order to solve the problems, we propose a new method which can process multiple large-scale point data captured from different viewpoints without merging them into a huge data which might not be able to be handled on a computer with a limited main memory size.

Keywords: large-scale point-clouds, laser scanner, data from different viewpoints.
DOI: 10.3722/cadaps.2011.519-530

1 INTRODUCTION

The recent progress on mid/long-range laser scanners has made it possible to capture large-scale point-clouds from a broad range of scenes. These laser scanners can be used for acquiring 3D shapes of large-scale artifacts, such as factories, power plants, heavy goods, buildings, transportation infrastructure, etc.

Large-scale artifacts require long-term maintenance, and large cost is spent for maintenance in their long lifecycle. Model-based simulation is very useful for reducing time and cost of maintenance tasks. It can effectively detect the interference of a new component with existing facilities during its setup process. However, old artifacts lack reliable drawings as well as 3D models in most cases, because they have been repeatedly renovated in their long lifecycles. In such cases, it is necessary to create new as-built solid models for the maintenance simulation.

Mid/long-range laser scanners are one of the most promising solutions. Two types of laser scanners are commonly used for measuring large artifacts in the field of surveying. One type is the time-of-flight scanner, which measures the round-trip travel time of the laser pulses. This type of scanner can measure in the range of a few hundred meters, but it takes relatively long time to measure many points that cover large artifacts. The other type is the phase-based laser scanner, which radiates

continuous modulated laser pulses and calculates distances using the phase difference between the emitted and received signals. The phase-based scanners can typically measure a range of 50~120 meters. In this paper, we call the former 'long-range' scanners, and the latter 'midrange'.

The reverse engineering of product shapes has been intensively studied in the CAD community. Most of the previous studies are based on the points precisely measured by triangular-based laser scanners. However, point clouds of large artifacts captured by mid/long-range scanners are quite different in the following aspects.

- The number of points is very large. In the case of the phase-based scanners, a single scan can capture 50M to 200M points. Naive algorithms easily exceed the limit of 32 bit PCs.
- Measuring precision is relatively low in comparison with point density. So generating surface mesh from the point cloud is not straight forward.
- Point data are very noisy and contain a lot of outliers. This is because the mid/long-range scanners cannot avoid the mixed pixel, at which a laser beam is reflected from multiple surfaces.
- It is difficult to obtain complete point sets of large artifacts, because it is very costly and time-consuming to construct stages for measuring artifacts from many directions. In addition, the intricate structure of engineering facilities restricts the locations for measurement.
- Large artifacts include a large number of components. Therefore, it is necessary to devide point-clouds into components.
- Many parts are overlapped, and they may be partly occluded.

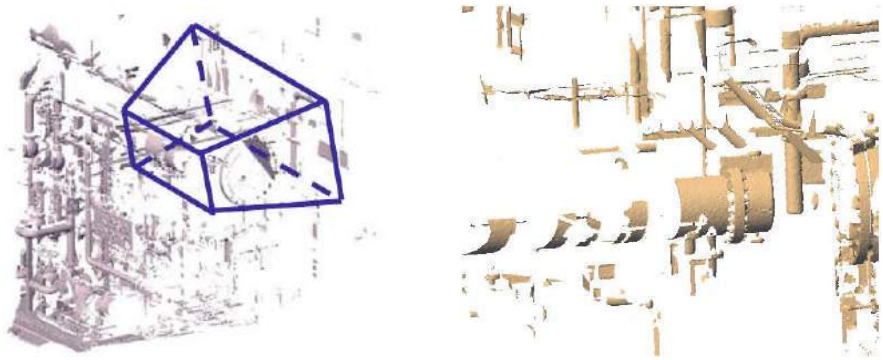
Masuda and Tanaka proposed a method for creating smooth mesh models from very noisy and large-scale point clouds [1]. This work has solved the first three problems described above. They introduced a robust smoothing operator based on the Lorentzian estimate and applied the operator to point data captured by a phase-based scanner. In addition, they extracted surface primitives and showed their method could be used for creating 3D mesh models of large facilities. They also discussed the remaining problems and proposed a new modeling method for creating 3D solid models based on images and mesh models generated from point-clouds [2]. Their contribution is to introduce image-based techniques for the purpose of compensating for missing portions in mesh models. In their method, while the user interacts with images, the computer estimates geometry using mesh models. This method is useful for generating solid models based on incomplete point data, especially when points are measured only from a limited number of viewpoints or many overlapping components are included.

Figure 1(a) shows a smooth mesh model generated by the method proposed in [1]. When components are carefully measured from appropriate positions, they can be converted to solid models by conventional reverse engineering techniques. However, when the object is located behind other components, its point-cloud is partially occluded and may be divided into fragments. In Fig. 1(b), cylindrical pipes are subdivided into fragments because of occlusions. In such cases, the user has to collect fragments and reconstruct the original shape to generate ideal shapes. However, it is often difficult for the user to recognize the original components from fragments. It is also difficult to develop a general algorithm that automatically reconstructs the original shape from fragments, although relatively small gaps can be filled by mesh processing [3,4].

The method in [2] combined image-based techniques and mesh processing techniques. Images and mesh models can be generated from a point-cloud, because mid/long-range laser scanners output the reflectance value as well as the coordinate at each point. The reflectance value represents the strength of reflected laser beam at each point, and is strongly influenced by the material properties of objects. Generally, the reflectance value becomes large for white objects, and low for black ones. So they make a black-and-white reflectance image by normalizing the logarithms of the reflectance value between 0 and 255.

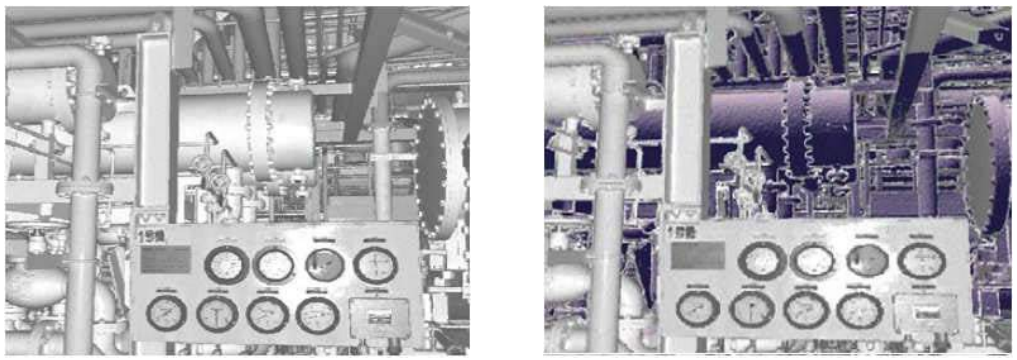
Figure 2(a) shows the reflectance image generated by the point-cloud of Fig. 1(b). The eye position is placed at the origin of the laser beams in this figure. We can obviously identify each component in Fig. 2(a), unlike Fig. 1(b). In Fig. 2(b), mesh models in Fig. 1(b) are projected on the reflected image. Fig. 2(b) shows that the image and the mesh models can be simultaneously used for modeling 3D shapes. This hybrid representation allows not only the user to intuitively work on 2D images, but also the computer to determine 3D geometry using mesh models.

Some of the previous works discuss point-clouds and images. Xu et al. [5] proposed a non-photorealistic rendering technique based on point-clouds. They also proposed a segmentation technique for a point-cloud using 2D images[6]. However, they did not discuss 3D shape modeling based on images and point-clouds. 3D reconstruction from stereo images is popular in computer vision [6], but Masuda and Tanaka method is quite different from those methods. They use a single reflectance image and a mesh model for generating solid models.



(a)Mesh model and a view volume. (b)Close-up disconnected mesh models in the view volume

Fig. 1: Disconnected mesh models generated from point-cloud.



(a) Reflectance image. (b) Reflectance image overlapped with mesh models.

Fig. 2: Reflectance image of point-cloud.

The interface proposed in [1] and [2] is excellent, but their system could not handle multiple point data captured from different viewpoints. If we merge these multiple point data after registration, we will have the following problems. We cannot use their interface anymore because it assumes one to one correspondence between a pixel in the image and a 3D measured point. If we merge two point data, this correspondence cannot be maintained. The other problem is that the data after merger might become too large to be handled by a PC with a limited memory size. Therefore we develop a new method using their interface which can process multiple large-scale point data captured from different viewpoints without merging them into a huge point-cloud.

In the following of this paper, we describe the framework of the modeling system proposed by Masuda and Tanaka [1,2] in Section 2 since our method is an extension of their methodology. We discuss the system overview and problem in Section 3, our proposed method in Section 4, and in Section 5. Finally, we conclude and mention future work in Section 6.

2 FRAMEWORK OF MODELING SYSTEM PROPOSED BY MASUDA AND TANAKA

Masuda and Tanaka proposed a new solid modeling methodology based on a reflectance image and mesh models. A lot of point-clouds are usually captured in surveying large artifacts, and each point-cloud contains 30~50 M points. Since the total number of points often reaches 109~1010 points, it is very time-consuming to process all point-clouds. Therefore, we developed an off-line point-processing system and an online 3D modeling system separately. In the off-line process, the system converts each point-cloud to a mesh model and a reflectance image. This phase is done by batch process. Then the user interactively generates solid models using the online modeling system.

In the off-line phase, noisy point data are smoothed using non-linear robust estimates [1], and then a large-scale mesh model is generated by using streaming Delaunay triangulation [8], which generates a very large mesh model represented in the streaming format [9]. However, the streaming mesh format is not suitable for interactively referring to the user-specified region in the large mesh model. Therefore, we introduce the grid mesh format to handle a large-scale mesh model in an out-of-core manner. We will discuss the grid mesh format in Section 4.

In the online process, the user interactively creates solid models using the reflectance image and the mesh model. Fig. 3 shows the user interface of our modeling system. Fig. 3(a) is a reflectance image. Fig. 3(b) shows a modeling view in which the user generates solid models. In our modeling system, the user interacts with the reflectance images for 3D modeling.

Solid models can be displayed in different views. In Fig. 3(b), solid models are drawn on the perspective image. The user can interactively generate, modify, and move 3D models on this view. Fig. 3(c) displays solid models with the reflectance image. We call this view as a spherical view, because the reflectance image is displayed as a texture on a sphere. Fig. 3(d) displays solid models using a Cartesian coordinate system. The user can switch these four views to confirm and generate solid models.

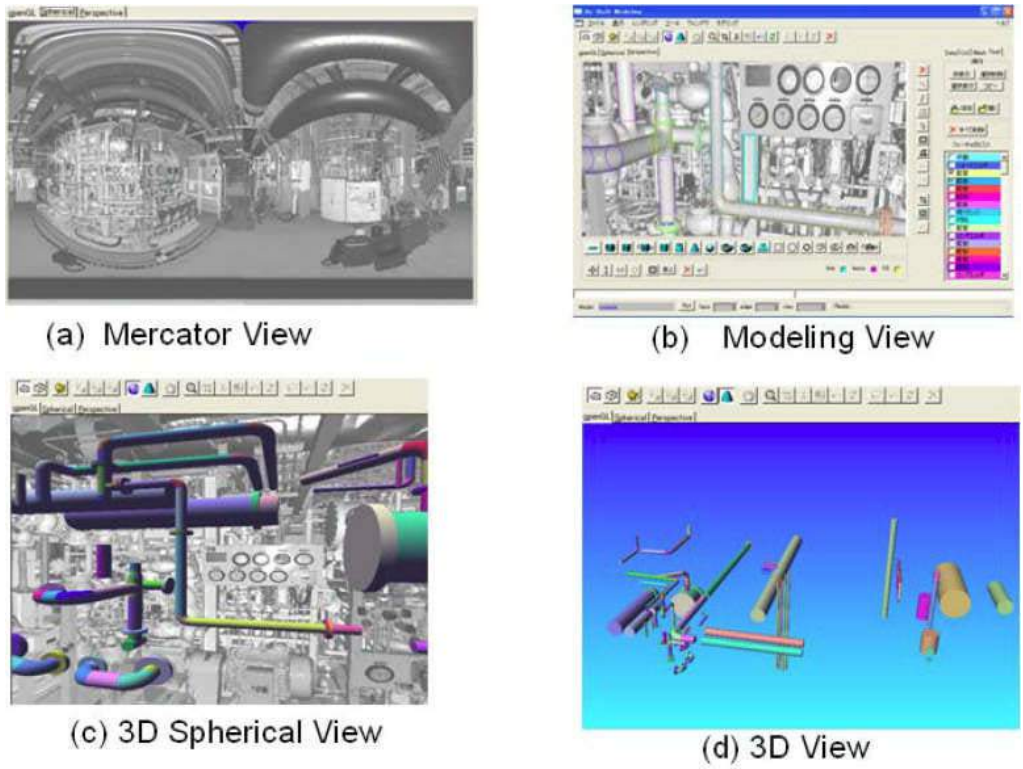


Fig. 3: User interface of our modeling system.

3 GENERATION OF REFRECTANCE IMAGE

3.1 Generation of Mercator Image

The directions of laser beams of a mid/long-range scanner are controlled by the azimuth angle θ and the zenith angle ϕ, as shown in Fig. 4. In principle, the coordinate of a point is determined by the distance r and the direction (θ, ϕ). Therefore, (x,y,z) coordinates can be converted to spherical coordinates (r, θ, ϕ), when the origin of the coordinate system is placed at the source of laser beams. This coordinate system is called the *scanner coordinate system*. Each scanned data set has its own scanner coordinate system.

After converting coordinates (x,y,z) to the scanner coordinate (r, θ, ϕ), the reflectance value of each point is mapped on the (θ, ϕ) of a unit sphere. Fig. 5(a) shows a spherical image. This spherical image can be converted to a rectangular image by the Mercator projection, as shown in Fig. 5(b). This rectangular image is called a Mercator image. A Mercator image consists of 8192×4096 pixels for 30M~50M points in our system.

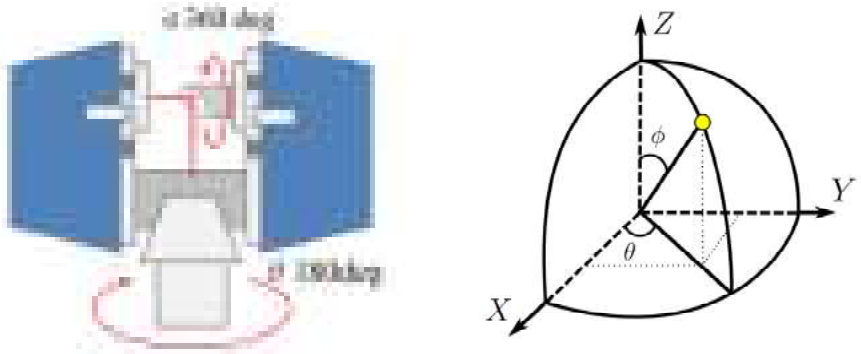
When multiple scanned datasets are merged by a registration tool, their transformation matrices are maintained. These translation matrices are used to calculate coordinates on the scanner coordinate system. Then a Mercator image can be generated for each scan by using the spherical coordinates.

Two different views are provided according to coordinate systems. In Fig. 3(c), 3D objects and a textured sphere are displayed on the scanner coordinate system. In Fig. 3(d), 3D objects from multiple scans are displayed using registered coordinates on the world coordinate system.

3.2 Generation of Perspective Image

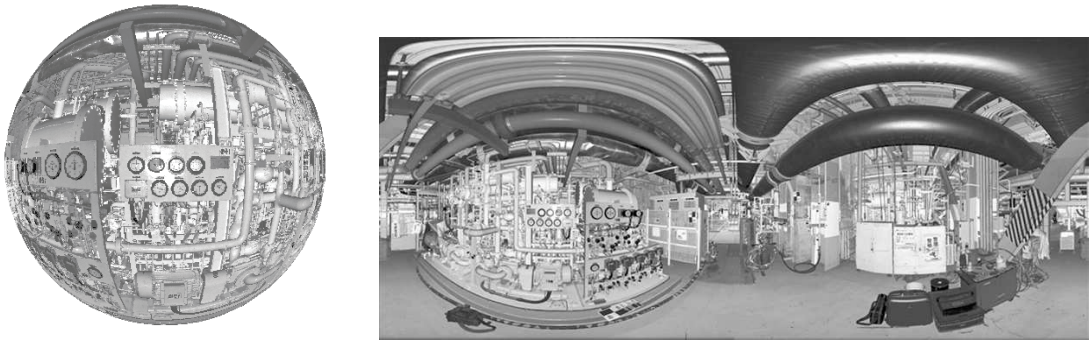
While the Mercator image is suitable for representing the whole point data of a single scan, it distorts straight lines to curved ones, as shown in Fig. 6(a). This distortion prevents the user to recognize each component and draw straight lines on the image. Therefore, we convert the Mercator image to perspective images, because the perspective projection always maps straight lines in 3D space to 2D straight lines, as shown in Fig. 6(b). However, a perspective image can cover only a limited range of a Mercator image. In our system, when the user selects a region of interest on a Mercator image, the selected region is projected onto the perspective screen.

Fig. 7 shows the relationship between 2D coordinate (θ, ϕ) on the Mercator image and 2D coordinate (i,j) on the perspective image. This figure shows that (i,j) and (θ, ϕ) can be mutually converted by calculating the intersection with the line from the origin. As each point (θ, ϕ) on the Mercator image is associated with the measured point in the same direction, we can associate (i,j) on the perspective image with 3D coordinate (x,y,z) on an object, because (r, θ, ϕ) and (x,y,z) can be mutually converted as $(x,y,z) = (r \cos\theta \sin\phi, r \sin\theta \sin\phi, r \cos\phi)$.



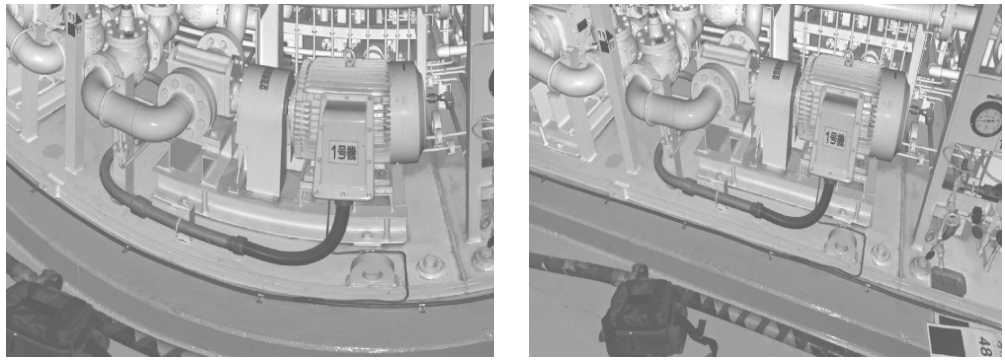
(a) Principle of mid/long-range scanners. (b) The azimuth angle and zenith angle

Fig. 4: The azimuth angle and zenith angle of mid/long-range laser scanners.



(a) Reflectance image projected onto a sphere. (b) Mercator image.

Fig. 5: Spherical image and Mercator image.



(a) Mercator image. (b) Perspective image.

Fig. 6: Generation of a perspective image from a Mercator image.

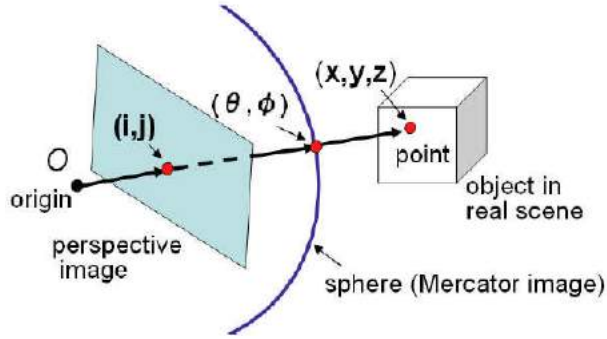


Fig. 7: Relationships among (i, j) and (θ , ϕ), and (x, y, z).

3.3 The Problems of the System Proposed by Masuda and Tanaka

By use of the system proposed by Masuda and Tanaka, the user can efficiently perform geometric modeling based on the image. However, since the system takes advantages of the property that a 3D point data can be mapped onto an 2D image data, if plural point-clouds captured from different viewpoints are combined, the points corresponding the 3D points are overlapped and one to one correspondence between a pixel in the image and a 3D point are broken. The methodology adopted by the system does not work anymore because the user cannot select a pixel to uniquely specify a 3D point. The other problem is that the merged data might become too large for, especially a lap-top pc with a limited size of the main memory at outside fields.

A point-cloud data captured from a single viewpoint usually does not contain a whole data of a component because of the complex alignment of the components and the data of a component are separated by the occlusions of the components. That makes very difficult to identify a single component which consists of several separated data and plural objects are independently recognized. Fig. 8 shows this type of mis-recognition and the system detects several cylinders which are parts of a single cylinder. It is desirable for a system to identify a single component. There are cases where this type of the problem can be solved by use of multiple data from different viewpoints since from a different viewpoint, the whole image of a component might be captured and the connectivity of other component can be recognized easily.

Fig. 8: Type of mis-recognition and the system detects several cylinders which are parts of a single cylinder.

In order to solve the problem mentioned above, we would like to use multiple point-clouds captured from different viewpoints without their merger of these data which will destroy the effectiveness of the interface provided by the system and hamper the efficiency of geometric modeling capability and propose a new method to handle these multiple point-clouds.

4 PROPOSED METHOD

Our method is introduced into the interface proposed by Masuda and Tanaka with no modification. For modeling a component with their system, it is necessary to show a point-cloud measured from a certain point on the interface and to select a surface type and a seed region for fitting a surface of the specified type to the point-cloud.

To solve a minimization problem by the non-linear least square method is necessary for fitting a surface. The objective function of the minimization problem is the sum of the distances between the surface of the component and the point-cloud. The seed region is an initial region including point data which come from the component in the point-cloud. Since the parameters of the specific surface are obtained by an iterative method, adequate initial values for the parameters are necessary. If the specified surface type is a cylinder, its axis vector, the coordinates of a point on the axis, and its radius are necessary. Masuda and Tanaka described a method of the initial guess of the parameters [1].

Our proposed method automatically determines from the seed region selected by the user in the main point-cloud other seed regions in other point-clouds measured from different viewpoints for the same component. It allows us to make a geometric model in an integrated manner from the seed regions in the plural point-clouds.

4.1 Outline of the Proposed Method

In this subsection, we explain the outline of our method. Here we call the main point-cloud using for the interface to the user D_A and another point data obtained from a different viewpoint D_B. We assume that these two point-cloud are registered each other and they are using the same coordinate system. In the explanation here, we use only one point-cloud as well as the main data, but it is possible to perform the same operations for other data if we have more that one.

We call the seed region of the main point-cloud D_A selected by the user S_A and the seed region of the other data D_B, S_B. The proposed method consists of the steps described below and automatically

determines S_B of D_B using S_A of D_A. Fig.8 shows the flow of the processes from selecting S_A to determining S_B. At first, the user specify a seed region in the point-cloud D_A (see Fig.9(a)).

1) As shown in Fig.9(b), by use of the seed region S_A, we estimate initial value of the surface parameters and construct a closed object B'_A made of the surface calculated by them. Then we generate a bounding box B_A of the surface.

2) 2. The following two substeps are repeated for each cell C_i (i=1,2,…,n) generated by subdividing the point-cloud D_B into grids in the θ - ϕ space (see Fig.9(c)).

   1) Define a bounding box B_{Bi} by obtaining the maximum and minimum values of the X, Y, and Z coordinates of the point data in C_i.

   2) Perform an intersection check between Ba and BBi and if the intersection occurs, we add the point data in B_{Bi} to a point data set $\boldsymbol{v}$, which is a candidate point data for the seed region S_B. Fig.9(c) shows both of the two cases, one is with intersection and the other is with no intersection.

3) Perform a test to check whether each points in $\boldsymbol{v}_{Bi}$ is inside or outside the estimated closed object B'_A and $\boldsymbol{v}_{Bi}$ and if the point in $\boldsymbol{v}_{Bi}$ is outside, it is removed from $\boldsymbol{v}_{Bi}$.

4) Define $\boldsymbol{v}_{Bi}$ as S_B as shown in Fig.9(e).

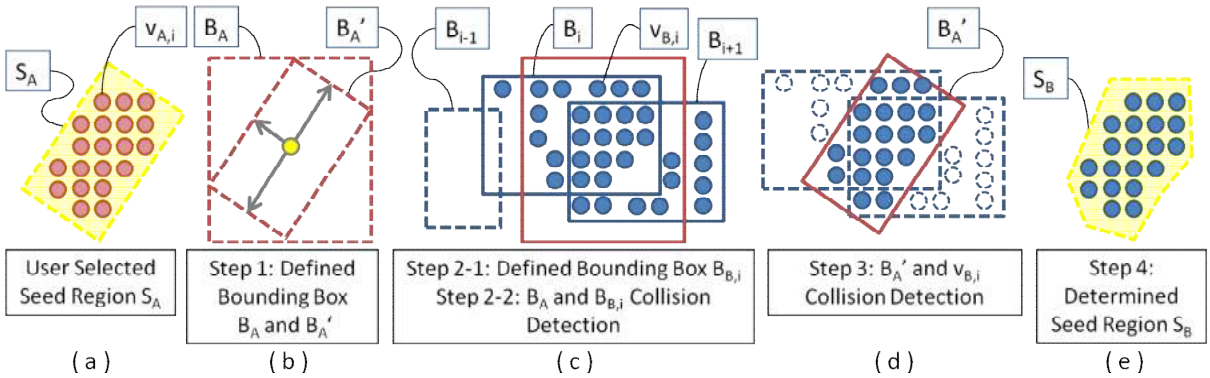


Fig. 9: Flow of the processes from selecting S_A to determining S_B.

The seed region determined by the above steps is enlarged by the region glowing method. In the region glowing, the point near the seed region is checked whether it lies or not on the surface generated by the initial estimated parameters and if it does, it is merged into the seed region. The region glowing is independently performed for each point-cloud and after glowing, the seed regions of all the point-clouds are merged and a surface is estimated by solving the non-linear least square method.

4.2 Example of Cylinder Modeling

Here we explain an example of the cylindrical surface modeling in our method. We use an AABB (axis-aligned bounding box) as the bounding box in Steps 1 and 2. For the intersection test, we adopt the intersection check between two AABBs. In Step 1, we assume that a point along the axis of the cylinder is given by $\boldsymbol{P}_0$, the unit vector whose direction is parallel to the axis $\boldsymbol{p}$ = (a, b, c), and the radius r, a set of points inside the seed region $\boldsymbol{v}_{Ai}$. The lengths along the axis l_+ and l_- are calculated by the following equations:

$$l_+ = \max\{(\boldsymbol{p}_0 - \boldsymbol{v}_{A,i}) \cdot \boldsymbol{p}\}$$

$$l_- = \min\{(\boldsymbol{p}_0 - \boldsymbol{v}_{A,i}) \cdot \boldsymbol{p}\}$$

Now we have obtained all the parameters to determine a closed object. Next we think about a rectangular parallelepiped circumscribing the closed object. Since we assume that the rectangular parallelepiped is aligned along the X, Y, and Z axes, to define the rectangular parallelepiped, it is enough to obtain the maximum and minimum values of the X, Y, and Z coordinates.

To calculate these maximum and minimum values, the axis $\boldsymbol{p}$ is assumed to be calculated by the following equation:

$$\boldsymbol{p} = M\begin{bmatrix}0\\0\\1\end{bmatrix}$$

Where matrix M is given by

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} = \begin{bmatrix} \cos\beta & \sin\alpha\sin\beta & \cos\alpha\sin\beta \\ 0 & \cos\alpha & -\sin\alpha \\ -\sin\beta & \sin\alpha\cos\beta & \cos\alpha\cos\beta \end{bmatrix}$$

In the above equation, α and β are rotation angles about the X and Y axes, respectively. The maximum and minimum values are given by the following equations.

$$x_{\max} = \begin{cases} l_{+}m_{13} + r\sqrt{m_{11}^2 + m_{12}^2} + a & (m_{13} \geq 0) \\ l_{-}m_{13} + r\sqrt{m_{11}^2 + m_{12}^2} + a & (m_{13} < 0) \end{cases} \quad x_{\min} = \begin{cases} l_{+}m_{13} - r\sqrt{m_{11}^2 + m_{12}^2} + a & (m_{13} \geq 0) \\ l_{-}m_{13} - r\sqrt{m_{11}^2 + m_{12}^2} + a & (m_{13} < 0) \end{cases}$$

$$y_{\max} = \begin{cases} l_{+}m_{23} + r\sqrt{m_{21}^2 + m_{22}^2} + b & (m_{23} \geq 0) \\ l_{-}m_{23} + r\sqrt{m_{21}^2 + m_{22}^2} + b & (m_{23} < 0) \end{cases} \quad y_{\min} = \begin{cases} l_{+}m_{23} - r\sqrt{m_{21}^2 + m_{22}^2} + b & (m_{23} \geq 0) \\ l_{-}m_{23} - r\sqrt{m_{21}^2 + m_{22}^2} + b & (m_{23} < 0) \end{cases}$$

$$z_{\max} = \begin{cases} l_{+}m_{33} + r\sqrt{m_{31}^2 + m_{32}^2} + c & (m_{33} \geq 0) \\ l_{-}m_{33} + r\sqrt{m_{31}^2 + m_{32}^2} + c & (m_{33} < 0) \end{cases} \quad z_{\min} = \begin{cases} l_{+}m_{33} - r\sqrt{m_{31}^2 + m_{32}^2} + c & (m_{33} \geq 0) \\ l_{-}m_{33} - r\sqrt{m_{31}^2 + m_{32}^2} + c & (m_{33} < 0) \end{cases}$$

The bounding box B_{Bi} in Step 2 is defined by the maximum and minimum values of the coordinates of the points in each Cell C_i obtained as a grid in the θ - ϕ space. The C_i's are constructed such that the number of the points in each C_i is about a few hundreds. The bounding boxes generated by the method mentioned above are used for the intersection check. Even in the case where the intersection is detected, each cell might include a set of points which should not be merged to the seed region. Hence we perform another intersection check using the bounding box B'_A defined by the initial parameters to detect points which should be merged to the seed region. In the cylinder case, we define the bounding box B'_A by $\boldsymbol{p}_0$, $\boldsymbol{p}$, $\boldsymbol{v}_{Ai}$, l_+ and l. Each point in the $\boldsymbol{v}_{Bi}$ is checked whether it is inside or outside of the B'_A and if it is classified to be inside, the point is regarded as a point of the seed region for the point-cloud D_B.

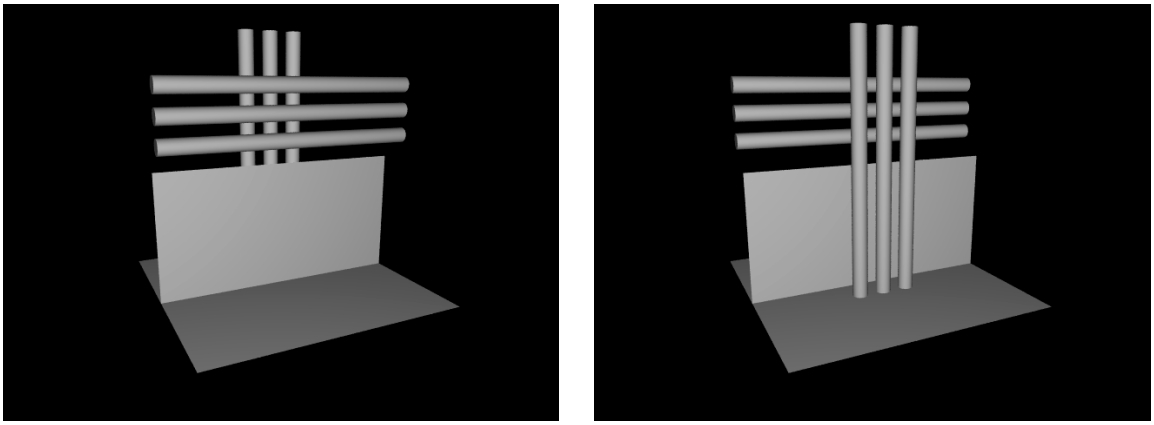
5 EXPERIMENT

We had experiments by making virtual measurement data to validate our proposed method. Fig. 9 shows a virtual environment which we used for the experiments. Fig. 10 shows Mercator images of the virtual point-clouds D_A and D_B from the viewpoints A and B calculated by a computer. The table 1 indicates parameters of these data.

Date	D_A	D_B
Number of point	200M	200M
Number of cell	20,000	20,000

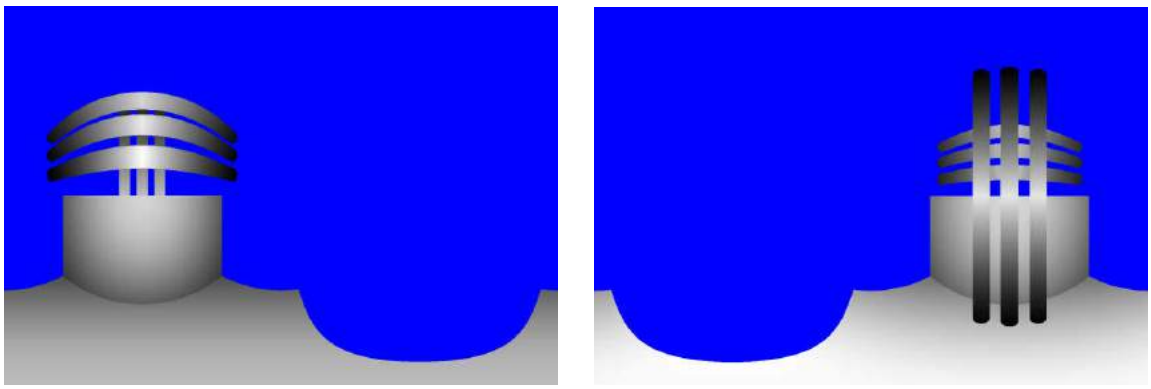
Tab. 1: Parameters of D_A and D_B.

As shown in Fig. 9 there are 6 cylinders in the environment. Three cylinders among them are not measured completely because of the existence of occluded objects. On the other hand, D_B measured from the viewpoint B has complete data of these cylinders.



(a) Virtual environment from A side. (b) Virtual environment from B side.

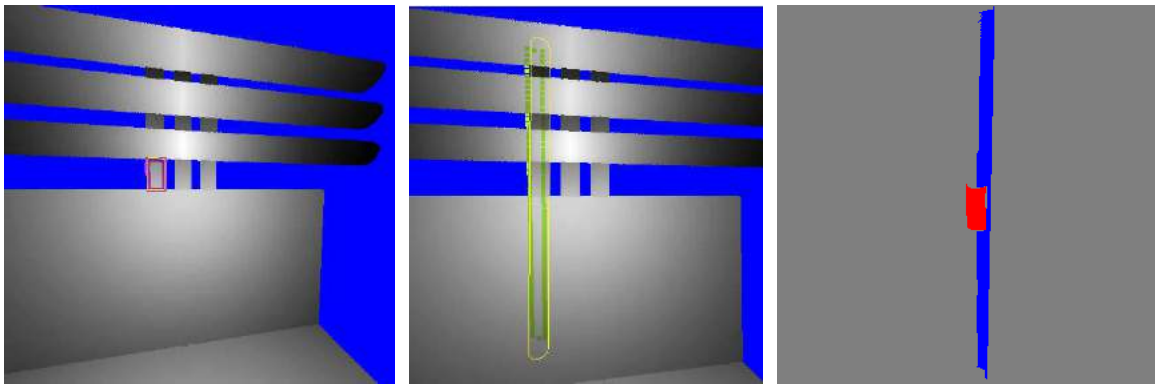
Fig. 9: Virtual environment.



(a) Mercator image of D_A from viewpoints A. (b) Mercator image of D_A from viewpoints B.

Fig. 10: Mercator images of the virtual point-clouds D_A and D_B from the viewpoints A and B.

By the original method proposed by Masuda and Tanaka, the models shown in Fig. 11(a) are obtained. Our method generates the models shown in Fig. 11(b). The original method cannot recognize the whole cylinders, but our method can generate whole cylinders as one components. The point set included in the seed region after growing are shown in Fig. 11(c).

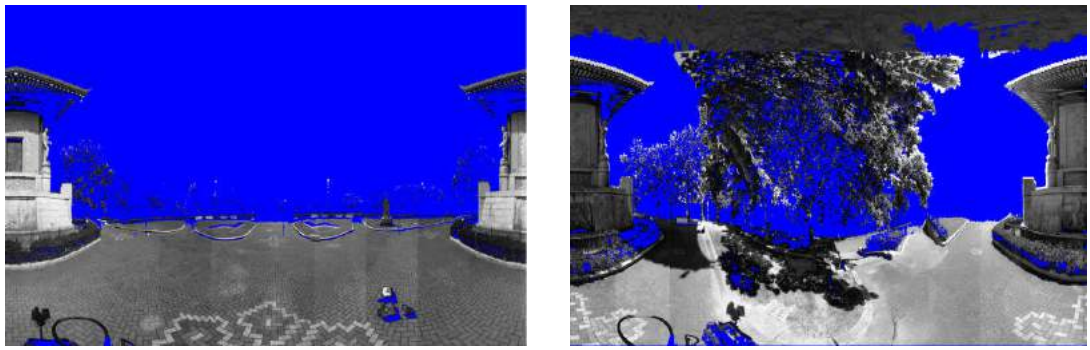


(a) Original method generates. (b) Our method generates. (c) Seed region after growing.

Fig. 11: Comparison of original method and our method.

The points painted in red color are from the point-cloud D_A and those in blue are from the point-cloud D_B. In our method, although it takes 17 seconds to perform the intersection tests for the point-cloud of 2 millions of points and 20,000 grids, if we generate bounding boxes and stored them in the look-up table as a pre-process, it takes only 0.7 seconds.

We have also applied our method to an actual measurement data and confirmed validity of our method. Fig. 12 shows the data which we have used for the experiments. Fig. 12 shows point-clouds of a temple measured from two different viewpoints and their Mercator images. We used the point-cloud shown in Fig. 12(a) as D_A and that shown in Fig. 12(b) as D_B. Fig. 13 shows the modeling result of a cylinder. It takes about 1 second to generate the cylindrical surface including the intersection tests.



(a) Mercator image of D_A from viewpoints A. (b) Mercator image of D_A from viewpoints B.

Fig. 12: Mercator images of the virtual point-clouds D_A and D_B from the viewpoints A and B.

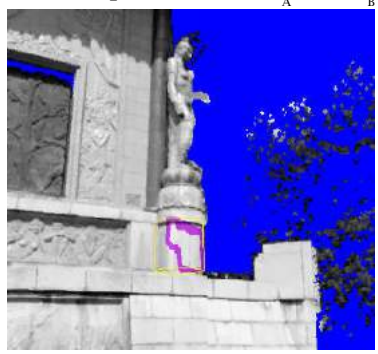


Fig. 13: Modeling result of a cylinder.

6 CONCLUTION AND FUTURE WORK

Our method proposed in the paper takes advantages of the previous method and with no modification of its interface we can reconstruct complete shapes of the components, which might not be obtained by use of one point-cloud measured from a specific viewpoint, by using plural point-clouds measured from several viewpoints without merging them. The processing time is comparable with the previous method because the calculation of the bounding boxes can be pre-processed.

We have validated our method by generating virtual measurement data and applying it to them and also we have applied it to an actual measurement data successfully. We are planning to apply our method to other more complicated actual measurement data, to validate it, and to improve it for practical use. In this paper we have dealt with only two set of the point-clouds. We think it is straightforward to use more than two point-clouds and we would like to apply our method for three or more point-clouds, that makes our method effectively.

REFERENCESERE

[1] Masuda, H.; Tanaka, I.: Extraction of Surface Primitives from Noisy Large-Scale Point-Clouds, Computer-Aided Design & Applications, 6(3), 2009, 47-57. DOI:10.3722/cadaps.2009.387-398

[2] Masuda, H.; Tanaka, I.: As-Built 3D Modeling of Large Facilities Based on Interactive Feature Editing, Computer-Aided Design & Applications, 7(3), 2010, 349-360. DOI:10.3722/cadaps.2010.349-360

[3] Sharf, A.; Alexa, M.; Cohen-Or, D.: Context-based Surface Completion, ACM Transaction of Graphics, 23(3), 2004, 875-884. DOI:10.1145/1015706.1015814

[4] Ju, T.: Robust Repair of Polygonal Models, ACM Transaction of Graphics, 23(3), 2004, 885-892. DOI:10.1145/1015706.1015815

[5] Xu, H.; Gossett, N.; Chen, B.: PointWorks: Abstraction and Rendering of Sparsely Scanned Outdoor Environments, Proc. of the 2004 Eurographics Symposium on Rendering, 2004, 45-52.

[6] Yuan, X.; Xu, H. ; Nguyen, M.; Shesh, A.; Chen, B.: Sketch-Based Segmentation of Scanned Outdoor Environment Models, Proc. of the 2nd Eurographics Workshop on Sketch-Based Interfaces and Modeling, 2005, 19-26.Debra, N. L.: Principles of Mechanical Design, Oxford University Press, New York, NY, 1990.

[7] Debevec, P.; Taylor, C.; Malik, J.: Modeling and Rendering Architecture from Photographs: A Hybrid Geometry- and Image-Based Approach, SIGGRAPH 96, 1996, 11-20. DOI:10.1145/237170.237191

[8] Isenburg, M.; Liu, Y.; Shewchuk, J.; Snoeyink, J.: Streaming Computation of Delaunay Triangulations, ACM Transactions on Graphics, 25(3), 2006, 1049-1056. DOI:10.1145/1179352.1141992

[9] Isenburg, M.; Lindstrom, P.: Streaming Meshes, Proc. of IEEE Visualization, 2005, 231-238. DOI:10.1109/VIS.2005.94

[10] Levin, D.: Mesh-Independent Surface Interpolation, Geometric Modeling for Scientific Visualization, 2003, 37-49.