www.cadanda.com
Computer-Aided Design AND APPLICATIONS

As-Built 3D Modeling of Large Facilities Based on Interactive Feature Editing

H. Masuda[1] and Ichiro Tanaka[2]

[1]The University of Tokyo, masuda@nakl.t.u-tokyo.ac.jp
[2]Tokyo Denki University, tanaka@cck.dendai.ac.jp

ABSTRACT

Phase-based and time-of-flight laser scanners can capture dense point-clouds of indoor and outdoor environment. However, such point-clouds usually lack the back-sides of objects and the portions occluded by other ones. In this paper, we propose a new modeling methodology which converts a point-cloud to a Mercator image and a mesh model, and enables reconstructing the missing portions interactively by using image-based techniques. The grid mesh format is proposed for quick access to large-scale mesh models. A prototype system is implemented to show the efficiency of our method.

Keywords: point-cloud, mesh processing, feature-base, image-based, out-of-core.
DOI: 10.3722/cadaps.2010.xxx-yyy

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 'mid-range'.

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.

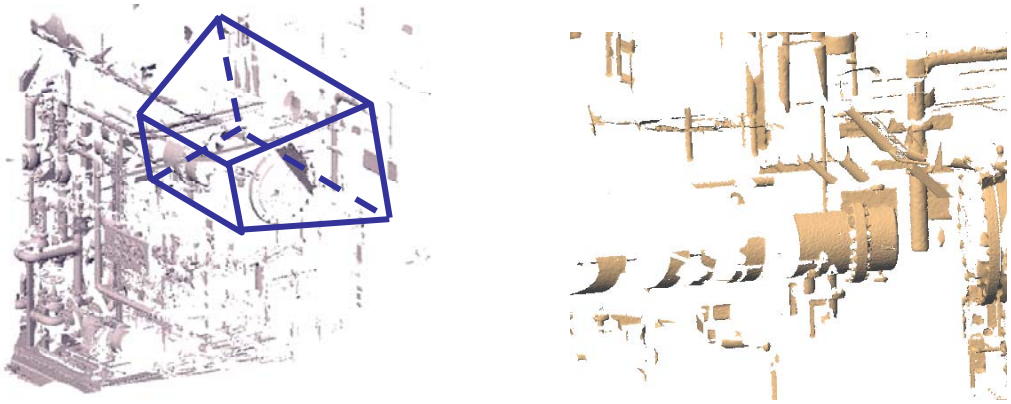
- The number of points is very large. In the case of the phase-based scanners, a single scan can capture 50M to 200M points. Naive algorithms easily exceed the limit of 32 bit PCs.
- Measuring precision is relatively low in comparison with point density. So generating surface mesh from the point cloud 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.

In our previous work, we 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. We 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, we extracted surface primitives and showed our method could be used for creating 3D mesh models of large facilities.

In this paper, we discuss the remaining problems and propose a new modeling method for creating 3D solid models based on images and mesh models generated from point-clouds. Our contribution is to introduce image-based techniques for the purpose of compensating for missing potions in mesh models. In our 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 view points or many overlapping components are included.

Fig. 1(a) shows a smooth mesh model generated by our previously proposed method. 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 by thinking of 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 [2,3].

Our method combines 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 we make a black-and-white reflectance image by normalizing the logarithms of the reflectance value between 0 and 255.



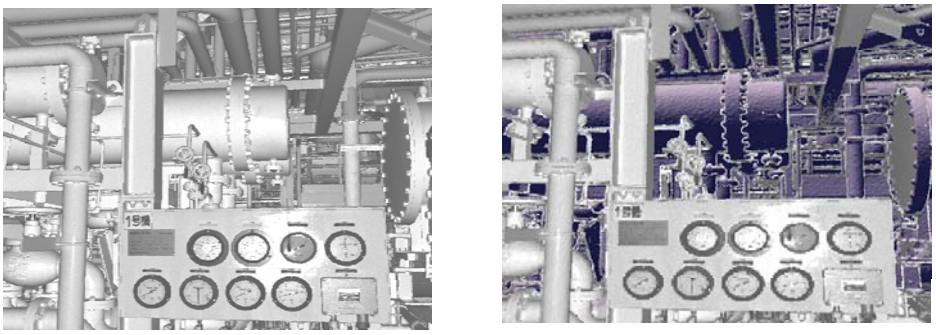
(a) Mesh model and a view volume. (b) Close-up of disconnected mesh models in the view volume

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

Fig. 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. [4] proposed a non-photorealistic rendering technique based on point-clouds. They also proposed a segmentation technique for a point-cloud using 2D images [5]. 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 our method is quite different from those methods. We use a single reflectance image and a mesh model for generating solid models.

In the following of this paper, we describe the framework of our modeling system in Section 2, the generation of reflectance images in Section 3, the surface extraction in Section 4, and feature modeling in Section 5. Finally, we conclude in Section 6.



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

Fig. 2: Reflectance image of point-cloud.

2 FRAMEWORK OF MODELING SYSTEM

We propose 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 10^9~10^{10} 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 [7], which generates a very large mesh model represented in the streaming format [8]. 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.

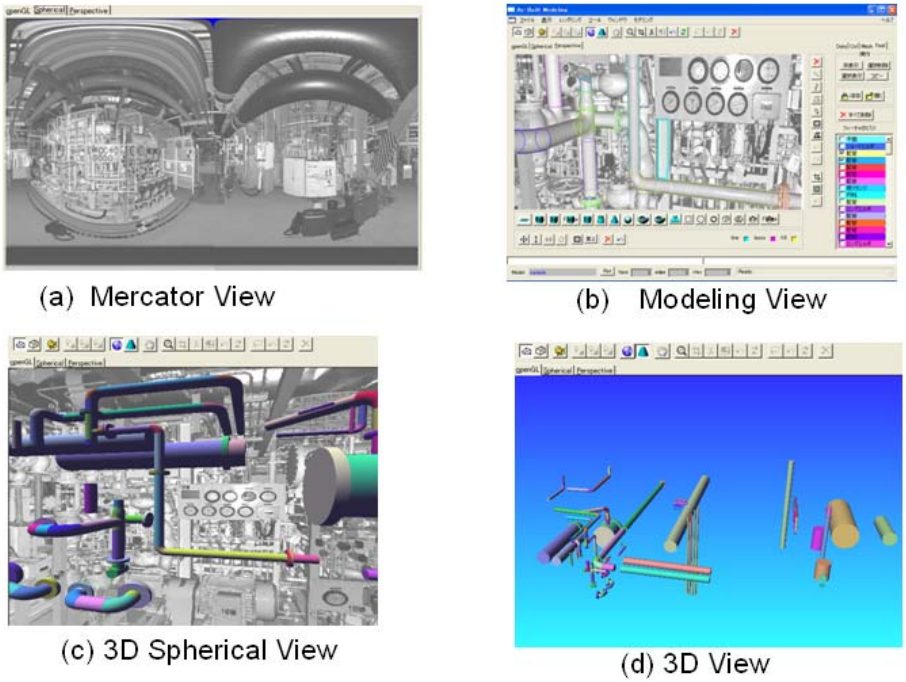


Fig. 3: User interface of our modeling system.

3 GENERATION OF REFLECTANCE 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. We call this coordinate system 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). We call this rectangular image as 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.

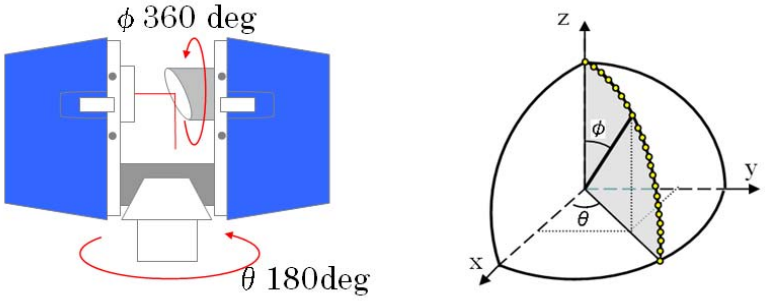
We provide two different views according to coordinate systems. In Fig. 3(c), 3D objects and a textured sphere are displayed on the scanner coordinate system. In Fig. 3(d), 3D objects from multiple scans are displayed using registered coordinates on the world coordinate system.

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,

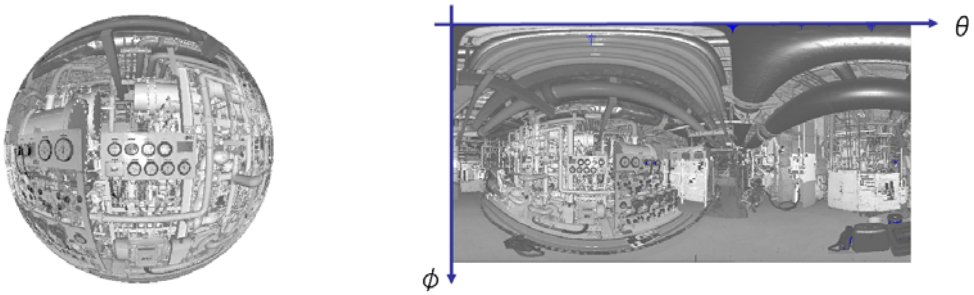
a perspective image can cover only a limited range of a Mercator image. In our system, when the user selects a region of interest on a Mercator image, the selected region is projected onto the perspective screen.

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 $(x,y,z)=(r\cos\theta\sin\phi, r\sin\theta\sin\phi, r\cos\phi)$.



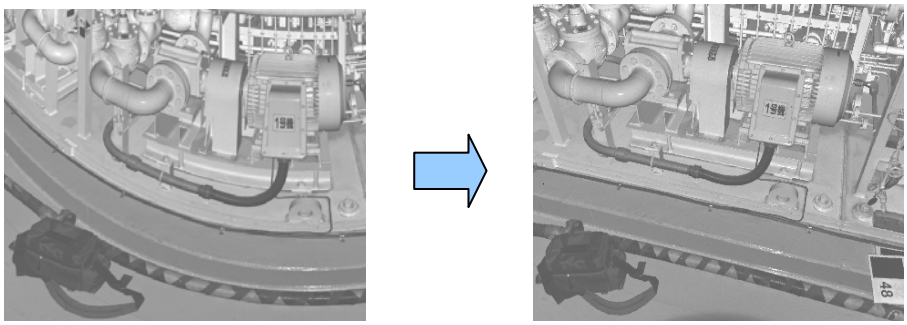
(a) Principle of mid/long-range scanners. (b) The azimuth angle and zenith angle

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.

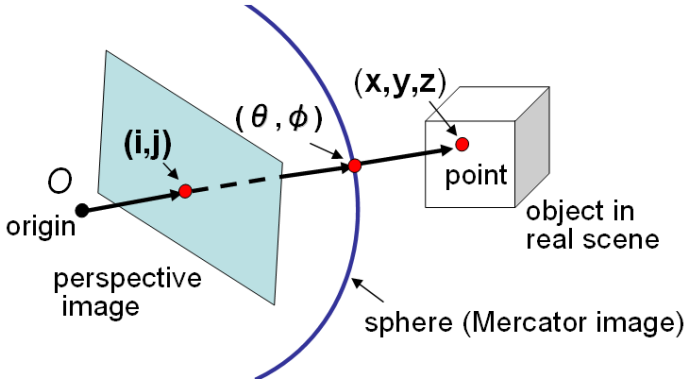


Fig. 7: Relationships among (i, j), (θ, ϕ), and (x, y, z).

4 EXTRACTION OF PRIMITIVE SURFACES

4.1 Smoothing of Point-Clouds

Mid/long-range scanners cannot avoid so-called mixed pixels, which are returned from two different surfaces. Since mixed pixels result in outliers, the mid/long range scanners produce quite a lot of outliers. Although the moving least-squares (MLS) projection [9] is a popular method for smoothing noisy point-clouds, MLS is sensitive to outliers. Therefore, we introduced the moving robust estimate to obtain smoothed point-clouds [1].

The moving robust estimate can be formalized as Eqn. (4.1), which maximizes the probability of the occurrence of residuals. Suppose a quadratic surface as $S(p;a) = 0$, where p is coordinate (x, y, z) and a is the parameters of the surface. Smoothing operators calculate smoothed points by projecting each coordinate on the quadratic surface.

In moving robust estimates, the parameters a of a quadratic surface can be calculated by the following simultaneous non-linear equations.

$$\sum_{i=0}^{n}\left(\frac{\partial S}{\partial a}(p_i)\cdot w(S(p_i))\cdot \chi(|\,\mathrm{p}_i - \mathrm{q}|)\right) = 0\,, \tag{4.1}$$

where $\{p_i\}$ are neighbor points to be fitted to a quadratic surface; n is the number of neighbor points; w is a weight function for residuals; χ is the Gaussian function; q is a reference point.

The weight function is determined by the supposed distribution of residuals. If the residuals are expected to follow the normal distribution, the weight function will be $w(r) = r$, which assigns large weights to outliers. In robust estimates, the weight functions should be determined so that outliers have small weights.

Various robust estimates can be defined according to the weight function $w(r)$. We adopted the Lorentzian distribution in our previous work, but we introduce the Tukey's bi-weight function in this paper. In our experiments, Tukey's bi-weight was 40% faster than the Lorentzian surface estimator.

Tukey's bi-weight function is defined as:

$$w_t(r) = \begin{cases} r\left(1 - r^2/c^2\right)^2 & |r| < c \\ 0 & |r| > c \end{cases} \tag{4.2}$$

Since this weight function assigns small values to outliers, the results of Eqn. (4.1) is robust to outliers.

Eqn. (4.1) requires the neighbor points of a point to calculate a quadratic surface. Since the mid/long-range scanners output the coordinates of points in a coherent order, as shown in Fig. 4-5, neighbor points can be obtained in a streaming manner using a relatively small buffer size. Therefore, we can apply our smoothing operator to a stream of coordinates in the order of measurement.

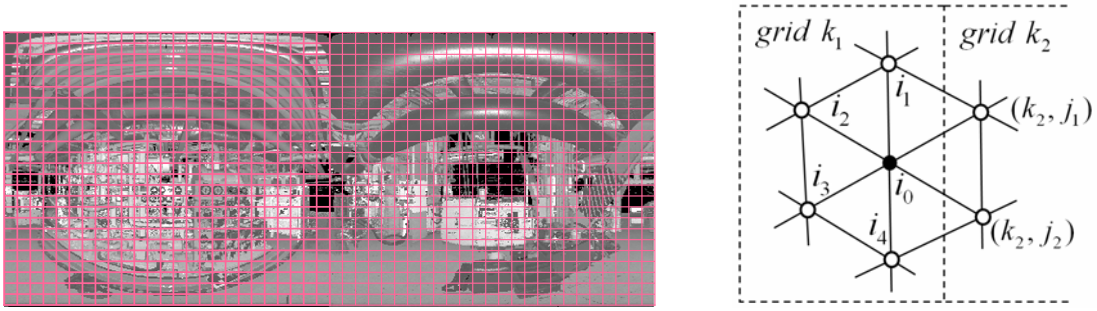
4.2 Grid Mesh Format

Since point data can be mapped on a Mercator image, they can be converted to a mesh model using 2D Delaunay triangulation. Isenburg, et al. proposed streaming Delaunay triangulation [7] and streamingly converted a very large point set to a triangular mesh model. Generated mesh models are very large. For example, the point-cloud in Fig. 2 consists of 50 million points which lead to 100 million triangles. Isenburg, et al. proposed the streaming mesh format [8] to represent very large mesh models, but this format is not suited for referring interactively to the user-specified region in a large mesh model. Therefore, we introduce the grid mesh format, which allows to accessing any potions of a large mesh model in an out-of-core manner.

Since every point can be mapped on a Mercator image, we can cluster points according to rectangle grids on the Mercator image, as shown in Fig. 8(a). The number of grids is determined so that each grid contains hundreds of points. We maintain a large mesh model on a hard disk, and load only a necessary region in the main memory. Since each grid can be independently loaded, any portion of a large mesh model can be quickly reconstructed.

Each grid maintains coordinates and adjacency relationships of vertices. When vertices of a mesh model are traversed, the adjacent vertices are investigated whether they are loaded on the main memory. If an adjacent vertex is not loaded yet, the grid that contains the vertex is retrieved from the hard disk.

Fig. 8(b) shows an example of vertex representation. We assign a unique number to each grid and represent the i th vertex in grid k as (k,i). To reduce the data size of a mesh model, we represent the connectivity in the same grid and the one between different grids in different ways. In Fig. 8(b), vertex i_0 has six adjacency vertices; four in the same grid and two in the neighbor grid. An adjacency vertex in the same grid is represented by its vertex-id, while an adjacency vertex out of the same grid is represented by the pair of its grid-id and vertex id. Though a regular triangular mesh model with 50M vertices requires about 1GB on a hard disk, the system can quickly load necessary grids into main memory. This method works well because necessary regions are relatively small for modeling components in facilities.



(a) Grids on the Mercator image (b) Adjacent relationship

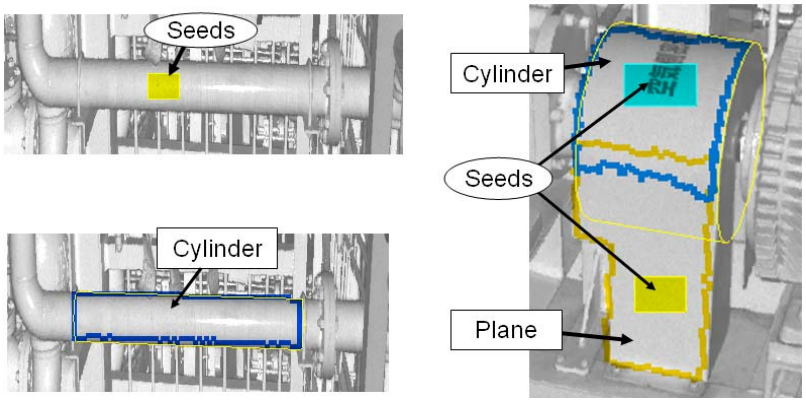
Fig. 8: Grid mesh format.

4.3 Extraction of Primitive Surfaces

In our system, the user first determines a base surface for creating 3D models. We call the underlying surfaces of points as base surfaces, which include planes, cylinders, cones, spheres, tori, generalized cylinders, and free-form surfaces.

Fig. 9 shows how to extract base surfaces by using the region growing method. When the user selects a seed region on a perspective image, the system detects the corresponding pixels on the Mercator image and loads appropriate grids in the mesh model. Then the system fits a surface to vertices in the seed region and searches the adjacent vertices that lie on the surface.

Bold lines in Fig. 9 show detected surface areas. A cylindrical region was detected in Fig. 9 (a), but the region growing was prevented by two holding fixtures. Such cases are very common in engineering facilities and they make it difficult to automatically reconstruct 3D shapes of facilities. In the following section, we introduce the modeling operations for generating 3D models of components using the base surfaces.



(a) Detection of a cylinder. (b) Detection of a plane and a cylinder.

Fig. 9: Region growing on a perspective image.

5 FEATURE-BASED MODELING

5.1 Modeling on Perspective Image

When an engineering facility is measured by laser scanners, it is often measured only from a limited number of view points because of the intricate structure of facilities. Therefore, large potions, such as back sides of components, may be missing and they may be partially occluded by other parts. The user has to create 3D models from these incomplete point-clouds.

In our modeling system, 3D models are determined by using 3D meshes and the user's strokes on the perspective image. Since our system provides perspective images of the scene, the user can easily recognize the types of components. In addition, the user can specify their locations and sizes by the mouse actions on the perspective image.

As shown in Fig. 7, the coordinate (x, y, z) of pixel (i, j) lies on a straight line from the origin. Therefore, if the base surface of an object is given, the corresponding coordinate can be uniquely determined by calculating the intersection between the base surface and the straight line. Since the resolution of an image is 8192×4096 pixels, we can determine 3D positions with considerable accuracy.

5.2 Industrial Standards

Many standard parts are commonly used in large facilities. They include pipes, structures, joints, flanges, valves, etc. The major dimensions of standard parts are specified by the industrial standards, such as ISO and JIS. We can use standard values for precisely calculating base surfaces when feature types are specified by the user. For example, the equation of a cylinder can be calculated by the non-linear minimization of an objective function with five variables. If some parameters (e.g. diameter) are given as standard values, unknown parameters are reduced and can be more stably calculated. Therefore, we first calculate base surfaces with no constraints and estimate standard values. Then the equation of the base surface with reduced number of variables is re-calculated using the estimated standard values. This method is useful for improving the accuracy of parameters, because point-clouds captured by mid/long-range scanners have relatively large noises. In our system, a catalog of representative standard parts is maintained. The user can select ANSI and JIS in the current implementation.

5.3 Feature Modeling

5.3.1 Feature Types

When the user recognizes a component in a perspective image, the user can specify the feature type and let the system calculate the base surface. Then the user creates 3D shapes on the image.

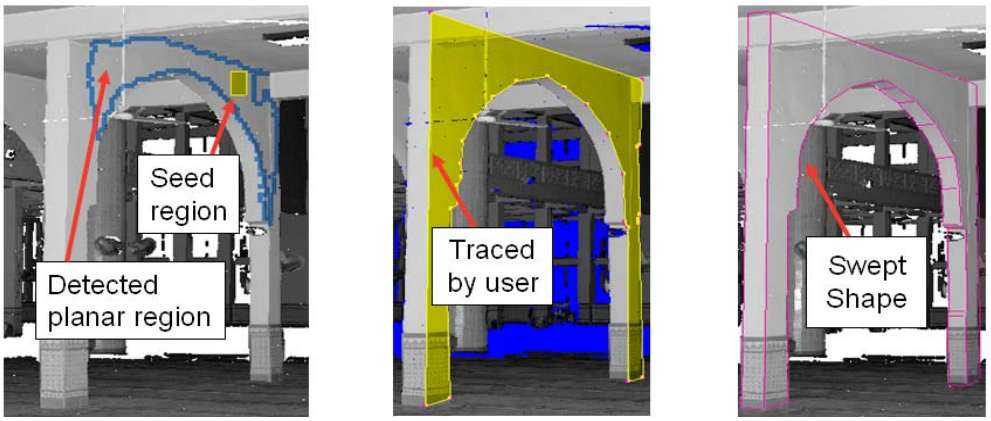
Feature classes are defined in an object-oriented manner. Therefore, we can easily add new feature types. In the current implementation, feature types for standard parts include pipes, bars, flanges, elbows, tees, reducers, balls, and structures. Their major dimensions are defined in the industrial standards. The user can also define

general shapes, such as cylinders, cones, spheres, tori, prisms, and planar polygons. It is also possible to apply Boolean operations to define more complicated 3D shapes.

5.3.2 Prism Feature

When a planar surface is extracted, the user can sketch 2D shapes on the surface. In Fig. 10(b,c), the user traces lines of a polygon on perspective image and sweeps the polygon face to specify the thickness. Then a right prism is constructed. Since it is difficult for the user to draw lines with right angles, we prepare a tool for drawing perpendicular lines.

This sketch-based method is convenient for generating faithful solid models using perspective images.



(a) Planar region is detected. (b) Polygon is traced. (c) The polygon is swept.

Fig. 10: Generating a right prism.

5.3.3 Cylindrical Features

Solid models of components can also be generated by editing base surfaces. Fig. 11 shows some examples.

In Fig. 11(a), two cylinders are extracted from a mesh model. Their lengths are not correctly calculated. If the user selects the “short elbow” as the feature type and specifies the two pipes, the lengths of the pipes and the shape of an elbow can be uniquely determined, because the radii of the torus is determined by the industrial standards.

In Fig. 11(b), it is difficult to precisely calculate the equation of the short cylinder of the flange. In this case, the user first detects the planar region and traces three points on the circular edge to define a circle face. Then the user specifies the height of the cylinder by dragging a mouse.

In Fig. 11(c), the user extracts two cylinders and specifies a tee feature. The shape of the tee is uniquely determined by the radii of two cylinders according to the industrial standards.

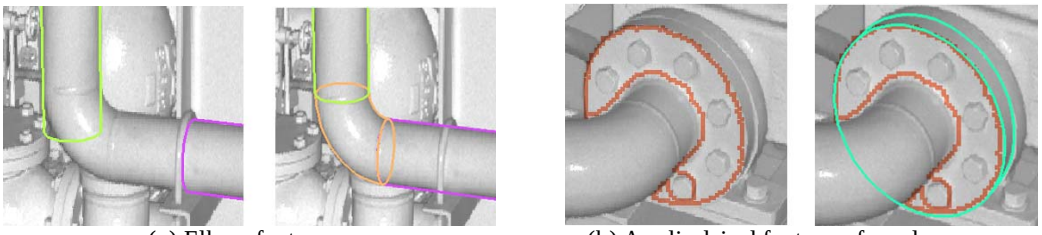
In Fig. 11(d), a stepped reducer is generated to connect two cylinders. This shape consists of two cylinders and a cone.

5.3.4 Example of Feature Modeling

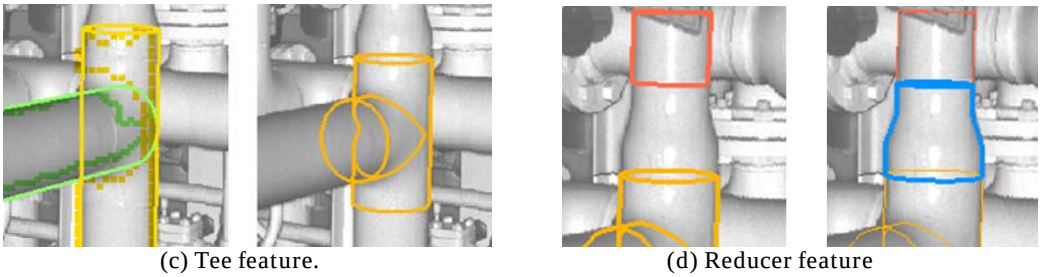
We provide tools for selecting, copying, translating, rotating and resizing features on perspective images. These tools are useful when the object is seriously occluded by other components and its base surfaces cannot be extracted. When the user observes the same types of features repeatedly, the user can copy and edit existing features.

Fig. 12 shows 3D models of a historic architecture. In this building, the same shapes appear repeatedly. Some arches are largely occluded, but they can be modeled by copying and editing 3D models of other arches.

Fig. 13 shows 197 features generated using a single scanned data. The user could generate 3D models intuitively and easily. All that the user had to do was to extract base surfaces and to edit them on the perspective images. Fig. 13(a) shows 3D models projected on the perspective image. Fig. 13(b) shows 3D models and a spherical image rendered with openGL. Our method allows the user to generate 3D models from incomplete point-clouds.

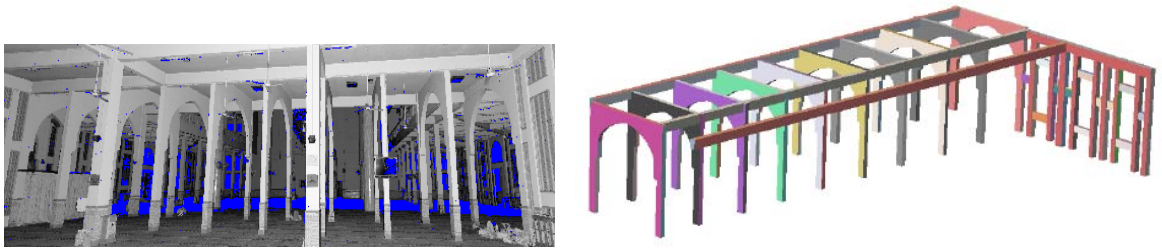


(a) Elbow feature. (b) A cylindrical feature of a valve.



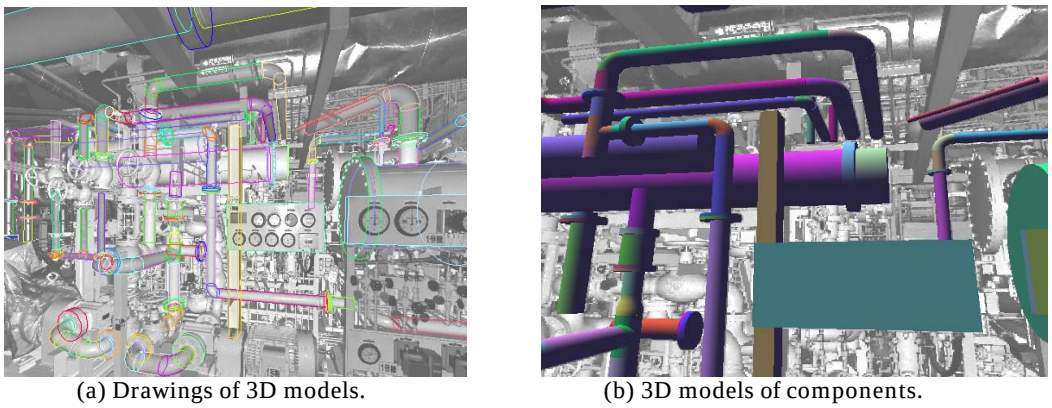
(c) Tee feature. (d) Reducer feature

Fig. 11: Generating cylindrical features.



(a) Perspective image (b) Faithful 3D models generated using a point-cloud.

Fig. 12: An example of a historic architecture.



(a) Drawings of 3D models. (b) 3D models of components.

Fig. 13: Modeling of a power plant.

6 CONCLUSION

In this paper, we proposed a new modeling method for creating 3D solid models based on images and mesh models generated from mid/long-range laser scanned point-clouds. We introduced image-based techniques to reconstruct missing potions in mesh models. The grid mesh format enabled an easy access to the necessary portion of the large mesh model in an out-of-core manner. We introduced a feature-based modeling approach and showed that the user could efficiently create 3D models of components in facilities.

In future work, we would like to improve the accuracy of base surfaces. In addition, we would like to automate feature modeling, because the user manually selects features in the current implementation. To solve these problems, we intend to introduce object recognition techniques in the field of computer vision.

ACKNOLEDGEMENTS

This work was supported by Grant-in-Aid for Scientific Research. Example point-clouds are courtesy of Shinsei-Giken and FARO International Japan.

REFERENCES

[1] Masuda, H.; Tanaka, I.: Extraction of Surface Primitives from Noisy Large-Scale Point-Clouds, Computer-Aided Design & Applications, 6(3), 2009, 47-57.
[2] Sharf, A.; Alexa, M.; Cohen-Or, D.: Context-based Surface Completion, ACM Transaction of Graphics, 23(3), 2004, 875-884.
[3] Ju, T.: Robust Repair of Polygonal Models, ACM Transaction of Graphics, 23(3), 2004, 885-892.
[4] 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.
[5] 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.
[6] Debevec, P.; Taylor, C.; Malik, J.: Modeling and Rendering Architecture from Photographs: A Hybrid Geometry- and Image-Based Approach, SIGGRAPH 96, 1996, 11-20.
[7] Isenburg, M.; Liu, Y.; Shewchuk, J.; Snoeyink, J.: Streaming Computation of Delaunay Triangulations, ACM Transactions on Graphics, 25(3), 2006, 1049-1056.
[8] Isenburg, M.; Lindstrom, P. : Streaming Meshes, Proc. of IEEE Visualization, 2005, 231-238.
[9] Levin, D.: Mesh-Independent Surface Interpolation, Geometric Modeling for Scientific Visualization, 2003, 37-49.