Title:
Point Cloud Segmentation for Pipelines in Industrial Facilities Using Recurrent Networks

Authors:
Kohei Shigeta, k.shigeta.uec@mail.uec.jp, The University of Electro-Communications
Takuma Nagumo, takuma.nagumo@uec.ac.jp, The University of Electro-Communications
Hiroshi Masuda, h.masuda@uec.ac.jp, The University of Electro-Communications

Keywords:
Point cloud, Terrestrial laser scanner, Reverse Engineering, Semantic segmentation, Recurrent Neural Network

DOI: 10.14733/cadconfP.2022.245-250

Introduction:

3D models are useful for simulating maintenance tasks in industrial facilities. For renovating complicated pipeline systems, 3D models make it easy to plan the routing of pipes. However, since most industrial plants were built decades ago, their faithful 3D models often do not exist. In such cases, terrestrial laser scanners (TLS) can effectively capture dense point clouds of industrial facilities, and 3D models of pipelines can be created from point clouds.

So far, many methods have been proposed for extracting components in industrial facilities from point clouds acquired by TLS[2], [6]. TLS can capture point clouds from a wide range, but point clouds of large facilities are often noisy, sparse, and partially missing. Therefore, most existing methods create 3D models using pre-defined knowledge, such as industrial standards. However, as shown in Fig. 1, there are various ways to combine components, and there are many components whose shapes are not defined by industrial standards. It is difficult to develop a general-purpose shape reconstruction method for such components.

Supervised machine learning is a promising method that can flexibly handle such a variety of components. Recently, a number of convolutional neural network (CNN) methods, such as PointNet[5], have been proposed for classification and segmentation of point clouds. These methods are applied to a relatively small number of point clouds, such as 1024 or 2048 points. In order to apply these methods to large-scale point clouds, it is necessary to divide the point cloud into fixed length of points.

In this paper, we discuss a method for classifying and segmenting point clouds of pipeline systems. A pipeline system is composed of various components connected in sequence. For such objects, it would be desirable for the neural network model to encode the order of points in the pipeline direction. Recurrent neural networks (RNN) are known for encoding time series data[1]. In this study, we regard the point cloud of a pipeline as time series data, and classify and segment the point cloud using recurrent neural networks. We also discuss a method for training the RNN model using automatically created virtual pipeline systems.

Slicing Point Cloud along The Center Axis:

As an RNN model for point clouds, RSNet [1] has been proposed for semantic segmentation. In this model, each point is converted to feature vectors using 1D convolutional layers. the features are sliced in the x, y, and z directions, and the sliced features are regarded as time series data in each direction. Then, the bi-directional gated recurrent unit (GRU) is applied to calculate the features including the data of the previous and next directions. Finally, each point is classified using the features.

(a) (b)

Fig. 1: Components with various shapes: (a) Flanges, (b) Valves.

Each pipeline has a central axis, and components in the pipeline, such as straight pipes and flanges, are arranged along the central axis. So far, many methods have been proposed to calculate the central axis from a point cloud of a generalized cylinder shape[2], [4], [6]. In this study, we calculate the central axis of a pipeline by detecting cylinders from a point cloud[3]. Then, we define moving frames with $\mathbf{e}_1, \boldsymbol{e}_2, \mathbf{e}_3$ axes along the central axis. The point cloud is sliced along each axis of the moving frame, and the features of each sliced data are treated as time series data. In this study, we use the RNN model with the same layers as RSNet.

Fig. 2 shows an outline of our method. In order to slice a point cloud along the central axis, we define a moving frame so that the origin is on the central axis and the $\mathbf{e}_3$ axis coincides with the orientation of the central axis. The axes $\mathbf{e}_1$ and $\mathbf{e}_2$ are orthogonal to $\mathbf{e}_3$. The direction of $\mathbf{e}_1$ and $\mathbf{e}_2$ are arbitrarily determined at the starting point of the pipeline. When the origin of the moving frame moves along the central axis, the $\mathbf{e}_1$ and $\boldsymbol{e}_2$ axes are determined so that they are perpendicular to $\mathbf{e}_3$ and the rotation after the move is minimized. The coordinates of points are represented in the moving frame coordinate system and the coordinates are input into the RNN model. Then, a time series of features are generated along the three axes of the moving frame.

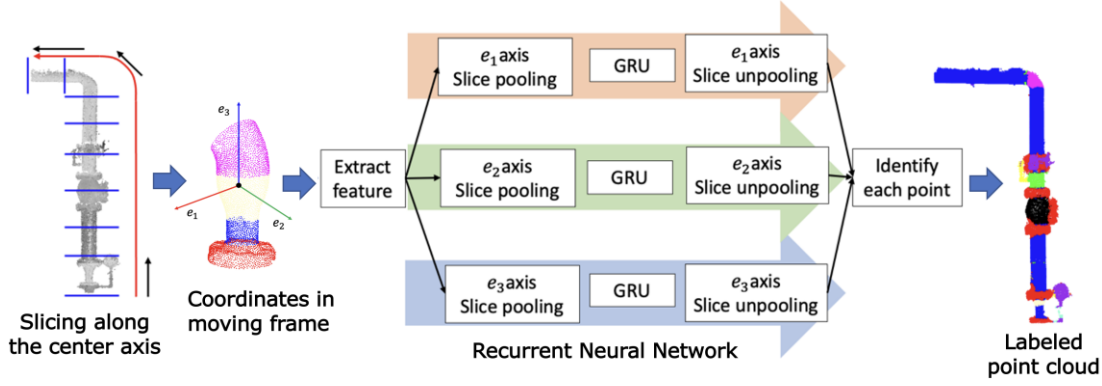


Fig. 2: Process for point cloud segmentation.

Augmentation of Training Data Using CAD Data:

The RNN model requires a large amount of training data to train a large number of parameters for semantic segmentation tasks. If the number of training data is small, overtraining would occur and the generalization of the RNN would be significantly degraded. However, it is difficult to obtain a large number of point clouds of pipeline systems in industrial facilities. In addition, each point of point clouds have to be manually labeled for creating training data. This process requires a lot of effort.

To solve this problem, we automatically create CAD models of virtual pipeline systems, and generate point clouds by irradiating laser beams to the CAD models from a virtual laser scanner. In our method, virtual pipeline systems are created by combining CAD models of components. For standard components, shapes of CAD models can be determined according to the industrial standards. For non-standard components, such as manometers, we selected typical shapes from point clouds of industrial facilities, and created their CAD models. Some components, such as flanges, are used in combination

with other components. In such cases, we created CAD models of combined components, as shown in Fig. 3(b)(c).

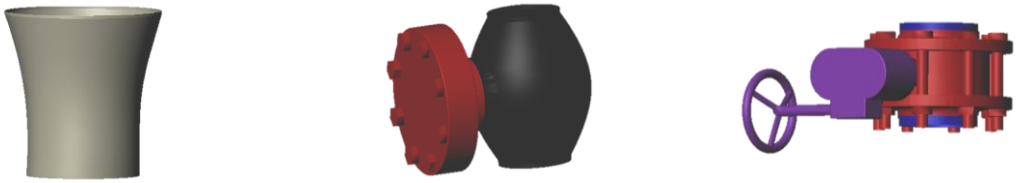
Virtual pipeline systems are generated by randomly selecting a component and connecting it to the pipeline. The shapes of straight pipes, elbows, flanges, and tees can be uniquely determined by the radius of the connecting component. As a result, we can automatically create any number of pipeline systems with piping routes as shown in Fig. 4.

However, when components are randomly selected and connected, they may cause self-interference, as shown in Fig. 5(a). The interference can be detected by calculating distances between the central axes of the components. If interference occurs, the piping route is changed by inserting or replacing elbows or tees so that interference is avoided.

Generating Points on CAD Models:

In capturing point clouds using a laser scanner, the backside of components are often missing. In our method, a virtual laser scanner is placed at a distance of several meters from the object, and laser beams are emitted to the object in equal intervals of the azimuth and zenith angles. Then, point clouds that are similar to actual ones can be obtained, as shown in Fig. 5(b). The calculated point-cloud is dependent on the relative position between the laser scanner and the CAD model. Therefore, many point clouds are generated from each CAD model by rotating it in various axes at various angles.

However, point clouds generated from CAD models are too clean compared to actual point clouds of pipelines, as shown in Fig. 6(a). Therefore, as shown in Figure 6(b), the irregular occlusion, outliers caused by laser spot splitting, and chipped edges are reproduced to create a point cloud that is similar to the actual point cloud.



(a) (b) (c)

Fig. 3: Templates for components of pipelines: (a) Reducer, (b) Torus, (c) Flange with valve.

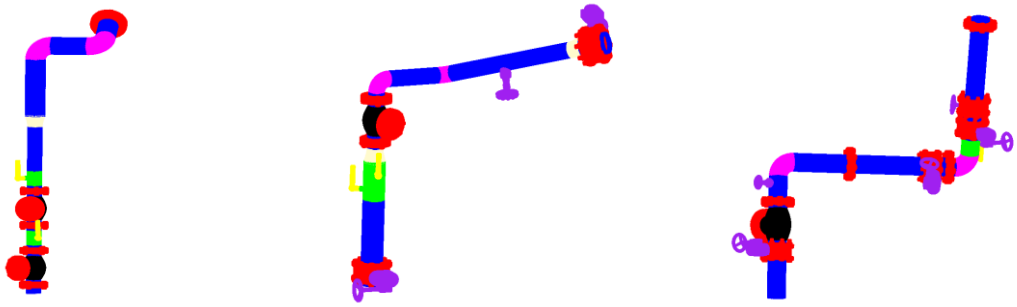
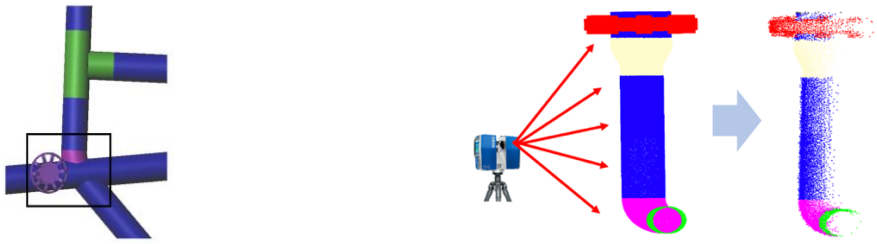


Fig. 4: Training data created using 3D models.

Experimental Results:

To validate our method, we trained the RNN model using three types of training data: (a) only actual point clouds captured using a laser scanner, (b) both actual and virtual point clouds, and (c) both the measured data and the virtual data with occlusions and outliners. In the case (c), occlusions and outliners were added to the half of the virtual data. After training RNN models using the three types of training data, we verified the accuracy of each model. F-measures were used as the evaluation index.



(a) (b)

Fig.5: Generation of training data from 3D models.: (a) Self-intersection and (b) Generation of point clouds from a CAD model

(a) (b)

Fig. 6: Add occlusions and outliers to the generated points: (a) generated points(aligned) and (b) generated points with occlusions and outliners.

For test data, we used actual point clouds that were not used in training data. The input data were cut by 0.5 m along the direction of the central axis of each pipeline, and 4096 points were selected uniformly from each block. Then, the coordinates were normalized to fit into the unit sphere. The training data were overlapped by 0.2m when they were cut into blocks, and augmented with Jitter noise [5]. The numbers of actual and virtual data are shown in Tab.1. The results of the three trained RNN models are shown in Tab.2.

In Tab.2, the RNN model (a), which were trained using only actual point clouds, could detect only straight pipes, elbows, and flanges. This is because the actual data had few components other than straight, elbow, and flange. Compared to this model, the models (b) and (c), which were trained using both actual and virtual point clouds, could significantly improve scores for most component classes. In particular, scores have improved significantly for a small number of components. In addition, the results shows that the model (c), which were trained using virtual data augmented by occlusions and outliers, could further improve the scores. Fig. 7 shows some results of point cloud segmentation.

We compared the RNN model with PointNet++ using training data(c). In this evaluation, the RNN model achieved better overall accuracy than PointNet++, and the F-values were better in six in nine classes. While the RNN model encodes point clouds along the pipeline route, the PointNet++ model encodes points in k-nearest neighbors. It is considered that these models extract different features from component shapes. Therefore, we created an integrated CNN model by concatenating the outputs from the RNN and PointNet++ models. The result is shown in Tab. 2. In all classes, the F-values were improved compared to the RNN-only discriminator. Fig. 7 shows some results of point cloud segmentation.

In this evaluation, the scores of tees and reducers were low in all models. This may be because all models including PointNet++ could not capture effective features from these component shapes. Solving this problem is a future issue.

Conclusion:

In this paper, we proposed a semantic segmentation method using RNN for point clouds of pipeline systems. We also proposed a method for training RNN models using virtual pipelines augmented by

occlusions and outliers. Our experimental results showed that our method could achieve sufficient accuracy for detecting components from point clouds of industrial facilities.

In future work, we would like to create more realistic point clouds from CAD models. Our experiments show that actual point clouds are necessary for training RNN models. We would like to investigate features that are effective for various types of components.

test	train	
measured	measured	CAD
8	15	186

Tab. 1: Number of pipeline's data used for training and evaluation.

	(a) Only actual point clouds	(b) Actual and virtual (not augmented)	(c) Actual and virtual (augmented)	Concatenated RNN & PointNet++
Elbow	39.3%	65.8%	67.3%	69.4%
Flange	55.3%	80.8%	87.6%	89.8%
Straight	73.9%	87.3%	87.6%	88.9%
Tee	0%	27.5%	16.4%	19.2%
Valve	0%	64.3%	73.1%	77.9%
Torus	0%	59.0%	62.1%	73.7%
Reducer	0%	11.0%	23.0%	28.2%
Handle	0%	25.2%	53.5%	62.5%
Manometer	0%	10.4%	61.5%	71.4%

Tab. 2: F-measures of classifiers trained using 3 types of data sets.

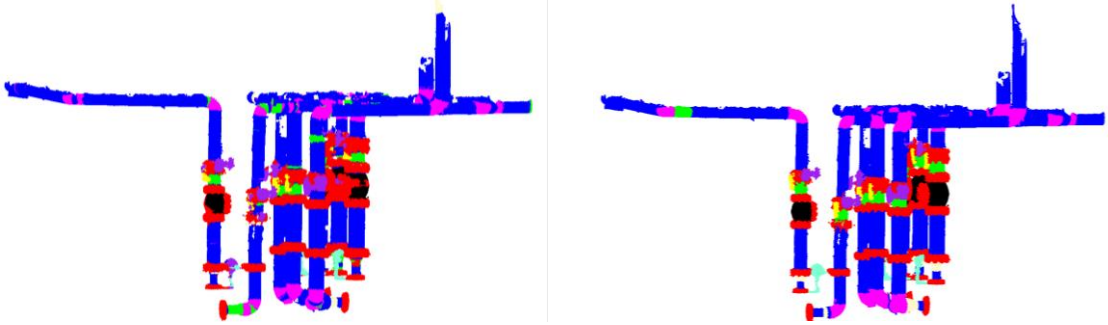


Fig. 7: Semantic segmentation results: (a) Predicted labels, (b) Ground truth labels.

References:

[1] Huang, W.; Wang, U.: Neumann Recurrent Slice Networks for 3D Segmentation of Point Clouds, Conference on Computer Vision and Pattern Recognition, 2018, 2626-2635. 10.1109/CVPR.2018.00278

[2] Masuda, H.; Tanaka, I.: As-Built 3D Modeling of Large Facilities Based on Interactive Feature Editing, Computer-Aided Design and Applications, Vol.7, No.3, 2010, 349-360. 10.3722/cadaps.2010.349-360

[3] Masuda, H.; Niwa, T.; Tanaka, I.; Matsuoka, R.: Reconstruction of Polygonal Faces from Large-Scale Point-Clouds of Engineering Plants, Computer-Aided Design and Applications, Vol.12, No.5, 2015, 555-563. https://doi.org/10.1080/16864360.2015.1014733

[4] Midorikawa, Y.; Masuda, H.: Extraction of Rotational Surfaces and Generalized Cylinders from Point-Clouds Using Section Curves, International Journal of Automation Technology, Vol.12, No.6, 2018, 901-910. 10.20965/ijat.2018.p0901

[5] Qi, C. R.; Su, H.; Mo, K.; Guibas, L. J.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, Conference on Computer Vision and Pattern Recognition, 2017, 652-660. https://doi.org/10.1109/CVPR.2017.16

[6] Son, H.; Kim, C.; Kim, C.: Automatic 3D Reconstruction of As-Built Pipeline Based on Curvature Computations from Laser-Scanned Data, Construction Research Congress 2014, 925-934. 10.1061/9780784413517.095

[7] Tagliasacchi, A.; Zhang, H.; Cohen-Or, D.: Curve Skeleton Extraction from Incomplete Point Cloud, ACM SIGGRAPH, Vol28, No.3, 2009, 1-9. https://doi.org/10.1145/1531326.1531377