JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 13, No. 3, September 2019, pp. 99-106

Point Cloud Segmentation of Crane Parts Using Dynamic Graph CNN for Crane Collision Avoidance

Hyeonho Jeong, Hyosung Hong, Gyuha Park, and Mooncheol Won*
Department of Mechatronics Engineering, Chungnam National University, Daejeon, Korea
zz4979@cnu.ac.kr, hyosung.hong@cnu.ac.kr, khp3927@naver.com, mcwon@cnu.ac.kr

Mingyu Kim and Hoyeong Yu
FM Electronics, Daejeon, Korea
mglov@futureman.co.kr, artisty123@futureman.co.kr

Abstract
In this study, we have developed a point cloud segmentation algorithm for a collision avoidance system between cranes and other objects in construction yards. We used the Dynamic Graph CNN (DGCNN) algorithm to segment the point cloud of the entire yard into crane parts and backgrounds. The point cloud data were obtained from several LIDAR sensors attached to the crane. All points were grouped into specific core clusters using the DBSCAN algorithm. The core clusters were used to train the DGCNN after labeling with corresponding part names. This network classified the point cloud into crane types and their part names. Experimental results show that the crane part segmentation performance of the suggested algorithm is accurate enough to be used for collision avoidance system. It is possible to estimate the pose of a crane by comparing the segmented point clouds with those of the CAD model.

Category: Smart and Intelligent Computing

Keywords: Crane; 3D point cloud; Segmentation; DBSCAN; Dynamic graph, CNN

I. INTRODUCTION

Crane is a commonly used equipment for industrial areas including shipyards and construction sites. However, accidents involving a crane result in extensive damage to humans and property. Usually, there is a signal person who monitors the safety of the crane from outside and transmits signals for operation of the crane. However, complex industrial sites may have blind spots that cannot be detected by the crane driver or signal person, which increases the possibility of an accident. Therefore, it is necessary to use an autonomous crane monitoring system to avoid collisions.

The collision avoidance systems that operate using GPS sensors [1] estimate the distance between the objects to alert and warn the risk of crash. However, if the system only measures the distance between installed objects, other objects that do not have the sensor will not be recognized as possible collision objects. In addition, since the collision alarm is sounded when the distance between two objects is close, a false alarm may occur during normal operation (for example, involving a crane and an object or when a crane lifts another crane).

To solve these problems, all objects near the crane should be included in the collision risk areas. Also, if the crane movement can be predicted, it is possible to reduce

Open Access http://dx.doi.org/10.5626/JCSE.2019.13.3.99 http://jcse.kiise.org

Received 05 August 2019; **Revised** 09 September 2019; **Accepted** 11 September 2019
*Corresponding Author

 pISSN: 1976-4677 eISSN: 2093-8020

false alarm during the normal operation of the crane. In this study, we used a light detection and ranging (LIDAR) sensor-based collision avoidance system instead of GPS to detect all objects near the crane, and constructed a system that allows the driver to visually check the surrounding environment. In addition, the part segmentation algorithm was applied to the point cloud data to detect the different types of crane and to segment the parts of each crane, including jib and mast.

The part segmentation algorithm divides the point cloud data of the crane into several clusters using density-based spatial clustering of applications with noise (DBSCAN) [2]. DBSCAN is a density-based clustering method used to classify the clusters into 13 classes applying the deep learning-based Dynamic Graph CNN (DGCNN) [3]. Classification of crane parts can help prevent crane collision and predict crane pose and movement path.

II. RELATED WORK

Machine learning techniques used to process point cloud data process images by projecting them into a 2D space instead of 3D coordinates [4]. However, the 2D projection method is limited by the complex and changing dimensions in addition to the associated losses and deformations of the 3D coordinates.

The PointNet [5] architecture is one way to resolve 3D point cloud problems. It can perform classification and segmentation while using original 3D coordinate data. Since there is no projection into 2D space, the calculation is fast without loss of 3D coordinate information, resulting in higher accuracy than the existing algorithms. Based on these advantages, PointNet architecture resulted in state-of-the-art performance in indoor environmental data classification and segmentation in 2017. However, one of the disadvantages of PointNet is that it cannot distinguish 3D and local features between points because they learn 3D coordinate points separately. In order to overcome the disadvantage, the PointNet ++ [6] architecture considers the distance between points by learning points hierarchically. However, it still does not directly determine the geometric features between points.

DGCNN [3] is able to distinguish the geometric features between points by adding an edge convolution layer to existing PointNet architecture. The part segmentation performance is improved because the local geometric features of the object can be learned.

III. LIDAR-BASED CRANE COLLISION AVOIDANCE SYSTEM

Unlike other crane collision avoidance systems, the LIDAR sensor-based crane collision prevention system of FutureMan (FM) Electronics [7] of Korea prevents collisions between cranes as well as with other objects. The LIDAR sensor applied to the system has been renovated using Velodyne LIDAR Puck sensor to rotate along the vertical and horizontal axes, allowing the sensor to see a wider range with denser point cloud data.

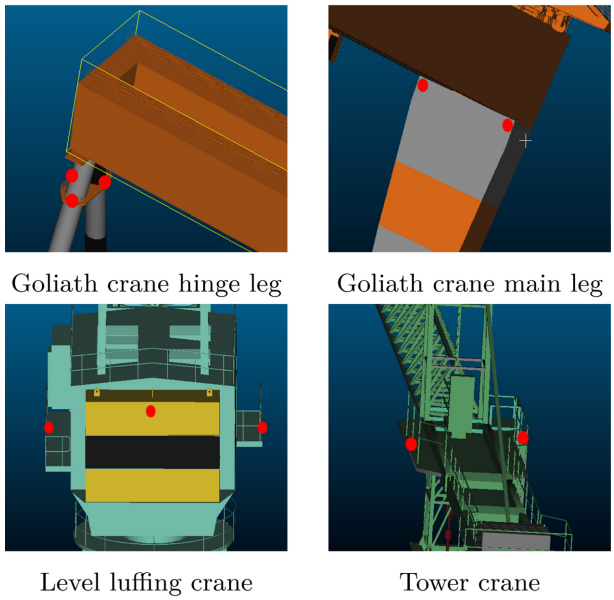


Fig. 1. The position of the sensor attached to the crane is marked with a red dot.

In this study, we have installed collision avoidance systems on three types of cranes as follows (Fig. 1).

(1) Goliath crane (GC), which has two legs, is characterized by simultaneous movement of the controller and the workpiece.

(2) Level luffing crane (LLC), in which the jib moves up and down.

(3) Tower crane (TC), which has a fixed mast and a jib rotating at 360º.

Five, three, and two LIDAR sensors were installed in the three cranes, respectively.

IV. CRANE PART SEGMENTATION ALGORITHM

The crane part segmentation algorithm consists of three steps. First, a sampling process is performed to reduce the number of points obtained from the LIDAR sensor. Then, the sampled point cloud is divided into clusters using DBSCAN, and the noise clusters are removed. Finally, a segmentation process is performed to classify the points belonging to each crane part. Fig. 2 depicts the part segmentation algorithm.

A. Sampling

When importing point cloud data from a LIDAR

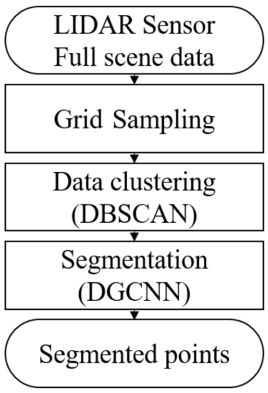


Fig. 2. Algorithm flow chart.

sensor, the number of points is approximately 150,000 to 600,000. The large number of points affects the real-time performance of the algorithm. Therefore, a sampling process is required to reduce the number of points.

We compared two sampling methods: Gaussian random sampling and grid sampling. We selected grid sampling because due to the LIDAR sensor characteristics, the point near the sensor is denser than the point away from the sensor. With Gaussian random sampling, points are sampled with probability, such that points that are far from the sensor can hardly be sampled due to low density. Grid sampling does not suffer from the density problems.

Grid sampling was performed by dividing the full scene point clouds obtained by the LIDAR sensor into 0.5 m × 0.5 m × 0.5 m voxels and obtaining the average values of the point coordinates in one voxel. We tried a heuristic method to find the appropriate size of the voxel to reduce the number of point clouds for efficient computation while maintaining the original shape. As a result of sampling, the number of points has been reduced to 10% (Fig. 3).

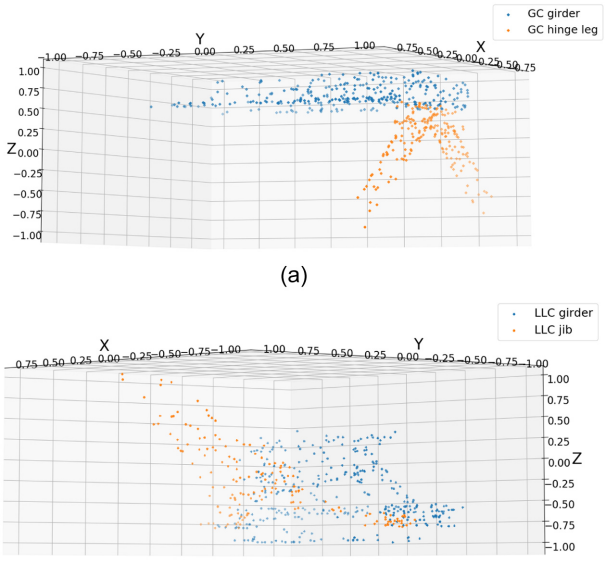


Fig. 3. Example of one cluster after grid sampling the point cloud obtained from LIDAR sensor and dividing the point cloud into multiple clusters through DBSCAN: (a) Cluster of Goliath crane and (b) cluster of level luffing crane.

B. Clustering

The purpose of clustering is two-fold. First, it filters out noisy line points, which means that clusters with a small population are neglected. Second, it considers only the clusters that are near the crane body, which implies that the clusters far from a specific crane are neglected in the segmentation process.

We selected DBSCAN as a clustering technique. Because we do not know the number of objects that are likely to collide with the crane, we cannot use a clustering method that pre-determines the total number of clusters (e.g., k-means). A disadvantage of DBSCAN is that clustering does not work well when the density differences of each point cloud are large. However, we have already alleviated the density difference via sampling process. Therefore, DBSCAN clustering is an appropriate method for this problem.

There are two parameters to set when using DBSCAN. To obtain a clustered density criterion, you must set the radius ε and the minimum number *m* of points that must be within a circle with radius ε. The method consists of three steps: (1) for every point, ensure that you have at least *m* points in a circle with radius ε, and consider these points as the core point. (2) If the core points are in a single circle, they are grouped into one cluster, except for points that are not core points. (3) Points that are not core points but within the cluster are considered as border

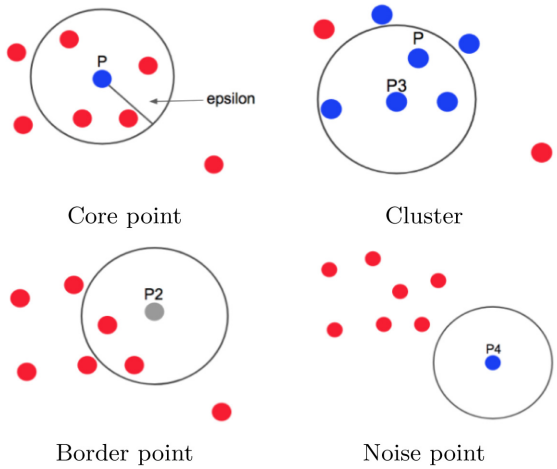


Fig. 4. DBSCAN point types. Core point: if there are more points than the minimum number of points p in radius ∈, Cluster: a collection of core points, Border point: a point that is not a core point but is within another core point area, Noise point: points that are neither core points nor border points.

points, and if they are outside the cluster, they are considered as noise points (Fig. 4).

In this study, we set the radius ε to 2 m and the minimum number m of points to 10. After DBSCAN, we only selected clusters within the maximum distance at which the crane was in danger of collision. In this work, the maximum distance was selected as 50 m. Also, a cluster containing fewer number of points than the reference point is regarded as a noise cluster, and was removed.

C. Segmentation

After finishing the data sampling and clustering process, the crane part segmentation was performed using the DGCNN [3] architecture (Fig. 5). DGCNN is a network structure inspired by PointNet [5] architecture. The basic structure is similar, but an edge convolution layer is added to compensate for the disadvantages of PointNet. By performing the edge convolution calculation, we can learn the geometric relation between points that cannot be learnt in PointNet architecture. Also, DGCNN exhibits both translation-invariant and non-locality characteristics.

The calculation of the edge convolution layer is as follows. First, an F-dimensional point cloud is considered with n points. Each point can be thought of as a 3D coordinate $X_i = (x_i, y_i, z_i)$, i.e., $F = 3$. We can add additional information, such as color and surface normals. The graph $G = (V, E)$, which is shown in the middle of Fig. 6, represents the partial local structure of the point cloud, where V and E are vertices and edges with the form $V = [1, \ldots, n]$, and $E = V \times V$, respectively. The graph G is constructed through the K-nearest neighbors (K-NN). The K-NN algorithm is an algorithm that finds the nearest K points based on a point.

Fig. 6 represents the calculation of operation edge convolution. From the point X_i, we find the proximity points (X_{j1}), (X_{j2})...(X_{jk}) through K-NN, and the edge feature as $e_{ij} = h_\theta(X_i, X_j)$, where $h_\theta()$ is a parametric non-linear function and θ are the learned parameters in the function $h_\theta()$.

Finally, $X' = \max_{j:(i,j)\in E} e_{ij}$ is the output, in which X' can be thought of as a new point that includes an edge feature. Then, if we consider a collection of multiple X' as another point cloud, we can construct a new graph G' and repeat this process to learn DGCNN.

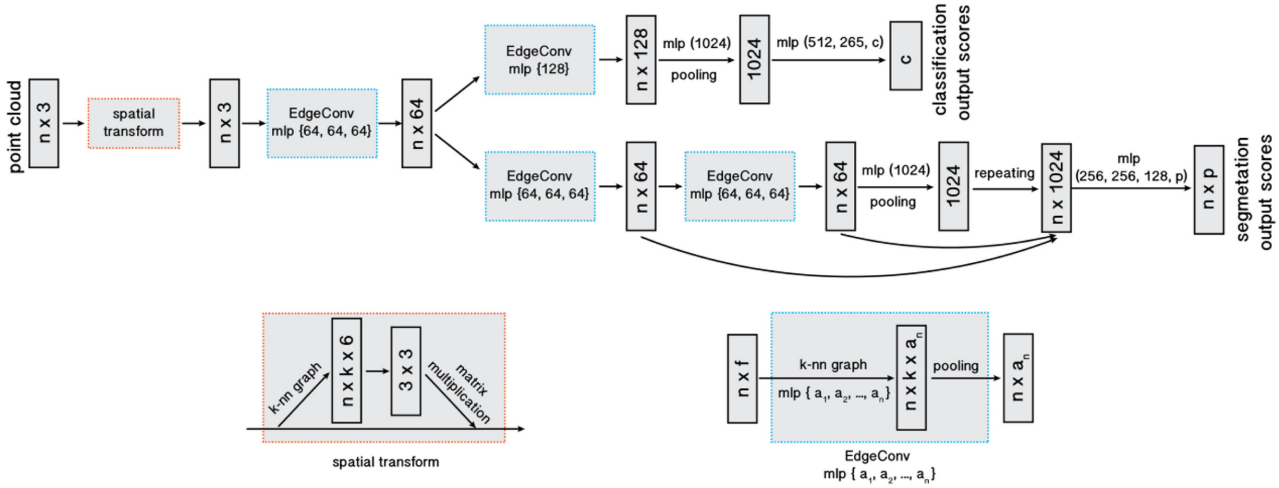


Fig. 5. Dynamic graph CNN model architectures.

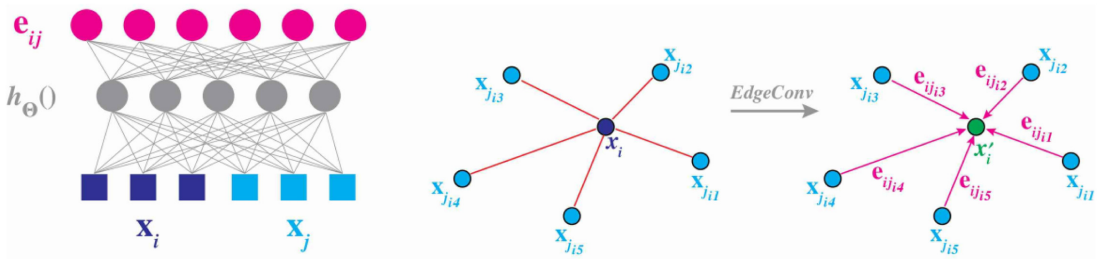


Fig. 6. Computing an edge feature and visualization of the Edge Convolution operation adapted from Wang et al., "Dynamic graph CNN for learning on point clouds," 2018 [3]. Edge convolution is the process of finding the edge feature e_{ij} through $h_\theta(X_i, X_j)$, and selecting the largest of them to create a new point called X′.

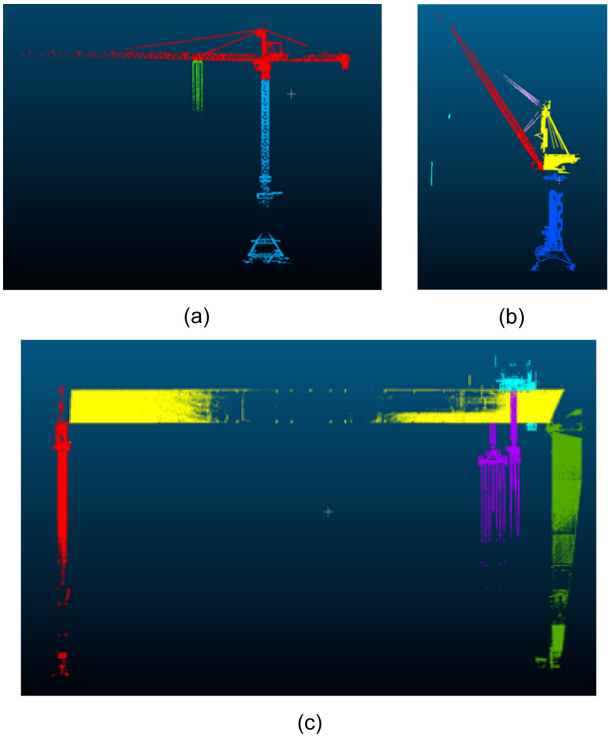


Fig. 7. Crane labeling information: (a) tower crane, (b) level luffing crane, and (c) Goliath crane.

V. EXPERIMENT

The two neural network algorithms based on PointNet and DGCNN were tested. The same training and testing data were used to test the two algorithms.

A. Data Training and Testing

The data used in the part segmentation algorithm were obtained by taking the point cloud obtained from the LIDAR sensors installed on the cranes and manually labeling them using the semantic segmentation editor program [8] (Fig. 7).

The goliath crane carries five labels: a girder on top of the crane, an I-shaped main leg, a triangular hinge leg, a driver's cockpit on the crane, and a hook hanging under the cockpit.

The level luffing crane contains four labels: a jib that corresponds to the arm of the crane, a girder that supports the jib, a hook that hangs at the end of the jib, and a mast that corresponds to a column of a crane.

The tower crane has three labels: a jib that rotates 360°, which corresponds to the crane's arm, a hook that hangs at the end of the jib, and a mast that corresponds to the crane column. Finally, all non-crane objects were labeled in the background and classified into a total of 13 classes.

The shape of the input data is (x_{global}, y_{global}, z_{global}, $x_{cluster}$, $y_{cluster}$, $z_{cluster}$).

We normalized the global coordinate value (x_{global}, y_{global}, z_{global}) in full scene and the local coordinate value ($x_{cluster}$, $y_{cluster}$, $z_{cluster}$) in one cluster to values between -1 and 1, and concatenated the two coordinate values. In addition, we removed the data below 6 m from the ground level, which is not the area of interest (Fig. 8).

As shown in Table 1, we have a severe data imbalance problem. If you compare the background area with the crane area, you can see additional points in the background area. When viewed by the type of cranes, you can see that the tower crane data are smaller than the other two crane data.

In order to solve the data imbalance problem, Gaussian noise was added to the coordinate value in the sampling process, and the sampling was repeated three times. The tower crane was repeated nine times because of insufficient data. The data obtained included 7,892 clusters of training data and 2,397 clusters of test data.

B. Results

Table 2 shows the segmentation performance of DGCNN and PointNet with the same pre-processed training and test data. Both algorithms showed relatively accurate segmentation results by recording more than 90% of recalls and precisions in most crane parts except LLC hooks.

When comparing the recall, DGCNN was found to be 1%–2% higher on average than PointNet. The precision shows that the results of the goliath and the tower crane hooks in the PointNet were 10% lower than those of the DGCNN. These results suggest that the DGCNN can segment the crane parts and background more accurately than the PointNet.

However, the two algorithms could not detect the hooks of LLC cranes. Based on the number of training

Table 1. Training and test data point numbers

	Background	Goliath crane					Level luffing crane				Tower crane		
		Girder	Main leg	Hinge leg	Hook	Control	Girder	Jib	Hook	Mast	Jib	Hook	Mast
Train data	1,971,869	413,406	97,424	221,895	52,375	59,587	384,297	190,184	25	286,437	271,678	8,150	129,457
Test data	574,987	135,877	32,551	74,370	18,496	19,438	126,556	70,052	726	117,590	41,314	426	14,881

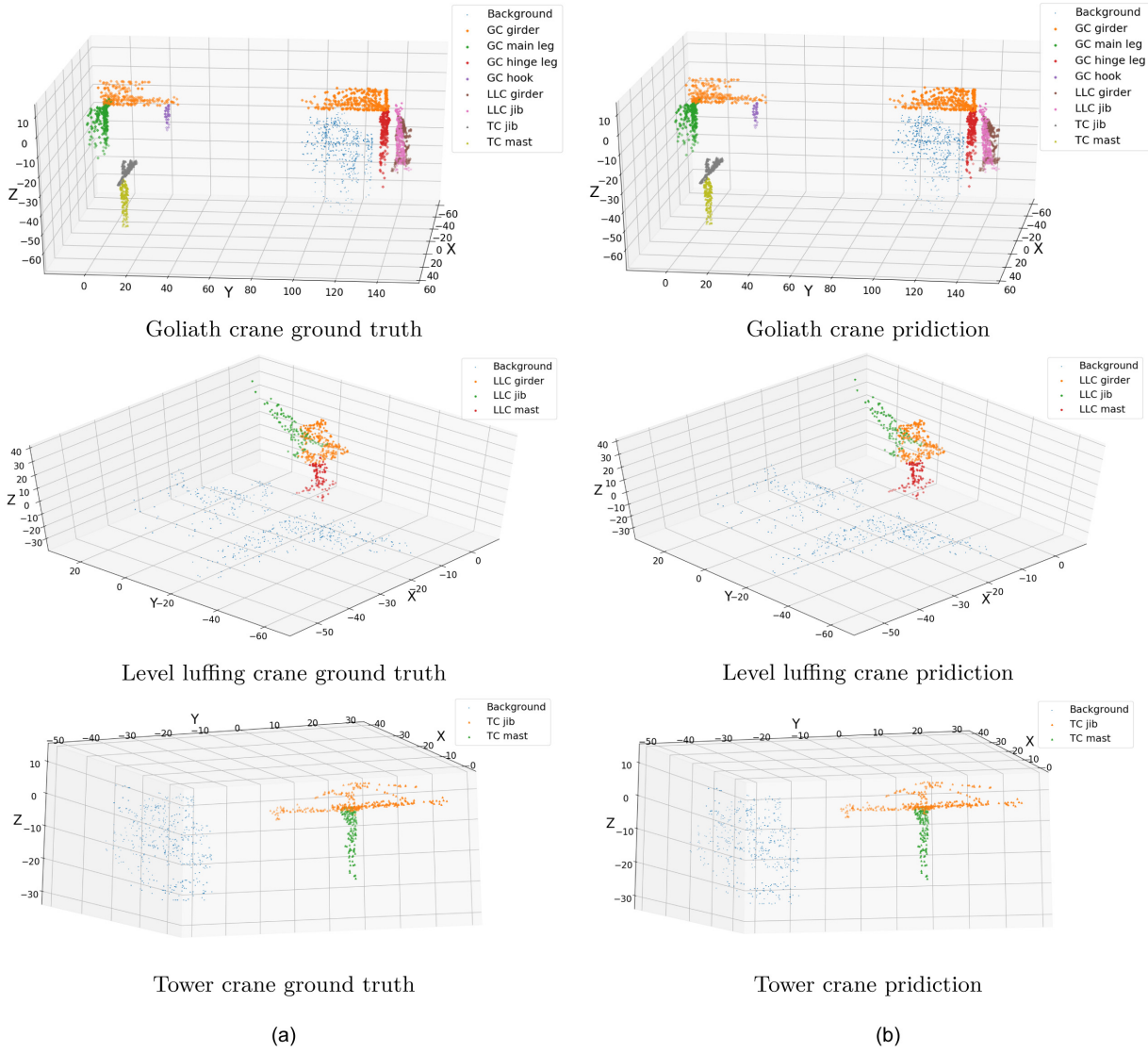


Fig. 8. (a) Test point cloud ground truth, GC, LLC, TC order from above. (b) Results of segmentation algorithm.

Table 2. Segmentation algorithm performance comparison (%)

		Background	Goliath crane					Level luffing crane				Tower crane		
			Girder	Main leg	Hinge leg	Hook	Control	Girder	Jib	Hook	Mast	Jib	Hook	Mast
Recall	PointNet	98.02	96.12	85.41	95.20	93.20	89.93	94.00	95.48	0	99.23	99.28	88.73	96.69
	DGCNN	98.94	96.95	87.70	96.37	96.34	91.92	96.11	95.77	0	98.96	98.54	90.38	98.17
Precision	PointNet	98.20	93.62	93.21	96.80	80.38	95.57	96.75	94.85	0	98.27	95.90	42.05	94.20
	DGCNN	98.88	95.10	90.30	95.20	97.86	96.07	97.90	98.99	0	97.89	98.99	95.53	96.27

data points, the number of hook data in LLC crane was too small to be trained properly. For the test data, about 2.6% of the crane parts were misidentified, and half of them were analyzed as other types of crane.

Labeling errors are one of the most likely factors contributing to outperformance by PointNet compared with DGCNN. Since humans do the labeling manually, 100% correct labeling is impossible. We think that incorrect labeling may decrease the performance of DGCNN more than PointNet since DGCNN utilizes local features.

VI. CONCLUSION

We designed a point cloud part segmentation algorithm applied to a crane collision avoidance system. The training and testing data were generated from raw point cloud data obtained from LIDAR sensors mounted on cranes. Using the DGCNN network structure, we classified crane parts with greater than 90% accuracy.

Additional data may facilitate better classification of the crane hooks and further subdivision of the background area. This algorithm can be applied to industrial places such as smart factories and unmanned construction sites. In a future study, we will utilize this algorithm to calculate the crane pose by matching the point cloud data from CAD model with the segmented crane parts. We also confirmed the possibility of designing a point cloud matched with a CAD model and a crane pose prediction algorithm.

ACKNOWLEDGMENTS

This study was funded and conducted under "The Competency Development Program for Industry Specialists" of the Korean Ministry of Trade, Industry and Energy (MOTIE), operated by Korea Institute for Advancement of Technology (No. P0008473, The development of highly skilled and innovative manpower to lead the innovation based on robot), and FM Electronics.

REFERENCES

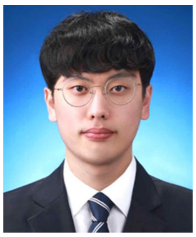
1. H. Wu, Y. Yin, S. Wang, W. Shi, K. C. Clarke, and Z. Miao, "Optimizing GPS-guidance transit route for cable crane collision avoidance using artificial immune algorithm," *GPS Solutions*, vol. 21, no. 2, pp. 823-834, 2017.
2. M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, Portland, OR, 1996, pp. 226-231.
3. Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," 2018; https://arxiv.org/abs/1801.07829.
4. E. Grilli, F. Menna, and F. Remondino, "A review of point clouds segmentation and classification algorithms," *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, pp. 339-344, 2017.
5. C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: deep learning on point sets for 3D classification and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, 2017, pp. 652-660.
6. C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: deep hierarchical feature learning on point sets in a metric space," *Advances in Neural Information Processing Systems*, vol. 30, pp. 5099-5108, 2017.
7. NOCOL: smart crane anti-collision system, http://www.futureman.co.kr/CraneAntiCollision.
8. Semantic Segmentation Editor: web labeling tool for camera and LIDAR data, https://github.com/Hitachi-Automotive-And-Industry-Lab/semantic-segmentation-editor.

Hyeonho Jeong

Hyeonho Jeong received bachelor's degree in Department of Mechatronics Engineering, Chungnam National University, Korea in 2018. He is currently a master's student at the same graduate school. His research interests are in point cloud processing and image processing using artificial intelligence.

Hyosung Hong

Hyosung Hong received bachelor's degree and master's degree in Department of Mechatronics Engineering, Chungnam National University, Korea in 2012 and 2017, respectively. He is currently a Ph.D. student at the same graduate school. His research interests are in autonomous vehicle control and object 6D pose estimation from 2D images.



Gyuha Park

Gyuha Park received bachelor's degree in Department of Mechatronics Engineering, Chungnam National University, Korea in 2019. He is currently a master's student at the same graduate school. His research interest is in image processing using artificial intelligence.

Mingyu Kim

Mingyu Kim received master's degree in Robot System, Chungnam National University, Korea in 2016. He also works for FM Electronics, Daejeon, Korea. His research interests are in panorama LIDAR, driverless crane, crane ADAS for construction and shipyard environments.

Hoyeong Yu

Hoyeong Yu received B.S. degree in Electronics Engineering, Dankook University, Korea in 2013. He is currently a master's student in Department of Mechatronics Engineering, Chungnam National University since March 2018. He also works for FM Electronics, Daejeon, Korea. His research interests are in machine learning and applications using LIDAR data.

Mooncheol Won

Mooncheol Won received bachelor's degree and master's degree in Department of Naval Architecture, Seoul National University, Korea in 1983 and 1985, respectively. He received Ph.D. degree in Department of Mechanical Engineering, University of California, Berkeley, CA in 1995. He is currently a professor at the Department of Mechatronics Engineering in Chungnam National University. His research interests are in control engineering and artificial intelligence.