EMITTER International Journal of Engineering Technology
Vol. 10, No. 1, June 2022, pp. 153~169
DOI : 10.24003/emitter.v10i1.704

Density-based Clustering for 3D Stacked Pipe Object Recognition using Directly-given Point Cloud Data on Convolutional Neural Network

Alfan Rizaldy Pratama[1], Bima Sena Bayu Dewantara[1], Dewi Mutiara Sari[1], Dadet Pramadihanto[1]

[1]Department of Informatics and Computer Engineering,
Politeknik Elektronika Negeri Surabaya, Indonesia
Correspondence Author: dadet@pens.ac.id

Received March 4, 2022; Revised April 6, 2022; Accepted May 7, 2022

Abstract

One of the most commonly faced tasks in industrial robots is bin picking. Much work has been done in this related topic is about grasping and picking an object from the piled bin but ignoring the recognition step in their pipeline. In this paper, a recognition pipeline for industrial bin picking is proposed. Begin with obtaining point cloud data from different manner of stacking objects there are well separated, well piled, and arbitrary piled. Then followed by segmentation using Density-based Spatial Clustering Application with Noise (DBSCAN) to obtain individual object data. The systems then use Convolutional Neural Network (CNN) that consume raw point cloud data. Performance of the segmentation reaches an impressive result in separating objects and network is evaluated under the varying style of stacking objects and give the result with average Accuracy, Recall, Precision, and F1-Score on 98.72%, 95.45%, 99.39%, and 97.33% respectively. Then the obtained model can be used for multiple objects recognition in one scene.

Keywords: Industrial object, Density-based clustering, 3D object recognition, Convolutional neural network

1. INTRODUCTION

The growth of robot applications in the industrial appliance is increased significantly. There are many purposes of robotic research, especially for an industrial appliance is to build a system that works independently and can be flexible for certain tasks [1]. Bin picking is one of many applications of automatic manufacturing that can be used for unloading industrial parts such as fitting pipes [2]. This task cannot be separated from the machine vision system that helps the robot see the environment of the work area. To obtain the visual information, Time-of-Flight camera is one of many 3D sensors or depth camera that provide depth information to represent the geometry and surface of the scene as well as its robustness in facing various lightning [3].

Generally, the object that has been used in bin picking is randomly piled. With these configurations, it might happen the object is overlapping

each other making the information of an object is not complete. The core task in bin picking is how the multiple objects are separated from each other although these are overlapping, afterward the individual data object can be further processed. There are many popular algorithms to solve this problem, like centroid-based clustering and hierarchical clustering. In the previous approach, the cluster is represented by a centroid which is not included in a dataset, and each point is grouped with the nearest centroid, this is widely known as K-means clustering algorithm which remains a problem that can't automatically determine the number of clusters. This condition will confuse the system that must manually determine the number of clusters which certainly vary [4]. The Euclidean clustering algorithm is neighborhood distance-based clustering that groups data from its neighborhood [5]. Unfortunately, these algorithms can't handle overlapping objects that make occlusion on data results though can determine the number of clusters automatically. Density-based clustering [6] attempted to cluster data by a density of a region based on radius and minimum points contained in its region. This algorithm is known-well for grouping data which is not based only on their distance, but the density is considered to form a cluster of data. With the benefit of density information that can help to form a cluster though the object is randomly piled.

Although pose information of an object is important, the system also must be can recognize every type of object that missing in the recognition object task [2], [7], [8]. When the many types of objects are used in industry, it's important for the system can understand which type of objects are further processed. Many techniques are used for object recognition using a handcrafted feature such as local descriptors like Signature of Histograms of OrienTations (SHOT) [9] and Fast Point Feature Histogram (FPFH) [10]. Also, global descriptors like Viewpoint Feature Histogram (VFH) [11] and Ensemble of Shape Functions (ESF) [12] are commonly used to recognize an object. By using a handcrafted feature extractor, it has a limitation that can't recognize multiple objects in a scene, [3] shows the limitation on recognizing objects is taken one-by-one object which decreases the efficiency of the system's pipeline. Moreover, recently deep learning has dominated many research areas such as speech recognition, natural language processing, and especially computer vision [13], [14]. These feature extractors do not require large data for training and usually used were make deep learning on point cloud is not popular. However nowadays availability of acquisition devices is increasing that make the usability of deep learning for further processing is reasonable [15]. Although point cloud data is preserving the original geometric information in 3D spatial space without any discretization, applying deep learning directly on point cloud data is remains challenges in the size of the dataset, high dimensionality, and unstructured data of point cloud itself [16]. Shown by [17] that point cloud data is represented into volumetric such 3D voxel and Multiview images, [18] represent point cloud data into a graph model. To avoid the negative impacts by using discretized

point cloud data, [19] propose PointNet which is an architecture of Convolutional Neural Network that consume point cloud data directly as input to the network. The ability to handle unordered properties of point sets by using Permutation and Transformation Invariance lead to avoid modified the global point cloud category and feature such as rotating and translating points.

The focus of this research is building a pipeline that can handle stacked industrial objects which used is a fitting pipe, starting with segmentation using Density-based Spatial Clustering Application with Noise (DBSCAN) and Convolutional Neural Network (CNN) for classifying and recognizing whether the objects type in a scene using point cloud data that directly taken using Time-of-Flight camera.

2. RELATED WORKS

L. Duc Hanh et al [20] and D. M. Sari et al [3] use Euclidean clustering algorithm [2] to separate all objects captured in the scene into partial clusters of objects. Based on the distance neighborhood of points using Euclidean distance, the algorithm can separate well the object that naturally separated by the distance itself, so the algorithm creates clusters based only on their distance. With this condition, the object that may overlap or have a varying density of each other will be difficult to be clustered because the distance of every cluster is may be different and cannot be determined by one value of distance threshold. This algorithm also applied by Z. Sun et al [21] uses improved Euclidean clustering that is applied to data obtained from LiDAR. In the captured environment is not possible the objects are stacked and have a huge overlap of each other.

S. M. Ahmed et al [22] proposed clustering based on DBSCAN in point cloud data. The author uses the novel DBSCAN that still dependent on choosing their parameter. The author used mean average predicted (mAP) to evaluate their module, in a large object such as bathtub, bed, bookshelf, sofa, and table shows larger epsilon value gives a more significant increase of mAP. With this condition, the system must be adjusted to obtain an impressive result and complicate users who will use the system or algorithm. The result of the clustering step further is processed by convolutional neural network that also consumes raw point cloud data without any discretization in another representation. T. Czerniawski et al [23] used DBSCAN for cluster point cloud data in a building environment. The author also uses normal point information instead of only x, y, and z data and further processed by bagging tree method for classifying the cluster of their labels. Both [22] and [23] use only offline data to evaluate the result of classification.

3. ORIGINALITY

In bin-picking task, most researchers focus on how the result of the pose is almost near perfect then the picking system can do the job well. Instead of only gaining the pose information, this research proposes a

recognition pipeline system that can understand whether the object type is based on the prediction even though the position lies in several different placements such as well separated, well piled, and arbitrary piled which can be seen in Table 2. By using density-based clustering that can well separate various stacked objects and convolutional neural network deep learning method for classifying the objects that consume raw point cloud data instead of interpreting in another representation. By doing so, the system can be understanding the label information of the object and can be further processed for perfect information of poses instead of only the position and orientation information that can be gained from all objects in a scene instead of processing it one-by-one recognition.

4. SYSTEM DESIGN

The main system consists of two main blocks, Reference and Target. As shown in Figure 1, in Reference block is an offline process that resulting a model for recognition purposes. Using CamBoard pico flexx Time-of-Flight camera, point cloud in cartesian coordinates $\mathbb{R}^3$ contained x, y, and z data have been obtained as the input of the system. Then followed by preprocessing and segmentation step, which results from an individual object that is used as the dataset for the training process. After that, the Target block that also offline process, is used to recognize object which the process is the same as the reference block until the segmented data is obtained then become input for the recognizing step.

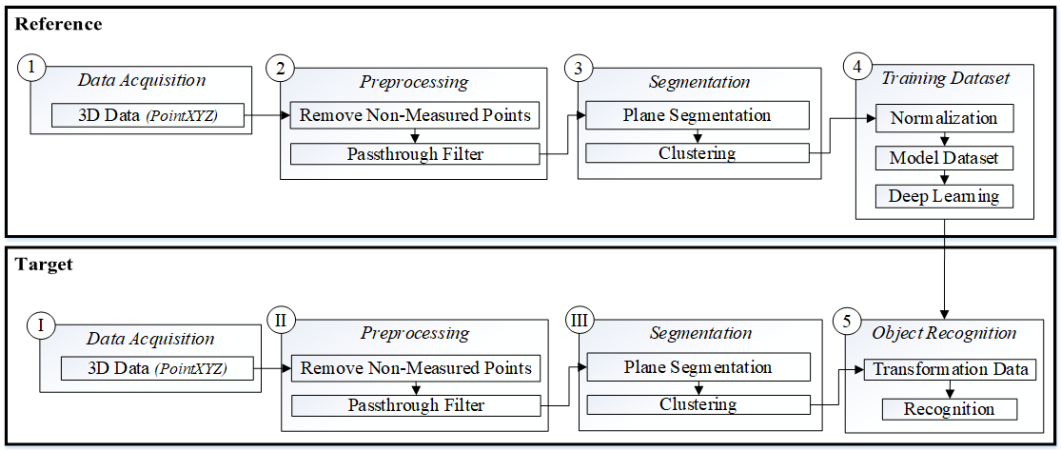


Figure 1. Block diagram of the proposed system

4.1 Preprocessing

Before the data is applied both in reference and target blocks, preprocessing is required to remove the undesirable data that come from camera as Non-measured data (NaN). Suppose that T is point cloud in $\mathbb{R}^3$, A is a scene that covered all the points T, and *i* is an index of point T. Removing non-measured data is formalized in Equation 1 and Equation 2. Then, for simplicity and effectiveness, Passthrough filter is used to dismiss unused static data outside of the box or target area by using Equation 3.

$$T_i \in A \tag{1}$$

$$T_i = \begin{cases} T_i, T_i! = NaN \\ null, otherwise \end{cases} \tag{2}$$

$$T_i = \begin{cases} T_i, LT > T_{i(x,y)} > GT \\ null, otherwise \end{cases} \tag{3}$$

By using two parameters of Passthrough filter *LT* and *GT* for the threshold whether the point is processed or being removed in 2-dimensional cartesian space x and y.

4.2 Segmentation

To get the individual data, plane segmentation is the first thing to do to separate between plane and objects in a scene that uses RAndom SAmple Consensus (RANSAC) algorithm. By choosing three random points for forming plane equation as shown in Equation 4.

$$ax + by + cz + d = 0 \tag{4}$$

a, b, c and *d* are constant of the plane equation and can be obtained using Equation 5 until Equation 8. Then, since the constant of the plane equation is obtained, then followed with Equation 9 to calculate the distance $\mathcal{D}$ of the all-new points in T.

$$a = [((y_2 - y_1)(z_3 - z_1)) - ((z_2 - z_1)(y_3 - y_2))] \tag{5}$$

$$b = [((z_2 - z_1)(x_3 - x_1)) - ((x_2 - x_1)(z_3 - z_2))] \tag{6}$$

$$c = [((x_2 - x_1)(y_3 - y_1)) - ((y_2 - y_1)(x_3 - x_2))] \tag{7}$$

$$d = -(ax + by + cz) \tag{8}$$

$$\mathcal{D} = \frac{ax_4 + by_4 + cz_4 + d}{\sqrt{a^2 + b^2 + c^2}} \tag{9}$$

Where x_4, y_4, z_4 is a new point that will be tested whether included in a plane or as an object. For determining the state of the points, then the threshold is given. If the point is below the threshold, then categorized as a plane and the opposite is categorized as an object. This process is done repeatedly until all points are processed and object can separate by its plane.

After doing the plane segmentation, clustering process is required which aims to separate between each object. In this research, DBSCAN algorithm is chosen because of its ability to handle data with various densities and robust to outliers [20] also works simply and effectively based on density. Some dependencies of DBSCAN are denoted definitions below:

- Eps-neighborhood of point p is denoted by $N_{eps}(p)$ which meet the requirement in Equation 10. Eps is a radius for the search of neighborhood points and $dist(p,q)$ is the Euclidean distance between point p and q that formed in Equation 11.

$$N_{eps}(p) = \{q \in \mathrm{T} \mid dist(p,q) \leq Eps\} \quad (10)$$

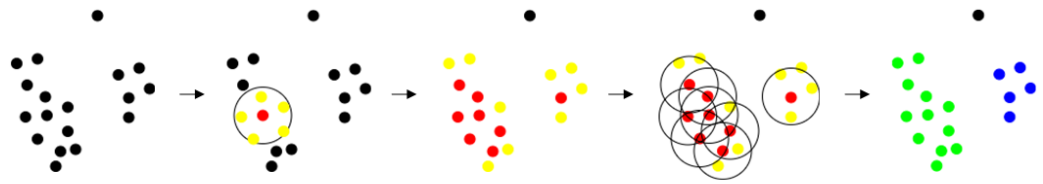
$$ED = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \quad (11)$$

- Directly density-reachable, which the relation of point p and other points wrt. Eps and MinPts. Single point q is directly density-reachable with point p if $q \in N_{eps}(p)$ and point p is a core point that meets Equation 12.

$$N_{eps}(p) \geq MinPts \quad (12)$$

- Density-reachable, which is the relation between point p and q wrt. Eps and MinPts if there are a chain p_1, p_2, p_3, ..., p_n where $p_1=q$, $p_n=p$ and p_{i+1} is directly density-reachable from p_i.
- Density-connected, the relation between point p and q wrt. Eps and MinPts if there are core point o that density-reachable into both two points p and q.
- Cluster, group of points C wrt. Eps and MinPts that meet the conditions stated below:
  - $\forall p,q$: if $p \in C$ and q are density-reachable from p wrt. Eps and MinPts, then $q \in C$ (Maximility).
  - $\forall p,q \in C$:p is density-connected into q wrt. Eps and MinPts (Connectivity).

Instead of using all the points definitions of DBSCAN which is the core point and border point to form a cluster, based on [24] concept of border points is eliminated due to its instability. Firstly, DBSCAN works with searching for core point using Eps-neighborhood statement which is based on Equation 10 and Equation 12 using Eps and MinPts with a certain value.



(a) (b) (c) (d) (e)

Figure 2. (a) Example dataset, (b) Finding core point, (c) Result of core points (d) Forming a cluster, (e) Formed cluster

The illustration of this statement can be seen in Figure 2b which the black point is the input point contained by the data, the red point is the core point, and the yellow point is Eps-neighborhood of the core point.

The process of finding the core point is kept doing until all the input points are done processing. The result of obtained core points shown in Figure 2c was then further processed to form a cluster that made a connection of Eps-neighborhood using a circle with Eps radius as shown in Figure 2d.

The cluster will be formed using density-connected from the core point and Eps-neighborhood. Points that do not meet the requirement of core point and not be Eps-neighborhood points will be noise and not included in any clusters. An example of a formed cluster from DBSCAN is shown in Figure 2e which the green and blue point are represent the formed cluster and the black point is a noise.

4.3 Training and Object Recognition

Object recognition takes an important role in the bin-picking process which can support the system identify which object has been targeted. Using the powerful feature extractor which is Convolutional Neural Network (CNN). Starting with downsampling to equalize the number of points using Voxel Grid that is based on centroid on each axis in a voxel using Equation 13 to represent all points in a voxel.

$$T_{centroid} = \left(\frac{\sum_{i=0}^{n} T_i}{n}\right) \tag{13}$$

Followed by zero-mean filtering, which aims the centered data as input for CNN's because the center distribution of data is avoiding the wrong decision weighting at the training process. Each axis has a different scale of each other, resizing the scale is required to equalize the scale of each feature by using Equation 14.

$$T_{scaled} = \frac{T}{\max\,(T)} \tag{14}$$

Finally, jittering is applied to add the random point between the minimum and maximum of each axis if the downsampling can't afford the stated amount number of points which is 400 in total. All the dataset for training, validation, and testing is stored in an HDF5 format file to accommodate requirement in training using CNN.

PointNet [17] is the first CNN architecture that can be consumed raw point cloud. Considering that condition, the architecture built emphasizes Permutation Invariance and Transformation Invariance. Given the unstructured point cloud data, scanning on N points has N! permutations, in the further process must be invariant to the different representations and output of classification should be unchanged if the object runs into transformation such as rotation and translation.

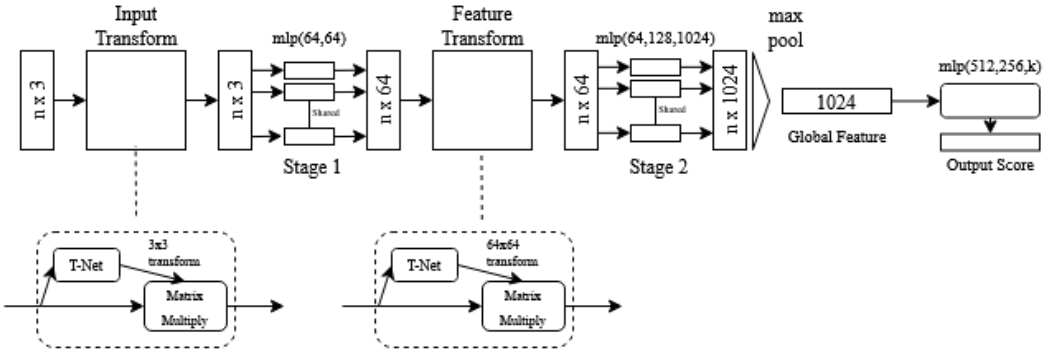


Figure 3. Classification architecture of PointNet

Input transformation built on T-Net and matrix multiplication. Figure 4 shows that the output is the same as originally input data with 3 features that are still handled. Consists of some multi-layer perceptron that includes 1D convolution, batch normalization, max pooling, ReLu for the activation function, and followed by matrix multiplication.

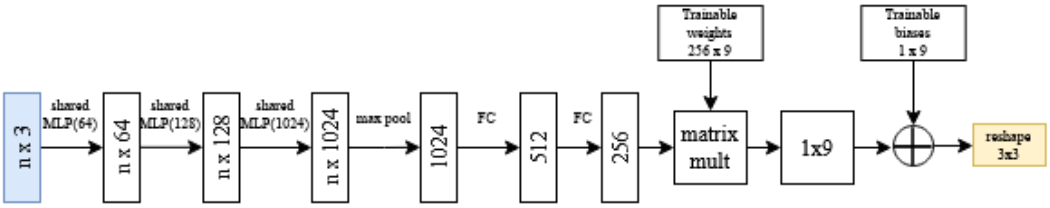


Figure 4. T-Net 3x3 architecture

After the input transformation is done, followed by multi-layer perceptron stage 1 to collect the feature of the data using 1D convolution and batch normalization until getting $\mathbb{R}^{64}$ of features. Then do the feature transformation based on T-Net and matrix multiplication. The difference between T-Net for input and feature transformation is feature obtained is the input obtained from MLP stage 1 in nx64. When the feature transformation is finished, multi-layer perceptron stage 2 is performed and again consist of 1D convolution and batch normalization that give $\mathbb{R}^{1024}$ features and assigned as a global feature for classification layer that builds from neural network layer with normalization and dropout. Then the final feature in $\mathbb{R}^{256}$ is assigned into a fully connected layer followed by dropout to eliminate possibly same interpretation of prediction that can help to avoid overfitting then with softmax classifier to obtain the probability score of each class.

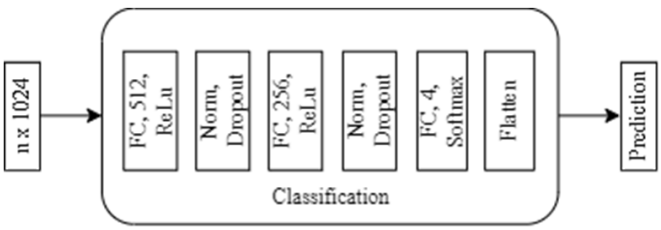


Figure 5. Classification layer

For evaluate the training result, confusion matrix which contain true positive, true negative, false positive and false negative between the actual label and predicted label. It's used to calculate evaluation metrics that represent the performance of the model to recognizing new data.

Table 1. Confusion Matrix

		Predict	
		Label A	**Label B**
Actual	**Label A**	TP	FN
	Label B	FP	TN

Some evaluation metrics that used in this research is accuracy, recall, precision, and F1-Scoore which the formula can be seen in Equation 15 until Equation 18 respectively.

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (15)$$

$$recall = \frac{TP}{TP+TN} \quad (16)$$

$$precision = \frac{TP}{TP+FP} \quad (17)$$

$$F1 - Score = 2 \times \frac{recall \times precision}{recall+precision} \quad (18)$$

5. EXPERIMENT AND ANALYSIS

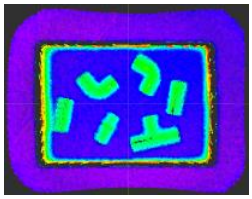
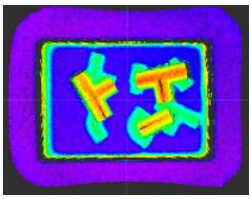
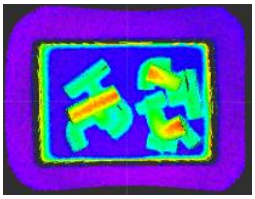
5.1 Data Acquisition

Using CamBoard pico flexx camera, four objects there are Tee pipe, L-pipe, Straight pipe, and Thread pipe. Consist of 3 models of forming the pipe that well separated, well piled, and arbitrary piled with each condition contains 50 data. The data acquisition process is due by placing the objects in a box sized 30.3 cm x 20.8 cm x 8.2 cm in length, width, and height respectively. The camera placed 42 cm perpendicular to the plane and its visualization can be seen in Figure 6.

Figure 6. Visualization of data acquisition

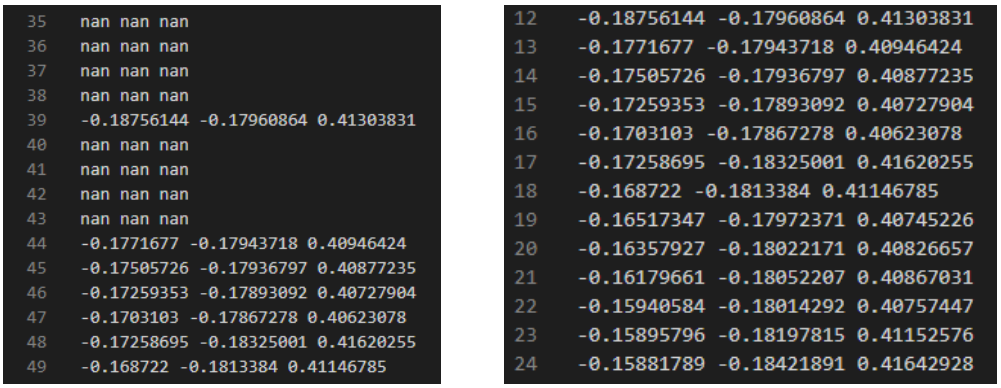
Then Table 2 shows the sample result of acquired data both point cloud and grayscale, but only the point cloud data is used in further process. The color shown in point cloud data is just the visualization of its depth because its obtained data has no color information.

Table2. Acquired data in the grayscale and point cloud with original image

Data	Well Separated	Well Piled	Arbitrary Piled
Original Image			
Grayscale			
Point Cloud			

5.1 Preprocessing

Hence the preprocessing step on removing NaN is not visually available and for the result of NaN in data representation can be seen on Figure 7.

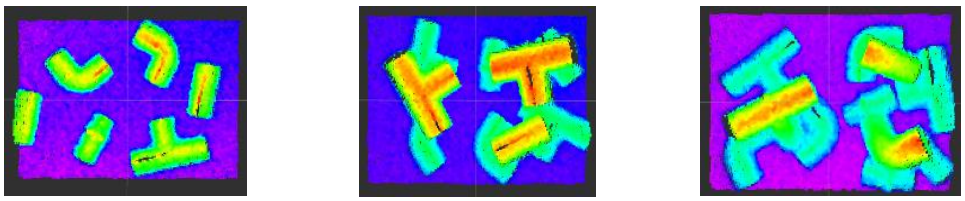


(a) (b)

Figure 7. (a) Point data before remove NaN, (b) Point data after remove NaN

Then the sample of Passthrough filter result can be seen in Figure 8 that remains the inside part of the box which contains some of the placed pipes.

Qualitative analysis by looking forward in each result is used to measure the success of this step. Consist of 50 data in each way of placing, the accuracy hit 100% for all conditions.

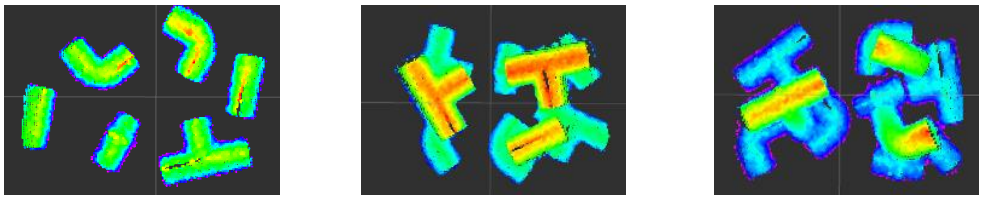


(a) (b) (c)

Figure 8. Sample visualization result on passthrough filter. (a) Well separated, (b) Well piled, (c) Arbitrary piled

5.2 Segmentation

To obtain objects individually, plane segmentation is the first-time task to separate between objects and their plane. By using the RANSAC algorithm, all of the objects in each condition can be successfully separated by their plane and can be processed by separating each object. The example visualization of the plane segmentation result can be seen in Figure 9.



(a) (b) (c)

Figure 9. Sample visualization result of plane segmentation, (a) Well separated, (b) Well piled, (c) Arbitrary piled

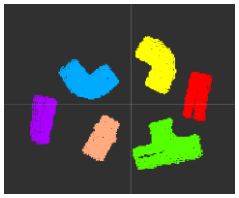
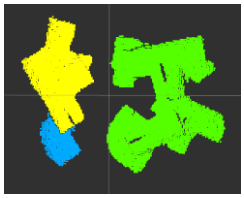
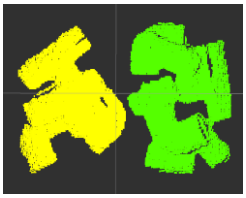
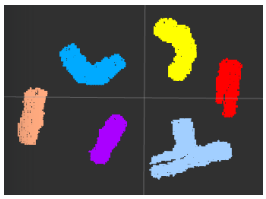
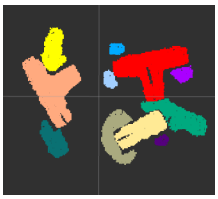
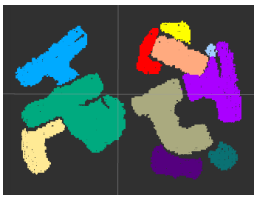
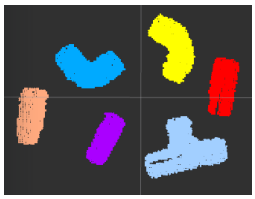
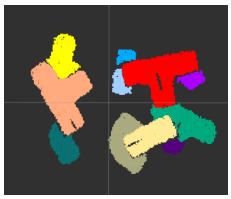
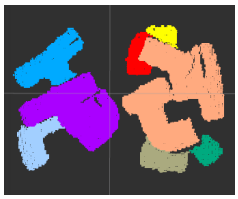
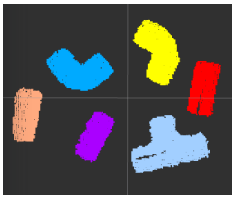
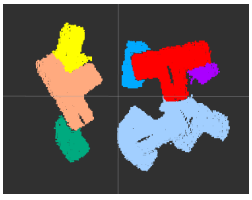
Then, is followed by a clustering step to obtain objects individually that can be used for the recognition task whether in the training process or testing process. In this research, the proposed algorithm which DBSCAN with various parameters ε and *m* as minpts are compared by a novel clustering algorithm that is based on distance only and usually called by Euclidean clustering. The accuracy for each algorithm is summarized in Table 3.

Table 3. Accuracy comparison of algorithms in clustering step

Algorithm	Accuracy (%)		
	Well Separated	Well Piled	Arbitrary Piled
Euclidean Clustering	98.55	34.58	15.13
DBSCAN (ε=0.008, *minpts*=40)	97.46	**98.83**	**80.43**
DBSCAN (ε=0.01, *minpts*=50)	**99.67**	92.80	58.47
DBSCAN (ε=0.012, *minpts*=60)	97.39	75.17	27.36

From Table 3, can be seen that by using DBSCAN algorithm, clustering accuracy is sharply increased than the Euclidean clustering, especially in well piled and arbitrary piled conditions. Table 4 shows the different result between the algorithms and emphasize the advantage of DBSCAN in handling stacked object. Meanwhile for the Euclidean clustering only considering the distance to form a cluster that can't handle that situation. The best accuracy for well piled and arbitrary piled is on 0.008 epsilon and 40 while on well separated 0.01 epsilon and 50 MinPts give the best accuracy.

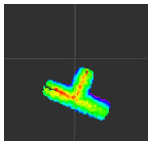
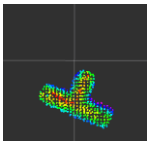
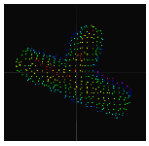
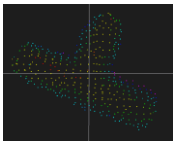
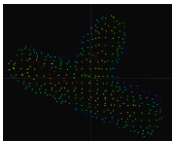
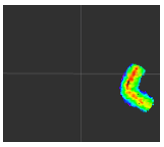
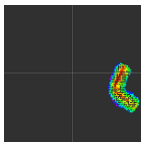
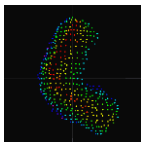
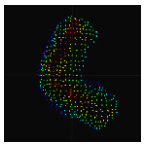
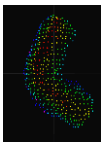
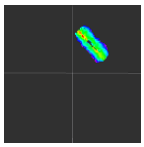
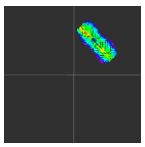
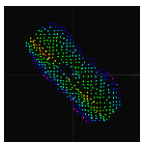
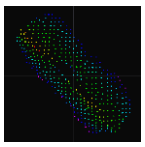
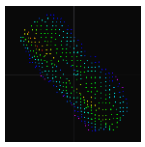
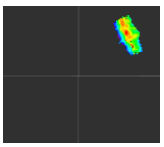
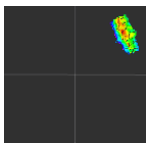
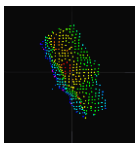
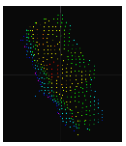
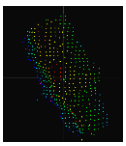
Table 4. Visualization of difference on clustering algorithm

Algorithm	Well Separated	Well Piled	Arbitrary Piled
Euclidean clustering			
DBSCAN (ε=0.008, *minpts*=40)			
DBSCAN (ε=0.01, *minpts*=50)			
DBSCAN (ε=0.012, *minpts*=60)			

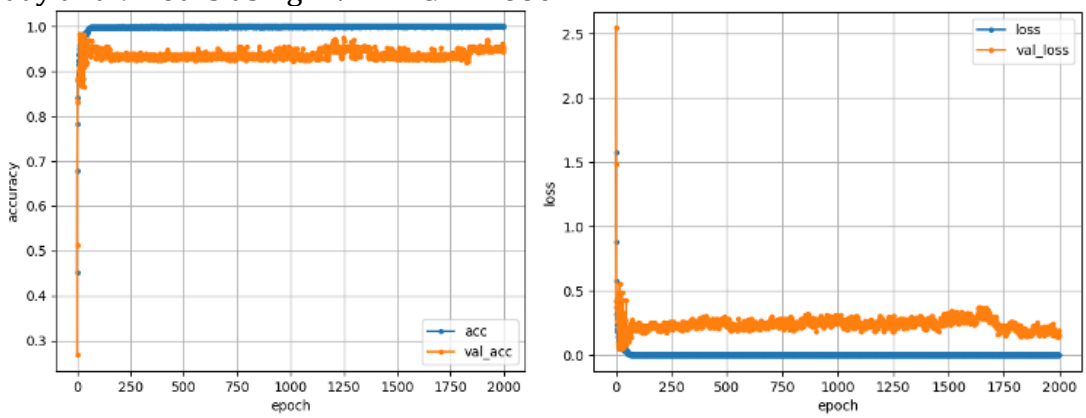
5.3 Training and Object Recognition

Before the training process begins, the dataset obtained from the clustering process needs to be preprocessed which contains downsampling, zero-mean filtering, resize, and jittering. The example results of preprocessed dataset are shown in Table 5.

Table 5. Preprocessing result step for training

No	Input	Downsampling	Zero-Mean Filtering	Resize	Jittering
1					
2					
3					
4					

After the data is preprocessed, then storing all the data in the .h5 file that is used for training, validation, and testing. Respectively, the total used data is 689, 60, and 176 divided into 4 objects. By using PointNet CNN's architecture, and some hyperparameters like the batch size in 32, learning rate at 0.0001, and using 2000 epoch. The optimizer used is the Adam optimizer, then the loss function is cross-entropy loss. Training time took 1 day and 9 hours using NVIDIA GTX 1660.



(a) (b)

Figure 10. (a) Graph of training accuracy, (b) Graph of training loss

After training finished, as can be seen in Figure 10a the model accuracy on training reach 99.99% and the validation accuracy reach 94.52%. With this result, a model is good enough to recognize the validation data considering the number of data used is not too much.

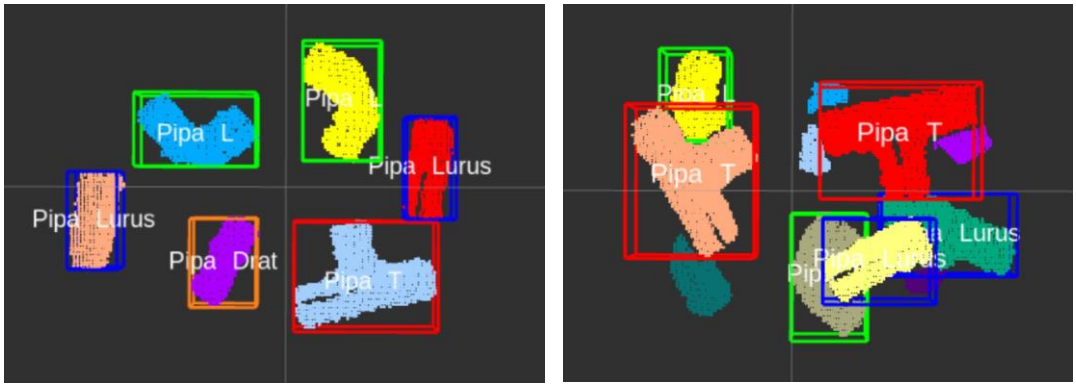
Besides the accuracy, another metric that was evaluated is loss and the graph can be seen in Figure 10b. Loss value indicates the probabilities between the actual and the predicted label. The loss on training got 0.00025 and the validation loss got 0.19. Although the difference is notable, for validating the model is robust into new data, testing is done and can be evaluated from confusion matrix that shown in Table 6.

Table 6. Confusion matrix of testing process

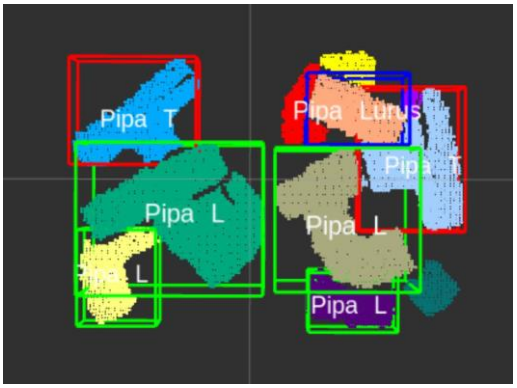
Actual \ Predict	Tee Pipe	L-Pipe	Straight Pipe	Thread Pipe
Tee Pipe	44	0	0	0
L-Pipe	2	40	2	0
Straight Pipe	0	0	44	0
Thread Pipe	0	1	3	40

From the confusion matrix in Table 6, can be obtained the evaluation on Accuracy, Recall, Precision, and F-1 Score that the calculation is based on Equation 15 until Equation 18 for Tee pipe are 1, 1, 1, and 1 respectively. For L-pipe are 0.9716, 0.9090, 0.9756, and 0.9411. Next, Straight pipe are 1, 1, 1, and 1. Finally, for Thread pipe are 0.9773, 0.9090, 1, and 0.9523.

Furthermore, the obtained model with good performance based on the above evaluation can be used to recognize multiple objects in a scene which can be seen on Figure 11.



(a) (b)



(c)

Figure 11. Sample multiple objects recognition in a scene. (a) Well separated, (b) Well piled, (c) Arbitrary piled

6. CONCLUSION

We have developed a pipeline for segmentation and recognition system on stacked pipe objects using density-based clustering and PointNet's CNN algorithm respectively. Based on the obtained result and evaluation, can be inferred that the system can be perfectly done for preprocessing and plane segmentation of all conditions, while the clustering steps which used the DBSCAN algorithm has the accuracy percentage of 99.67% for well separated, 98.83% for well piled, and 80.84% for arbitrary piled.

For the training process, the systems can validate data in 94.52% accuracy and 0.19 loss. Then from the good model obtained from the training process, the recognition step hit the average Accuracy, Recall, Precision, and F1-Score on 98.72%, 95.45%, 99.39%, and 97.33% respectively. This condition made the model can be robust on new data that will be further tested on the system and return highly accurate recognized objects. Also, the obtained model can be used for recognizing multiple objects in one scene that in this condition made the proposed pipeline further can be used in bin picking on industrial applications to recognize the target object.

Acknowledgments

This research was partially funded by Ministry of Education, Culture, Research, and Technology Indonesia under the Penelitian Dasar Unggulan Perguruan Tinggi (PDUPT) program, contract number 082/E4.1/AK.04.PT/2021, with the title is "B-FLoW, bipedal humanoid sebagai platform kolaborasi manusia dan robot dalam tata gerha". This research also supported by the Robotics and Intelligent Systems Center (RoISC), Politeknik Elektronika Negeri Surabaya.

REFERENCES

[1] Le T-T and Lin C-Y, **Bin-Picking for Planar Objects Based on a Deep Learning Network: A Case Study of USB Packs**, *Sensors*, vol. 19, no. 16, 2019.

[2] Yan W, Xu Z, Zhou X, Su Q, Li S, and Wu H, **Fast Object Pose Estimation Using Adaptive Threshold for Bin-Picking**, *IEEE Access*, vol. 8, pp. 63055-63064, 2020.

[3] Sari DM, Pratama AR, Pramadihanto D, and Marta BS, **3D Object Detection Based on Point Cloud Data**, *Inform : Jurnal Ilmiah Bidang Teknologi Informasi dan Komunikasi*, vol. 7, no. 1, pp. 59-66, 2022.

[4] Kameshwaran K and Malarvizhi K, **Survey on Clustering Techniques in Data Mining**, *(IJCSIT) International Journal of Computer Science and Information Tehcnologies,* vol. 5, no. 2, pp. 2272-2276, 2014.

[5] Rusu RB, **Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments**, Ph.D thesis, Technische Universität München, 2010.

[6] Ester M, Kriegel H-P, Sander J, and Xu X. A, **Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise**, *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining*, pp. 226-231, 1996.

[7] Lim GH, Lau N, Pedrosa E, Amaral F, Pereira A and Luís Azevedo J, **Precise and efficient pose estimation of stacked objects for mobile manipulation in industrial robotics challenges**, *Advanced Robotics*, vol. 33, no. 13, pp. 636-646, 2019.

[8] Zongming L, Jianxun L, Guodong L and Dong Y, **Pose Estimation of Rigid Object in Point Cloud**, *9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics*, pp. 708-713, 2016.

[9] Tombari F, Salti S, and Stefano LD, **Unique Signatures of Histograms for Local Surface Description**, *ECCV'10: Proceedings of the 11th European Conference on Computer Vision*, pp. 356-369, 2010.

[10] Rusu RB, Blodow N and Beetz M, **Fast Point Feature Histograms (FPFH) for 3D registration**, *2009 IEEE International Conference on Robotics and Automation*, pp. 3212-3217, 2009.

[11] Rusu RB, Bradski G, Thibaux R and Hsu J, **Fast 3D recognition and pose using the Viewpoint Feature Histogram**, *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2155-2162, 2010.

[12] Wohlkinger W and Vincze M, **Ensemble of shape functions for 3D object classification**, *2011 IEEE International Conference on Robotics and Biomimetics*, pp. 2987-2992, 2011.

[13] Fernandes D, Silva A, Névoa R, Simões C, Gonzalez D and Guevara M, **Point-cloud based 3D object detection and classification methods for self-driving applications: A survey and taxonomy**, *Information Fusion*, vol. 68, pp. 161-191, 2020.

[14] Khaliluzzaman Md, Abu Bakar Siddiq Sayem Md, and Misbah KL, **HActivityNet: A Deep Convolutional Neural Network for Human Activity Recognition**. *EMITTER International Journal of Engineering Technology*, vol. 9, no. 2, pp. 357-376, 2021.
[15] Bello SA, Yu S, Wang C, Adam JM and Li J, **Review: deep learning on 3D point clouds**, *Remote Sensing,* vol. 12, no. 11, pp. 1729, 2020.
[16] Guo Y, Wang H, Hu Q, Liu H, Liu L and Bennamoun M, **Deep Learning for 3D Point Clouds: A Survey**, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 12, pp. 4388-4364, 2020.
[17] Qi CR, Su H, Niessner M, Dai A, Yan M and Guibas LJ, **Volumetric and Multi-View CNNs for Object Classification on 3D Data**, *2016 IEEE Conferene on Computer Vision and Pattern Recognition (CVPR)*, 2016.
[18] Zhang Y and Rabbat M, **A Graph-CNN for 3D Point Cloud Classification**, *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.
[19] Qi CR, Su H, Mo K and Guibas LJ, **PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation**, *2017 IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
[20] Hanh LD and Duc LM, **Planar Object Recognition For Bin Picking Application**. *2018 5th NAFOSTED Conference on Information and Computer Science (NICS)*, pp. 211-215, 2018.
[21] Sun Z, Li Z and Liu Y, **An Improved Lidar Data Segmentation Algorithm Based on Euclidean Clustering**, *Proceedings of the 11th International Conference on Modelling, Identification and Control (ICMIC2019)*, pp. 1119-1130, 2019.
[22] Ahmed SM and Chew CM, **Density-Based Clustering for 3D Object Detection in Point Clouds**, *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10608-10617, 2020.
[23] Czerniawski T, Sankaran B, Nahangi M, Haas C and Leite F, **6D DBSCAN-based segmentation of building point clouds for planar object classification**, *Automation in Construction*, vol. 88, pp. 44-58, 2018.
[24] Campello RJGB, Moulavi D and Sander J, **Density-Based Clustering Based on Hierarchical Density Estimates**, *Advances in Knowledge Discovery and Data Mining*, vol. 1819, pp. 160-172, 2013.