Computing and Informatics, Vol. 41, 2022, 1446–1464, doi: 10.31577/cai_2022_6_1446

POINTVOTES: A DEEP LEARING POINT CLOUD MODEL FOR TIRE BUBBLE DEFECT DETECTION

Hualin Yang, Yuanzheng Jiang, Wenxue Nie, Fang Deng*

College of Mechanical and Electrical Engineering
Qingdao University of Science and Technology
266100 Qiangdao, China
e-mail: `younghualin@163.com, 4019030009@mails.qust.edu.cn, {nwx19991004, dengfhelen}@163.com`

Maozhen Li

Brunel University London
Kingston Lane, Uxbridge, Middlesex UB8 3PH
London, UK
e-mail: `Maozhen.Li@brunel.ac.uk`

Abstract. In order to eliminate the hidden dangers caused by tire bubble defects, considering that the two-dimensional technology is sensitive to light, the 3D point cloud technology is used to obtain the tire surface morphology. This paper proposes a 3D point cloud network model named PointVotes, a point based target detection method. The designed structural framework includes: the fusion sampling layer, the voting layer and the proposal refinement layer. By observing the spatial characteristics of the detected target, a new point sampling method named C-farthest point sampling (C-FPS) is proposed. Combining with the fusion sampling strategy, the FPS and the C-FPS are sampled in a certain proportion. It solves the problem that the proposal box cannot be generated due to less available prospect information when generating suggestions for small targets. The network model uses Set Abstraction layers in multiple PointNet++ to extract features, arranges and combines features of different scales, forms high-dimensional features of points and votes, judges whether there are bubble defects through classification, and then generates proposals and regression to the prediction frame. Experiment results show that

* Corresponding author

the mean average precision of the model can reach 82.8 % with a detection time of 0.12 s.

Keywords: 3D point cloud, target detection, point sampling method, neural network, vote

1 INTRODUCTION

In recent years, 3D point cloud technology has become a more and more important element in the field of computer vision, it has been applied in many directions, especially in auto driving scenes and target detection. Compared with 2D technology, 3D point cloud has natural advantages in expressing the whole picture of the target, which is insensitive to illumination changes and has good robustness. Its accuracy is directly determined by the point cloud sensor.

The structure of automobile tire is complex. The bubble defects shall be easily produced due to the influence of materials and production technology in the formation process. As shown in Figure 1, its existence will seriously endanger the driving safety of automobiles, thus it must be eliminated in time. Considering the advantages of 3D point cloud, Kinect 2.0 is used as a sensor to obtain the dense point cloud of the target. The difference between point cloud and image lies in the irregularity of point cloud, which makes the traditional convolutional neural network (CNN) unable to deal with point cloud directly. However, current point cloud processing methods, whether voxel, volume, mesh or Bird's Eye View (BEV), have the phenomenon of synthesizing local point areas into one point. Although these methods can generate proposals for targets with the help of Region Proposal Network (RPN) in real time, these methods will lose local information and are not obvious in some places with subtle spatial changes. Therefore, they are not conducive to the detection of small targets. Herein, the point-based method is used to make up for the local information and avoid the loss of information.

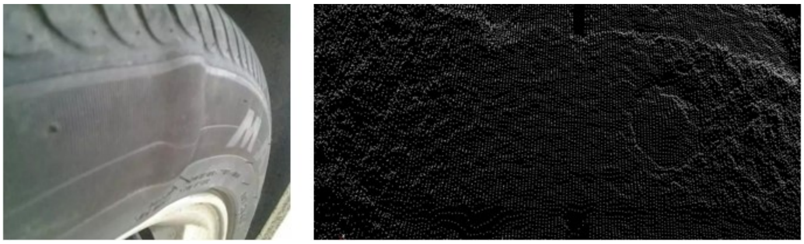

Figure 1. Schematic diagram of tire bubble defect

Inspired by [1], this paper proposes a defect detection method using 3D point cloud named PointVotes. It can directly process the original point cloud, extract the final center point, arrange and combine the features of different scales and vote to predict the possible defects. By observing the spatial characteristics of the detected targets, a new point sampling method named C-farthest point sampling (C-FPS) is proposed. Combining with the fusion sampling strategy, the FPS and C-FPS are sampled in a certain proportion, which solves the problem that the proposal box cannot be generated due to the less available foreground information of small targets when generating suggestions. The Set Abstraction (SA) layer in multiple PointNet++ is used to extract features, arrange and combine the features of different scales to form the high-dimensional features of points, and to vote, group and propose the cluster points after voting.

The main contributions of the paper are as follows.

1. PointVotes samples the point cloud geometry to preserve the edge points of the defect.
2. Unlike voxels or BEV, PointVotes can save the local feature information of the point cloud and indirectly increase its detection accuracy by directly operating the original point cloud. Compared with other networks, it has a good effect on detecting target defects in self-made data sets, and its detection time reaches 0.12 s.

The rest of the paper is organized as follows. Section 2 introduces a number of related works. Section 3 presents the PointVotes network architecture and details the implementations of the components. Section 4 validates the proposed work and describes the experimental settings and results. Section 5 concludes the paper.

2 RELATED WORK

The target detection network can be divided into one stage and two stages. Compared with the first stage, the second stage has more selection and regression of the proposal box, while the first stage is usually the direct prediction result, so the detection accuracy of the second stage is higher than the first stage. At present, the target detection network is mostly used in traffic, furniture and other directions, and this paper uses the self created data set for bubble defect target detection.

Although the accuracy of convolution is similar to that of 2D image, it is also more complex than that of 3D image. In 2014, RCNN [2], as a pioneering work, applies convolutional neural network to extract the features of 2D images. Fast RCNN [3] proposes feature pooling to realize the end-to-end training process and greatly improves the training speed. Later, the faster RCNN [4] improves the detection speed to achieve real-time detection. In particular, the RPN network proposed by fast RCNN also has many applications in target detection of 3D point cloud, and can even be used as the basis of two-stage target detection.

At present, 3D target detection can be roughly divided into the following categories.

1. Point based methods. PointNet [5] solves the difficult problems of point cloud, and PointNet++ [6] adds the extraction of local features as an improvement of PointNet. Then PointNet and PointNet++ are widely used in various point cloud models. VoteNet uses the SA layer as the backbone to generate central points and vote, and groups the voted points to form clusters for generating suggestions. It has good results in ScanNet [7] and SUN RGB-D [8] data sets. PointRCNN [9] first separates the foreground and background of the point cloud, standardizes the coordinates of each local point, and completes the refinement of the bounding box in combination with the global features. As a single-stage detector, 3DSSD [10] takes PointNet++ as the backbone layer and uses improved SA layer fusion sampling to replace the removed FP layer, which greatly improves the detection speed on the premise of considerable performance. The sampling strategy of StarNet [11] is different from that before. When selecting the center point, it only generates two-dimensional center point coordinates in a Z-dimensional plane. PointConv [12] regards the weight of convolution kernel as a continuous function, and constructs convolution network by combining local coordinates, density and input characteristics of points. ELF-Net [13] proposes Local Points Encoding Module (LPEM), which is used to encode the information of eight orientations and 3D coordinate information of local points.

   Point based method is a direct point cloud processing algorithm, which is also a more widely used method. Its core is to directly operate the local and global features of points and better retain the detailed information. This study is also designed based on point clouds and voting to detect the target.

2. Voxel or mesh based method. In this method, all point clouds are divided into a voxel or three-dimensional grid, and then detected by CNN. As a classic work of voxel target detection, literature [14] proposed the characteristics of voxels encoded in VFE layer and combined with RPN to generate detection. STD [15] and SECOND [16] also use VFE to encode voxels. Literature [17] applies CNN to voxels, and the weight of the voting mechanism is obtained from the weight of flip convolution, while document [18] votes on the divided grid in the form of sliding window. PV-RCNN [19] down sampled 1/16 of voxels to obtain the characteristics of each stage and aggregated the characteristics in combination with BEV, while literature [20] down sampled only 1/8. PointPillars [21] represented the point cloud with pillars, stacked the sparse pillars to form dense pillars, and used efficient 2D convolution operation. However, whether it is voxel, volume or mesh, the local regional features of points will be lost, which will reduce the detection accuracy. EPC-Net [22] proposes a lightweight network module to aggregate the local geometric features for lower memory consumption. GSV-NET [23] converts the regions of the 3D point cloud into color representation and captures region features with a 2D wide-inception network.

3. Method based on aerial view. BirdNet [24] generates the BEV view from the point cloud. The maximum height of the midpoint of each grid represents the height information. The down sampling method is similar to PV-RCNN [19]

and PIXOR [25]. YOLO3D [26] takes the BEV diagram as the input to establish the height and density of a grid containing point. However, the disadvantage of this method is that the generation of BEV will lead to a large loss of Z-dimensional information, which will reduce the network performance. Literature [27] also projects the preprocessed point cloud onto a 2D surface and generates masks.

4. Method of combining RGB image and point cloud. Literature [28, 29, 30, 31] combines the feature information obtained by 2D detector in RGB image with BEV representation or original 3D point cloud to generate suggestions, so as to save the detection time. Literature [32] transforms 3D point cloud into front view and aerial view as input and convoluted to extract features. Literature [33] adopts multiple views and adds the mlpconv layer to obtain the characteristics of the object. This method may cause loss in the process of combining 2D and 3D, and taking into account RGB image and point cloud at the same time will inevitably lead to the decline of detection speed. Literature [34] fuses the point cloud and RGB image, and voxelizes the raw point cloud which can form a frustum. Literature [35] uses Res2Net to extract the features from multiple 2D views and achieves higher classfication accuracy and better performance.

3 NETWORK FRAME

As shown in Figure 2, the PointVotes architecture described in this paper is mainly composed of three parts: the fusion sampling layer, the point voting layer and the proposal and refinement layer. Fusion sampling uses SA layer to sample under the point cloud, and uses different sampling methods to sample according to a certain proportion, so that some front scenic spots can be retained. The voting layer arranges different scale characteristics of the points, votes and groups each central point. The last part is the generation of proposals and the regression of the prediction box.

3.1 Fusion Layer

As mentioned in Section 2, most of the steps of the point-based method include the process of proposal generation and box refinement. In this study, the SA layer in PointNet++ is used as the backbone to sample the original point cloud and obtain feature information. An SA layer includes feature point sampling, grouping and local feature extraction (PointNet).

The common point cloud sampling method is the farthest point sampling (FPS), which is a uniform sampling method. Although some local information can be retained, the sampling effect is general for places with curvature. Moreover, since the detection defect target in this paper is small, FPS may take less front scenic spots, which will result in poor detection effect. Therefore, this paper proposes

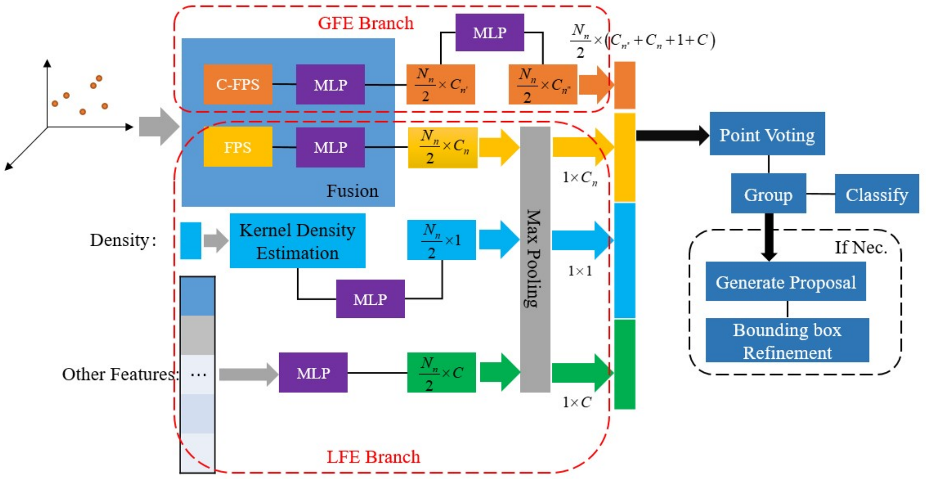

Figure 2. Schematic diagram of PointVotes network model

a sampling method, named C-farthest point sampling (C-FPS), based on curvature features extracting in Euclidean space, which can effectively retain the front scenic spots with small detection targets.

In the Euclidean space, N_s represents neighborhood with s number of points at point $N_i, i \in R$ of point cloud, the surface normal vector at point N_i is v_i, point x_j represents other points in the neighborhood, and its surface normal vector is v_j. Then the equation can be expressed as $\theta(i,j) = a\cos\left(\frac{v_i v_j}{||v_i||||v_j||}\right)$, $\theta(i,j) \in (0,\pi)$. As shown in Figure 3, when there are no defects on the surface of the point cloud, the normal vector of each point will be parallel and the included angle will be 0° in the ideal state. When there is a defect on the surface of the point cloud, the value of part $\theta(i,j)$ will change in the edge neighborhood of the defect. Setting the threshold $\theta(i,j)$ to 10°, if the value exceeds the threshold, the point is regarded as the front scenic spot; else if the value is less than the threshold, it will be regarded as the background point. Therefore, the points in each neighborhood with such characteristics can be retained, which will greatly increase the number of foreground points in the example, remove a large number of useless background points, and provide favorable conditions for subsequent detection.

However, if all C-FPS are used to sample the point cloud, a large number of front scenic spots will be concentrated in the target area, and duplicate proposal boxes will be generated in a small range, which will result in reduced accuracy and efficiency. In order to obtain sufficient points in the sampling process, this work uses the fusion sampling strategy to collect a certain proportion of points in the sampling layer of SA by FPS and the proposed C-FPS, as shown in Equation (1).

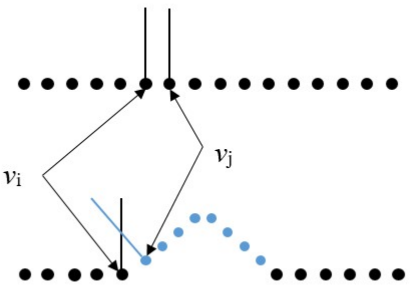

Figure 3. Arrangement diagram of normal and bubble defects

$$N_{total} = \lambda N_{C\text{-}FPS} + \beta N_{FPS} \lambda + \beta = 1 \tag{1}$$

where N_{total} represents total number of points, $N_{C\text{-}FPS}$ and N_{FPS} represent the set of points extracted using the methods of C-FPS and FPS, respectively, λ and β represent the corresponding sampling proportion. This method preserves more front spots for smaller target detection, thus has enough background points for classification, and does not use FP layer to preserve front spots. Furthermore, it can provide the sampled points as input to subsequent grouping steps. Generally speaking, the problem of large gap between foreground and background points due to small detection targets still exists. This study uses the focus loss function [36] to balance the gap between the number of points.

In this paper, the feature extraction of the input point cloud is completed with the help of multi-layer perceptron. After completing the C-P fusion sampling, the density and other features of $\frac{N_n}{2}$ number of points are added. The density characteristics are estimated by kernel density, which is represented by $\frac{N_n}{2} \times 1$. Other features of points are represented by $\frac{N_n}{2} \times C$.

Considering that the number of each group of point clouds in the data set is about 24 000, and the number of corresponding annotation box points is about 200, 1/20 of the original point number is initially taken as the network input, which is the ratio of the number of points included in the detection target to the total number of points. In order to make up for the shortage of former scenic spots as much as possible, set an appropriate number of SA layers to 3, as shown in Table 1. Table 2 shows the comparison between four layers of SA and three layers of SA. When $n = 4$, the parameters increase, thus the running time and training time also increase accordingly, and the mean average precision (mAP) is also higher than that of $n = 3$.

The input point cloud can be expressed as $N \times 3$. Let $\lambda = \beta = 1/2$, that is, half quantity points are used for FPS and C-FPS respectively. After the first feature extraction in the C-P fusion sampling layer, the outputs of the two methods are expressed as $\frac{N_1}{2} \times C_1$ and $\frac{N_1}{2} \times C_1'$, respectively, and half number of points are taken,

SA_n	Input Point Feature	Output Point Feature	MLP_n	Search Radius/m
$n = 1$	about $24\,000 \times 3$	$1\,024 \times (3 + 256)$	$(128, 128, 256)$	0.01
$n = 2$	$1\,024 \times (3 + 256)$	$512 \times (3 + 512)$	$(256, 256, 512)$	0.02
$n = 3$	$512 \times (3 + 512)$	$256 \times (3 + 512)$	$(256, 256, 512)$	0.04

Table 1. Parameters of SA layer

Number of Layers	Running Time/s	mAP/%
$n = 3$	0.12	82.80
$n = 4$	0.17	83.15

Table 2. Performance comparison of SA layers with different layers

respectively. After the second feature extraction, it can be expressed as $\frac{N_2}{2} \times C_2$ and $\frac{N_2}{2} \times C_2'$, and half of the points are taken again. Until it passes through the n^{th} feature extraction layer, the output includes N_n points, which is composed of two parts. One part is generated by FPS, represented by $\frac{N_n}{2} \times C_n$, and the other part is generated by C-FPS, represented by $\frac{N_n}{2} \times C_n'$. They use different sampling methods, so their characteristics will be different. So use C_n and C_n' to distinguish them. Therefore, the final result consists of $\frac{N_n}{2} \times C_n'$ from GFE Branch and $\frac{N_n}{2} \times (C_n + C + 1)$ from LFE Branch.

3.2 Voting Layer of Points (PV Layer)

In order to ensure the detection accuracy and fully capture the local and global features of points, this section arranges and combines the points after feature extraction at different scales to form a new feature vector set (similar as [18]). Its output is a feature oriented quantum set of n points. Each point in the subset will correspond to a vote, and each vote will be voted through the MLP network. The voting layer of points can be divided into two branches: the local feature branch (LFE branch) and the global feature branch (GFE branch).

LFE branch consists of three parts to extract global features at different scales.

1. In this paper, parts of N_{FPS} points and their characteristics after FPS sampling are retained, and the output through the maximum pool layer is regarded as a global feature. This global feature is added to the feature vector set as the first part of the new feature vector set.
2. Due to the density change of the detected defect target in some local areas, this paper adds the density feature of points and other features as the subsequent part of the feature vector set. For the density features of points, the kernel density is used to estimate the density features, and the density features and other features are extracted into the vector set with the help of multi-layer perception and maximum pooling operation.

The function of GFE branch is to extract the points processed by C-FPS as local features. After the original point cloud is processed in the SA_n layer (multiple SA layers) by using the fusion sampling strategy, a subset $\{x_i | x \in (1, 2, \ldots, n)\}$ and corresponding features processed by the C-FPS sampling method are obtained. Taking this subset as the initial center point, the initial center point moves relatively under the supervision of the loss function. This process is the same as VoteNet, it can generate the final center points. These points will be output as local features which shall be added in the LFE branch.

3.3 Refinement Layer

In the point cloud sampling process, even after the sampling process described in Section 3.1, some points in the point cloud are not accurately located on the target surface position they should represent, or even deviate far, and the existence of the error will make some background points detected as front scenic spots. This error point is called pseudo front scenic spot. This problem is caused due to the accuracy error of the depth camera and the error caused by shooting factors, which results in non-ideal position between the point clouds. In this situation, the proposal is generated after voting, which undoubtedly does great harm to the detection results, and detecting such pseudo defects in the point cloud without target defects will seriously reduce the efficiency.

To solve this problem, the following methods are used. Since the detection type of this paper is defect detection, the target is single. It belongs to two categories of target detection. There are only bubble defects and background in the required categories. In order to avoid generating and refining the proposal frame for each collected 3D point cloud, and to save the detection time, this paper proposes a simple method to judge whether there is a defective target in the frame point cloud, and put the classification of the target object before the 3D box proposal. If the target object is classified, then make the proposal and refinement, otherwise the process will not take effect.

Firstly, the cluster points $C_P = \{C_i, F_i\}, i = 1, \ldots, n$ and the centroid b of the cluster points are defined. Among them, $C_i \in R^3$ represents the coordinates of cluster points and $F_i \in R$ represents the characteristics of cluster points. If there are defective targets in the point cloud, the number of cluster points generated is much larger than that generated by the point cloud without defective targets. Defining $C_i - b$ and applying it to transform the cluster points into the centroid local normalized coordinate system. Since the cluster points of defective targets are relatively concentrated, taking the centroid as the center point, for Euclidean space with defective targets, there exists $r_i = \frac{\sum_{n=1}^{i} ||C_i - b||}{i} < r$, which means that the average value of the neighborhood point and the central point near the center point is less than a threshold value r. However, the cluster points without defect targets will irregularly disperse in the whole point cloud space, so there will be a small amount of characteristics and a large r_i.

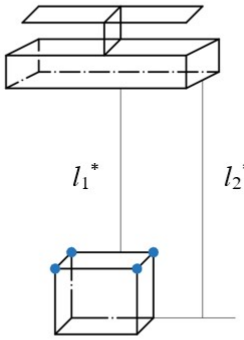

Figure 4. Schematic diagram of improved bounding box coding method

Each point in the cluster generates a 3D bounding box. Since it is relatively complex to directly encode the eight corners of the bounding box, the coding method of the bounding box needs to use the ground plane given by the sensor. The experimental device in this paper is carried out on the cylinder, thus the ground plane is a curved surface that cannot be given. We still use the method similar to [29] to encode the four corners and two heights of the bounding box to reduce the high-dimensional number to 10, as shown in Figure 4. The difference is that the two heights refer to the distance from the upper and lower planes of the bounding box to the sensor (it can be obtained by the sensor), and the four corners are the four points of the upper plane of the bounding box, so the target to be regressed is $(x_1^* \sim x_4^*, y_1^* \sim y_4^*, l_1^*, l_2^*)$. The information useful to us is distinguished by comparing the IoU (Intersection over Union) between the bounding box and the ground real box. If the IoU is less than 0.4, it is considered as the background point; other if the IoU is greater than 0.6, it is divided into the front scenic spot. For redundant bounding boxes, 3DNMS is used to improve the accuracy.

3.4 Loss Function

After the 3D bounding box is proposed, the target to be regressed is seven parameters such as $(x_\mu, y_\mu, z_\mu, l_\mu, w_\mu, h_\mu, \theta_\mu), \mu \in (g, a)$, where (x, y, z) represents the center coordinate value of the box, (l, w, h) represents the size of the box, θ represents the angle, the subscript g indicates the real box of the ground and the subscript a indicates the bounding box. The regression residuals of the ground real box and bounding box can be expressed as Equation (2).

$$\begin{aligned}
\Delta x &= \tfrac{x^{gt}-x^{an}}{d^{an}}, & \Delta y &= \tfrac{y^{gt}-y^{an}}{d^{an}}, & \Delta z &= \tfrac{z^{gt}-z^{an}}{h^{an}}, \\
\Delta w &= \log \tfrac{w^{gt}}{w^{an}}, & \Delta l &= \log \tfrac{l^{gt}}{l^{an}}, & \Delta h &= \log \tfrac{l^{gt}}{l^{an}}, \\
& & \Delta \theta &= \sin \theta^{gt} - \theta^{an} & &
\end{aligned} \tag{2}$$

where $d^{an} = \sqrt{(w_a)^2 + (l_a)^2}$, the superscript gt is the real ground frame and the superscript an is the bounding box. The regression residual loss of the two can be expressed as: $L_{reg} = \sum(Smooth - L_1(\Delta(x, y, z, l, w, h, \theta))$.

Due to the large difference between the number of foreground and background points, the focus loss function is used in the classification loss, which can be expressed as: $L_{cls} = -\alpha_{an}(1 - p_{an})^{\gamma} \log p_{an}$, where p_{an} is the category probability of the bounding box. The target defect does not have azimuth requirements like the vehicle, so the proportion of classification loss function is reduced to 0.001. In addition, the corner loss is the Euclidean distance between the four corners on the bounding frame and the GT box and the distance between the upper and lower planes of the bounding frame and the sensor, so it can be expressed as $L_{corner} = \sum_{i=1}^{4}(||x_i - x_i||) + \sum_{j=1}^{2}(l_i - l_i)$. The total loss function can be expressed as $L_{total} = L_{reg} + L_{cls} + L_{corner} + L_{votenet}$.

4 EXPERIMENTAL DEVICE AND RESULTS

In order to establish bubble defect data set independently, the experimental device shown in Figure 5 is built. The length and width of bubble defects vary from a few millimeters to a few centimeters. In this paper, clay is used to simulate a variety of bubble defects with different shapes, as shown in Figure 1, so as to maintain the diversity of the data set. The computer configuration used in the experiment is: GTX1660s-6 GB, 16 GB memory capacity.

The data set contains 2 500 samples. The data set is divided into training set and test set, with a ratio of 9:1. The training set has 2 250 samples and the test set has 250 samples. During the training process, the global parameters are as follows: the sample batch_size is 4, the maximum sample training times (epochs) is 210, and the initial learning rate is set to 0.001.

In order to verify the quality of the proposed network model, this paper uses indexes such as average precision (AP), mean average precision (mAP), recall and average recall (AR) to evaluate the model, where precision is defined as: precision = number of correct information extracted/number of information extracted; and recall is defined as: recall = number of correct information extracted/number of information in the sample. Since the target detection classification described in this paper only includes background and detection target, AP and mAP, recall and AR have the same meaning. As shown in Table 3, when the IoU threshold is 0.3 and 0.6, respectively, the model is evaluated every 10 times of training. When the thresholds are different, there are some differences in the law of mAP. It can be seen that at the beginning of training, mAP has a large shock between 10 to 20 and 40 to 50, which can reach 30 %. When the training times reach 70, there will be no large shock, but a stable rise. After the training times reach 90, there will be a small shock, which means that the performance of the model reaches the peak, and the maximum shock value reaches 80 %. The training times in the table fluctuate to some extent before 50, and then rise steadily until it is stable. Through the ex-

Training Times/Epochs	mAP/%	
	IoU Threshold of 0.3	IoU Threshold of 0.6
10	33.54	22.97
20	65.69	46.05
30	38.49	29.52
40	52.33	41.43
50	29.67	25.12
60	61.61	41.01
70	57.72	42.47
80	63.45	46.52
90	78.71	49.93
100	74.90	47.69
110	79.91	49.74
120	74.17	47.81
130	76.20	51.18
140	78.66	48.44
150	79.46	50.70
160	75.18	52.36
170	76.22	50.88
180	80.80	53.23
190	80.28	50.99
200	82.80	52.34
210	80.88	50.04

Table 3. Changes of mAP

perimental results, it can also be proved that a large learning rate will lead to the oscillation of mAP, and with the continuous decline of learning rate, mAP also rises steadily. As it can be seen from Table 4, the target AR decreases with the increase of threshold.

IoU Threshold	0.3	0.6
AR/%	97.14	99.43

Table 4. Comparison of AR under different IoU thresholds

Figure 6 shows the change of loss function in the training process. The solid line indicates the total loss value. The initial value is high. With the increase of training times, the function value continues to decline, and there is a shock between 40–90 training times. After reaching the 100th training, the downward trend of the loss function slows down and converges to about 0.01. The double solid line indicates the size loss. The initial value is 0.02, which is slightly lower than the total loss, and its trend is basically the same as the total loss. Without considering the direction loss, it can be considered that the size loss guides the trend of the total loss, and the value converges to 0.0078. The dotted line indicates the central loss, which finally converges to 0.0003, and its value is about 1/10 of the total loss, so the

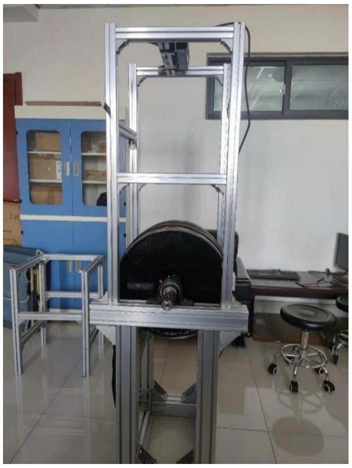

Figure 5. Experimental device

trend fluctuation is not obvious. The average size of the sample frame marked in the data set is (0.045, 0.047, 0.03). After training, the size loss accounts for 16–26 % of the average size, while the center loss accounts for 0.1–6.38 %.

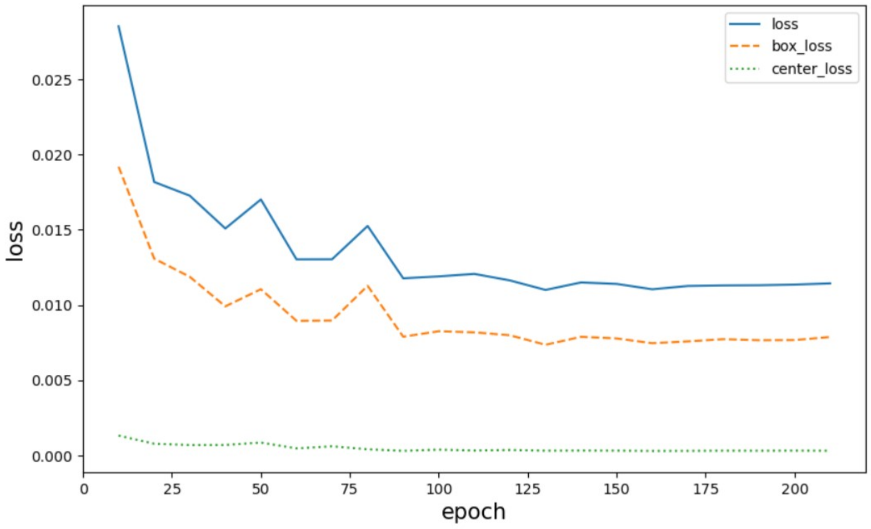

Figure 6. Variation trend of loss function in different training stages

Table 5 shows the performance comparison between the PointVotes network model proposed in this paper and other point cloud models (including one-stage and two-stage algorithms). When the IoU threshold is 0.3, the mAP of PointVotes has a high improvement of 16.08 % comparing with MV3D, and has a small improvement of 1.86 % comparing with PointRCNN. When the IoU threshold is 0.6, the mAP of this method is 8.36 % higher than that of MV3D, but there is still a small gap comparing with Voxel, PointRCNN and PointPiers. This is mainly due to PointVotes adding C-FPS to the fusion sampling, so that the features in the foreground can be fully captured. The processing time is 0.5 s faster than Vote3Deep, but 0.1 s slower than PointPillars. By selecting an appropriate number of SA layers, the running time can be stabilized at about 0.12 s. On the whole, PointVotes model still shows good performance.

Methods	0.3 IoU	0.6 IoU	Run Time/s
PointVotes	82.80	51.47	0.12
MV3D	66.72	43.11	0.33
Vote3Deep	69.43	48.38	0.62
Frustum	77.33	49.19	0.17
Joint3D	79.45	50.84	0.10
Voxel-FPN	81.35	52.05	0.03
PointRCNN	80.94	52.10	0.11
PointPillars	77.02	51.65	0.02

Table 5. Comparison of AR under different IoU thresholds

5 CONCLUSION

Compared with two-dimensional images, 3D point cloud has better robustness to light and can more accurately express the appearance shape of tire bubble defects. This paper proposed the PointVotes model, a defect detection method that can directly process 3D point cloud and extract the features of different scales for group merging and voting, to achieve the purpose of detecting bubble defects. By observing the spatial structure characteristics of the target, the C-FPS method was proposed, and it was sampled in proportion to the FPS to increase the foreground information. Even small targets can generate a proposal box when generating suggestions. Before generating the proposal box, they were classified and confirmed to have bubble defects before proposal operation.

The designed structural framework can be divided into three parts: fusion sampling layer, voting layer and proposal refinement layer. The SA layer in multiple PointNet++ was used to extract features. The global and local features were obtained by establishing LFE branches and GFE branches, and the features of different scales were arranged and combined to form the high-dimensional features of points and vote, and then the cluster points after voting were grouped and proposed.

In this paper, the performance of the training model was evaluated. Results show that the network model has the following advantages. The essence of the network model is based on points, which completely retained local features, while adding global features to improve the detection accuracy. Using the new sampling method C-FPS, the edge position of bubble defects can be more accurately expressed, and the appropriate evaluation interval was set to observe the training process of the network model under different thresholds. The mAP can reach 82.8 % with a detection time of 0.12 s.

Acknowledgement

This research was funded by the National Natural Science Foundation of China (No. 52101401).

REFERENCES

[1] Qi, C. R.—Litany, O.—He, K.—Guibas, L.: Deep Hough Voting for 3D Object Detection in Point Clouds. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 9276–9285, doi: 10.1109/ICCV.2019.00937.

[2] Girshick, R.—Donahue, J.—Darrell, T.—Malik, J.: Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 580–587, doi: 10.1109/CVPR.2014.81.

[3] Girshick, R.: Fast R-CNN. 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1440–1448, doi: 10.1109/ICCV.2015.169.

[4] Ren, S.—He, K.—Girshick, R.—Sun, J.: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 39, 2017, No. 6, pp. 1137–1149, doi: 10.1109/TPAMI.2016.2577031.

[5] Qi, C. R.—Su, H.—Mo, K.—Guibas, L. J.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 77–85, doi: 10.1109/CVPR.2017.16.

[6] Qi, C. R.—Yi, L.—Su, H.—Guibas, L. J.: PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In: Guyon, I., Von Luxburg, U., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.): Advances in Neural Information Processing Systems 30 (NIPS 2017). Curran Associates Inc., 2017, pp. 5099–5108.

[7] Dai, A.—Chang, A. X.—Savva, M.—Halber, M.—Funkhouser, T.—Niessner, M.: ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2432–2443, doi: 10.1109/CVPR.2017.261.

[8] Song, S.—Lichtenberg, S. P.—Xiao, J.: SUN RGB-D: A RGB-D Scene Understanding Benchmark Suite. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 567–576, doi: 10.1109/CVPR.2015.7298655.

[9] Shi, S.—Wang, X.—Li, H.: PointRCNN: 3D Object Proposal Generation and Detection from Point Cloud. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 770–779, doi: 10.1109/CVPR.2019.00086.

[10] Yang, Z.—Sun, Y.—Liu, S.—Jia, J.: 3DSSD: Point-Based 3D Single Stage Object Detector. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 11037–11045, doi: 10.1109/CVPR42600.2020.01105.

[11] Ngiam, J.—Caine, B.—Han, W.—Yang, B.—Chai, Y. et al.: StarNet: Targeted Computation for Object Detection in Point Clouds. 2019, doi: 10.48550/arXiv.1908.11069.

[12] Wu, W.—Qi, Z.—Fuxin, L.: PointConv: Deep Convolutional Networks on 3D Point Clouds. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 9613–9622, doi: 10.1109/CVPR.2019.00985.

[13] Chen, L.—Wei, M.: ELF-Net: Enriching Local Features Network for 3D Point Cloud Classification and Semantic Segmentation. Journal of Intelligent and Fuzzy Systems, Vol. 41, 2021, No. 2, pp. 3973–3983, doi: 10.3233/JIFS-210065.

[14] Zhou, Y.—Tuzel, O.: VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 4490–4499, doi: 10.1109/CVPR.2018.00472.

[15] Yang, Z.—Sun, Y.—Liu, S.—Shen, X.—Jia, J.: STD: Sparse-to-Dense 3D Object Detector for Point Cloud. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 1951–1960, doi: 10.1109/ICCV.2019.00204.

[16] Yan, Y.—Mao, Y.—Li, B.: SECOND: Sparsely Embedded Convolutional Detection. Sensors, Vol. 18, 2018, No. 10, Art. No. 3337, doi: 10.3390/s18103337.

[17] Engelcke, M.—Rao, D.—Wang, D. Z.—Tong, C. H.—Posner, I.: Vote3Deep: Fast Object Detection in 3D Point Clouds Using Efficient Convolutional Neural Networks. 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 1355–1361, doi: 10.1109/ICRA.2017.7989161.

[18] Wang, D. Z.—Posner, I.: Voting for Voting in Online Point Cloud Object Detection. In: Kavraki, L. E., Hsu, D., Buchli, J. (Eds.): Proceedings of Robotics: Science and Systems XI. 2015, doi: 10.15607/RSS.2015.XI.035.

[19] Shi, S.—Guo, C.—Jiang, L.—Wang, Z.—Shi, J.—Wang, X.—Li, H.: PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 10529–10538, doi: 10.1109/CVPR42600.2020.01054.

[20] Li, B.: 3D Fully Convolutional Network for Vehicle Detection in Point Cloud. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017, pp. 1513–1518, doi: 10.1109/IROS.2017.8205955.

[21] Lang, A. H.—Vora, S.—Caesar, H.—Zhou, L.—Yang, J.—Beijbom, O.: PointPillars: Fast Encoders for Object Detection from Point Clouds. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 12689–12697, doi: 10.1109/CVPR.2019.01298.

[22] Hui, L.—Cheng, M.—Xie, J.—Yang, J.—Cheng, M. M.: Efficient 3D Point Cloud Feature Learning for Large-Scale Place Recognition. IEEE Transactions on Image Processing, Vol. 31, 2022, pp. 1258–1270, doi: 10.1109/TIP.2021.3136714.

[23] Hoang, L.—Lee, S. H.—Lee, E. J.—Kwon, K. R.: GSV-NET: A Multi-Modal Deep Learning Network for 3D Point Cloud Classification. Applied Sciences, Vol. 12, 2022, No. 1, Art. No. 483, doi: 10.3390/app12010483.

[24] Beltrán, J.—Guindel, C.—Moreno, F. M.—Cruzado, D.—Garcia, F.—De La Escalera, A.: BirdNet: A 3D Object Detection Framework from LiDAR Information. 2018 21st International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2018, pp. 3517–3523, doi: 10.1109/ITSC.2018.8569311.

[25] Yang, B.—Luo, W.—Urtasun, R.: PIXOR: Real-Time 3D Object Detection from Point Clouds. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 7652–7660, doi: 10.1109/CVPR.2018.00798.

[26] Ali, W.—Abdelkarim, S.—Zidan, M.—Zahran, M.—El Sallab, A.: YOLO3D: End-to-End Real-Time 3D Oriented Object Bounding Box Detection from LiDAR Point Cloud. In: Leal-Taixé, L., Roth, S. (Eds.): Computer Vision – ECCV 2018 Workshops. Springer, Cham, Lecture Notes in Computer Science, Vol. 11131, 2019, pp. 716–728, doi: 10.1007/978-3-030-11015-4_54.

[27] Uddin, K.—Jeong, T. H.—Oh, B. T.: Incomplete Region Estimation and Restoration of 3D Point Cloud Human Face Datasets. Sensors, Vol. 22, 2022, No. 3, Art. No. 723, doi: 10.3390/s22030723.

[28] Qi, C. R.—Liu, W.—Wu, C.—Su, H.—Guibas, L. J.: Frustum PointNets for 3D Object Detection from RGB-D Data. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 918–927, doi: 10.1109/CVPR.2018.00102.

[29] Ku, J.—Mozifian, M.—Lee, J.—Harakeh, A.—Waslander, S. L.: Joint 3D Proposal Generation and Object Detection from View Aggregation. 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 1–8, doi: 10.1109/IROS.2018.8594049.

[30] Hou, J.—Dai, A.—Niessner, M.: 3D-SIS: 3D Semantic Instance Segmentation of RGB-D Scans. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4416–4425, doi: 10.1109/CVPR.2019.00455.

[31] Liang, M.—Yang, B.—Chen, Y.—Hu, R.—Urtasun, R.: Multi-Task Multi-Sensor Fusion for 3D Object Detection. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 7337–7345, doi: 10.1109/CVPR.2019.00752.

[32] Chen, X.—Ma, H.—Wan, J.—Li, B.—Xia, T.: Multi-View 3D Object Detection Network for Autonomous Driving. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6526–6534, doi: 10.1109/CVPR.2017.691.

[33] Qi, C. R.—Su, H.—Niessner, M.—Dai, A.—Yan, M.—Guibas, L. J.: Volumetric and Multi-View CNNs for Object Classification on 3D Data. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 5648–5656, doi: 10.1109/CVPR.2016.609.

[34] Jiang, H.—Lu, Y.—Chen, S.: Research on 3D Point Cloud Object Detection Algorithm for Autonomous Driving. Mathematical Problems in Engineering, Vol. 2022, 2022, Art. No. 8151805, doi: 10.1155/2022/8151805.

[35] Wang, W.—Wang, T.—Cai, Y.: Multi-View Attention-Convolution Pooling Network for 3D Point Cloud Classification. Applied Intelligence, Vol. 52, 2022, No. 13,

pp. 14787–14798, doi: 10.1007/s10489-021-02840-2.

[36] Lin, T. Y.—Goyal, P.—Girshick, R.—He, K.—Dollár, P.: Focal Loss for Dense Object Detection. Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2980–2988.

Hualin Yang received his M.Sc. degree in mechanical manufacturing and automation from the Qingdao University of Science and Technology, Qingdao, China, in 2003, and his Ph.D. in chemical machinery from the ZheJiang University, Hangzhou, China, in 2006. His current research interests include intelligent manufacturing, computer vision, mechanical design, control and automation.

Yuanzheng Jiang received his M.Sc. degree in mechanical engineering from the Qingdao University of Science and Technology, Qingdao, China, in 2022. His current research interests include intelligent manufacturing and computer vision.

Wenxue Nie studied in the Qingdao University of Science and Technology for master's degree in mechanics in 2021. At present, the research direction is digital twinning in the direction of virtual simulation and digital twinning in the direction of visualization.

Fang Deng received her M.Sc. degree in chemical machinery from the ZheJiang University, Hangzhou, China, in 2006, and her Ph.D. in mechanical engineering from the Qingdao University of Science and Technology, Qingdao, China, in 2019. Her current research interests include nonlinear control and estimation, adaptive control, and marine crafts control.

Maozhen Li is Professor in the Department of Electronic and Computer Engineering, Brunel University London, UK. He received his Ph.D. from the Institute of Software, Chinese Academy of Sciences in 1997. His main research interests include high performance computing, big data analytics and intelligent systems with applications to smart grid, smart manufacturing and smart cities. He has over 180 research publications in these areas including 4 books. He has served over 30 IEEE conferences and is on the editorial board of a number of journals. He is Fellow of the British Computer Society and the IET.