Point Cloud Pre-training with Natural 3D Structures

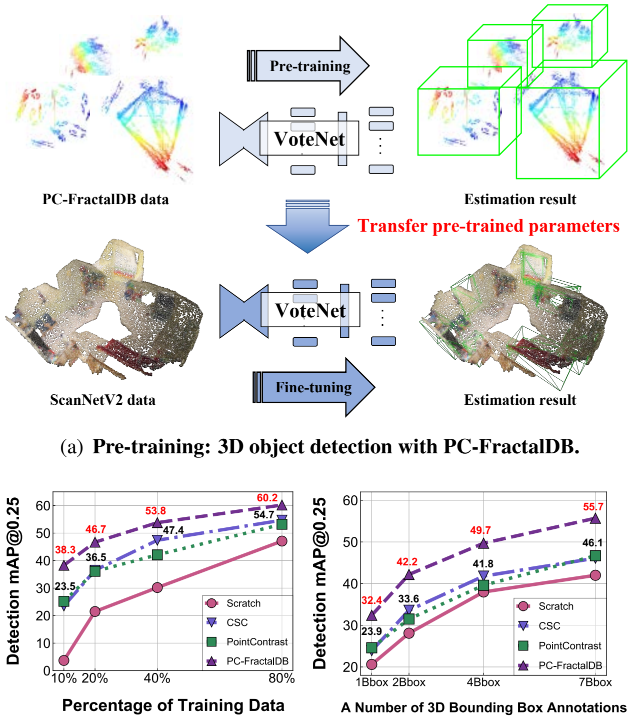
Ryosuke Yamada[1]* Hirokatsu Kataoka[1]* Naoya Chiba[2] Yukiyasu Domae[1] Tetsuya Ogata[1,2]
[1]National Institute of Advanced Industrial Science and Technology (AIST) [2]Waseda University

Abstract

The construction of 3D point cloud datasets requires a great deal of human effort. Therefore, constructing a large-scale 3D point clouds dataset is difficult. In order to remedy this issue, we propose a newly developed point cloud fractal database (PC-FractalDB), which is a novel family of formula-driven supervised learning inspired by fractal geometry encountered in natural 3D structures. Our research is based on the hypothesis that we could learn representations from more real-world 3D patterns than conventional 3D datasets by learning fractal geometry. We show how the PC-FractalDB facilitates solving several recent dataset-related problems in 3D scene understanding, such as 3D model collection and labor-intensive annotation. The experimental section shows how we achieved the performance rate of up to 61.9% and 59.0% for the ScanNetV2 and SUN RGB-D datasets, respectively, over the current highest scores obtained with the PointContrast, contrastive scene contexts (CSC), and RandomRooms. Moreover, the PC-FractalDB pre-trained model is especially effective in training with limited data. For example, in 10% of training data on ScanNetV2, the PC-FractalDB pre-trained VoteNet performs at 38.3%, which is +14.8% higher accuracy than CSC. Of particular note, we found that the proposed method achieves the highest results for 3D object detection pre-training in limited point cloud data. [1]

1. Introduction

Recently, 3D object recognition with 3D point clouds is expected to become increasingly helpful in real-world applications, such as mobile robots and self-driving cars. In particular, 3D object detection estimates the location and category of an object from 3D scenes. Compared with image-based detection models, 3D point clouds enable robust detection of real-world objects without relying on appearance. However, a limitation of constructing 3D datasets is that this requires a significant workforce to create and annotate the 3D model, and such 3D models cannot usually be collected in large quantities via the Internet. For this reason, it is necessary to either create 3D models via computer-aided design (CAD) software or scan 3D scenes from the real environment using sensors such as LiDAR scans. In addition, constructing a point cloud dataset based on a 3D scene will result in human annotators and cross-validators. Training with limited data or annotation tends to cause overfitting of the detection model. Therefore, the present study focuses on pre-training with the point cloud dataset to solve the abovementioned problems.



(a) **Pre-training: 3D object detection with PC-FractalDB.**

(b) **Fine-tuning results for a limited ScanNet data.**

Figure 1. Pre-training effects on the PC-FractalDB as a family of formula-driven supervised learning. Although the proposed method does not use real data, it is a better pre-training approach to understand a 3D scene, especially in a limited data scenario.

We have already witnessed the effectiveness of pre-training in point cloud processing. In order to address the

*indicates equal contribution.

[1] Dataset release: https://ryosuke-yamada.github.io/PointCloud-FractalDataBase/

human annotation problem for point cloud datasets, self-supervised learning (SSL) has been proposed [1, 27, 35, 52, 54, 59, 63, 67, 70]. In particular, PointContrast [67] demonstrates the possibility of pre-training for the first time higher-level scene understanding tasks, namely 3D object detection and 3D object segmentation. After the advent of PointContrast, self-supervised learning using contrastive learning has been proven the best performance on point cloud datasets such as ScanNetV2 [17] and SUN RGB-D [58] in 3D object detection. A limitation of these methods, since pre-training is restricted to a backbone network only, and training data depend on the scale of a point cloud dataset. Therefore, to achieve accurate detection of 3D objects, we need to develop approaches to reduce the annotation effort on datasets and effective pre-training methods.

The present paper describes a point cloud pre-training method that automatically constructs a point cloud dataset under the laws governing natural 3D structures. More specifically, we implement the concept of formula-driven supervised learning (FDSL) to 3D vision that generates infinite training data based on a mathematical formula proposed by Kataoka *et al.* in 2D vision [32]. The present study uses a mathematical formula based on fractal geometry [41], which is assumed to be highly applicable to natural and artificial objects in real-world 3D scenes. Since fractal geometry has two essential properties, self-similarity and non-integer dimensions, we believe it can generate fine-grained 3D structures that CAD models cannot represent.

Our proposed point cloud fractal database (PCFractalDB) enables users to significantly improve the representation learning for 3D object detection. By focusing on fractal geometry, a piece of background knowledge in the real-world, it is possible to automatically generate 3D models and 3D scenes that resemble real-world nature. Thus, we do not require human labor to auto-construct a point cloud dataset by following natural law.

We summarize the contributions in the present study as follows: (i) We propose the PC-FractalDB automatically generated by natural 3D structures with fractals. Notably, this framework does not require data collection and annotation. The PC-FractalDB directly enables the acquisition of feature representation for 3D object detection in the pre-training phase, shown in Fig. 1(a). (ii) By creating the PC-FractalDB pre-trained detector, we have improved performance rates in 3D object detection tasks on representative point cloud datasets, such as ScanNetV2 and SUN RGB-D. (iii) Our proposed PC-FractalDB pre-training assists when fine-tuning for the dataset limited to the number of training data and annotation, shown in Fig. 1(b).

2. Related work

3D point cloud datasets. 3D scene understanding with point clouds has been rapidly progressing with the increase of public point cloud datasets [3,10,20,29,57,61,62,65,66] with rich annotations. However, the most frequently used datasets, ScanNetV2 and SUN RGB-D, consist of scanned models, for which significant human efforts have been spent creating scanned models and annotations. Therefore, we can easily assume that current point cloud datasets contain a more limited number of data and annotations for training, validation, and test sets as compared to 2D vision datasets [18,33,53,72].

On the other hand, deep learning relies on a large quantity of training data, the restrict of learning with limited data and annotations. Pre-training is one of the most promising methods by which to tackle this problem [5]. This concept has been validated successfully in video recognition [23] and 2D image recognition [19]. However, in order to succeed in pre-training, we need large-scale datasets in each domain, such as Kinetics-700 [9] and JFT-300M [60]. Namely, we believe that the performance level of 3D object recognition with point cloud would be improved if it were possible to construct a million-order-instance dataset.

3D object detection. In 3D object detection, there are two main types of approaches: architectures based on 2D-CNN or 3D-CNN [14, 34, 38, 43, 49, 55, 68, 73] and architectures that directly input 3D scenes consisting of 3D point clouds [11, 16, 22, 24, 39, 42, 46–48, 56, 71]. The present study focuses on architectures that directly input 3D scenes. In particular, VoteNet [48] uses Hough voting for sparse point cloud input to perform 3D bounding box detection via feature sampling, clustering, and voting operations designed for 3D scene data.

Self-supervised learning. Self-supervised learning has made significant progress and received great attention in 2D vision [6–8, 12, 13, 21, 25, 45, 69]. As such, there have been attempts to adapt the pre-text task proposed for 2D vision to 3D vision in order to address the human annotation problem on 3D datasets [1, 2, 26, 27, 30, 36, 37, 40, 64, 67, 70]. The most well-known method of self-supervised learning is PointContrast. PointContrast uses contrastive learning to learn geometric features by registering point cloud pairs on a 3D scene. The advantage provided is that optimizations contrastive loss between paired corresponding points in feature space from two different viewpoints.

Formula-driven supervised learning. Formula-driven supervised learning [28, 31, 32, 44] automatically generates large-scale datasets based on mathematical formulas and does not require human image collection and manual annotation. Kataoka *et al.* [32] showed that the 2D-FractalDataBase pre-trained model performs and performs as an ImageNet pre-trained model in a part of the image classification task. Remarkably, these methods achieve results by pre-training synthetic images rendered fractal without natural images.

We speculate that these results may come from pre-

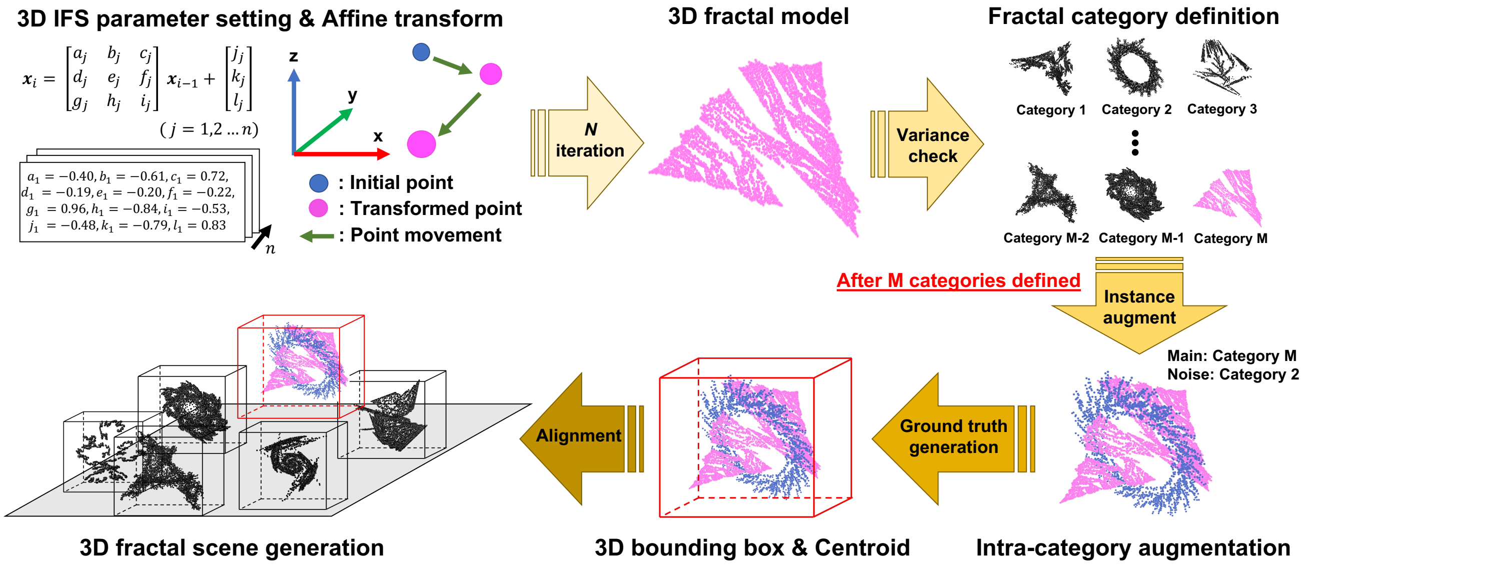

3D IFS parameter setting & Affine transform
$x_i = \begin{bmatrix} a_j & b_j & c_j \\ d_j & e_j & f_j \\ g_j & h_j & i_j \end{bmatrix} x_{i-1} + \begin{bmatrix} j_j \\ k_j \\ l_j \end{bmatrix}$
$(j = 1,2 \ldots n)$
$a_1 = -0.40, b_1 = -0.61, c_1 = 0.72,$
$d_1 = -0.19, e_1 = -0.20, f_1 = -0.22,$
$g_1 = 0.96, h_1 = -0.84, i_1 = -0.53,$
$j_1 = -0.48, k_1 = -0.79, l_1 = 0.83$
n
z
y
x
: Initial point
: Transformed point
: Point movement
N iteration
3D fractal model
Variance check
Fractal category definition
Category 1
Category 2
Category 3
Category M-2
Category M-1
Category M
After M categories defined
Instance augment
Main: Category M
Noise: Category 2
Ground truth generation
Alignment
3D fractal scene generation
3D bounding box & Centroid
Intra-category augmentation

Figure 2. Overview of the formula-driven supervised learning framework for 3D object detection with 3D point clouds. We generate a 3D fractal model using the 3D iterated function system [4] (see Sec. 3.1). The proposed PC-FractalDB is automatically constructed by difiniting a fractal category using variance threshold and instance augmentation with FractalNoiseMix (see Sec. 3.2 and 3.3). A 3D fractal scene is generated by randomly selecting 3D fractal models and translating these from the origin on the z-plane (see Sec. 3.4).

training based on fractal, which is common in the real-world, and thus covers more real-world patterns than large-scale datasets such as ImageNet. In addition, the present study focuses on fractal because we consider that succeeding pre-train with natural 3D structure is the assisted understanding of 3D scenes in the real-world.

3. Point cloud fractal database (PC-FractalDB)

We introduce the PC-FractalDB in terms of auto-generated 3D fractal models and 3D fractal scenes. We construct the PC-FractalDB through four procedures. First, we provide a method for automatic 3D generation based on a 3D iterated function system (3D IFS) [4] (see Sec. 3.1). Second, we define the categories based on the data distribution of the 3D fractal model (see Sec. 3.2). Third, we generate instances for each category using a novel augmentation method, which we call FractalNoiseMix (see Sec. 3.3). Finally, we automatically generate a 3D fractal scene using 3D fractal models (see Sec. 3.4). The overview of our framework is presented in Fig. 2.

3.1. Automatic 3D fractal model generation

The PC-FractalDB which is generated 3D fractal scenes from infinite pairs of 3D fractal models and their fractal categories using the 3D IFS. By exploiting fractal geometry, common in the real-world, we hypothesize that we can easily represent complex patterns in 3D scenes using the 3D IFS and can assist in 3D scene understanding in the real-world. A 3D fractal model is automatically generated using the following five steps. (1) Multiple affine transforms and select probabilities are randomly set. (2) The initial point cloud is indicated by the origin coordinates and is set as the current point cloud. (3) One of the affine transforms is selected based on select probabilities. (4) The current point cloud is affine transformed for the next point cloud using the selected affine transformation. (5) Steps 3 and 4 are recursively performed up to the set N iterations.

A 3D fractal model is generated by iteratively applying a 3D affine transform T_j to an initial point. In the present study, for the sake of simplicity, we introduce homogeneous coordinates to handle affine transforms. In homogeneous coordinates, a 3D point cloud $\mathbf{x} = \begin{bmatrix} x & y & z \end{bmatrix}^\top \in \mathbb{R}^3$ is described as $\hat{\mathbf{x}} = \begin{bmatrix} x & y & z & 1 \end{bmatrix}^\top \in \mathbb{R}^4$, where the notation $\hat{\cdot}$ indicates that the point is considered in homogeneous coordinates. Note that a 3D affine transform includes rotations, translations, scaling, and skewing. In order to generate a 3D fractal model automatically, we make affine transforms randomly. In order to construct a 3D IFS set, an affine transforms $\{T_j \in \mathbb{R}^{4\times4} | 1 \leq j \leq N\}$ are generated, where the elements of affine transform matrices are sampled by a uniform distribution in the range of $[-1.0, 1.0]$. When an initial point x_0 is given, a 3D affine transform T_j makes a 3D fractal model $P = \{\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_n\}$ by

$$\hat{\mathbf{x}_i} = T^i \hat{\mathbf{x}}_{i-1} \tag{1}$$

for i from 0 to n, where n is the number of iterations. The probability of selecting T_j is denoted as P_{T_j}. Here, $p_j =$

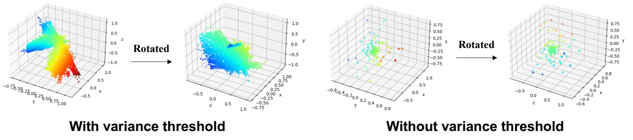


Figure 3. **Difference of 3D fractal models on with and without variance threshold.** Depending on the set parameter as a property of 3D IFS, a part of a 3D fractal model is biased and aggregated, resulting in a significant sparse. By binning a category with a variance threshold, the shape of a 3D fractal model can be distinct.

$|\det T_j|/\sum_{j=0}^{N}|\det T_j|$. Note that the scaling factor of an affine transform T_j is given by $|\det T_j|$. Next, we set the original coordinate as the initial point cloud P_0 and select an affine transform from 3D IFS by following the probabilities p_j. A 3D fractal model is generated of 4,000 iterations.

3.2. Binning by variance to assign category

After generating the 3D fractal model, it is necessary to define a category for it. By using a framework with 3D IFS, we can create an infinite number of categories with randomly generated affine transform parameters T_j for each fractal category. However, simply setting a category definition without performing a quality check may establish a wrong category. In contrast, the proposed method includes quality checks for 3D fractal models by using a variance. The point cloud distribution gives the shape features corresponding to a 3D fractal model (see Fig. 3). When the 3D fractal model's calculated variance is above the threshold, it is registered as a new fractal category. By setting this variance threshold, we expect to create a clear natural 3D structure in 3D space and expand the differences between fractal categories. The variance threshold are formulated as follows:

$$\min(Var[x], Var[y], Var[z]) > \sigma \tag{2}$$

where the present paper set the bins of variance by thresholding σ from 0.0 to 0.2 with 0.05 increments per step because of taking a longer time to define a fractal category if the variance threshold is greater than 0.20. In addition, all point clouds of the 3D fractal model are translated with the center set as the origin. Furthermore, depending on the affine transformation parameters, the scale of the 3D fractal model may become divergent. Therefore, we normalized the 3D fractal model scale to [-1.0, 1.0].

3.3. Instance augmentation by mixing fractal noise

The variance binning defines the fractal category, and each fractal category has only one 3D fractal model. In order to assist the increase of in the 3D fractal model, we propose the FractalNoiseMix (FNM) for instance augmentation inspired by PointMixup [15] as shown in Fig. 4. Different from PointMixup, where the instance augmentation is interpolating between training samples to crate middle category, our approach attempts to augment intra-category and improve effectiveness for PC-FractalDB pre-training.

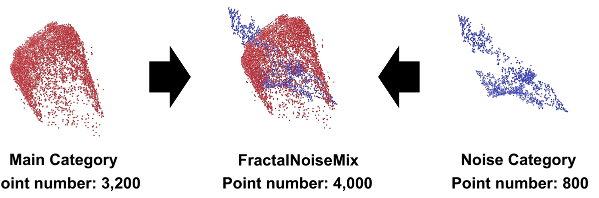


Figure 4. **FractalNoiseMix: Intra-category augmentation.** We mix two different 3D fractal models. One is the main fractal category, whereas the other is a fractal category used as fractal noise.

The FNM of the proposed method involves mixing major and minor fractal categories. For example, once a major fractal category is fixed, we set this category as 80% of the full 3D fractal model and randomly select and add 20% of the minor point cloud in the 3D fractal model to fill in the major point cloud the 3D fractal model. Note that let a major fractal category be a fractal category when classifying 3D fractal models. Random point cloud could be given to augment, but we consider that the important fractal shape feature will be lost in that case, so the present paper uses FNM.

3.4. Automatic 3D fractal scene generation

To generate a 3D fractal scene, we first need to sample multiple objects from 3D fractal models randomly—the number of objects per 3D fractal scene by Poisson distribution. Next, we generate 3D bounding boxes and rotate the 3D fractal model around the z-axis. We begin by randomly setting the scale factor at the x-axis from 0.75 to 1.25, and we multiply by a coefficient set as the aspect ratio from 0.9 to 1.1 in the y-axis and z-axis based on the set x-axis scale factor. The reason why 3D indoor datasets tend to be a small variance in each object scale. At the same time, the orientation of each 3D fractal model can be randomly rotated around the z-axis to gain training variations. However, since the 3D fractal model does not have a front, the rotation angle is set randomly between [-180, 180] degrees. Finally, the 3D fractal models should be translated onto the z-plane to align the structure with the existing datasets such as ScanNetV2 [17] and SUN RGB-D [58]. In order to accomplish this, we randomly set the x and y coordinates of an instance generated by the 3D fractal models as the centroid of the 3D fractal model and redefine the centroid. In this case, the x and y positions to be redefined should be within the range of [-7.5, 7.5].

Note that the minimum z coordinates for each 3D fractal model are aligned in the same z-plane. Because in the real world, objects cannot be floating in the air due to gravity.

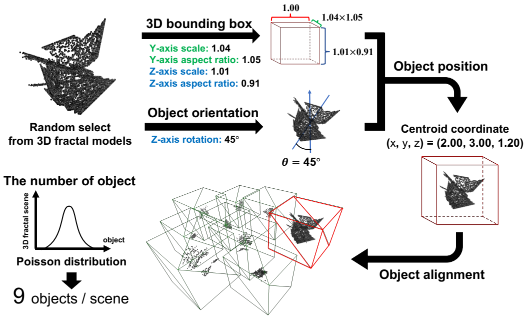


Figure 5. **3D bounding box / 3D fractal scene generation for 3D object detection.**

Additionally, note that the 3D fractal models are placed in the 3D fractal scene in non-overlapping positions. Visualization of 3D fractal scenes is provided in the appendix.

4. Experiments

In this section, we firstly introduce how to pre-train our PC-FractalDB and fine-tune it for downstream datasets (see Sec. 4.1). We then provide analysis experiments to understand the importance of pre-training by object detection, the effects of 3D fractal model variations, and show our method's advantages over a 3D scene consisting of CAD models (see Sec. 4.2). We then explore the optimal parameters of PC-FractalDB from exploratory experiments (see Sec. 4.3). From these results, we compare PC-FractalDB on the best parameter with previous methods on two 3D indoor object detection benchmarks (see Sec. 4.3). Finally, we experiment with the effectiveness of our method when only limited training data and annotation is available (see Sec. 4.5). More analysis explored PC-FractaDB parameters and visualizations are provided in the appendix.

4.1. Experimental setting

Pre-training on PC-FractalDB. In the present paper, we employ VoteNet, an end-to-end 3D object detection network based on a synergy of deep point set networks and Hough voting [48]. In experiments, we use both PointNet++ [50] and Sparse Res-UNet (SRU-Net) [67] as the backbone network. Unlike previous work, our proposed method enables the acquisition of feature representation for object detection in the pre-training phase. In order to construct the PC-FractalDB pre-trained VoteNet, the following training parameters are assigned. Pre-training is carried out with 1.8M iterations at minimum, a batch size of 64, and a learning rate of 0.004 as the hyperparameters. The input point clouds are randomly sampled at 40,000.

For example, the construction of the PC-FractalDB (Category: 1k, Instance: 500, Scene; 10k) can be completed in two days, and the pre-training can be completed in six days using four NVIDIA Tesla V100 GPUs. Given that ScanNetV2 (Category: 18, Scene; 1.5k) takes approximately 23 days ((22 [min] * 1,500 [scene]) / (60 [min] * 24 [hour])) to generate due to the fact that the generation process takes 22 [min] per 3D scene [17], we found the construction and pre-training of the PC-FractalDB to be very fast.

Table 1. Pre-training the PC-FractalDB and the fine-tuning SUN RGB-D and ScanNetV2 in our experiments.

Dataset	Supervision	Category	#Scene	#Model
ScanNetV2 [17]	Human	18	1.5k	–
SUN RGB-D [58]	Human	37	10.2k	65k
PC-FractalDB	Formula	1k	100k	1M

Fine-tuning on downstream datasets. Next, we evaluated the PC-FractalDB pre-trained model using fine-tuning datasets. Fine-tuning datasets are used ScanNetV2 [17] and SUN RGB-D [58], and these datasets' details are summarized in Table 1. These datasets, which captured indoor scenes, are frequently used in 3D object detection. The fine-tuning is carried out with 180 epochs, a batch size of 64, and a learning rate of 0.01 as the hyperparameters. The learning rate is 0.01 for each interval of 40, 80, 120, and 160 epochs. The input point clouds were randomly sampled at 40,000 (ScanNetV2) and 20,000 (SUN RGB-D).

4.2. Preliminary study

In this subsection, in order to understand the effects of pre-training tasks, 3D fractal model variations, and show our method's advantages over a 3D scene consisting of CAD models, we performed three preliminary experiments. Specifically, we experimentally investigated the answers to the following three questions. (i) Which pre-training task is better: 3D object classification or 3D object detection as pre-train?, (ii) How vital are 3D pattern variations in pre-training for 3D object detection?, (iii) Which is more effective of pre-training, 3D fractal models or CAD models?

(i) Which pre-training task is better: 3D object classification or 3D object detection as pre-train? (see Table 2). This preliminary experiment (i) attempts to clarify which is more effective as pre-training 3D object classification or 3D object detection tasks. We executed by pre-training on PC-FractalDB (w/ and w/o 3D bounding box /3D fractal scene) and fine-tuning on the SUN RGB-D / ScanNetV2 dataset.

In the case of the 3D object classification task, it allows pre-training a backbone network (PointNet++) by classifying 3D fractal models. The hough voting module and object candidate proposal module are optimized in the fine-tuning phase. On the other hand, the pre-training on the 3D object detection task can optimize the whole network including hough voting module and object candidate proposal module in VoteNet. The number of input points for a single 3D fractal model to 2,048, and for a 3D fractal scene that consists was randomly sampled to 40,000.

Table 2. The comparison for pre-training part of classification and detection tasks.

	ScanNetV2 mAP@0.25	SUN RGB-D mAP@0.25
PointNet++	48.8	49.8
VoteNet	**61.1**	**57.6**

Table 3. Effects of 3D fractal model variations.

#model	#cat.	#ins	ScanNetV2 mAP@0.25	SUN RGB-D mAP@0.25
1	1	1	57.2	56.4
1k	1k	1	60.3	57.5
1k	1	1k	59.3	56.6
1M	1k	1k	**61.6**	**59.2**

Table 4. The comparison for our proposed PC-FractalDB and ModelNet.

	ScanNetV2 mAP@0.25	SUN RGB-D mAP@0.25
ModelNet	59.9	55.0
PC-FractalDB	**60.4**	**58.0**

As shown in Table 2, the detection pre-training showed on ScanNetV2 was +12.3% more accurate than the classification pre-training on the same dataset. The same tendency was observed for SUN RGB-D, the detection pre-training was +7.8% better than classification pre-training. These results consider that detection pre-training is more effective than classification pre-training because the 3D object detection task can also pre-train the Hough voting and object candidate proposal module.

(ii) How vital are 3D pattern variations in pre-training for 3D object detection? (see Table 3). This preliminary study (ii) attempts to reveal how vital 3D pattern variations are in pre-training for 3D object detection. Preliminary study (ii) compares using the PC-FractalDB consisting of 10,000 3D scenes using only one 3D fractal model, the PC-FractalDB consisting of 10,000 3D scenes using 1,000 3D fractal models, and the PC-FractalDB consisting of 10,000 3D scenes using 100,000 3D fractal models. Here, concerning 1,000 3D fractal models, it is with 1,000 categories and one instance and one category and 1,000 instances.

As shown in Table 3, the performance is confirmed that the PC-FractalDB (Category: 1k, Instance: 1k) pre-trained model is the best score of other PC-FractalDB pre-trained models. In particular, we observed +4.4% and +2.8% performance improvement compared to PC-FractalDB (only one 3D fractal model) for ScanNetV2 and SUN RGB-D.

(iii) Which is more effective of pre-training, 3D fractal models or CAD models? (see Table 4). This preliminary study (iii) aims to evaluate 3D fractal scene generation effectiveness in 3D fractal models. We compare pre-training performance in both 3D fractal scenes and 3D scenes by the CAD models included in ModelNet [65].

As can be seen in Table 4, the PC-FractalDB pre-trained VoteNet outperformed the 3D scenes produced with ModelNet. The performance gaps were +0.5% and +3.0% on ScanNetV2 and SUN RGB-D, respectively. Note that the 3D scenes with ModelNet (Category: 40, Instance: average 243, Scene; 10k) are larger than PC-FractalDB (Category: 40, Instance: 243, Scene; 10k) used in this experiment. Consequently, we can confirm that generated 3D fractal scenes based on fractal geometry are more effective than 3D scenes generated by CAD model well-organized surface data such as the ModelNet.

4.3. Exploration study

In this subsection, to explore optimized parameters of the PC-FractalDB, we performed six exploration studies. Specifically, we explore how to construct a PC-FractalDB for variance threshold, FNM, #instance, #category, #scene, and #object.

Effects of variance threshold (see Table 5). This experiment clarifies whether or not a variance threshold σ was needed (w/ and w/o variance) in the fractal category definition under the PC-FractalDB (Category: 1k, Instance: 500, Scene; 10k) condition. Table 5 shows that the w/ variance threshold σ is better than the w/o setting. In addition, we found that 0.15 is better than 0.10 for variance threshold σ, and the variance threshold of 0.20 requires a larger amount of time to search the fractal category. The exploration experiment details compare the performance on each variance threshold are in supplement.

Effects of FractalNoiseMix (see Table 6). This experiment clarifies whether or not the FNM was needed (w/ and w/o variance) in the intra-category augmentation under the PC-FractalDB (Category: 1k, Instance: 500, Scene; 10k) condition. Table 6 shows that the w/ FNM is better than the w/o FNM. Moreover, we explored that the fractal noise ratio of 20 % gave the best effective parameter. The exploration experiment details of the ratio of fractal noise are in supplement.

Effects of #instance (see Table 7). This experiment explored the best effective #instance in PC-FractalDB pre-training under the PC-FractalDB (Category: 1k, Scene; 10k) condition. Table 7 shows that 1,000 instances provide the best results.

Effects of #category (see Table 8). This experiment explored the best effective #category in PC-FractalDB pre-training under the PC-FractalDB (Instance: 500, Scene; 10k) condition. Table 8 shows that 1,000 categories provide the best results.

Effects of #scene (see Table 9). This experiment explored the best effective #scene in PC-FractalDB pre-training under the PC-FractalDB (Category: 1k, Instance: 500) condition. Table 9 shows that 10,000 scenes provide the best results.

Effect of #object per scene (see Table 9). This experiment explored the best effective #object per scene in PC-FractalDB pre-training under the PC-FractalDB (Category:

Table 5. The comparisons for with (w/) and without (w / o) variance threshold.

	ScanNetV2 mAP@0.25	SUN RGB-D mAP@0.25
w / o variance	58.9	55.4
w / variance	**61.9**	**59.0**

Table 6. The comparisons for with (w/) and without (w / o) FractalNoiseMix (FNM).

	ScanNetV2 mAP@0.25	SUN RGB-D mAP@0.25
w / o FNM	60.3	57.5
w / FNM	**61.9**	**59.0**

Table 7. Effects of #instance.

	ScanNetV2 mAP@0.25	SUN RGB-D mAP@0.25
10	60.8	58.2
100	60.6	57.7
1,000	**61.6**	**59.2**

Table 8. Effects of #category.

	ScanNetV2 mAP@0.25	SUN RGB-D mAP@0.25
10	60.8	57.8
100	61.0	58.3
1,000	**61.9**	**59.0**

Table 9. Effects of #scene.

	ScanNetV2 mAP@0.25	SUN RGB-D mAP@0.25
1k	60.0	55.3
10k	**61.9**	**59.0**
100k	61.5	58.3

Table 10. Effects of #object.

	ScanNetV2 mAP@0.25	SUN RGB-D mAP@0.25
5	59.4	57.9
15	**61.9**	**59.0**
25	58.3	56.8

1k, Instance: 500, Scene; 10k) condition. The number of 3D fractal models to be placed in a 3D fractal scene is determined according to the Poisson distribution. In this experiment, we set {5, 10, 15}, which is assumed to be a realistic number of objects in indoor scenes, as the mean value of the Poisson distribution. Table 10 shows that 15 objects provide the best results.

4.4. Comparison with other pre-training methods

Based on the exploration study in Sec. 4.3, we list the 3D object detection scores in Table 11. Here, we compared the proposed PC-FractalDB with self-supervised learning methods (PointContrast [67], CSC [26], and RandomRooms [51]) in terms of pre-training. This experiment used backbone networks such as PointNet++ and SR-UNet.

As shown in Table 11, when the backbone network is PointNet++, pre-training with the PC-FractalDB improved the score by +4.0% on ScanNetV2 and +2.0% on SUN RGB-D at mAP@0.25 as compared to training from scratch. In addition, when the backbone network is SR-UNet, pre-training with the PC-FractalDB improved the score by +2.4% on ScanNetV2 and +1.0% on SUN RGB-D at mAP@0.25 as compared to training from scratch.

Next, we confirmed that the performance of PC-FractalDB is relatively higher than that of previous state-of-the-art self-supervised learning methods. The performance rate with PC-FractalDB (PointNet++) is +0.6% and +0.2%, which is better than RandomRooms on ScanNetV2 and SUN RGB-D. We also confirmed that PC-FractalDB (SR-UNet) is approximately equivalent to CSC and PointContrast. The performance rate with PC-FractalDB (PointNet++×2) is +2.1% and +3.7%, which is better than PointContrast on ScanNetV2 and SUN RGB-D. On the other hand, when comparing PointNet++ and SRUNet with equal parameters, PC-FractalDB (PointNet++×2) showed the highest accuracy in all evaluations except SUN RGBD at mAP@0.50.

4.5. Additional experiments

We performed three additional experiments, including (i) limited training data with {10% , 20%, 40%, 80%} subsets, (ii) limited 3D bounding box annotations with {1, 2, 4, 7} objects, and (iii) Effects of supervisor label in pre-training.

Limited fine-tuning data (see Fig. 1). We verified the effectiveness of the PC-FractalDB pre-trained model on smaller fine-tuning datasets. We sample {10% , 20%, 40%, 80%} from ScanNetV2 training data and use the official ScanNetV2 validation set for evaluation (for details, refer to [26]). As seen in Fig. 1, the PC-FractalDB pre-trained model produced higher scores than the PointContrast pre-trained model and CSC pre-trained model on all limited training subsets. The results show that the proposed method can acquire effective features compared to previous self-supervised learning methods for limited training data on fine-tuning datasets.

Limited 3D bounding box annotations (see Fig. 1). In addition, we also evaluated the effectiveness of the PC-FractalDB pre-trained model on limited 3D bounding box annotations. We randomly sample {1, 2, 4, 7} 3D bounding boxes per scene from ScanNetV2 training data and use the official ScanNetV2 validation set for evaluation (for details, refer to [26]). As seen in Fig. 1, the PC-FractalDB pre-trained model produced higher scores than the PointContrast pre-trained model and CSC pre-trained model on all limited annotation subsets.

Effects of supervisor label in pre-training. (see Table 12). We investigated the pre-training task regarding which formula-driven and self-supervised learning are more effective in the PC-FractalDB. For self-supervised learning, the PC-FractalDB is given pseudo-labels from two different viewpoints based on the implementation of PointContrast. Table 12 shows that the formula-driven score improved by +1.8% on ScanNetV2 and +2.8% on SUN RGB-D compared to self-supervised learning.

Table 11. 3D object detection comparisons on representative datasets. We employed architecture with the basic VoteNet model and used them to compare network pre-training methods, including training from scratch, PointContrast [67], CSC [26], RandomRooms [51], and the PC-FractalDB. The **Underlined bold** and **bold** scores indicate the best and second best values, respectively.

Pre-training	Backbone	Parameter	Input	ScanNetV2		SUN RGB-D	
				mAP@0.25	mAP@0.50	mAP@0.25	mAP@0.50
Scratch	PointNet++	0.95M	Geo + Height	57.9	32.1	57.4	32.8
Scratch	SR-UNet	38.2M	Geo	57.0	35.8	56.1	34.2
RandomRooms [51]	PointNet++	0.95M	Geo + Height	61.3	36.2	59.2	35.4
PointContrast [67]	SR-UNet	38.2M	Geo	59.2	38.0	57.5	34.8
CSC [26]	SR-UNet	38.2M	Geo	-	**39.3**	-	**36.4**
PC-FractalDB	PointNet++	0.95M	Geo + Height	**61.9**	38.3	**59.4**	33.9
PC-FractalDB	PointNet++ ×2	38.2M	Geo + Height	**63.4**	**39.9**	**60.2**	35.2
PC-FractalDB	SR-UNet	38.2M	Geo	59.4	37.0	57.1	**35.9**

Table 12. Effects of supervisor label in pre-training.

Supervisor label	ScanNetV2	SUN RGB-D
	mAP@0.25	mAP@0.25
PointContrast (SSL)	57.6	54.3
3D IFS (FDSL)	**59.4**	**57.1**

5. Discussion

The fractal geometric feature is essential. Table 4 and Table 11 shows that the PC-FractalDB is more effective than 3D scenes constructed by a single-object CAD model such as RandomRooms in pre-training. The proposed PC-FractalDB can pre-train complex geometric shapes than a CAD model. This leads us to consider that the PC-FractalDB can learn relatively more diverse variations and common 3D patterns in real-world than conventional 3D datasets because of constructing based on fractal geometry, it is important for effective pre-training.

Pre-training with 3D object detection task is effective. Table 2 shows that pre-training considering the entire 3D object detection task is more effective than pre-training the backbone network only. This leads us to consider that the previous self-supervised learning approach enabled only the backbone network to be initialized with the pre-trained model, but our proposed method enabled the entire network to be initialized with the PC-FractalDB pre-trained model is contributing importance to effective pre-training. Furthermore, Fig. 1 shows that the PC-FractalDB is more effective for limited data and annotation than the previous self-supervised learning. This leads us to consider that pre-training with a large quantity of 3D scenes is important for limited datasets. The concept of constructing 3D datasets with FDSL, which does not require manual data collection and annotation, is up-and-coming for 3D vision.

How to assign a supervisor label is essential. Table 12 shows the PC-FractalDB recorded better scores by 3D fractal data and supervisor labeled pair from the mathematical formula than an external label with PointContrast. This leads us to consider that the object-label relationship is essential to acquiring better feature representations in pre-training. Furthermore, our method can assign consistent supervisor labels to a great quantitive of auto-generated 3D fractal data based on the mathematical formula.

6. Conclusion

In order to address the challenging problem of pre-training in 3D point clouds, we proposed a method designed to simplify the construction of 3D datasets under the formula-driven supervised learning framework. We designed PC-FractalDB, a novel FDSL family inspired by fractal geometry encountered in natural 3D structures. The most important is 3D dataset automatically construction so as not to require scanned data and human annotation differ from previous self-supervised learning. We showed that our proposed PC-FractalDB significantly improved the performance of 3D object detection. In addition, we discovered important parameters for use in the pre-training dataset construction by comprehensively investigating the categories, instances, scenes, *etc.*, of 3D datasets. In particular, the PC-FractalDB pre-trained model indicates more effectiveness for limited training data and annotation than previous self-supervised learning since the entire network is available to pre-train. As a result, we discovered construct conception for the effective pre-training dataset for 3D detection, and we believe that our PC-FractalDB will provide an essential key to increasing understanding of 3D scenes in the future.

Acknowledgement: This paper is based on results obtained from a project, JPNP20006, commissioned by the New Energy and Industrial Technology Development Organization (NEDO). Computational resource of AI Bridging Cloud Infrastructure (ABCI) provided by National Institute of Advanced Industrial Science and Technology (AIST) was used. We want to thank Ryosuke Araki and Kai Watabe for their helpful research discussions.

References

[1] Idan Achituve, Haggai Maron, and Gal Chechik. Self-supervised learning for domain adaptation on point clouds. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 123–133, 2021.

[2] Antonio Alliegro, Davide Boscaini, and Tatiana Tommasi. Joint supervised and self-supervised learning for 3d real world challenges. In *Proceedings of the 25th International Conference on Pattern Recognition (ICPR)*, pages 6718–6725, 2021.

[3] Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1534–1543, 2016.

[4] Michael F. Barnsley. Fractals Everywhere. *Academic Press. New York*, 1988.

[5] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 35(8):1798–1828, 2013.

[6] Piotr Bojanowski and Armand Joulin. Unsupervised learning by predicting noise. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 517–526, 2017.

[7] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[8] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9650–9660, 2021.

[9] Joao Carreira, Eric Noland, Chloe Hillier, and Andrew Zisserman. A short note on the kinetics-700 human action dataset. *arXiv preprint arXiv:1907.06987*, 2019.

[10] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[11] Jintai Chen, Biwen Lei, Qingyu Song, Haochao Ying, Danny Z Chen, and Jian Wu. A hierarchical graph network for 3d object detection on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 392–401, 2020.

[12] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 1597–1607, 2020.

[13] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15750–15758, 2021.

[14] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1907–1915, 2017.

[15] Yunlu Chen, Vincent Tao Hu, Efstratios Gavves, Thomas Mensink, Pascal Mettes, Pengwan Yang, and Cees GM Snoek. Pointmixup: Augmentation for point clouds. In *Proceedings of the 16th European Conference on Computer Vision (ECCV)*, pages 330–345, 2020.

[16] Bowen Cheng, Lu Sheng, Shaoshuai Shi, Ming Yang, and Dong Xu. Back-tracing representative points for voting-based 3d object detection in point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8963–8972, 2021.

[17] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5828–5839, 2017.

[18] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[19] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 647–655, 2014.

[20] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, pages 3354–3361, 2012.

[21] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

[22] JunYoung Gwak, Christopher Choy, and Silvio Savarese. Generative sparse detection networks for 3d single-shot object detection. In *Proceedings of the 16th European Conference on Computer Vision (ECCV)*, pages 297–313, 2020.

[23] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6546–6555, 2018.

[24] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3d object detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11873–11882, 2020.

[25] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9729–9738, 2020.

[26] Ji Hou, Benjamin Graham, Matthias Nießner, and Saining Xie. Exploring data-efficient 3d scene understanding with contrastive scene contexts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15587–15597, 2021.

[27] Siyuan Huang, Yichen Xie, Song-Chun Zhu, and Yixin Zhu. Spatio-temporal self-supervised representation learning for 3d point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6535–6545, 2021.

[28] Nakamasa Inoue, Eisuke Yamagata, and Hirokatsu Kataoka. Initialization using perlin noise for training networks with a limited amount of data. In *Proceedings of the 25th International Conference on Pattern Recognition (ICPR)*, pages 1023–1028, 2021.

[29] Allison Janoch, Sergey Karayev, Yangqing Jia, Jonathan T Barron, Mario Fritz, Kate Saenko, and Trevor Darrell. A category-level 3d object dataset: Putting the kinect to work. In *IEEE/CVF International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1168–1174. 2011.

[30] Longlong Jing, Yucheng Chen, Ling Zhang, Mingyi He, and Yingli Tian. Self-supervised modal and view invariant feature learning. *arXiv preprint arXiv:2005.14169*, 2020.

[31] Hirokatsu Kataoka, Asato Matsumoto, Ryosuke Yamada, Yutaka Satoh, Eisuke Yamagata, and Nakamasa Inoue. Formula-driven supervised learning with recursive tiling patterns. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4098–4105, 2021.

[32] Hirokatsu Kataoka, Kazushige Okayasu, Asato Matsumoto, Eisuke Yamagata, Ryosuke Yamada, Nakamasa Inoue, Akio Nakamura, and Yutaka Satoh. Pre-training without Natural Images. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2020.

[33] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv preprint arXiv:1811.00982*, 2018.

[34] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12697–12705, 2019.

[35] Xinhai Liu, Xinchen Liu, Zhizhong Han, and Yu-Shen Liu. Spu-net: Self-supervised point cloud upsampling by coarse-to-fine reconstruction with self-projection optimization. *arXiv preprint arXiv:2012.04439*, 2020.

[36] Yunze Liu, Li Yi, Shanghang Zhang, Qingnan Fan, Thomas Funkhouser, and Hao Dong. P4contrast: Contrastive learning with pairs of point-pixel pairs for rgb-d scene understanding. *arXiv preprint arXiv:2012.13089*, 2020.

[37] Yueh-Cheng Liu, Yu-Kai Huang, Hung-Yueh Chiang, Hung-Ting Su, Zhe-Yu Liu, Chin-Tang Chen, Ching-Yu Tseng, and Winston H Hsu. Learning from 2d: Pixel-to-point knowledge transfer for 3d pretraining. *arXiv preprint arXiv:2104.04687*, 2021.

[38] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. *arXiv preprint arXiv:1907.03739*, 2019.

[39] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3d object detection via transformers. *arXiv preprint arXiv:2104.00678*, 2021.

[40] Xiaoyuan Luo, Shaolei Liu, Kexue Fu, Manning Wang, and Zhijian Song. A learnable self-supervised task for unsupervised domain adaptation on point clouds. *arXiv preprint arXiv:2104.05164*, 2021.

[41] Benoit Mandelbrot. The fractal geometry of nature. *American Journal of Physics*, 51(3), 1983.

[42] Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2906–2917, 2021.

[43] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7074–7082, 2017.

[44] Kodai Nakashima, Hirokatsu Kataoka, Asato Matsumoto, Kenji Iwata, and Nakamasa Inoue. Can vision transformers learn without natural images? *arXiv preprint arXiv:2103.13023*, 2021.

[45] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *Proceedings of the 16th European Conference on Computer Vision (ECCV)*, pages 69–84, 2016.

[46] Xuran Pan, Zhuofan Xia, Shiji Song, Li Erran Li, and Gao Huang. 3d object detection with pointformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7463–7472, 2021.

[47] Charles R Qi, Xinlei Chen, Or Litany, and Leonidas J Guibas. Imvotenet: Boosting 3d object detection in point clouds with image votes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4404–4413, 2020.

[48] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3d object detection in point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9277–9286, 2019.

[49] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 918–927, 2018.

[50] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[51] Yongming Rao, Benlin Liu, Yi Wei, Jiwen Lu, Cho-Jui Hsieh, and Jie Zhou. Randomrooms: Unsupervised pre-training from synthetic shapes and randomized layouts for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3283–3292, 2021.

[52] Jonathan Sauder and Bjarne Sievers. Self-supervised deep learning on point clouds by reconstructing space. *arXiv preprint arXiv:1901.08396*, 2019.

[53] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8430–8439, 2019.

[54] Charu Sharma and Manohar Kaul. Self-supervised few-shot learning on point clouds. *arXiv preprint arXiv:2009.14168*, 2020.

[55] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10529–10538, 2020.

[56] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–779, 2019.

[57] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Proceedings of the 16th European Conference on Computer Vision (ECCV)*, pages 746–760, 2012.

[58] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 567–576, 2015.

[59] Riccardo Spezialetti, Federico Stella, Marlon Marcon, Luciano Silva, Samuele Salti, and Luigi Di Stefano. Learning to orient surfaces by self-supervised spherical cnns. *arXiv preprint arXiv:2011.03298*, 2020.

[60] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 843–852, 2017.

[61] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2446–2454, 2020.

[62] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1588–1597, 2019.

[63] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matthew J Kusner. Pre-training by completing point clouds. *arXiv preprint arXiv:2010.01089*, 2020.

[64] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matthew J Kusner. Unsupervised point cloud pre-training via view-point occlusion, completion. *arXiv preprint arXiv:2010.01089*, 2020.

[65] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, 2015.

[66] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. Sun3d: A database of big spaces reconstructed using sfm and object labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1625–1632, 2013.

[67] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *Proceedings of the 16th European Conference on Computer Vision (ECCV)*, pages 574–591, 2020.

[68] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7652–7660, 2018.

[69] Richard Zhang, Phillip Isola, and Alexei A Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1058–1067, 2017.

[70] Zaiwei Zhang, Rohit Girdhar, Armand Joulin, and Ishan Misra. Self-supervised pretraining of 3d features on any point-cloud. *arXiv preprint arXiv:2101.02691*, 2021.

[71] Zaiwei Zhang, Bo Sun, Haitao Yang, and Qixing Huang. H3dnet: 3d object detection using hybrid geometric primitives. In *Proceedings of the 16th European Conference on Computer Vision (ECCV)*, pages 311–329, 2020.

[72] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 40(6):1452–1464, 2017.

[73] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4490–4499, 2018.