2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

PC-NBV: A Point Cloud Based Deep Network for Efficient Next Best View Planning

Rui Zeng[1], Wang Zhao[1], and Yong-Jin Liu[1†]

***Abstract*— The Next Best View (NBV) problem is important in the active robotic reconstruction. It enables the robot system to perform scanning actions in a reasonable view sequence, and fulfil the reconstruction task in an effective way. Previous works mainly follow the volumetric methods, which convert the point cloud information collected by sensors into a voxel representation space and evaluate candidate views through ray casting simulations to pick the NBV. However, the process of volumetric data transformation and ray casting is often time-consuming. To address this issue, in this paper, we propose a point cloud based deep neural network called PC-NBV to achieve efficient view planning without these computationally expensive operations. The PC-NBV network takes the raw point cloud data and current view selection states as input, and then directly predicts the information gain of all candidate views. By avoiding costly data transformation and ray casting, and utilizing powerful neural network to learn structure priors from point cloud, our method can achieve efficient and effective NBV planning. Experiments on multiple datasets show the proposed method outperforms state-of-the-art NBV methods, giving better views for robot system with much less inference time. Furthermore, we demonstrate the robustness of our method against noise and the ability to extend to multi-view system, making it more applicable for various scenarios.**

I. INTRODUCTION

Active robotic reconstruction has considerable significance in many fields including medical, agriculture, industrial applications. Generally, obtaining a digital model of a three-dimensional object is important for many downstream applications. To achieve efficient robotic reconstruction, view planning is inevitable. It studies the problem of how to efficiently find the next best view (NBV) for the robot system sensor after each scanning. A practical view planner can quickly plan a robot's effective scan view sequence, enabling the robot system to complete a full scan of a 3D object with fewer views, thus makes the whole reconstruction faster.

Due to its importance for a wide range of robotic applications, the NBV problem has drawn much attention for a long time [1][2][3][4]. Most works follow the classic generate-and-test pipeline [2] to find the NBV. In this pipeline, first, a couple of candidate views are sampled according to the system constraints. Then, the view planner evaluates every single view to predict their information gain, and decides

*This work was partially supported by the Natural Science Foundation of China (61725204, 61521002) and Grant from Science and Technology Department of Jiangsu Province, China.

[1]R. Zeng, W. Zhao and Y-J Liu are with BNRist, MOE-Key Laboratory of Pervasive Computing, Department of Computer Science and Technology, Tsinghua University, Beijing, China. `zengr17@mails., zhao-w19@mails. and liuyongjin@}tsinghua.edu.cn`

[†]Corresponding author

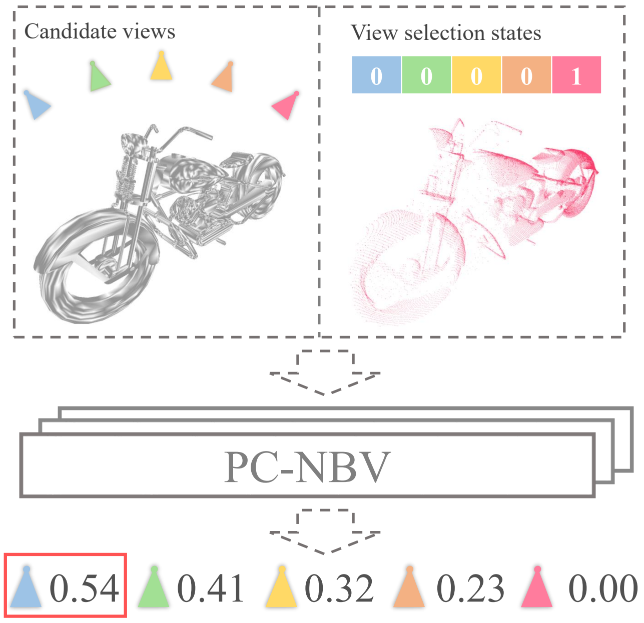


Fig. 1. The complete motorbike model in the left is the target model for reconstruction and the right partial point cloud is the current reconstruction result. Our proposed PC-NBV network predicts the information gain of each candidate view through partial point cloud and view selection states.

the next best view. The class of volumetric methods [5] is the mainstream for view evaluation. It transforms raw point cloud data — acquired by depth cameras — into voxel representation, and performs simulated ray projection at each candidate view to estimate the information gain. While this kind of volumetric methods can reasonably evaluate the contribution of views for object reconstruction, the computational costs in data transformation and ray casting are quite high, making it inefficient for time-critical systems.

Recently, deep learning techniques have made breakthroughs in 3D computer vision, including point cloud processing [6][7] such as classification, detection and segmentation. As a data-driven approach, the neural network could improve run-time efficiency at the expense of pre-training calculations and model memory footprint, which is suitable for the NBV problem. In this paper, we introduce a novel PC-NBV network to efficiently solve the NBV problem based on point cloud representation. As shown in Fig. 1, the network takes the point cloud data of the currently scanned object surface and the view selection states as input, and directly evaluates the information gain of all candidate views. By avoiding costly data transformation and ray-casting operations, the PC-NBV network achieves significant improvement on efficiency compared to previous baselines. Moreover, the network implicitly learns 3D structure priors from large-scale training data, and thus gives better view planning results. The PC-NBV network also has strong generalization ability. Once trained on a dataset, it can directly apply on various 3D models to infer NBV without fine-tuning. To the best of the authors' knowledge, PC-NBV is the first work to directly process point cloud data for inferring the next best view.

In addition to predicting the next best view one by one, our method can naturally extend to the multi-view setting. For example, in a multi-robot collaboration system, the view planner needs to select the best view simultaneously for every robot. Our system uses an iterative erasing strategy to enable the planning of arbitrary number of views at once. Our experiments show that PC-NBV well utilizes the capacity of multi-views, making it feasible for multi-robot collaboration applications.

In this paper, extensive experiments are conducted on the ShapeNet dataset [8], ABC dataset [9] and 11 complex models from The Stanford 3D Scanning Repository and MIT CSAIL Textured Models Database. We perform a simulated reconstruction for 3D objects, and report the surface coverage with inference time to check the performance of proposed method. Results show that our PC-NBV network outperforms state-of-the-art methods on both effectiveness and efficiency. Specifically, we achieves 20 times faster inference speed with better reconstruction quality compared to previous methods. Moreover, by randomly permuting the point cloud input, we demonstrate the strong robustness of our proposed method against noise.

The contributions of this work are summaried as follows: 1) We introduce a novel PC-NBV network to handle the next best view problem, which is the first work to directly process point cloud data to infer the next best view for reconstruction; 2) Extensive experiments show that our PC-NBV network improves the inference speed of NBV by several magnitudes, and achieves better reconstruction results compared to previous representative NBV methods; 3) The proposed PC-NBV network generalizes pretty well to unseen test data and has strong resistance to noise; 4) A simple multi-view extension of the PC-NBV network is provided to utilize multi-robot capacity.

II. Related Work

A. Next best view methods

NBV is a long-standing problem in robotics and has received much attention (e.g., [1], [2], [3], [4]) . Based on different planning strategies, the NBV methods can be broadly classified in synthesis methods and generate-and-test methods. The synthesis methods directly calculate the pose of next best view under certain system and task constraints. Although they have low computational cost, the robustness of these methods is poor and makes them not very stable. Most works prefer the generate-and-test method because they can achieve good balance between efficiency and the quality of planning views. Our method proposed in this paper falls in the generate-and-test methods.

Different 3D data structures have been used in different NBV methods to represent the workspace, including point cloud, voxels and meshes, etc. Some works use triangular meshes to represent the surface of the scanned 3D object [10][11]. However, the computation and storage costs of meshes are high. Compared to mesh-based approaches, volumetric approaches are more widely used. As early as the 1980s, Connolly [1] proposed the idea of sampling views under spherical constraints and evaluating the information gain of candidate view in the voxel space. Later bunch of works followed this idea, evaluating sensor views through simulated ray casting in the voxel space [12][13][14][15][16][17]. Daudelin *et al.* [18] integrated object probability to weight cost functions in view evaluations. Monica *et al.* [19] further utilized point cloud segmentation to detect saliency for scoring the candidate views. Recently, Delmerico *et al.* [5] summarized the volumetric approaches for the NBV problem, and compared several formulation of evaluating information gain by simulation experiments. Despite of the simplicity of the idea and reasonably good view planning performance, the volumetric methods still suffer from costly data transformation from raw point cloud to voxel and ray casting, which makes it hard to apply on time-critical tasks. As for point cloud data, to the best of the authors' knowledge, so far the method that directly predicts NBV from point cloud data for reconstruction does not exist.

B. Deep learning for point cloud processing

Recently deep learning has been sucessfully applied on point cloud processing. PointNet [6] and PointNet++ [7] are pioneers in this direction. They make use of multi-layer perceptron (MLP) and the global symmetric function to handle unordered point set, and achieve impressive performance on point cloud classification and segmentation.

Following these two works, a variety of methods [20][21][22][23][24] utilize deep neural network for a wide range of point cloud applications such as detection, instance segmentation, completion, generation, etc. The success of these applications demonstrates the powerful ability of neural network to learn valuable 3D shape priors from training data.

C. Deep learning for view planning

Deep learning has also been introduced into the field of NBV planning. Wu *et al.* [25] proposed a convolutional deep belief network to achieve object completion based on depth image information. Then the completion prediction from the network is used to guide view planning for recognition. The same idea is applied in [26] to complete the selection of view groups through the point cloud completion network. In [26], voxelization and ray projection are involved to evaluate the candidate views, which are quite time-consuming. Different from their work, our unified method directly produces the NBV without point cloud completion and further evaluation.

In the work [27], the convolutional neural network (CNN) is used to find the view that matches with current view to maximize the object recognition rate. The work proposed by Hepp *et al.* [28] uses a 3D convolution network to learn the utility function for evaluating the candidate sensor views. However, they focus on scene exploration rather than object reconstruction. Recently, Mendoza *et al.* [29] proposed NBV-Net to solve the NBV problem for 3D object reconstruction. NBV-Net adopts voxel representation and thus cannot overcome the time-consuming bottleneck. Different from the above mentioned methods, our proposed PC-NBV network directly inputs point cloud data, and predicts the NBV in an efficient way.

III. Problem Identification

Due to self-occlusion of complex 3D objects and physical limitation of sensors, 3D object reconstruction requires the sensor to move across different views around the object to capture new information. The task of view planner is to decide a serials of views $V_{plan} = \{v_i | i = 0,1,2,...,n\} \subset V$ for the sensor to completely cover the surface of the object with as few views as possible, where $V = \mathbb{R}^3 \times SO(3)$ represents the sensor view space. However, even with the prior knowledge of object geometry, the problem of finding the best view sequence V_{best} is still an NP-C problem [30].

It is common to use greedy search for this problem. That is, at the beginning of the entire reconstruction process, a number of candidate views have been sampled according to the system constraint, which form the candidate viewpoint space V_c. Initially, the first view v_0 is randomly selected in V_c to start the scanning process. After each round of scanning, all candidate viewpoints will be re-evaluated by utility function f_{util} to select the next best view $v_{best} \in V_c$. Different works define f_{util} in different ways. In this paper, we propose a point cloud network called PC-NBV to learn the utility function, which measures the improvement of object surface coverage.

IV. Learning NBV

In this section, we first present the technical details of PC-NBV network, including the details of training data, the definition of utility function and the network architecture. Then we present the iterative-erasing strategy that enables PC-NBV to be naturally extended to multi-view systems.

A. Training supervision

To train PC-NBV network for NBV prediction, it is vital to build effective training supervision, i.e. pairing input data and target ground-truth. We perform a simulated reconstruction process on synthetic 3D object models to get these training pairs. The process is summarized in Alg. 1.

This preparation process needs the object mesh model O, the complete point cloud P_o, the candidate view space V_c, and the maximum scanning number max_{iter} as input. Mesh model O and complete point cloud P_o of objects are easy to acquire from synthetic 3D datasets like ShapeNet [8]. For the candidate view space V_c, we choose a common spherical sampling method to define it. By uniformly sampling viewpoints on sphere around object, a generous view space V_c can be obtained, where $\|V_c\| = m$. The sensor could obtain a depth map of the object at each view $v_i \in V_c$ and project it to get point cloud $P(v_i, O)$.

Algorithm 1: NBV training data preparation

```
Input: $O$, $P_o$, $V_c$, $max_{iter}$.
1  $V^k_{state} \leftarrow 0$ $(k = 1,2,3,...,m)$
2  $iter \leftarrow 0$
3  $P_{part} \leftarrow \varnothing$
4  $i \leftarrow Random(1:m)$
5  while $iter < max_{iter}$ do
6      $V^i_{state} \leftarrow 1$
7      $P_{part} += P(v_i, O)$
8      $max \leftarrow 0$
9      foreach $j \leftarrow 1:m$ do
10         if $C(P^j_{new}) > max$ then
11             $max \leftarrow C(P^j_{new})$
12             $i \leftarrow j$
13         end
14     end
15     $Save(P_{part}, V_{state}, C(P_{new}))$
16     $iter++$
17 end
```

To make the network learn to efficiently select views and reconstruct objects, we directly use surface coverage score as the supervision signal. Given a partial point cloud P and its target complete point cloud P_o, the surface coverage is defined as:

$$C(P) = \frac{1}{|P_o|} \sum_{p \in P} \mathrm{U}\left(\min_{p_o \in P_o} ||p - p_o||_2 - \varepsilon \right) \tag{1}$$

Where U is the Heaviside step function, and ε is a distance threshold.

The training supervision pair for PC-NBV network is defined as $(P_{part}, V_{state}, C(P_{new}))$. The network takes (P_{part}, V_{state}) as input and is supposed to output prediction close to groundtruth $C(P_{new})$. The input P_{part} indicates the partial point cloud of the object, and the V_{state} indicates the selection status of the candidate view space and is defined as follows:

$$V^i_{state} = \begin{cases} 1, & v_i \in V_{selected} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Where $V_{selected}$ is the set of candidate views that are already used by sensor. The groundtruth $C(P_{new}) \in \mathbb{R}^m$ is a vector containing m surface coverage scores for each newly added point cloud P^j_{new} from candidate view v_j. The newly added point cloud P^j_{new} from view v_j is defined as:

$$P^j_{new} = \{p_v | p_v \in P(v_j, P_o), \min_{p \in P_{part}} ||p_v - p||_2 > \varepsilon\} \tag{3}$$

In each iteration, after fusing the point cloud (concatenation is used for simplicity) from the current view into P_{part}, the values of all the views in $C(P_{new})$ are updated. So a tuple $(P_{part}, V_{state}, C(P_{new}))$ is saved as a training pair for PC-NBV network. After repeating this process for each model

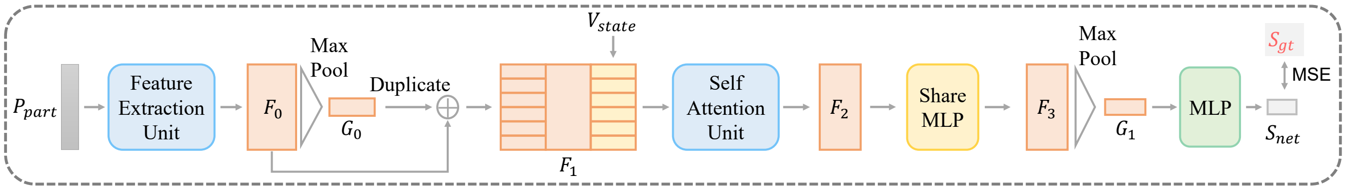


Fig. 2. Network Architectures. The network first extracts the global feature of the partial point cloud. Then, the global feature, partial point cloud, and view selection state are fused together to predict the information gain of each candidate view.

from a 3D model dataset, we can obtain the entire NBV training data. The training pairs come from all stages of the reconstruction process, so the network can learn to evaluate views in different situations.

B. Network architecture

The architecture of the PC-NBV network is illustrated in Fig. 2. First, a partial point cloud P_{part} is processed by feature extraction unit proposed by [24]. This unit extracts features from different network layers and integrates them through dense connections to get point-wise feature F_0. Then the global feature G_0 is obtained through max pooling. Subsequently, the network duplicates G_0 and another input V_{state}, concatenates them with F_0 to get augmented point-wise feature F_1. A self-attention [23] module is used to better integrate the features from the partial point cloud, global feature and view selection state. Then, followed by another shared MLP and max-pooling, a final global feature G_1 is computed, which contains all the essential information about current reconstruction. Finally, views scores $S_{net} \in \mathbb{R}^m$ are obtained by MLP from G_1.

We use the mean squared error (MSE) to calculate the loss between S_{gt} and S_{net}. The S_{gt} is exactly the coverage of new point cloud $C(P_{new})$, which is defined in the previous section. At the testing phase, the next best view is chosen according to the scores S_{net} predicted by PC-NBV network. Specifically, $v_{best} = v_{i^*}, \quad i^* = \arg\max_i S^i_{net}$.

Experiments in Section V show that the network structure can effectively learn the ideal utility function and quickly provide a reasonable score for all candidate viewpoints during reconstruction. By picking the NBV according to S_{net}, reconstruction tasks can be performed in an efficient manner.

C. Extension to multi-view systems

We propose an iterative erasing strategy, so that PC-NBV can efficiently select multi-NBVs for multi-view systems, without modifying the network architecture or retraining the network. The pseudo-code of multi-NBVs selection is summarized in Alg. 2. The details of this strategy is as follows.

After selecting one view v_i based on the output S_{net} of PC-NBV network, we mark it as used in V_{state}. Then, the updated view space state V_{state} and partial point cloud P_{part} are fed to the network again to get new scores. This process is iterated for r times until the system obtains all r planing views for the next round. Through the pseudo-marking, the network is forced to avoid those selected views when picking a new NBV. Our experiments in Section V demonstrate that by only updating the view selection state, our PC-NBV model can make good predictions for view planning, i.e., efficiently selecting a group of views before each round scanning, which enables the multi-robot collaborations.

Algorithm 2: Multi-view selection though PC-NBV

```
  Input: $P_{part}$, $V_{state}$
  Output: $V_{best}$
1 foreach $i \in \{1, 2, ..., r\}$ do
2     $S_{net} \leftarrow f_{PC-NBV}(P_{part}, V_{state})$
3     $j^* \leftarrow argmax_j S^j_{net}$
4     $V_{best} += v_{j^*}$
5     $V^{j^*}_{state} \leftarrow 1$
6 end
7 return $V_{best}$
```

V. Experiment

In this section, all experiments are conducted on a PC with Inter(R) Xeon(R) CPU E5-2640 v4 @ 2.40GHz and Nvidia Geforce RTX 2080 Ti GPU. We train PC-NBV using the training dataset built from ShapeNet [8] and perform tests on our testing split of ShapeNet [8] , ABC dataset [9] and 11 scanned complex object models. Scanned complex models are from Stanford 3D Scanning Repository[1] and MIT CSAIL Textured Models Database[2]. We compare PC-NBV with state-of-the-art volumetric methods [14][5] and voxel-based deep learning method called NBV-Net [29]. We also verify the noise-insensitivity and check the NBV performance of our multi-view extension. Code will be made publicly available.

A. Network training

NBV dataset. We choose the large 3D CAD model dataset ShapeNet [8] as the main model dataset. Following the train/test split of the ShapeNet object categories of [21], we randomly pick 4,000 models from the 8 categories (airplane, cabinet, car, chair, lamp, sofa, table, and vessel) as training models, 400 models as validation models, and 400 as similar testing models. Likewise, we randomly pick 400 models from another unseen 8 categories (bus, bed, bookshelf, bench, guitar, motorbike, skateboard, pistol) as novel testing models. Fig. 3 shows some object examples

[1]`graphics.stanford.edu/data/3Dscanrep/`
[2]`people.csail.mit.edu/tmertens/textransfer/data/`

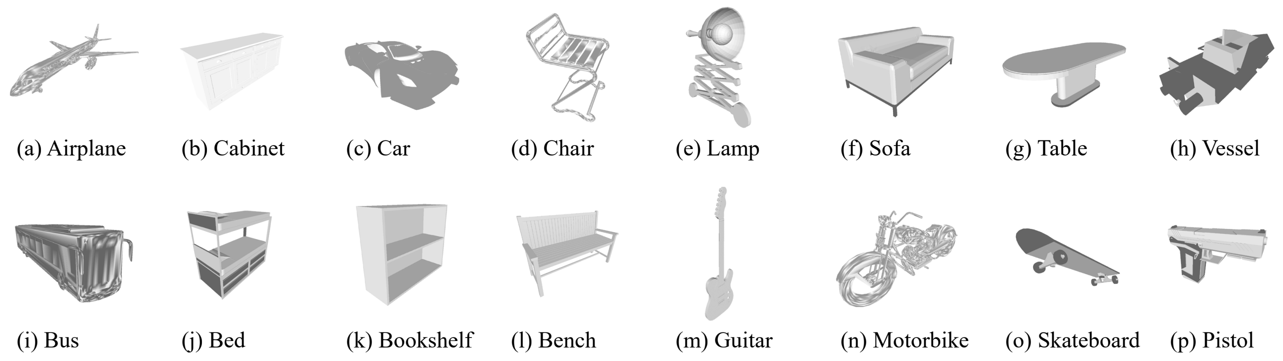


Fig. 3. Example models from ShapeNet dataset. The first row shows sample models from the 8 categories used for both training and testing (similar testing dataset), and the second row contains models from the 8 novel testing categories (novel testing dataset).

in ShapeNet from 16 categories. The training and testing models both contain some complex shape categories such as Lamp, Vessel, Bookshelf, Motorbike, etc.

Parameters details. For training, validating and testing, each model follows Alg. 1 to perform simulated reconstruction and collect input and ground truth data. We uniformly sample 16,384 points from object mesh O as P_o. We set $\varepsilon = 0.00707m$ for surface coverage calculation, the candidate view number $m = 33$ and $max_{iter} = 10$. When performing the back-projecting to obtain P_{part}, we adopt the camera intrinsic parameters from the Kinect sensor. The network is trained by Adam [31] optimizer with base learning rate of 0.0001 and mini-batch size of 32. The input point cloud are randomly down-sampled to 512 points for training, and 1,024 points for testing. After every 50K iterations, the learning rate is decayed by 0.7. We use L2 loss for weight regularization and set $\lambda = 0.0001$. When testing the NBV methods, we iteratively scan the object with the predicted camera views and terminate the reconstruction after 10 iterations. Fixed number of iterations is used in experiments for clear comparison of different NBV methods. For the real-world practice, the reconstruction process would be terminated when the information gain of a new view is less than a defined threshold.

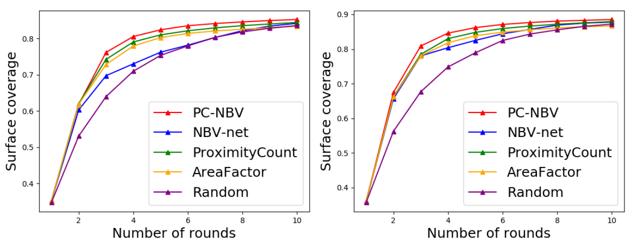
B. Network ablation study

We evaluate the performance of PC-NBV and conduct ablation experiments to test the contribution of each module in PC-NBV network. We use the loss value and spearman's rho to measure the performance of PC-NBV. The loss value can help judge how well the network fits in the validation and testing datasets. The spearman's rho value indicates the rank correlation between the network output and ground truth scores, which is directly related to view selection.

Table III summarizes the ablation study results. "Valid" represents the validation dataset. "Test S" represents the similar testing dataset, and "Test N" represents the novel testing dataset. "No V_{state}" represents removing V_{state} input. "No P_{part}" represents only feed V_{state} to MLP for obtaining S_{net}. "Baseline" represents removing feature extraction unit [24] and self-attention unit [23] from our PC-NBV network.

The results in Table III show that the input of V_{state} is significant for score prediction, and the addition of P_{part} can greatly reduce the loss of the network and improve spearman's rho value by about 0.1. Feature extraction [24] and self-attention unit [23] generally upgrade the network capacity and extract better features from point cloud by fusing global and local information, thus make the network more robust.

C. Reconstruction performance on ShapeNet



(a) Similar testing dataset (b) Novel testing dataset

Fig. 4. Comparison of reconstruction process on ShapeNet testing dataset. Results clearly show that PC-NBV has the highest reconstruction performance in both similar testing dataset and novel testing dataset.

We perform simulation experiments on ShapeNet testing dataset. We also include a random selection method as baseline. To give a quantitative analysis of reconstruction efficiency, following [5], we use area under the curve (AUC) to measure the performance. The value of AUC varies between 0.0 and 1.0, and is higher for better and faster reconstruction. It is worth mentioning that we only use the training set of ShapeNet to train the PC-NBV model once, and then test it on ShapeNet testing dataset and other datasets without further fine-tuning. For baseline methods [5][14], we use the default parameter setting provided in their codes. For learning-based method [29], we train their network under the same setting with ours and report the performance.

Table I reports the average AUC values of different methods in each category of ShapeNet testing dataset. The first eight categories are from the similar test dataset. The rest are from novel test dataset. Results show that our method

TABLE I

COMPARISON ON SHAPENET TEST DATASET. AUC RESULTS ARE REPORTED ON EACH CATEGORY. PC-NBV ACHIEVES THE BEST PERFORMANCE ON MOST CATEGORIES COMPARED TO STATE OF THE ARTS.

	Airplane	Cabinet	Car	Chair	Lamp	Sofa	Table	Vessel	Bus	Bed	Bookself	Bench	Guitar	Motorbike	Skateboard	Pistol	Average
Random	0.745	0.545	0.542	0.724	0.770	0.589	0.710	0.674	0.609	0.619	0.695	0.795	0.795	0.672	0.768	0.614	0.678
Proximity Count [5]	**0.800**	0.596	0.591	0.772	**0.803**	0.629	0.753	0.706	0.646	0.645	**0.749**	0.829	**0.854**	0.705	0.828	0.660	0.723
Area Factor [14]	0.797	0.585	0.587	0.751	0.801	0.627	0.725	0.714	0.629	0.631	0.742	0.827	0.852	0.718	0.799	0.660	0.715
NBV-Net[29]	0.778	0.576	0.596	0.743	0.791	0.599	0.693	0.667	0.654	0.628	0.729	0.824	0.834	0.710	0.825	0.645	0.706
PC-NBV	0.799	**0.612**	**0.612**	**0.782**	0.800	**0.640**	**0.760**	**0.719**	**0.667**	**0.662**	0.740	**0.845**	0.849	**0.728**	**0.840**	**0.672**	**0.733**

TABLE II

COMPARISONS ON COMPLEX MODELS. AUC RESULTS ARE REPORTED. WE CONDUCT THE RECONSTRUCTION FOR 10 TIMES AND THE RESULTS ARE AVERAGED TO REDUCE RANDOMNESS.

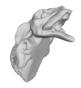
	Lion	Dragon	Gargoyle 	Bust	Bunny	Head	Bird	Buddha	Column	Owl	Armadillo	Average
Random	0.781	0.777	0.778	0.802	0.773	0.786	0.817	0.740	0.809	0.776	0.795	0.785
Proximity Count [5]	0.838	0.803	0.813	**0.846**	0.835	**0.831**	**0.852**	0.774	0.833	0.827	0.811	0.824
Area Factor [14]	0.833	0.798	0.802	0.761	0.797	0.732	0.838	0.769	0.767	0.711	0.805	0.783
NBV-Net [29]	0.828	0.790	0.779	0.825	0.818	0.776	0.788	0.782	0.779	0.824	0.764	0.796
PC-NBV	**0.842**	**0.822**	**0.822**	0.843	**0.840**	0.829	0.849	**0.818**	**0.848**	**0.838**	**0.834**	**0.835**

TABLE III

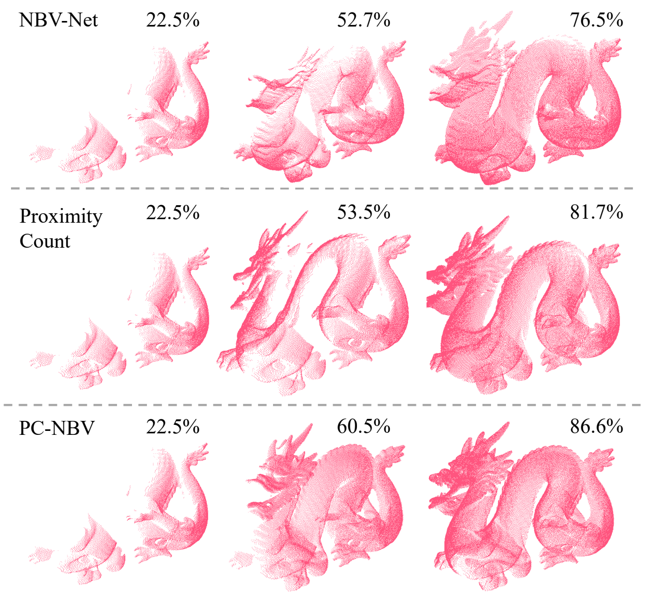
ABLATION STUDY OF PC-NBV ON SHAPENET. THE LOSS VALUE AND SPEARMAN'S RHO VALUE ARE REPORTED.

Network	Loss value			Spearman's rho		
	Valid	Test S	Test N	Valid	Test S	Test N
No V_{state}	0.711	0.612	1.754	0.395	0.396	0.248
No P_{part}	0.412	0.394	0.667	0.758	0.761	0.706
Baseline	0.194	0.180	0.333	0.853	0.864	0.796
PC-NBV	**0.189**	**0.176**	**0.316**	**0.857**	**0.866**	**0.799**

TABLE IV

COMPARISON OF INFERENCE TIME FOR EACH NBV PREDICTION. PC-NBV IS FASTER THAN TRADITIONAL METHODS BY ABOUT 20 TIMES. COMPARED WITH DEEP LEARNING METHOD NBV-NET, OUR METHOD STILL RUNS IN MUCH LESS INFERENCE TIME.

	PC	AF	NBV-Net	Ours(CPU)	Ours(GPU)
Time(s)	2.126	2.166	0.952	0.106	0.034



(a) 1 round (b) 2 rounds (c) 3 rounds

Fig. 5. Reconstruction progress comparison on Dragon model. The surface coverage number is labeled on the upper right corner of each point cloud. PC-NBV makes more effective selections for NBV on this complex model.

outperform all the other methods in most categories, except for in several categories where PC-NBV is only slightly inferior to Proximity Count method [5].

Fig. 4 shows the reconstruction process of all five methods. It shows that our PC-NBV model can reconstruct the complete object faster and has higher efficiency than all other methods. Note that since the inside of models cannot be illuminated, the surface coverage typically converges to a value smaller than 90%.

D. Reconstruction performance on other datasets

To demonstrate that the network model trained on ShapeNet can successfully apply in more practical cases and handle more complex shapes, we further conduct experiments on ABC dataset [9] and 11 object models selected from Stanford 3D Scanning Repository and MIT CSAIL Textured Models Database.

ABC dataset [9] is an huge collection of Computer-Aided Design (CAD) models, and mainly includes various industrial models (See Fig. 6). We use the 10k testing patches subset introduced in its Normal Estimation Benchmark for testing. We only train the network on our training split of ShapeNet and directly test on the ABC testing subset without

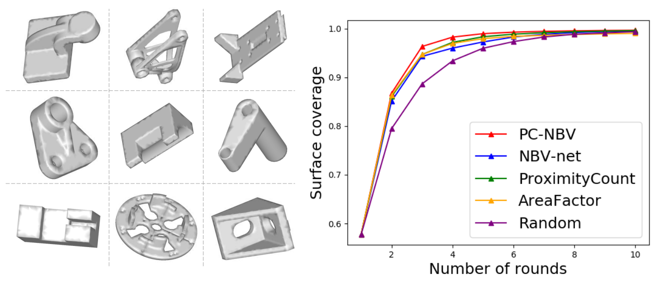


Fig. 6. Left: Some example models in ABC dataset. Right: Comparison of reconstruction on ABC dataset [9]. The results show that PC-NBV still outperfroms other methods in the new dataset without retraining.

re-training or further fine-tuning. As shown in Fig. 6, PC-NBV still outperforms other methods, demonstrating great potentials for mechanical model scanning and reconstruction.

To verify the performance of PC-NBV on complex models, we perform experiments on 11 object models with complex surfaces and asymmetrical shape. The object models and results are shown in Table II. For each object, we repeat the reconstruction for 10 times and average the results to reduce the random noise. Results show that PC-NBV performs best on most of the models and achieves the highest score for averaged AUC value. Fig 5 shows a comparison of the reconstruction process on the dragon model. Compared with traditional voxel methods or voxel-based deep learning methods, the views chosen by PC-NBV is more conducive to quickly improving surface coverage and reconstructing objects.

E. Inference time

In Table. IV, averaged inference time per round are reported for four methods. Our proposed PC-NBV network improves the speed of NBV prediction significantly. The CPU-based PC-NBV method is nearly 20 times faster than the traditional voxel method. By utilizing the computation capacity of GPU, our method can even run faster. The reason is that (1) the network does not need to perform complex data transformations and ray projection operations, and (2) more importantly, the network can evaluate all views through one calculation in parallel.

F. Noise resistance results

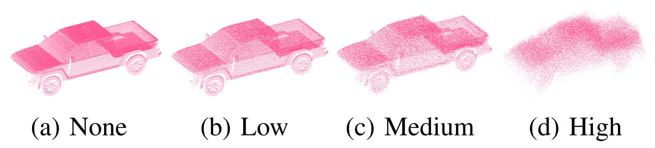


Fig. 7. Noisy point cloud with varying degrees of disturbance. The point cloud gradually loses model details and becomes fuzzy. At the high level of noise interference, we could see many outliers.

TABLE V
EVALUATION OF NOISE RESISTANCE.

Degree of Noise	None	Low	Medium	High
AUC value	0.733	0.733	0.733	0.732

Following the work of [28], We test the noise immunity of our method. When performing the same reconstruction on ShapeNet testing dataset, we perturb the depth image value using normal distribution. Our method is naturally resistant to the random discard of point clouds. This is mainly because that our network only uses a small number of points sampled from original point cloud to predict in both training and testing stage. Specifically, the projected depth maps often contain more than 10,000 points while we only randomly sample 512 or 1024 points to feed into network for training or testing. Thus the network only needs the approximate shape of partial point cloud to infer the NBV.

Fig. 7 shows the noisy point cloud under different disturbance levels. To define the disturbance degree of noise, we set $\sigma = 0.002m$ for low degree, $\sigma = 0.01m$ for medium degree and $\sigma = 0.05m$ for high degree. We conduct this experiment on ShapeNet testing dataset. Table V shows the performance of PC-NBV under different noise degrees. AUC values are calculated by ground truth depth images using the same view sequences. The results show that undergoing the noise interference, the AUC values remain almost the same.

G. Multi-view system evaluations

We also verify the effectiveness of PC-NBV on multi-view systems. Experiments are performed on our test split of ShapeNet. As shown in Fig 8, iterative erasing is our proposed method, and "random" represents picking multiple candidate views in a time with a random manner. "Ideal" indicates the PC-NBV result that is obtained by updating point cloud after selecting each view. Updating point cloud after selecting each view is not feasible in multi-view setting since the system needs to select multiple views before the next scanning, so we use it as an ideal results. Our proposed iterative erasing method is very close to the result of Ideal, which shows the efficiency of this algorithm. Furthermore, when using multiple views, the reconstruction speed is improved significantly when compared to using a single robot. This makes multi-robots collaboration possible.

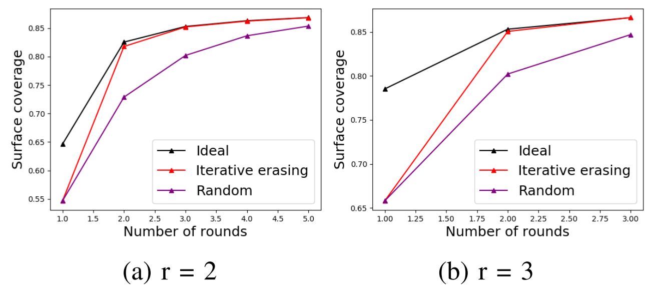


Fig. 8. Multi view results of PC-NBV on all test models of ShapeNet. r is the view number. The results show that our proposed iterative erasing method is very close to the ideal upper bound.

VI. Conclusions

In this article, we introduce a point-cloud-based network called PC-NBV to solve the NBV problem. The network uses the scanned point cloud information to directly score all candidate viewpoints without any other pre-processing.

Experimental results show that our method is nearly 20 times faster than traditional methods and has higher reconstruction effectiveness. The PC-NBV network only needs to be trained once and then can perform well on a variety of models that PC-NBV has never seen before. Experiments on large mechanical dataset and complex object models verify its ability in various real world applications. In addition, the PC-NBV network is insensitive to noise and can be easily extended to a multi-view system. Integrating the PC-NBV network into practical robot systems with careful designs about sensor noise and evaluating the whole view planning system considering the time to execute predicted camera trajectory would be interesting future directions.

References

[1] C. Connolly, "The determination of next best views," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1985, pp. 432–435.

[2] K. A. Tarabanis, P. K. Allen, and R. Y. Tsai, "A survey of sensor planning in computer vision," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 1, pp. 86–104, 1995.

[3] W. R. Scott, G. Roth, and J.-F. Rivest, "View planning for automated three-dimensional object reconstruction and inspection," *ACM Computing Surveys (CSUR)*, vol. 35, no. 1, pp. 64–96, 2003.

[4] S. Chen, Y. Li, and N. M. Kwok, "Active vision in robotic systems: A survey of recent developments," *The International Journal of Robotics Research*, vol. 30, no. 11, pp. 1343–1377, 2011.

[5] J. Delmerico, S. Isler, R. Sabzevari, and D. Scaramuzza, "A comparison of volumetric information gain metrics for active 3d object reconstruction," *Autonomous Robots*, vol. 42, no. 2, pp. 197–208, 2018.

[6] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 652–660.

[7] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 5099–5108.

[8] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[9] S. Koch, A. Matveev, Z. Jiang, F. Williams, A. Artemov, E. Burnaev, M. Alexa, D. Zorin, and D. Panozzo, "Abc: A big cad model dataset for geometric deep learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9601–9611.

[10] R. Pito, "A solution to the next best view problem for automated surface acquisition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 10, pp. 1016–1030, 1999.

[11] S. Chen and Y. Li, "Vision sensor planning for 3-d model acquisition," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 35, no. 5, pp. 894–904, 2005.

[12] N. A. Massios, R. B. Fisher, *et al.*, *A best next view selection algorithm incorporating a quality criterion*. University of Edinburgh, Department of Artificial Intelligence, 1998.

[13] C. Potthast and G. S. Sukhatme, "A probabilistic framework for next best view estimation in a cluttered environment," *Journal of Visual Communication and Image Representation*, vol. 25, no. 1, pp. 148–164, 2014.

[14] J. I. Vasquez-Gomez, L. E. Sucar, R. Murrieta-Cid, and E. Lopez-Damian, "Volumetric next-best-view planning for 3d object reconstruction with positioning error," *International Journal of Advanced Robotic Systems*, vol. 11, no. 10, p. 159, 2014.

[15] J. I. Vasquez-Gomez, L. E. Sucar, and R. Murrieta-Cid, "View/state planning for three-dimensional object reconstruction under uncertainty," *Autonomous Robots*, vol. 41, no. 1, pp. 89–109, Jan 2017.

[16] S. Kriegel, C. Rink, T. Bodenmüller, and M. Suppa, "Efficient next-best-scan planning for autonomous 3d surface reconstruction of unknown objects," *Journal of Real-Time Image Processing*, vol. 10, no. 4, pp. 611–631, 2015.

[17] F. Bissmarck, M. Svensson, and G. Tolt, "Efficient algorithms for next best view evaluation," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 5876–5883.

[18] J. Daudelin and M. Campbell, "An adaptable, probabilistic, next-best view algorithm for reconstruction of unknown 3-d objects," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1540–1547, July 2017.

[19] R. Monica and J. Aleotti, "Contour-based next-best view planning from point cloud segmentation of unknown objects," *Autonomous Robots*, vol. 42, no. 2, pp. 443–458, 2018.

[20] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3d object reconstruction from a single image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 605–613.

[21] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "PCN: Point completion network," in *Proceedings of the International Conference on 3D Vision (3DV)*, 2018, pp. 728–737.

[22] R. Li, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "Pu-gan: a point cloud upsampling adversarial network," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 7203–7212.

[23] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," *arXiv preprint arXiv:1805.08318*, 2018.

[24] W. Yifan, S. Wu, H. Huang, D. Cohen-Or, and O. Sorkine-Hornung, "Patch-based progressive 3d point set upsampling," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5958–5967.

[25] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1912–1920.

[26] C. Wu, R. Zeng, J. Pan, C. C. Wang, and Y.-J. Liu, "Plant phenotyping by deep-learning-based planner for multi-robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3113–3120, 2019.

[27] E. Johns, S. Leutenegger, and A. J. Davison, "Pairwise decomposition of image sequences for active multi-view recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3813–3822.

[28] B. Hepp, D. Dey, S. N. Sinha, A. Kapoor, N. Joshi, and O. Hilliges, "Learn-to-score: Efficient 3d scene exploration by predicting view utility," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 437–452.

[29] M. Mendoza, J. I. Vasquez-Gomez, H. Taud, L. E. Sucar, and C. Reta, "Supervised learning of the next-best-view for 3d object reconstruction," *arXiv preprint arXiv:1905.05833*, 2019.

[30] G. H. Tarbox and S. N. Gottschlich, "Planning for complete sensor coverage in inspection," *Computer Vision and Image Understanding*, vol. 61, no. 1, pp. 84–111, 1995.

[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.