2016 23rd International Conference on Pattern Recognition (ICPR)
Cancún Center, Cancún, México, December 4-8, 2016

3D Point Cloud Object Detection with Multi-View Convolutional Neural Network

Guan Pang
Department of Computer Science
University of Southern California
Los Angeles, CA 90089
Email: gpang@usc.edu

Ulrich Neumann
Department of Computer Science
University of Southern California
Los Angeles, CA 90089
Email: uneumann@usc.edu

***Abstract*—Efficient detection of three dimensional (3D) objects in point clouds is a challenging problem. Performing 3D descriptor matching or 3D scanning-window search with detector are both time-consuming due to the 3-dimensional complexity. One solution is to project 3D point cloud into 2D images and thus transform the 3D detection problem into 2D space, but projection at multiple viewpoints and rotations produce a large amount of 2D detection tasks, which limit the performance and complexity of the 2D detection algorithm choice. We propose to use convolutional neural network (CNN) for the 2D detection task, because it can handle all viewpoints and rotations for the same class of object together, as well as predicting multiple classes of objects with the same network, without the need for individual detector for each object class. We further improve the detection efficiency by concatenating two extra levels of early rejection networks with binary outputs before the multi-class detection network. Experiments show that our method has competitive overall performance with at least one-order of magnitude speed-up comparing with latest 3D point cloud detection methods.**

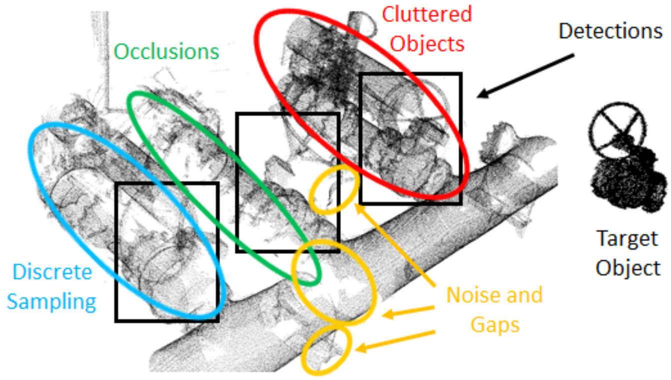


Fig. 1. 3D object detection in point clouds is a challenging problem due to discrete sampling, noisy scans, occlusions and cluttered scenes.

I. Introduction

3D object detection in point clouds is a challenging problem due to discrete sampling, noisy scans, occlusions and cluttered scenes, as illustrated in fig. 1. Many existing methods focus on small-scale data [4], [5], [6], [7], [8] using 3D descriptors. A few others work with large-scale data, mostly urban street scans [9], [10], [12], [13], [14], [15]. Relatively fewer take on industrial part detection[14], [1], where objects are often more densely arranged, making segmentation more difficult. Most methods utilize machine learning to select the best description for a specific type of 3D object, so they can be recognized reliably in a large point cloud scene. Many methods, especially those based on descriptors, require prior segmentation or preprocessing of input data, to reduce the matching complexity of descriptors. Regardless of domain focus, most methods perform the detection process in 3D, either using 3D local descriptors [4], [9], [25], [5], [6], [7] or exhaustive 3D scanning-window search with object detector [14], [18], [1]. Both types of approaches are time-consuming due to the 3-dimensional search. Large-scale industrial or street data contain 100's of millions or billions of 3D points, motivating the exploration for faster 3D detection methods.

On the other hand, 2D object detection in images has improved dramatically, especially in the recent works with convolutional neural network [2], [3]. This motivates a transformation of the 3D object detection problem into a series of 2D detection problems [1]. The 3D-to-2D strategy has been previously used for 3D object model retrieval [26], [19], [20], but our target is unsegmented noisy large-scale 3D point cloud which is much more complex. To transform the 3D problem into 2D space, 3D point cloud will be projected based on depth information at multiple viewpoints and rotations. This will generate a large amount (at least 10,000) of 2D detection task between projections of scenes and objects, which put a requirement on the speed of single 2D detection algorithm that it must be very fast in processing all 2D images to finish the overall 3D detection task in a reasonable time. As a result, this limits the complexity of 2D detection algorithm, and thus the overall performance of 3D detection.

To solve this speed-complexity trade-off, we propose to use convolutional neural network (CNN) for 2D detection, which has already been proved [2], [3], [16], [17] to be the most powerful method for 2D detection. We use CNN to handle multiple viewpoints and rotations for the same class of object together with a single pass through the network, thus reducing the total amount of 2D detection tasks dramatically. Moreover, while the existing strategies usually require an individual detector for each class of object, CNN can be trained with a multi-class output, further saving tremendous processing time when there are multiple objects to detect. To enable multi-class CNN to detect object classes with varied sizes, we unify the training sample sizes with padded boundary so the detector will search for all object classes in a uniform-sized window. On top of these, we further improve the detection efficiency by concatenating two extra levels of early rejection networks with binary outputs, simplified architecture and smaller image sizes, before the final multi-class detection network. Experiments show that our method has competitive overall performance with at least one-order of magnitude speed-up in comparisons with state-of-the-art 3D point cloud object detection methods.

Our main contributions include:

- Introduce CNN to process all viewpoints together in multi-view projection-based 3D object detection.
- Unify detector for multiple object classes with multi-class CNN and uniform-size training samples.
- Increase detection speed by concatenating two early rejection networks with binary outputs, simplified architecture and smaller image sizes.

II. Related Work

A. 3D Object Detection

Existing 3D object detection methods usually require prior segmentation, and they are slow due to 3D complexity. Methods for object recognition in urban street data often require segmenting objects from the ground [10], [12], [13], [15]. A set of object types is then defined to train either a global detector or a set of local descriptors. Golovinskiy et al. [12] extend the targets to over twenty types of street objects, using classifiers trained with global features, while requiring the scene to be pre-processed based on ground estimation, so that candidate objects are segmented before applying recognition algorithms. Pang et al. [14] employ Adaboost to train a combination of weighted 3D Haar-like features for detectors and exhaustively searches for objects in 3D space, thus avoiding the requirement for segmentation. However, this method only handles limited rotation changes. Song et al. [18] use depth maps for object detection with a 3D detector scanning in 3D space, which is similar to the depth-based projections we use, but their method focuses on RGB-D data rather than point clouds, and it is very time-consuming due to the extensive costs for detector training.

B. 3D Descriptor

Local 3D shape descriptors are frequently used by existing methods. Most popular are spin images (SI) [4] which encodes surface properties in a local object-oriented system, as well as others such as 3D shape context [9], fast point feature histogram [25], signature of histograms of orientations [5] and unique shape context [6]. However, 3D descriptor-based recognition methods require prior segmentation of background points, as well as descriptor computation and matching in 3D space, time-consuming processes that make these methods inefficient.

C. CNN for Object Classification and Detection

Since the work of Krizhevsky et al. [3] on ImageNet, convolutional neural network (CNN) [2] has became the most successful method for image classification problem. For object detection, R-CNN [16] is the state-of-the-art on 2D RGB images, while Depth-RCNN [17] expanded the R-CNN algorithm to adapt to RGB-D images. In the 3D domain, 3D ShapeNet [21] represented geometric 3D shape on 3D volumetric grids and applied CNN for classification. For 3D CAD model classification, Su et al. [20] took a view-based

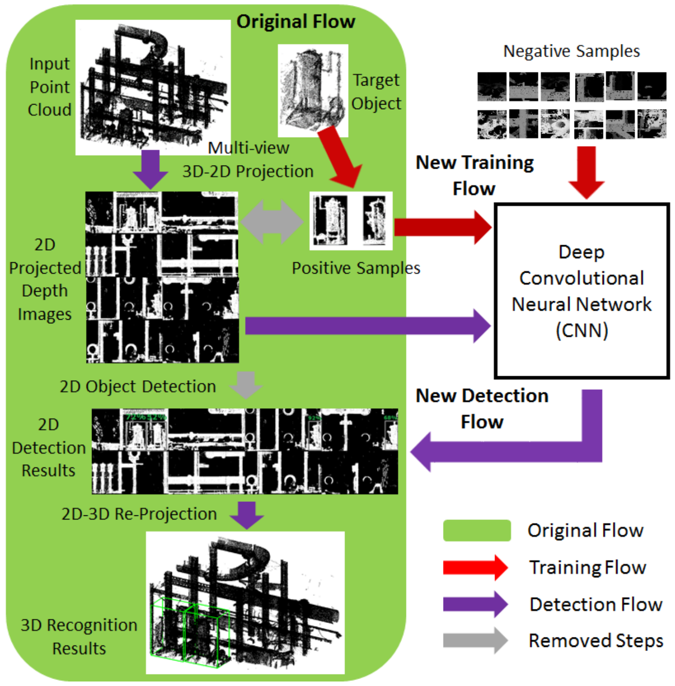


Fig. 2. Comparison of pipelines for the original multi-view 3D point cloud object detection algorithm [1] and our proposed algorithm with concatenated CNNs.

deep learning approach by rendering 3D shapes as 2D images. This method shares some similarities with ours, but 3D mesh model classification and retrieval is a much different problem than 3D object detection in large point clouds. For 3D point cloud, Prokhorov [22] and Habermann et al. [23] explored 3D point cloud object classification with CNN, but only focused on pre-segmented street objects. VoxNet [24] converted 3D point cloud into volumetric data and trained a 3D-CNN to classify them, but the method still focused only on point cloud segment from mostly street data, instead of more complex large-scale industrial point cloud that we're working on.

III. 3D Object Detection with CNN

A. Multi-View 3D Object Detection

To detect 3D object in point cloud, we follow the idea of multi-view projection-based 3D detection method as described in [1], though many details have to be changed to adapt to the introduction of CNN.

The core idea of [1] is to transform a 3D detection problem into a series of 2D detection problem, thereby reducing the complexity of an exhaustive 3D search into a fixed number of 2D searches. As shown in the (green-shaded) algorithm flow in fig. 2, this is achieved by projection of 3D point clouds at multiple viewpoints to decompose it into a series of 2D images. To ensure that the original 3D information is not lost, the 3D to 2D projection is done at multiple viewing angles (evenly chosen on a sphere). Depth information is utilized when projecting 2D images for each view, and kept for later re-projection back into 3D for fusion of 2D results. After the input 3D point cloud is projected into 2D images from multiple views, each view is used to locate the target object. Lastly, all

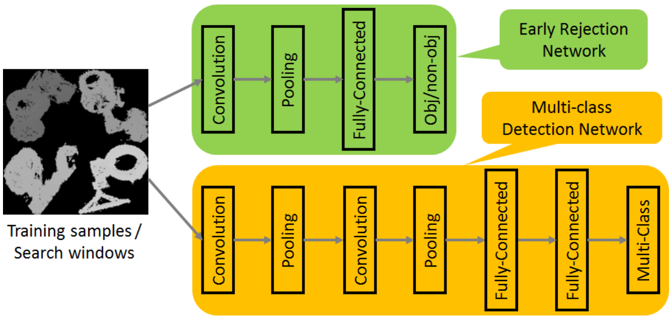


Fig. 3. Two types of CNN network structures we used. Note the different number of classes in output.

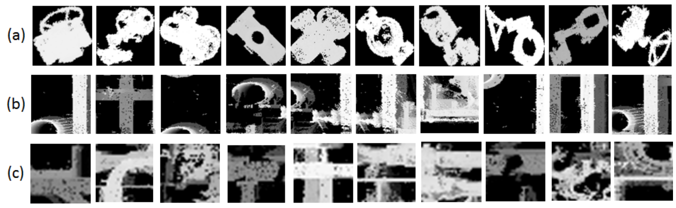


Fig. 4. Some examples of CNN training samples: (a) Positive; (b) "Easy" negative; (c) "Hard" negative.

2D detection results are re-projected back into 3D space for a fused 3D object location estimate.

B. Detect 2D Projections with CNN

In the original multi-view 3D detection method [1], the 3D object will be projected in about 50 viewpoints and searched with 12 in-plane rotations to approximately cover all possible orientation, and each of these view projections and rotations need to be detected individually, resulting in thousands of 2D detection tasks.

To alleviate this complexity, we propose to use convolutional neural network (CNN) for the task of 2D detection. CNN can be trained with all viewpoints and rotations together for one object class. CNN actually synergize with the multi-view projection method very well, because the large amount of projections will produce enough training samples for the CNN to train with. In the training phase, projections in all viewpoints and rotations will be produced and supplied as training samples for CNN so it can learn all possibilities. However, in the detection phase, CNN-based detection can handle all viewpoints and rotations for one object class together, saving tremendous amount of detection tasks. Figure 2 shows the new training and detection flow based on the original flow.

In most existing 3D object detection algorithms [14] [1], each object class has its own detector or classifier, and they have to be applied individually, costing proportionally more time when there're several different object classes to detect. We use CNN-based detection to improve this as well, because CNN can be trained with a multi-class output, capable of detecting all object classes together in one pass.

In implementation, we used two types of network architecture, as shown in fig. 3. One type of network has less layers with a 2-class output for fast object/non-object classification, in order to efficiently reject most non-object negative windows. The other is a more complicated network with multi-class output to decide the specific class of objects, and reject much harder non-objects. The relationship and setup of the two types of networks are explained in sec. IV-A.

C. Training Sample Generation

The positive training samples are generated by projecting a raw 3D point cloud object instance into 2D images at different viewpoints with different in-plane rotations. The projections are performed based on depth as seen from each viewpoint. The 3D space is discretized into cells. Cells with at least one point are considered occupied. The cell size is set so that each projected object has roughly 100-150 pixels in size on average, and fixed for each dataset. Parallel projection rays from each view then sample the scene by extending rays from pixel array. The occupied cell closest to the viewpoint sets the depth value of that pixel, similar to the idea of z-buffering.

CNN generally requires a large amount of training samples to be effective. Therefore, we further expand the size and diversity of our positive samples with more random "jittering". This includes adding random depth shift, small in-plane translation, noise, dilation or erosion on the edges, and synthetic occlusions. In implementation, we projects the objects at 100 viewpoints distributed on a 3D sphere, and rotate each projection in 20 rotations, then generate 5 samples for each rotations with random "jittering". This gives us 10000 positive samples for each raw 3D object instance. Figure 4(a) shows some examples of the positive training samples.

In order to perform detection for multi-view and multi-class together, an uniform search window size must be enforced for all viewpoints and classes. This requires the training samples to all have the same size without distorting the object. Our solution is to find a minimal window size that can enclose all views and classes, then pad empty pixels on image border so that all positive training samples are enlarged to that size. The empty pixels may further be filled during the random "jittering" process.

Negative training samples are generated in point clouds without any object class.The negative point cloud is projected according to depth in the same way at random viewpoints, then cropped at random location of the required size into negative samples, as shown in fig. 4(b). These randomly generated negative samples are usually not very representative. They are used with the positive samples to train CNN, then perform detection on negative projected image, and all false alarms are considered "harder" negative samples, as shown in fig. 4(c), which are fed back into the training set to fine-tune the network. This process will be repeated 2-3 times until the network has been trained with good classification capability.

IV. Concatenated Networks

A. Concatenated CNN for Fast Negative Rejection

Object detection in the projected 2D images is executed as an exhaustive scanning window search. However, due to the nature of the point cloud data, many search windows are actually almost blank or contain mostly primitive shapes. The

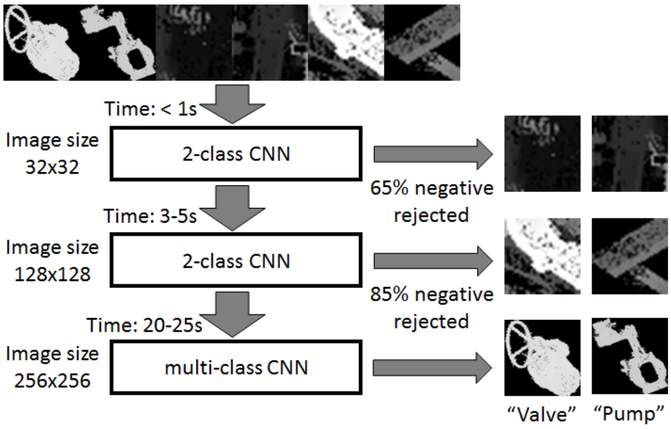


Fig. 5. Concatenate two levels of early rejection network for fast negative window filtering, before the final level of multi-class detection network.

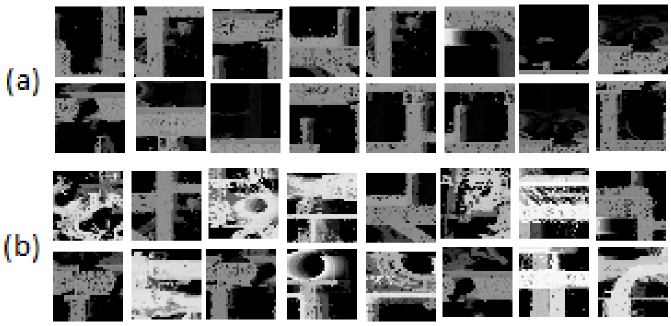


Fig. 6. (a) Negative windows rejected in the first level of early rejection network are mostly simpler backgrounds. (b) Negative windows rejected in the second level of early rejection network are more complicated non-objects.

CNN classifier will perform the same amount of convolutions and spend the same time no matter the complexity of the search window. Therefore, we propose to concatenate multiple CNNs trained with different network architectures and training samples, aiming for different objectives.

We currently use a three-level structure with three networks, as shown in fig. 5. The first two levels use gradually smaller training images (the same size of search window but resized for fast computation), and the network has less layers with an output of only two classes, object or non-object. The final level is a more complicated CNN with more layers and multi-class output, in order to decide the final classification for different object classes or much harder negatives.

The first level of CNN use 32x32 sized images mostly for fast rejection of very simple negative windows. The second level of CNN use 128x128 sized images to deal with slightly harder negative windows. Figure 6 shows some examples of the negative windows rejected by the first two levels of early rejection networks, demonstrating different patterns between them, with the first level filtering out mostly simpler backgrounds, and the second level dealing with more complicated negatives. During training, the class probability threshold is set so that 99 percent of positive samples must pass through the first two CNNs. Experiments show that the first CNN can efficiently reject about 65 percent of total negative windows, while the second CNN can reject roughly half of the remaining ones, which means a total of 85 percent of negative windows are rejected in the first two layers.

TABLE I. Speed Analysis for 3D object detection methods

(Speed in seconds)	Multi-view [1]	Single CNN	Multi-level CNN
Single 2D projection	0.005	5	0.8
Single class 3D	60	120	25
6-class 3D	350	130	28

* Detect objects with about 20k points in a 500k-point scene.

B. Speed Analysis

This section analyze the speed of the original multi-view projection-based method [1], the single-level CNN method, and multi-level concatenated CNN method. Table I provides some numbers for the three methods.

The original multi-view method is much faster in terms of detection speed in single 2D projected image because it utilized binary operations. The CNN-based methods are much slower due to the complexity of neural network, but the early rejection networks can still bring a significant speed improvement.

However, the multi-view method has hundreds of different viewpoints and rotations that all require individual detections, while the CNN-based methods can handle them all together. Therefore, for detection of a single-class object, the multi-view method need about 1 minute, while single-level CNN need about 2 minutes, which is already comparable to multi-view method. With early rejections, it's further reduced to about 25 seconds, faster than multi-view method even for single-class detection. The first two levels of early rejection networks spend about 5 seconds out of 25, to reject roughly 85 percent of negative windows.

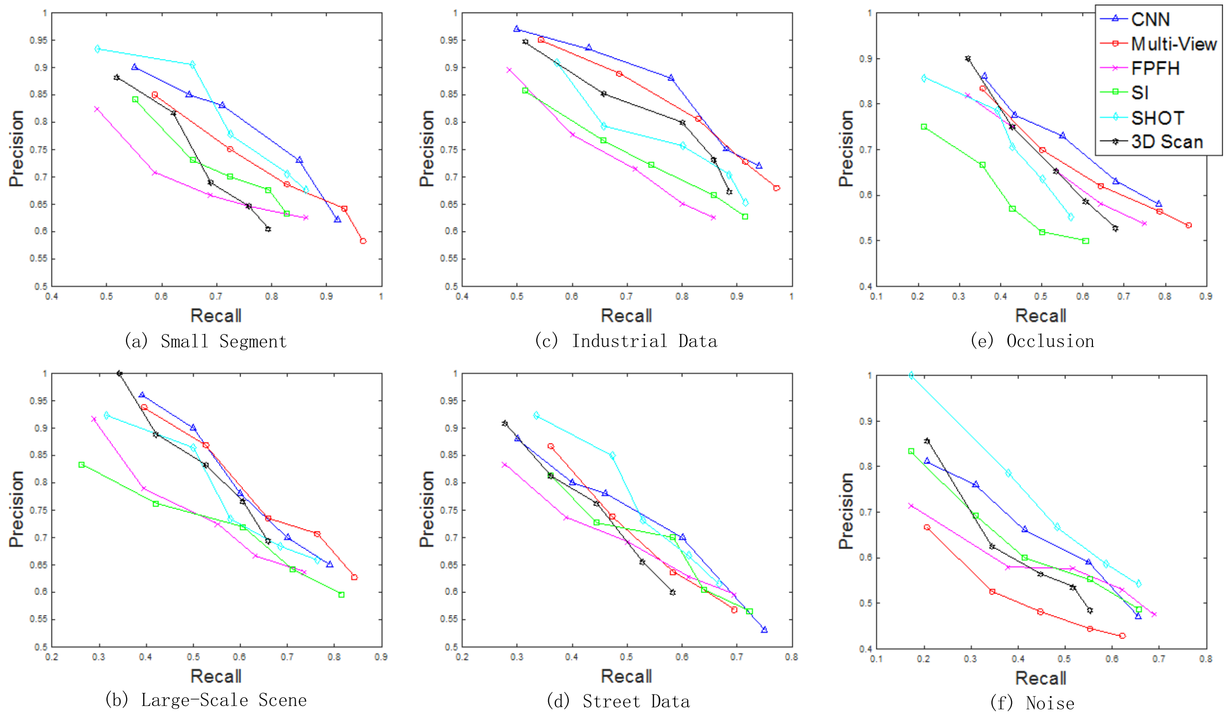
When there are much more classes of objects to detect, the time cost of multi-view method scales up proportionally, since each class requires a individual detector. Single CNN can handle multi-class problem in almost the same time as single-class problem just with a multi-class softmax layer as output. multi-level CNN is even faster, with one-order of magnitude speed-up compared to the original multi-view algorithm.

V. Experiments

A. Experiment Settings

Our evaluation dataset consists three types of data, including single objects, street data and industrial data, similar to the setup in [1]. We incorporate some existing public datasets, such as UWA 3D Object Dataset [8] for single object classification, and CMU Oakland 3-D Point Cloud Dataset [27], Washington Urban Scenes 3D Point Cloud Dataset [13] for street data. For industrial data, there is no public data available (as far as we know) so we use our own data. Original point cloud scans are used for all street and industrial data, but virtual scans are used for object retrieval data [8] with only mesh models.

The dataset contains varied data size and object density to reflect different scale and complexity, including small segments with two or three objects, and large scenes with more than five objects and many background data points. Some scan conditions are also tested, including occlusions with partially



(a) Small Segment (c) Industrial Data (e) Occlusion

(b) Large-Scale Scene (d) Street Data (f) Noise

Fig. 7. Precision-Recall Curves on various test cases, compared to the original multi-view method [1] Spin Images [4], FPFH [25], SHOT [5] and 3D window-scanning [14]. (a) Small segments with two or three object instances and few background points. (b) Large scenes with more than five object instances and many background points. (c) Industrial sites scan. (d) Street level LiDAR. (e) Occluded scene with partially scanned objects. (f) Noisy scene with random noisy points.

scanned objects, and noise from random points or point-shifting with normal distribution.

We compare our algorithm with three state-of-the-art 3D point cloud descriptors, including Spin Images [4], FPFH [25] and SHOT [5]. The PCL 1.6.0 [11] implementations of these descriptors are used, with scene segmentation, feature extraction and matching implemented following [15]. Besides descriptor-based methods, we also compare with the original multi-view projection-method without CNN [1], and a 3D window-scanning method using Adaboost and 3D Haar-like features [14]. The resulting statistics are compared in recall rate and precision curves, and also detection speed.

B. Precision-Recall Evaluation

The first set of experiments compares our algorithm with others on different data sizes. As shown in fig. 7(a)(b), our algorithm performs at about the same level as others, with more advantages on larger scenes. This is because larger data creates increasing numbers of feature points to be matched for descriptor-based methods, especially when there is no good criteria for prior segmentation. Compared to the original multi-view algorithm, our algorithm shows more improvement on smaller segments, thanks to the more stable CNN-based 2D detection.

The second set of experiments compares our algorithm with others on specific types of data, either from industrial sites or urban street LiDAR. As shown in figure 7(c)(d), our algorithm performs much better than others on industrial data, since the generic scene shapes result in lower descriptive power for descriptor-based algorithms. On street data, our algorithm achieves similar levels of detection performance as others. Our algorithm is generally an improvement in both scenarios compared to the original multi-view algorithm.

The final set of experiments compares our algorithm with others under occlusion or noise, both very common in real world scan data. As shown in figure 7(e)(f), our algorithm performs noticeably better than others under occlusion, thanks to the mechanisms of multi-view projections and depth sections. The original multi-view algorithm performs quite badly under noise, as its simple 2D detection algorithm is susceptible to noise. Instead, our CNN is trained with samples containing noise, and thus bringing a significant improvement under noise.

C. Time Efficiency Evaluation

An important goal in the design of our algorithm is fast detection speed while maintaining a good detection rate. Table II lists the detection times of our algorithm, the original multi-view method without CNN [1], the 3D window-scanning method [14] and the descriptor-based methods. The task is to detect 6 classes of objects (20k points each) in a mid-size scene (500k points). The 3D Shape Context (3DSC) [9] and Unique Shape Context (USC) [6] are also included, though they are not included in the precision-recall comparison because they are too slow. All experiments, including the compared descriptors, are executed with the same 2.5GHz Intel Core i7 CPU.

As shown in table II, the speed of our method is at least two-order of magnitudes faster than all the descriptor-based methods, and one-order of magnitudes faster than the

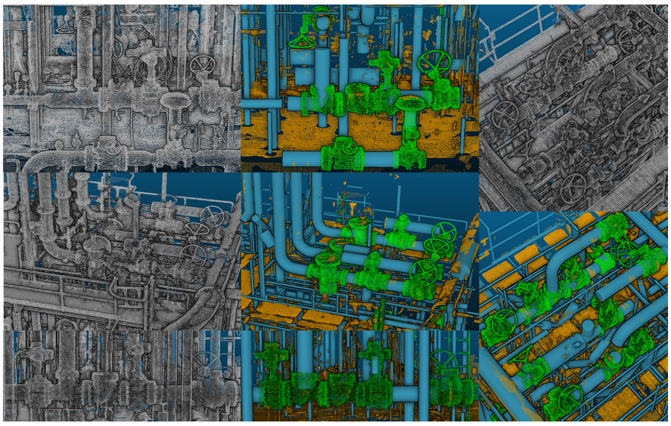


Fig. 8. Example results of our 3D object detection algorithm on large-scale 3D industrial point cloud.

TABLE II. TIME COMPARISON FOR DETECTING 6 OBJECT CLASSES

Time	Multi-level CNN	Multi-View[1]	3D-Scan[14]	FPFH[25]
6-Class 3D	28s	350s	450s	2400s
	SpinImage[4]	SHOT[5]	3DSC[9]	USC[6]
	2100s	2700s	39000s	30000s

others, thanks to the use of CNN and concatenated early rejection networks. This provides our method significant speed advantage in large-scale applications for 3D object detection such as industrial site or urban street data, without sacrificing detection performance. Figure. 8 shows some 3D object detection example results when applying our algorithm on large-scale 3D industrial point cloud [28].

VI. CONCLUSION

In this work, we propose to use convolutional neural network (CNN) for 2D detection, when following the idea to transform 3D object detection problem into a series of 2D detection problems. A trained network can handle all viewpoints and rotations together for the same object class, as well as predicting multiple object classes, without the need for individual detector for each object class, thus reducing the amount of 2D detection tasks dramatically. To make the multi-class CNN able to detect object classes with varied sizes, we unify the training sample sizes with padded boundary so the detector will search for all object classes in a uniform-sized window. In addition, we further improve the detection efficiency by concatenating two extra levels of early rejection networks with binary outputs, simplified architecture and smaller image sizes, before the final multi-class detection network. Experiments show that our method has competitive overall performance with at least one-order of magnitude speed-up in comparisons with latest 3D point cloud detection methods.

REFERENCES

[1] G. Pang and U. Neumann. Fast and Robust Multi-View 3D Object Recognition in Point Clouds. International Conference on 3D Vision (3DV), Oct 2015

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, vol. 86, no. 11, pp. 2278C2324, 1998.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems (NIPS), pp. 1097C1105, 2012.

[4] A. Johnson and M. Hebert. Using Spin Images for Efficient Object Recognition in Cluttered 3D Scenes. PAMI, 21(5): 433-449, 1999.

[5] F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. Proc. 11th European Conf. Computer Vision (ECCV), pp. 356C369, 2010.

[6] F. Tombari, S. Salti, and L. Di Stefano. Unique shape context for 3-D data description. ACM Workshop on 3-D Object Retrieval, pp. 57-62, 2010.

[7] C. Papazov and D. Burschka. An Efficient RANSAC for 3D Object Recognition in Noisy and Occluded Scenes. In Proceedings of the 10th Asian Conference on Computer Vision (ACCV'10), Nov. 2010.

[8] A. Mian, M. Bennamoun and R. Owens. On the Repeatability and Quality of Keypoints for Local Feature-based 3D Object Retrieval from Cluttered Scenes. IJCV, vol. 89(2), 348-361, 2010.

[9] A. Frome, D. Huber, R. Kolluri, T. Bulow, and J. Malik. Recognizing Objects in Range Data Using Regional Point Descriptors. ECCV, 2004.

[10] A Patterson, P Mordohai, K Daniilidis. Object Detection from Large-Scale 3D Datasets using Bottom-up and Top-down Descriptors. ECCV, pp.553-566, 2008.

[11] R. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, May 2011.

[12] A. Golovinskiy, V.G. Kim, T. Funkhouser. Shape-based Recognition of 3D Point Clouds in Urban Environments. ICCV, 2009.

[13] K. Lai and D. Fox. Object Recognition in 3D Point Clouds Using Web Data and Domain Adaptation. IJRR, vol. 29, no. 8, pp. 1019-1037, 2010.

[14] G. Pang and U. Neumann. Training-based Object Recognition in Cluttered 3D Point Clouds. International Conference on 3D Vision (3DV), June 29-30, 2013, Seattle, WA, USA.

[15] J. Huang and S. You. Detecting Objects in Scene Point Cloud: A Combinational Approach. International Conference on 3D Vision (3DV), June 29-30, 2013, Seattle, WA, USA.

[16] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. CVPR, 2014.

[17] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik. Learning rich features from RGB-D images for object detection and segmentation. ECCV, 2014.

[18] S. Song and J. Xiao. Sliding Shapes for 3D Object Detection in Depth Images. ECCV, 2014.

[19] M. Aubry, D. Maturana, A. Efros, B. Russell and J. Sivic. Seeing 3D chairs: exemplar part-based 2D-3D alignment using a large dataset of CAD models. CVPR, pp. 3762-3769, 2014.

[20] H. Su, S. Maji, E. Kalogerakis, E. Learned-Miller. Multi-view Convolutional Neural Networks for 3D Shape Recognition. ICCV, 2015.

[21] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang and J. Xiao. 3D ShapeNets: A Deep Representation for Volumetric Shape Modeling. IEEE Conference on Computer Vision and Pattern Recognition CVPR, 2015.

[22] A Convolutional Learning System for Object Classification in 3-D Lidar Data. D. Prokhorov. IEEE Transactions on Neural Networks, 21 (5), pp. 858 - 863, 2010.

[23] Artificial Neural Nets Object Recognition for 3D Point Clouds. D. Habermann, A. Hata, D. Wolf and F. S. Osorio. Brazilian Conference on Intelligent Systems (BRACIS), 2013.

[24] VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition. D. Maturana and S. Scherer. IEEE/RSJ International Conference on Intelligent Robots and Systems, 2015.

[25] R.B. Rusu, N. Blodow, M. Beetz. Fast Point Feature Histograms (FPFH) for 3D registration. ICRA, pp. 3212-3217, 2009.

[26] D.Y. Chen, X.P. Tian, Y.T. Shen and M. Ouhyoung. On Visual Similarity Based 3D Model Retrieval. Computer Graphics Forum (EUROGRAPHICS'03), Vol. 22, No. 3, pp. 223-232, Sep. 2003.

[27] D. Munoz, J. A. (Drew) Bagnell, N. Vandapel, and M. Hebert. Contextual Classification with Functional Max-Margin Markov Networks. CVPR, June 2009.

[28] G. Pang, R. Qiu, J. Huang, S. You and U. Neumann. Automatic 3D Industrial Point Cloud Modeling and Recognition. The Fourteenth IAPR International Conference on Machine Vision Applications, May, 2015.