Object Recognition in Indoor Scenes Using 3D Shape Contexts

Yanran Guan
Carleton Immersive Media Studio
yanran.guan@carleton.ca

January 31, 2020

Abstract

We introduce a tool that can recognize the specific objects chosen by the user in indoor scenes derived from range scans, based on the regional point descriptor called *3D shape contexts*. We further extend the use of our tool to enable a simple form of 3D instance segmentation.

1 Introduction

3D object recognition is a fundamental research question in 3D computer vision. In this work, we explore the problem of recognizing user-chosen 3D objects in indoor scenes derived from range scans. Given a point cloud representing an indoor scene captured by a range scanner that scans a room, our goal is to recognize the specific objects chosen by the user in the scene, such as *chairs*, by comparing them to reference objects that serve as prototypes. We use a feature-based geometric approach, based on the regional point descriptor called *3D shape contexts* [1], to compute the similarity between reference objects and queried regions in the scene. Our results show that 3D shape contexts can encode the similarity between objects and thus provide a reasonable object recognition. The 3D shape contexts of objects can also be reused for further experiments, such as 3D instance segmentation [2, 3].

2 Background and Related Work

In this section, we discuss the research related to this work, including regional point descriptors and instance segmentation.

2.1 Regional Point Descriptors

Regional point descriptors have been adopted in many 2D and 3D recognition tasks [4]. A regional point descriptor constructs features that characterize the

geometric information of a regional neighborhood surrounding a basis point. Here, we mainly discuss two kinds of descriptors, namely, the shape context descriptor and the spin image descriptor.

2.1.1 Shape Context

The shape context descriptor [5] is a means that calculates the point-wise correspondences between a query shape and a reference shape. The shape context uses a set of points sampled from the contours of the object. Given a shape represented by a point set $\mathcal{P} = \{p_1, p_2, ..., p_n\}$, for each point p_i $(1 \leq i \leq n)$, the shape context considers the set of the $n-1$ vectors that originate from p_i to all other points on the shape as the distribution of relative positions of points that describes the feature of the shape. Thus, for the point p_i, the shape context is defined as the coarse histogram h_i of the relative coordinates of the remaining $n-1$ points, where the k^{th} bin of h_i is written as:

$$h_i(k) = \#\{q \neq p_i : (q - p_i) \in \text{bin}(k)\}. \tag{1}$$

Because the histogram defined in Equation 1 is more sensitive to differences in nearby points, all points on the shapes are converted to log-polar coordinates [5]. With modifications to the shape context, the descriptor with rotation invariance [6] and the extension to 3D shapes [1] are also provided. In this experiment, we use the 3D shape context with rotation invariance to recognize 3D objects.

2.1.2 Spin Image

The spin image descriptor [7] is a regional point descriptor based on surface matching [8]. The idea of the spin image descriptor is to construct 2D images associated with each oriented point on the surface of shapes, where an oriented point refers to the 3D point with surface normals. The points in the 2D images are accumulated into histogram bins, which provide a descriptive image for each oriented point. The similarity measure of shapes is made based on the similarity between the images from the surface points. In our experiment, however, since the point cloud is derived from range scans, we consider the surface normals of the points to be less reliable. Therefore, we base our object recognition method on the shape context descriptor, which only requires coordinates of point positions.

2.2 Instance Segmentation

Recently, there has been an increasing research interest towards fine-grained segmentation of 3D data using deep learning approaches [9], where instance segmentation is seen as an important direction that provides a more comprehensive and detailed understanding of 3D scenes/objects. Instance segmentation refers to the task of providing point-level annotations for each instance given a 3D scene/object, similar to semantic segmentation, which aims to label

points according to its class and can also be associatively employed to enable instance segmentation [10]. Instance segmentation further differentiates between different instances of the same class, including both object instances and part instances. Typically, most 3D instance segmentation methods use a similarity measure for instance differentiation. For example, SGPN by Wang et al. [2], which uses PointNet/PointNet++ [11, 12] as a baseline architecture, designs a double-hinge loss function to learn a pair-wise similarity matrix of points based on the point features learned by the PointNet model. And similarly, the instance segmentation method proposed by Zhang and Wonka [3] uses the probability density function (PDF) to learn a probabilistic spatial embedding that encodes the point-wise similarity within a shape.

3 Experiment

In this section, we introduce our experiment, including the method, experimental data, and results.

3.1 Method

Our object recognition method is based on the 3D shape context. First, we calculate the shape context for the reference shapes, which serves as the reference descriptor. Then, we look for the regions in the query scene that provide a shape context most similar to the reference descriptor, using a brute force search. The similarity between shape contexts is decided by the Euclidean distance. The tunable parameters in our method are summarized in Appendix A.

3.1.1 3D Shape Context

Similar to the shape context in 2D, the support region for a 3D shape context is defined in the polar coordinate system, which can be illustrated as a sphere. The support region is divided into bins by equally spaced boundaries in the azimuth and altitude dimensions and logarithmically spaced boundaries along the radial dimension [1]. An example of the support region is illustrated in Figure 1.

Bin divisions For a support region in the polar coordinate system, the radial dimension is divided into J bins, the azimuth dimension K bins, the altitude dimension L bins. The radial divisions are denoted as $R = \{r_1, r_2, ..., r_J\}$, the azimuth divisions as $\Theta = \{\theta_1, \theta_2, ..., \theta_K\}$, and the altitude divisions as $\Phi = \{\phi_1, \phi_2, ..., \phi_L\}$. With the first division r_1 being the minimum radius $r_{\min}$ and the last division r_J being the maximum radius $r_{\max}$, the radius boundaries r_j, with $1 \leq j \leq J$, is calculated as:

$$r_j = \exp\left\{\ln(r_{\min}) + \frac{j}{J}\ln\left(\frac{r_{\max}}{r_{\min}}\right)\right\}. \tag{2}$$

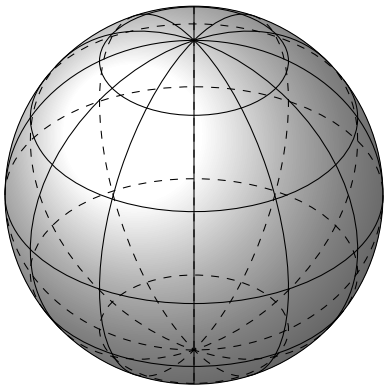


Figure 1: A support region for the shape context descriptor, with the radial dimension undivided, the azimuth dimension equally divided into 12 bins, and the elevation dimension equally divided into 6 bins.

In our experiment, we divide the support region divided into $5 \times 12 \times 6 = 360$ bins, with the radial dimension 5 bins, the azimuth dimension 12 bins, and the altitude dimension 6 bins, and set the extent of the support region with $r_{\min} = 1$ and $r_{\max} = e^4$. We accumulate each bin in two manners: (i) using the count of points that fall inside the bin, as described in Equation 1, and (ii) using the normalized the number of points, i.e., calculating the proportion of the number of points inside the bin to the total number of points.

Rotation invariance Though there are some ready-made methods in 2D, such as measuring angles at each point relative to the direction of its tangent line, we adopt a more straight forward approach to enable rotation invariance for shape contexts. 3D shape contexts can be rotated in two different directions, i.e., azimuthally and altitudinally, while we only consider the azimuthal rotation as important, as no upside-down objects are observed in the scanned scenes. For each reference shape, we rotate it azimuthally for 12 times and compute a shape context for it in each rotation. Thus, no matter how a queried region in the scene is rotated, its shape context is statistically comparable to one of the rotated shape contexts of the reference shape.

3.1.2 Scene Querying

We adopt a brute force search to find the corresponding regions in the query scene that fit the reference objects. Specifically, we use a sliding window, which is a block that has sufficient space to contain the support region of the reference descriptor. We move the block at each step for a small distance along the direction of one axis, to traverse all the consecutive regions in the query scene. For each region captured by the sliding window, we compute the shape context for the points in it and compare it to the reference descriptor.

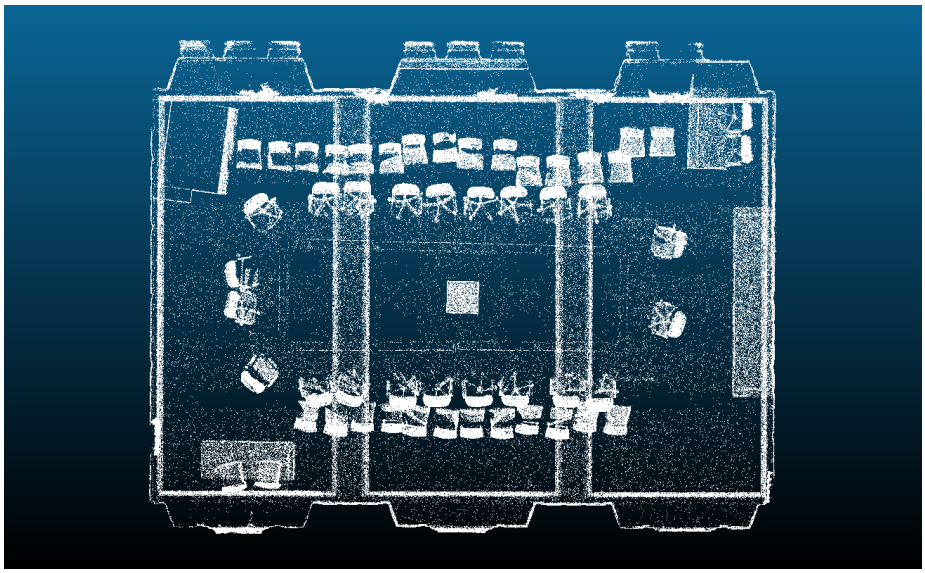


Figure 2: The query scene.

3.1.3 Similarity Measure

We use the Euclidean distance to measure the similarity between the reference descriptor h^* and the shape context h of the queried region. The distance between h^* and h is computed as:

$$d(h^*, h) = \sqrt{\sum_{k=1}^{N} (h^*(k) - h(k))^2}, \tag{3}$$

where k $(1 \leq k \leq N)$ is the index of bins. In our work, we use $N = 360$.

3.2 Experimental Data

We experiment with a query scene having 54 chairs in it, as shown in Figure 2, where the chairs can be roughly divided into two types, i.e., straight leg chairs and rolling leg chairs. From the scene, we choose two example chairs, i.e., one from each type, as the reference objects, as shown in Figure 3(a) and Figure 3(b), respectively.

3.3 Results

We seek to use our method for two applications: (i) object recognition and (ii) a simple form of instance segmentation.

(a) (b)

Figure 3: The two example chairs that we choose as reference objects: (a) a straight leg chair and (b) a rolling leg chair.

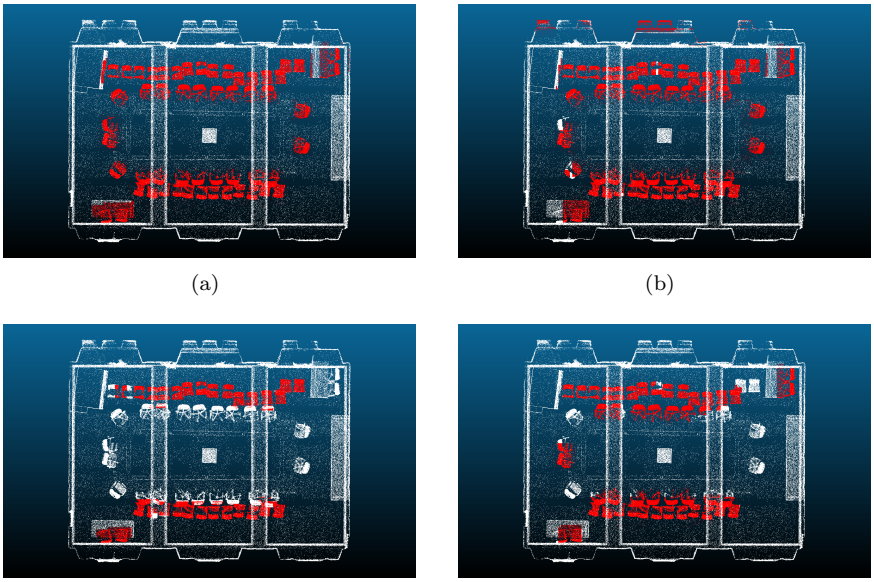
3.3.1 Object Recognition

We first test our method for recognizing all chairs, no matter the type, in the query scene. Assuming that averaging the shape contexts of the example chairs can generalize the features of the two types of chairs, we use the averaged shape context as the reference descriptor. We accumulate the histogram bins of shape contexts in two ways as described in Section 3.1.1 and label the points as chairs in the top 400 best matching blocks that are most similar to the reference descriptor. We show that accumulating bins by count can provide a more reasonable object recognition result, as shown in Figure 4(a), while accumulating bins by proportion may cause points to be mislabeled, as shown in Figure 4(b), where we see some structures with corners, such as some parts of the windows, are mistakenly recognized as chairs, and some chairs are failed to be recognized.

We also test our method for recognizing the two types of chairs separately, using the count of points to accumulate the histogram bins. The results for straight leg chairs and rolling leg chairs are shown respectively in Figure 4(c) and Figure 4(d), where we label the points in the top 200 best matching blocks. We see that mislabeling happens when recognizing rolling leg chairs in Figure 4(d), this may be caused by the missing points in the example chair.

3.3.2 Instance Segmentation

We enable a simple form of instance segmentation with the help of 3D shape contexts, by iteratively finding and removing the best matching region of each query. Specifically, we start with the best match of a query, remove the matching points from the scene, and then repeat this process for the new scene. In this way, we can avoid the matching of overlapping points, which are caused by



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

Figure 4: The results of object recognition in the query scene, with points recognized as chairs marked in red.

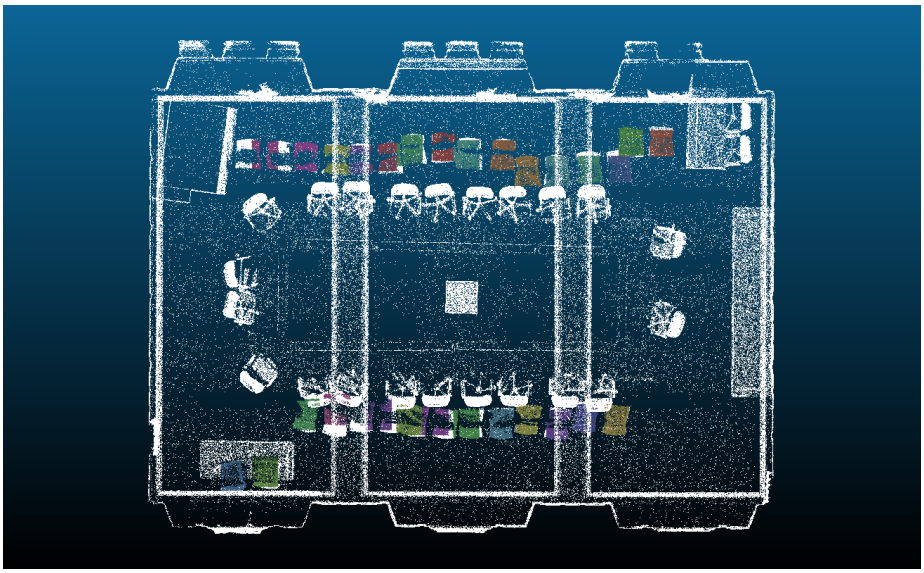


Figure 5: The result of instance segmentation, with each retrieved instance chair marked in a different color.

the sliding window. This method can retrieve the instance chairs from all the straight leg chairs in the query scene, as shown in Figure 5, within 30 iterations.

4 Conclusion

In this experiment, we show the feasibility of recognizing objects in indoor scenes using 3D shape contexts, which can effectively encode the point-wise similarities between shapes. We find that using the count of points to accumulate the histogram bins of shape contexts can provide a more reasonable recognition result than using the proportion of points. We also show the 3D shape contexts of objects can be used to enable a simple form of instance segmentation.

References

[1] A. Frome, D. Huber, R. Kolluri, T. Bülow, and J. Malik, "Recognizing objects in range data using regional point descriptors," in *Proceedings of the European Conference on Computer Vision*, pp. 224–237, 2004.

[2] W. Wang, R. Yu, Q. Huang, and U. Neumann, "SGPN: Similarity group proposal network for 3D point cloud instance segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2569–2578, 2018.

[3] B. Zhang and P. Wonka, "Point cloud instance segmentation using probabilistic embeddings," *CoRR*, vol. abs/1912.00145, 2019.

[4] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, 2005.

[5] S. J. Belongie, J. Malik, and J. Puzicha, "Shape context: A new descriptor for shape matching and object recognition," in *Advances in Neural Information Processing Systems*, pp. 831–837, 2001.

[6] S. J. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–522, 2002.

[7] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3D scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 5, pp. 433–449, 1999.

[8] A. E. Johnson and M. Hebert, "Surface matching for object recognition in complex three-dimensional scenes," *Image and Vision Computing*, vol. 16, no. 9-10, pp. 635–651, 1998.

[9] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su, "PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 909–918, 2019.

[10] X. Wang, S. Liu, X. Shen, C. Shen, and J. Jia, "Associatively segmenting instances and semantics in point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4096–4105, 2019.

[11] 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*, pp. 652–660, 2017.

[12] 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*, pp. 5099–5108, 2017.

A Parameters

In our tool, we allow the user to tweak a few parameters so as to derive different results. The parameters are listed as below:

- `nbins_r`: the number of bins in the radius dimension;
- `nbins_theta`: the number of bins in the azimuth dimension;
- `nbins_phi`: the number of bins in the altitude dimension;
- `rlog_min`: the logarithm of the boundary of the first bin;
- `rlog_max`: the logarithm of the boundary of the last bin;
- `normalize`: the flag for turning on/off the normalization of the number of points when accumulating the bins;
- `rotation`: the flag for turning on/off the rotation invariance;
- `length_x`: the edge length of the sliding window on the x-axis;
- `length_y`: the edge length of the sliding window on the y-axis;
- `length_z`: the edge length of the sliding window on the z-axis;
- `step_x`: the moving distance of one step of the sliding window on the x-axis;
- `step_y`: the moving distance of one step of the sliding window on the y-axis;
- `step_z`: the moving distance of one step of the sliding window on the z-axis.