2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Semantic Labeling of 3D Point Clouds with Object Affordance for Robot Manipulation

David Inkyu Kim Gaurav S. Sukhatme

***Abstract*— When a robot is deployed it needs to understand the nature of its surroundings. In this paper, we address the problem of semantic labeling 3D point clouds by object affordance (*e.g.*, 'pushable', 'liftable'). We propose a technique to extract geometric features from point cloud segments and build a classifier to predict associated object affordances. With the classifier, we have developed an algorithm to enhance object segmentation and reduce manipulation uncertainty by iterative clustering, along with minimizing labeling entropy. Our incremental multiple view merging technique shows improved object segmentation. The novel feature of our approach is the semantic labeling that can be directly applied to manipulation planning. In our experiments with 6 affordance labels, an average of 81.8% accuracy of affordance prediction is achieved. We demonstrate refined object segmentation by applying the classifier to data from the PR2 robot using a Microsoft Kinect in an indoor office environment.**

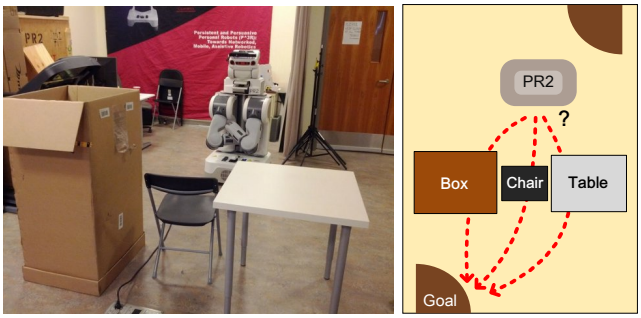


Fig. 1: Example of the PR2 robot situated in front of unknown obstacles. Unless the robot understands each object's affordance, it cannot decide which object to move in order to clear the path.

I. Introduction

Imagine a robot just entered a crowded office room, with tables, chairs, and boxes all around (Fig. 1). The robot wants to travel to the nearest door. Obstacles are blocking the way. When a robot is deployed in such an unknown environment and wants to navigate, a plausible course of action should be to perceive the environment to figure out where, what, and how objects are placed. Unless the robot finds an obstacle-free path (if one exists) it must find a way to move things, reorganize, and create a viable path. This is a complicated problem for a robot because it not only requires segmentation of the environment into objects but associating affordances [3] with objects that describe the ways an object can be manipulated.

Significant research has been done in the area of object recognition, especially in the computer vision community, to solve the problem of identifying objects. But in a situation like above, knowing object affordance, which tells how an object can be moved, is as important as recognizing object identity. With knowledge of whether an object is 'pushable forward' or 'liftable', a robot can plan to achieve desired reconfiguration of objects.

In this paper, we propose and evaluate an algorithm to explore an unknown environment, using semantic labels of object affordance. Specifically, we use acquired 3D point clouds and sort them into segments. Geometric features (*e.g.*shape, location, and geometric relationship) are examined, and machine learning techniques are applied to train a classifier. We compare the outcomes with manually labeled ground truth (81.8% average precision and 81.7% average recall). With the obtained classifier, we propose an iterative clustering algorithm to improve object segmentation and reduce manipulation uncertainty. We also provide evidence that incrementally merging multiple views refines object segmentation.

All authors are with the Robotic Embedded Systems Laboratory, Dept. of Computer Science, University of Southern California, Los Angeles, CA, USA. davidink@usc.edu, gaurav@usc.edu

II. Related work

When a robot needs to navigate among movable obstacles [14], it must understand the configurations and characteristics of objects in order to plan a manipulation task. Given an unknown environment, Wu et al. [16] proposed an navigation algorithm by interacting with the obstacles to learn the characteristics of objects. They used a simple pushing motion primitive to test whether an obstacle is static. Our work focuses on understanding affordances of obstacles, by processing a 3D point cloud to predict the semantic affordances of objects with which a robot has to interact.

There has been rigorous research in the area of scene understanding and object recognition from 2D images. Much of this body of work is focused on finding good features such as SIFT (Scale Invariant Feature Transform) [9], HOG (Histogram Of Gradients) [1], and contextual features [15]. Fritz et al. [2] used SIFT features to directly infer object affordances. Since 2D images are projections of the 3D world, parts of the geometric characteristics are lost during the projection. To overcome this limitation, 3D layouts or sequential multiple views were taken into consideration. Hoiem et al. [6] tried to capture the geometric features by modeling the interdependency of objects, surface orientations, and camera viewpoints. However, it could only be applied with certain

978-1-4799-3684-7/14/$31.00 ©2014 IEEE

relations of objects, and still suffered from lack of geometric representation.

Recently, RGB-D cameras like Microsoft Kinect became popular, which can capture the world in 3D depth images along with colors. Object recognition has been improved by using such RGB-D cameras with both visual and shape information (e.g. [4]). 3D features like NARF [13] were also developed to capture geometric characteristics in 3D depth images. In the area of object classification using 3D point clouds, Shapovalov et al. [12] classified outdoor scenes into ground, building, tree, and low vegetation. Xiong and Huber [17] labeled indoor scene as walls, floors, ceiling, and clutter by using CRF with geometrical relationships such as orthogonal, parallel, adjacent and coplanar. Koppula et al. [7] introduced semantic labels and applied associative relationships between objects to predict labels such as 'chair back rest' and 'chair base'.

Our research differs from previous works as we solve the object classification problem by labeling object affordance. The advantage of affordance labeling is that labels can directly be adopted for manipulation planning and it does not a require a prior database of objects identities. Also, the predicted labels can be utilized as new features to refine object segmentation and reduce affordance uncertainty.

III. Classifier for Semantic Labeling

We develop a classifier that predicts the object affordance for a given 3D point cloud. We perform the classification in three steps:

1) Segmentation of the 3D point cloud
2) Extraction of geometric features
3) Training and learning parameters of the classifier

A. Segmentation method

There are various segmentation methods such as planar, cylindrical, and Euclidean clustering [11]. Since our proposed algorithm aims to label object affordance, the segmentation method does not need to precisely extract the outlines of objects. Therefore, we apply the region growing method from the Point Cloud Library (PCL) [11], which captures continuous surfaces of a 3D point cloud. The method starts with randomly selected seed points and grows them by grouping neighboring points of similar normal angle and curvature. As a result, we can sort point cloud clusters, shown in (Fig. 2) with the 2D image from Kinect and the corresponding segmented 3D point cloud.

B. Geometric features

Among many features such as color, texture, and material, the geometric properties of objects play a crucial role when considering object affordance. For example, the PR2 robot can push a box forward as long as it is large and tall enough to be reachable by the arms of the PR2. Additionally, the surface of the box should be facing the PR2. To be 'graspable', an object must have a thin part that fits within the gripper of the PR2.

Fig. 2: Example of segmentation: 2D image captured from Kinect (left) and result of segmented 3D point cloud with randomly colored segments (right).

TABLE I: The list of geometric features

Features	Description
Scatter	How round a segment is
Linearity	How long and thin a segment is
Planarity	How planar a segment is
Normal	Average normal of a segment
Span	How large a segment is and how far it spans in XYZ
Centroid	XYZ coordinates of centroid of a segment
Occupancy	Relative locations of neighboring segments

To determine the affordance of objects, we consider two types of geometric features. First, we derive unary features from a single point cloud segment in terms of the shape, normal, and location of the segment. Second, we consider pairwise features that capture the geometric relation between neighboring segments. Pairwise geometric features are especially useful in explaining object affordance by capturing the relative position of segments. Suppose there is a table in front of a wall and the table is not pushable toward the wall. To classify several types of object affordance, we use the unary and pairwise features shown in Table I. In the following, we describe the affordance information contained within each feature, and provide details about how to compute feature values.

- Saliency features of point cloud describe the shape of a point cloud cluster. It is computed with the method inspired by tensor voting [8, 10]. A symmetric positive definite covariance matrix for a set of N 3D points $X_i = (x_i, y_i, z_i)^T$ is defined as

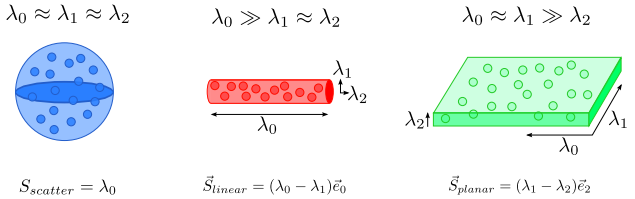


Fig. 3: Scatter, linearity, and planarity of point cloud, adapted from [8]. Each saliency feature can be computed with eigenvalues and eigenvectors of the covariance matrix

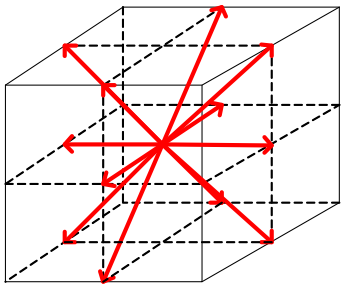


Fig. 4: Occupancy binning of neighboring segments. Each vector between centroids of segment pairs is binned into one of these 12 directions (3 layers of planes, each with 4 directions).

$$\frac{1}{N}\sum_{i=1}^{N}(X_i - \bar{X})(X_i - \bar{X})^T$$

where $\bar{X}$ is the centroid of the segment. From the covariance matrix, we can get three eigenvalues $\lambda_0, \lambda_1, \lambda_2$ in a descending order and three corresponding eigenvectors e_0, e_1, e_2. $\lambda_0, \lambda_0 - \lambda_1, \lambda_1 - \lambda_2$ from Fig. 3 represent scatter, linearity, and planarity features of 3D point cloud segment. These saliency features explain object affordances, for example, if a segment has a high linearity value, it is likely to be graspable.

- Normal is derived from eigenvector of covariance matrix. The normal represents which direction the segment is facing. Therefore it is a distinct geometric feature, indicating the direction in which the segment can be manipulated.
- Span implies how far each segment reaches in the XYZ coordinate. This gives an idea of how large the point cloud is and where it is located. For example, a wall will have long z-axis span from the floor to the ceiling, which would likely be static and not pushable.
- Occupancy is the relative pairwise geometric feature between segment neighbors. To express the relative position, we first compute the nearest neighbors of each segment by computing shortest distances between segments and thresholding them. Then we bin the direction of the vector between two centroids of segment pairs into one of 12 bins (3 split in z axis, each with 4 directions in xy plane, shown in Fig. 4). The shortest distances between two segments are scaled and normalized to express the measure of closeness.

C. Training with logistic regression

Affordances describe what a robot can do with the object. For example, a mobile robot could perform simple motions like pushing an object, while a humanoid robot can perform complicated multi-step motions like folding a chair or opening a door by turning the knob. In our work, we define 6 semantic labels with simplified motion primitives: pushable forward, pushable backward, pushable left, pushable right, liftable, and graspable. Each label except 'graspable' is a single step, omnidirectional motion with discretized orientations (forward, backward, left, right, up). All directions are relative to the current orientation and position of the robot. In order to build ground truth samples for training and evaluation, we define these criteria to manually label point cloud segments:

Fig. 5: Binary ground truth label of 'pushable forward'. The left table in front of the box and the chair in front of the right table are 'not pushable forward' as there are objects behind them (red: pushable forward, blue: not pushable forward).

- All segments belonging to a single object share the same labels.
- A segment is not pushable in a given direction if neighboring segments exist in that direction (*e.g.*a table in front of a box is not pushable forward, a table with an object on is not liftable).
- A segment is graspable if it has a thin and narrow part that the PR2 can grasp with its gripper.
- Labels represent physically feasible primitives (*e.g.*an object on a table is pushable in all directions, even though it might fall off).

For training a 3D point cloud classifier, Xiong and Huber [17] used a CRF model to label indoor point cloud with geometric labels and Koppula et al. [7] used a MRF model with log-linear model and pairwise edge potentials for semantic labeling. The shortcomings of those techniques are their computation time: in [7], it took average 18 minutes per scene to predict the labels with a mixed-integer-solver. To speed up, we use logistic regression which is simple and fast. Logistic regression predicts the probability of a segment having a label l by computing

$$p^l(x) = \frac{e^{f^l(\mathbf{x})}}{e^{f^l(\mathbf{x})}+1}, \quad f^l(\mathbf{x}) = \beta_0^l + \sum_{i=1}^{n}\beta_i^l x_i$$

where $\mathbf{x} = \{x_1, x_2, ...x_i\}$ and x_i stand for the ith feature value of the segment. To get parameters $\beta^l = \{\beta_0^l, \beta_1^l, ..., \beta_i^l\}$, we manually labeled point cloud segments with binary classification and trained them by logistic regression. Fig. 5 shows an example of labeled ground truth data.

IV. Iterative K-means clustering for entropy minimization

Now we have the semantic classifier of object affordances which predicts the characteristics of segments. But if a robot is to manipulate objects to change their locations or orientations, it is important to recognize an object as an

entity. This is because when a robot interacts with a (rigid) object, all segments belonging that object move together. As the region growing segmentation method usually identifies parts of an object, segments should be stitched together to represent a single object. To find such segments, the predicted labels can be used as feature vectors. As every segment from a single object shares same characteristics, those segments would have similar labels. We can thus refine object segmentation by grouping segments with similar label vectors.

For manipulation planning, it is also crucial to identify a segment with the highest affordance certainty in order to minimize manipulation failure. As each label represents the predicted probability, the uncertainty in affordance can be expressed by the entropy of label vectors:

$$H(X) = E[-ln(P(X)] = -\sum_{l=1}^{n} p^l(x) ln\, p^l(x)$$

where $p^l(x)$ is the probability obtained from the classifier of the label l. Higher accuracy of prediction (either close to 0 or 1) will have a lower entropy value. We can find the segment of most certain affordance by searching for the lowest entropy.

Algorithm 1 sorts segments into groups with two objectives: 1) finding similar label vectors to isolate single object, 2) finding low entropy segments to reduce manipulation uncertainty.

Algorithm 1 Iterative k-means clustering for entropy minimization

1: **for** Segment Set $S = \{S_1, S_2, ..., S_n\}$ **do**
2: Split S into two subsets S^1, S^2 by 2-means clustering
3: Start with two random point vectors m_1, m_2
4: **while** $\frac{1}{N_j} \sum_{S_i \in S^j} ||S_i - m_j||^2 > d_{th}$ **do**
5: **if** $||S_i - m_1||^2 < ||S_i - m_2||^2$ **then**
6: $S_i \subset S^1$
7: **else**
8: $S_i \subset S^2$
9: **end if**
10: **Update** $m_j = \frac{1}{N_j} \sum_{S_i \in S^j} S_i$
11: **end while**
12: **if** $H(S) > (H(S^j))$ **then**
13: Goto 4 and split iteratively
14: **else**
15: Group segments in the subset as an object
16: **end if**
17: **end for**
18: **return** Groups of segments S^j

The algorithm starts from all segments in a single set and tries to separate a group of segments with similarity. One iteration is shown in Fig. 6. As the similarity between segments can be expressed by the Euclidean distance between label vectors, k-means clustering algorithm can be applied to group segments. The problem with k-means clustering is

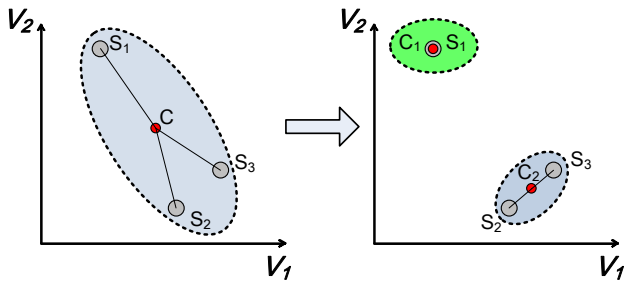


Fig. 6: Example step of iterative k-means clustering, with two label vectors V_1, V_2. Set $S = \{S_1, S_2, S_3\}$ is split into two subsets by k-means clustering and the entropies are computed from the centroids of the sets. If $H(C) > H(C_1)$ or $H(C_2)$, subsets are iteratively split again.

that the exact number of objects is unknown and Finding the optimal k of k-means clustering is NP-hard. Thus, we set k=2 and run clustering iteratively to pick out groups of similar segments at each iteration. 2-means clustering starts with two random points and splits the segment set S into two subsets by comparing Euclidean distances of label vectors from seed points. The seed points are updated by averaging the label vectors of the subset and the process is repeated until convergence. We also consider the entropy change in each split, where the entropy of a set is computed from the centroid of the set. If splitting lowers the entropy, it means the subset has higher certainty of object affordance. Therefore, segments with lower uncertainty can be searched through iterations. The result of the algorithm is a tree structure with increasing depth for each iteration. Object segmentation is obtained at each depth of the tree and the entropy is minimized at the leaves of the tree.

V. Incremental Object Segmentation Refining by Merging Multiple Views

Object segmentation can also be enhanced by merging predicted segments from multiple views. In each scene, the labels of a single object should be consistent no matter from which viewpoint they were seen. Also, those segments should be close to each other (sometimes they might even overlap).

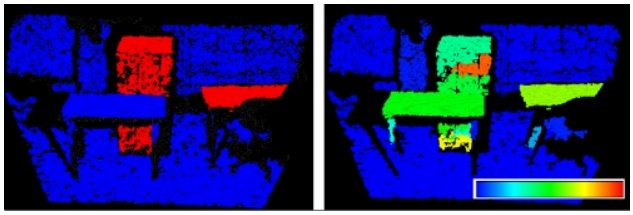
We propose an algorithm to find groups of segments having both proximity and similarity. We run the classifier on individual scenes and merge all the predicted segments into the global frame. Euclidean distances in Cartesian space and label vector space are computed among segments for proximity and similarity respectively. The process incrementally refines object segmetation as merging additional views improves prediction consensus.

VI. Experimental Results

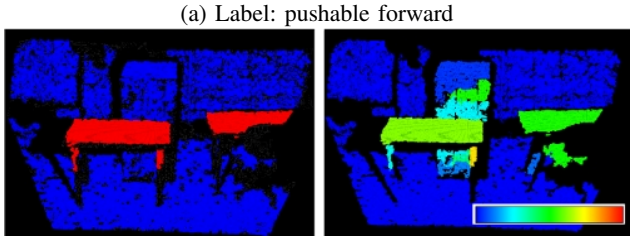
In our experiment, the PR2 robot from Willow Garage was used. With its two 7-DOF gripper equipped arms and a mobile base, the PR2 can perform simple tasks like pushing, grasping, and lifting (upto 1.8kg payload). All 3D point clouds were captured by a Microsoft Kinect mounted on

TABLE II: Result of logistic regression for semantic labels

Label	Precision	Recall
Pushable Forward	82.0	81.0
Pushable Backward	82.4	81.9
Pushable Left	82.4	81.9
Pushable Right	78.0	78.5
Liftable	78.9	80.4
Graspable	86.8	86.7
Average	**81.8**	**81.7**



(a) Label: pushable forward



(b) Label: liftable

Fig. 7: Result of predicted classification. The ground truth is shown (left) with binary classification (red: possible, blue: not possible) and the probabilities of prediction are shown in a heatmap (right).

the head of the PR2 and algorithms were implemented in Robot Operating System (ROS). Since the capability of the robot strongly affects object affordances, we assume all experimental settings to be specific to PR2 in our work.

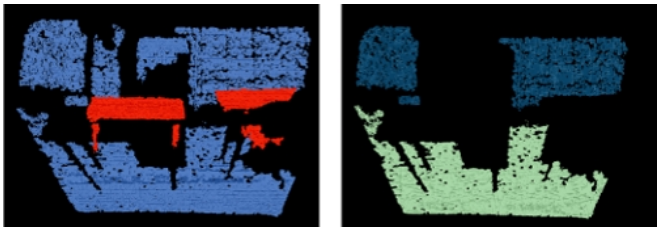
A. Logistic regression for semantic classification

In our experiment, the captured 3D point cloud was transformed into the PR2 frame, with the base link of PR2 at the origin. In the PR2 frame, the z axis corresponds to up and the x axis corresponds to the frontal direction of the PR2. We imaged 10 office scenes with different configurations of tables, chairs, and boxes. Each scene was segmented by the region growing method, resulting in 195 total segments. We manually labeled each segment with 6 semantic labels and logistic regression (using WEKA [5]) to learn the parameters $\beta_0^l, ..., \beta_n^l$ for each label l=1,...,6, with number of features n=25. 4-fold cross-validation was used to evaluate the classifier. The result is shown in Table II with average of 81.8% precision and 81.7% recall.

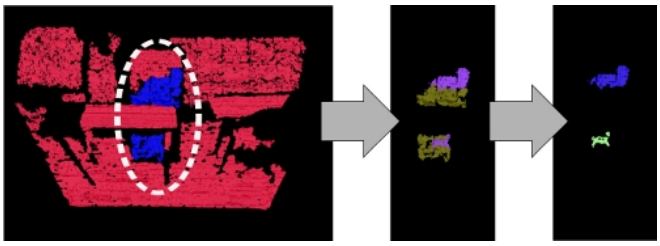
Fig. 7 shows examples of the classifier prediction, with hand labeled ground truth (left) and the probabilities of the predicted label (right).

B. Iterative k-means clustering with entropy minimization

Fig. 8 shows the result of iterative k-means clustering for entropy minimization. In each step, we can see the segments



(a) Refinement of object segmentation.



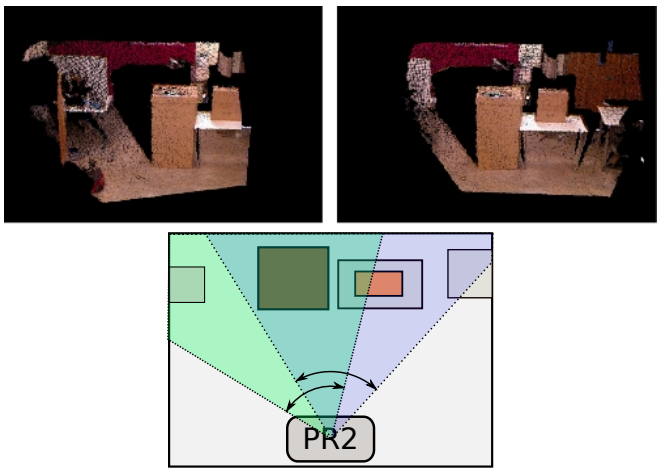
(b) Entropy minimization by each iteration

Fig. 8: Results of iterative k-means clustering. (a) object segmentation is refined as two tables and a chair are separated from other segments (left) and a wall and a floor are separated (right). (b) At every iteration split, the entropies are lowered. The last segments have the lowest uncertainties among segments of a box.

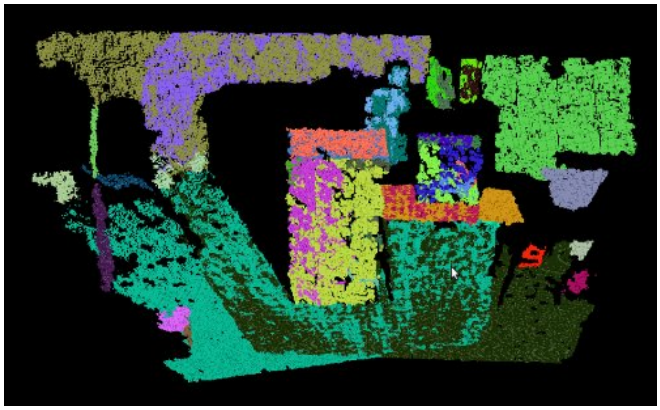
split into two subgroups and object segmentation gets refined at each step. In Fig. 8(a), two tables and a chair have been separated from the environment first, then a wall and a floor were divided in other iteration. The two tables were not split as entropy did not reduce by split. In this kind of example, further interactive manipulation should be performed in order to get better segmentation results. Also we can increase certainty of segment label (Fig. 8(b)). Among segments belonging to a box, entropy gets lower at every iteration and the final two segments show the least uncertainty of prediction. As the goal for iterative clustering is not solely concentrated on object segmentation, some of the result could not distinguish segments belonging to different objects. But still, by reducing entropy, a robot can have better prior knowledge of affordance for future manipulation.

C. Incremental object segmentation refining by merging multiple views

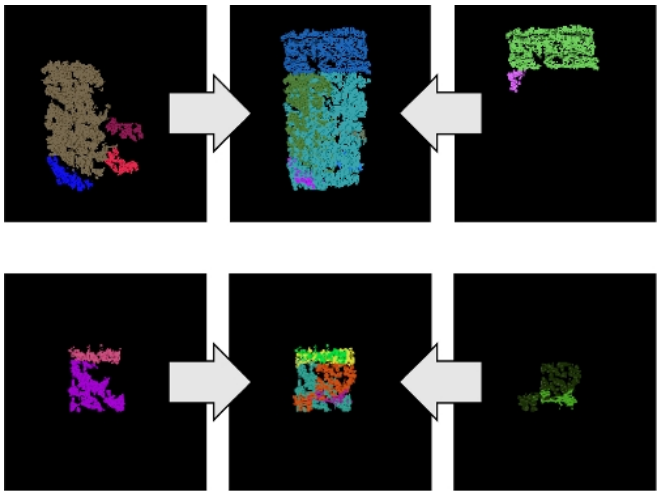
An example of incremental classification refinement is shown in Fig. 9. Since our directions of object affordances (e.g 'pushable forward') are relative to the position and orientation of the PR2 robot, multiple views were collected by rotating the head-mounted sensor with fixed body position. Individually classified segments were merged into the global frame (Fig. 9(b)) and we filtered XYZ Euclidean distances between segments with threshold of 0.05m for a proximity measure. We filtered similarity to be within a threshold of 0.7 (relatively, 11.6% of difference). As shown in Fig. 9(c), the segmentation of two boxes in the merged frame improved, compared to the individual segmentations.



(a) Point clouds taken from two different views of a scene



(b) Merged Point clouds with randomly colored segments



(c) Examples of object segmentation refinement

Fig. 9: Incremental classification with multiple views. (a) two views obtained by rotating the head-mounted Kinect on the PR2. (b) segments merged into the global frame. (c) segments representing a big box (top) and a small box (bottom) are shown from left view (left), right view (right), and merged cloud (middle).

VII. Conclusions and Future Work

In this paper, we presented a semantic 3D point cloud classifier of object affordance. The point cloud was divided into segments and classified by logistic regression. Object segmentation was refined with iterative k-means clustering and incrementally merging multiple point clouds. As a result, a robot can plan how to interact with objects. In the next step, we plan to improve the accuracy of the classifier and apply it to manipulation task planning.

- Since data from the Kinect are 3D RGB point clouds, we can also consider RGB color values of the point cloud as features. Even though most affordance characteristics are represented by geometric features, colors can also imply affordances (especially in human environments *e.g.*materials, textures).
- The affordances we captured for PR2 cannot be generalized for all other robots. Robot specification should also be considered to estimate the affordances for a certain robot platform. Further interaction with the environment will be necessary to confirm and improve the classification parameters.
- For each object, affordances can vary according to the conditions like relative positions. In our work, we only captured instantaneous affordances, but we can expend the work to distinguish between generic and instantaneous affordance to accomplish better manipulation planning.
- Other than just merging views taken from a single point of view, a robot can wander around the environment to collect and predict various segments' affordances. As these affordances should be consistent, they can be served as features, which can be applied for object matching, SLAM, etc.
- Navigation, exploration, and planning can be based on the outcomes of our algorithms. Interaction with the environment can provide feedback to improve the classifier suggesting avenues for future manipulation-aided perception

Acknowledgment

This work was supported in part by the US National Science Foundation Robust Intelligence Program (grant IIS-1017134), and by the Defense Advanced Projects Research Agency under the Autonomous Robot Manipulation program (contract W91CRBG-10-C-0136).

References

[1] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition (CVPR)*, pages 886–893, 2005.

[2] G. Fritz, L. Paletta, R Breithaupt, and Rome E. Learning predictive features in affordance based robotic perception systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3642–3647, 2006.

[3] J.J. Gibson. The concept of affordances. *Perceiving, acting, and knowing*, pages 67–82, 1977.

[4] S. Gould, P. Baumstarck, M. Quigley, A.Y. Ng, and D. Koller. Integrating visual and range data for robotic object detection. In *Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications*, 2008.

[5] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P Reutemann, and I.H. Witten. The weka data mining software: An update. *SIGKDD Explorations*, 11(1), 2009.

[6] D. Hoiem, A.A. Efros, and M. Hebert. Putting objects in perspective. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2137–2144, 2006.

[7] H. Koppula, A. Anand, T. Joachims, and A. Saxena. Semantic labeling of 3D point clouds for indoor scenes. In *Advances in Neural Information Processing Systems*, pages 244–252, 2011.

[8] J. Lalonde, N. Vandapel, D. Huber, and M. Hebert. Natural terrain classification using three-dimensional ladar data for ground robot mobility. *Journal of Field Robotics*, 23:839–861, 2006.

[9] D.G. Lowe. Object recognition from local scale-invariant features. In *Proc. of the Int. Conf. on Computer Vision 2*, pages 1150–1157, 1999.

[10] G. Medioni, M. Lee, and C. Tang. *A Computational Framework for Segmentation and Grouping*. Elsivier, 2000.

[11] R.B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–4, 2011.

[12] R Shapovalov, A. Velizhev, and O. Barinova. Non-associative markov networks for 3D point cloud classification. In *Photogrammetric Computer Vision and Image Analysis*, 2010.

[13] B. Steder, R. B. Rusu, K. Konolige, and W. Burgard. NARF: 3D range image features for object recognition. In *Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, volume 44, 2010.

[14] M. Stilman and J.J. Kuffner. Navigation among movable obstacles: Real-time reasoning in complex environments. *International Journal of Humanoid Robotics*, 2 (04):479–503, 2005.

[15] A. Torralba. Contextual priming for object detection. *International Journal of Computer Vision (IJCV)*, 53: 2003, 2003.

[16] H.N. Wu, M. Levihn, and M. Stilman. Navigation among movable obstacles in unknown environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1433–1438, 2010.

[17] X. Xiong and D. Huber. Using context to create semantic 3D models of indoor environments. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2010.