Furniture Classification using WWW CAD Models

Vladyslav Usenko, Florian Seidel,
Zoltan-Csaba Marton
Intelligent Autonomous System
TU Muenchen, Germany
vsu1991@gmail.com,
seidel.florian@googlemail.com,
marton@cs.tum.edu

Dejan Pangercic
Robert Bosch LLC
Palo Alto, CA
dejan.pangercic@gmail.com

Michael Beetz
Intelligent Autonomous Systems/Artificial Intelligence
Department of Computer Science and
Centre for Computing Technologies (TZI)
University of Bremen, Germany
beetz@informatik.uni-bremen.de

***Abstract*— In this paper, we revisit the approach by Mozos et al. [1] to address the problem of exploiting the structure in today's workplace interiors in order for service robots to add semantics to their sensor readings and to build semi-static models of their environment by learning generic descriptors from online object databases. These world models include information about the location, the shape and the pose of furniture elements like chairs, armchairs, tables and sideboards, which allow robots to perform their tasks more flexibly and efficiently.**

To recognize the different objects in real environments, where high clutter and occlusions are common, the method automatically learns a vocabulary of object parts from CAD models downloaded from the Web. After a segmentation and a probabilistic Hough voting step, likely object locations and a list of its assumed parts can be obtained without full visibility and without any prior about their locations. These detections are then verified by finding the best fitting object model, filtering out false positives and enabling interaction with the objects.

In this paper we alter the original method in that we use a low-cost and low-accuracy Kinect sensor, propose an alternative clustering algorithm based on the Regularized Information Maximization (RIM) [2] and reduce the pose fitting computation time by introducing part subset classification using Viewpoint Feature Histograms (VFH) [3] before it. The implementation of the system is available as open-source in ROS [1].

I. Introduction

We expect the future World Wide Web to include a shared web for robots, in which they can retrieve data and information needed for accomplishing their tasks. Among many other information, this web will contain models of robots' environments and the objects therein. Today's web already contains such 3D object models on websites such as Google 3D Warehouse or catalogs of online furniture stores.

The key idea advocated here is that autonomous mapping of human living environments should utilize the CAD (Computer Aided Design) models from furniture Web catalogs. In the approach presented by Mozos et al. [1] these models are first converted using a realistic simulation of a laser range finder scan. These point clouds are then over-segmented and the resulting parts from the different training objects are clustered to create a vocabulary of parts. In the classification step, similar parts are matched and probabilistic Hough voting [4] is applied to get initial estimates about the location and categories of objects found in the scene. Finally, CAD models are fitted to the results of the classification in order to a) obtain pose hypotheses and to b) reject the false positive detections.

In this paper, one hand we provide an open-source implementation of the algorithm by Mozos et al. [1] and on the other hand adapt their implementation to work with a low-cost low-accuracy RGBD sensor, integrate a substaintally better clustering algorithm and propose a speed up for the pose fitting step by introducing an additional part subset classification step.

The final goal of this work is to recognize so-called semi-static objects such as chairs and integrate them into the Semantic Object Maps, which are the information resource structures derived in our earlier work [5]. We represent Semantic Object Maps as symbolic knowledge bases that contain facts about objects in the environment (obtained from WWW sources such as common sense databases) and that link objects to data structures such as point clouds which then allows e.g. for the recognition of the objects. This combination enables the robot to apply this knowledge to reason about objects in the map, for example to infer that the chair might be in the way when setting the table for breakfast. While currently our implementation of the Semantic Object Maps captures static objects, the inclusion of semi-static objects such as the ones dealt with in this paper requires their detection and pose estimation.

In the rest of the paper we report about the related approaches (Sec. II), briefly recapitulate the algorithm of Mozos et al. (Sec. III, Sec. IV), describe the novelties introduced in this paper (Sec. III-C.2, Sec. IV-E), present the results (Sec. V) and finally conclude and give the pointers to the future work (Sec. VI).

II. Related Work

Part-based object classification in 3D point clouds has also been addressed by Huber *et al.* [6], using point clouds partitioned by hand. In contrast, we partition the objects in an unsupervised manner. Ruiz-Correa *et al.* [7] introduce an

[1] www.ros.org/wiki/furniture_classification

abstract representation of shape classes that encode the spatial relationships between object parts. The method applies point signatures, whereas we use descriptors for complete segments. Tombari et al. [8] use Hough Voting for the recognition and pose estimation of free-form objects. Their method differs from ours in that they use the 3D features (keypoint detector plus descriptor) as opposed to the patches we use and therefore introduce the notion of the additional local reference frame since their features are not translation and rotation invariant. Further, they use exactly the same models in the training and testing phase.

Many of the techniques in our approach come from the vision community. The creation of a vocabulary is based on the work by Agarwal and Roth [9], and its extension with a probabilistic Hough voting approach is taken from Leibe *et al.* [4]. Voting is also used by Sun *et al.* [10] to detect objects by relating image patches to depth information. Basing our approach on geometrical information allows us to have a single 3D CAD model of an example object in the WWW database, since the different views can be generated by the robot. Combinations of 3D and 2D features for part-based detection would definitely improve the results [11].

For matching problems, RANSAC and its variations are widely used due to the flexibility and robustness of the algorithm [12], [13]. To register different views of an object, local tensors are applied by Mian *et al.* [14]. Moreover, Rusu *et al.* [15] limit the point correspondences by using local features.

Finally, using synthetic data for training data is an idea that appears in several works [16], [17]. Lai and Fox [18] combine scans from real objects with models from Google 3D Warehouse to create an initial training set. In our approach, we solely base our classifier on synthetic models, and use those for getting object poses and to verify detection.

III. Training

The training part of the system consists of the following steps: i) virtual scanning of the CAD models and oversegmentation of the generated point clouds into segments S, ii) computation of the Simple Geometric Features (SGF), iii) clustering using k-means and RIM, and learning of the shape model.

A. Virtual Scanning and Segmentation

We download CAD models from the Google 3D Warehouse [19], Vitra's Furnish.net database for office furniture [20], and EasternGraphics' web catalogs [21]. To obtain realistic point cloud representations for these objects, we simulate the Kinect sensor behaviour and perform ray-casting to interesect each simulated beam with the CAD model of the object. Further we also add realistic noise to the depth measurements. The example CAD model and the corresponding point cloud are depicted in the Fig. 1.

In the next step clustering of the synthetically generated point clouds is performed. Our segmentation method follows a criterion based on a maximum angle difference between the surface normals. For each point, we thus calculate its normal

Fig. 1: Complete mesh model of a chair and a partial view point cloud generated from it.

by robustly identifying a tangent plane at the selected point and approximating the point's neighborhood (inside a radius of 3cm) using a height function relative to this plane, in the form of a 2^{nd} order bi-variate polynomial defined in a local coordinate system [22]. Using the obtained normals for each point in the cloud, we apply a region growing algorithm where we mark a point p as belonging to a part S_i if the distance between the point p and some point in S_i is closer than $\delta = 5$ cm, and if the angle formed by the normal of p and the seed normal of S_i is less than $\alpha = 40°$. Seed points are iteratively selected as points with the lowest curvature that do not belong to any part yet. The result of this step is depicted in Fig. 2.

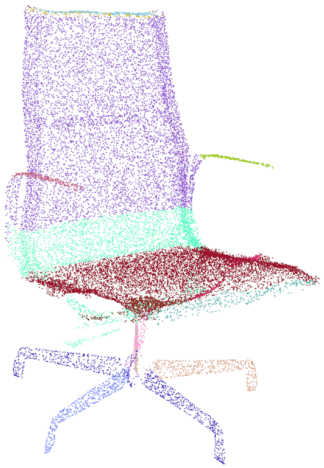


Fig. 2: Segments $S_1 \ldots S_n$ of a point cloud of a chair.

B. Simple Geometric Features

We use exactly the same features as in the original paper [1] which we list here for the sake of self-containedness of the paper. The features are size variant but translation and rotation invariant.

1) Proportion of boundary points in S_i computed as in [23].
2) Average curvature of S_i computed as the smallest eigenvalue's proportion to the sum of eigenvalues in the local neighborhoods of all points.
3) Volume occupied by the voxelized points in S_i.
4) We calculate three eigenvalues, e_1, e_2, e_3, of S_i and calculate six proportions: e_1/e_2, e_2/e_3, e_1/e_3, e_1/sum, e_2/sum, e_3/sum, where $sum = e_1 + e_2 + e_3$.
5) We obatin three eigenvectors, $\vec{e_1}$, $\vec{e_1}$, $\vec{e_1}$, of S_i, project the points onto each of them, and calculate three metric variances v_1, v_2, v_3 (which we used instead of e_1, e_2, e_3).
6) Orientation of the eigenvector corresponding to the smallest eigenvalue, indicating the orientation of S_i.
7) We project the points onto each eigenvector and get the distance to the farthest point from the medium in both directions $l^1_{\vec{e_1}}$, $l^2_{\vec{e_1}}$, $l^1_{\vec{e_2}}$, $l^2_{\vec{e_2}}$, $l^1_{\vec{e_3}}$, $l^2_{\vec{e_3}}$. We then calculate the following values: $(l^1_{\vec{e_1}} + l^2_{\vec{e_1}})$, $(l^1_{\vec{e_2}} + l^2_{\vec{e_2}})$, $(l^1_{\vec{e_3}} + l^2_{\vec{e_3}})$, $(l^1_{\vec{e_1}}/l^2_{\vec{e_1}})$, $(l^1_{\vec{e_2}}/l^2_{\vec{e_2}})$, $(l^1_{\vec{e_3}}/l^2_{\vec{e_3}})$, $(l^1_{\vec{e_1}} + l^2_{\vec{e_1}})/(l^1_{\vec{e_2}} + l^2_{\vec{e_2}})$.
8) Three proportions between features 5 and 7: $v_1/(l^1_{\vec{e_1}} + l^2_{\vec{e_1}})$, $v_2/(l^1_{\vec{e_2}} + l^2_{\vec{e_2}})$, $v_3/(l^1_{\vec{e_3}} + l^2_{\vec{e_3}})$.
9) Proportion between the occupied volume (feature 3) and the volume of the oriented bounding box of the part.

Each part S_i is finally represented by a vector f_i containing 24 features normalized to the range [0,1].

C. Vocabulary Learning

1) K-means Clustering: The approach to vocabulary learning in [1] is based on clustering of the Simple Geometric Feature (SGF) descriptors of the parts $f_i \in F$ into several clusters using k-means. The cluster centers are then used as the vocabulary A. Probabilistic Hough voting requires the probability of a feature vector matching a word in the vocabulary $p(A_i|f_i)$. This probability is obtained via

$$p(A_i|f_i) = \frac{w(A_i, f_i)}{\sum w(A_j, f_i)} \tag{1}$$

where $w(A_i, f_i) = \frac{1}{||A_i - f_i||_2}$.

In a fully automated vocabulary learning scenario like this it is crucial that the vocabulary learned from the training data is of consistently high quality for a wide range of parameter settings and datasets. The standard approach of using k-means cluster centers as vocabulary does not fit these requirements. It is known to produce highly varying results, depending on the number of clusters chosen and the initialization. As a consequence the precision and recall rates will vary greatly as well.

2) Regularized Information Maximization-based Clustering: Instead of representing the vocabulary in terms of exemplars A and deriving the cluster assignment probability $p(A|f_i)$ based on the distance of an SGF part descriptor f_i to these exemplars as done in [1], we in this paper introduce a novel softmax regression model to represent $p(A|f_i)$ and use the recently proposed Regularized Information Maximization (RIM) [2] to learn the model parameters from data in a fully unsupervised manner.

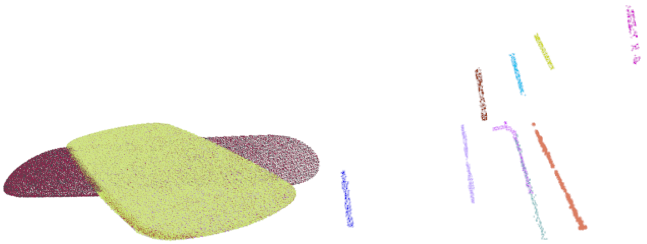


Fig. 3: Two example groups of parts in RIM-based clustering.

In the RIM framework a functional $\mathcal{F}(p(A|f, W), F, \lambda)$ is defined in such a way that it evaluates the quality of a conditional probability distribution p over labels A given the input $f_i \in F$ and parameters W. The parameter λ is a tuning parameter that is fixed during optimization and can be interpreted as a Langrage multiplier for connecting a regularization term with the objective. In the original RIM formulation the goal is to find a discriminative probabilistic model which clusters the data. Therefore, the three characteristics of the solution which determine the quality are class balance to avoid degenerate solutions, decision boundaries of sensible complexity to promote parsimony and desision boundaries that are not located in densely populated regions of the input space.

The authors of [2] were able to show that maximizing an empirical estimate of the mutual information between labels and data

$$I_W[A; F] = H[\frac{1}{N}\sum_i^N p(A_i|f_i; W)] - \frac{1}{N}\sum_i^N H[p(A_i|f_i; W)] \tag{2}$$

subject to a regularizing penalty term $R(W; \lambda)$ is suitable for expressing these requirements. The complete functional is:

$$\mathcal{F}(p(A|f, W), F, \lambda) = I_W[A; F] - \lambda R(W; \lambda) \tag{3}$$

In [2] an L2 penalty is used for regularization. This penalty is only applied to the weights and not to the bias, which leads to the property that during optimization the biases for some of the classes are driven to large negative values, which in turn means that no data points are assigned to these classes. This mechanism can be interpreted as model selection by selecting the number of classes from the data.

This functional is applied to the softmax model of $p(A|f)$ and optimized using L-BFGS [24]. Since the resulting cost function is highly non-linear a good initialization is required. We follow the original work and use k-means for initialization. Examples of the learned clusters are depicted in Fig. 3.

RIM initialized with k-means does improve results over the simple k-means approach considerably. However, higher

consistency in the results while maintaining clustering quality can be achieved by initializing RIM with a deterministic Principal Component Analysis (PCA) based initialization procedure [25]. This initialization method iteratively splits the cluster with the largest Sum of Squared Error to the cluster center along the mean of the first principle component coordinates of the datapoints in the cluster.

After the clustering and following [4], we learn a shape model for each training object view. This model specifies the distance relations among the different parts that compose the object. We extract the center of mass $s \in \Re^3$ of each part S, and the center of mass $p \in \Re^3$ of the complete point cloud. We then calculate each relation as the 3D vector $\vec{d} = p - s$.

IV. Classification

The classification part of the system consists of the following main steps: i) pre-processing, ii) generation of a set of hypotheses which indicate possible locations for objects in the point cloud, iii) a selection of the best hypotheses via a set of criteria and iv) model fitting and verification.

A. Pre-processing and Segmentation

After obtaining the point cloud of a real scene (Fig. 4, left) we first filter out the discretization error by applying a moving least squares algorithm [22]. Next we allign the floor with the XY plane and remove it. Finally we discard all the measurements beyond 3m from the camera origin due to inacceptable large depth error. To over-segment the scene we apply the same algorithm as described in Sec. III-A which results in a segmented scene as depicted in the right part of the Fig. 4.

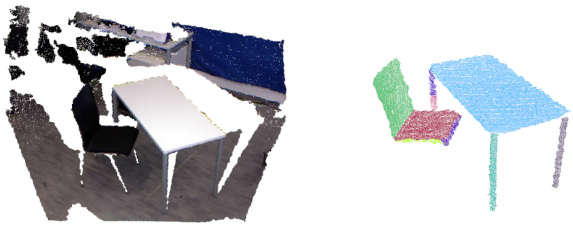


Fig. 4: Segmented real scene.

B. Probabilistic Hough Voting

Following the segmentation we obtain the corresponding feature vector f_i for each segmented part S_i. Each feature vector is then matched to a subset of words $\mathcal{A} \subset \mathcal{C}$ from the learned vocabulary (activation), which constitute possible interpretations of the part S_i. Each element of an activated word casts a vote for a possible object location. This scheme is known as probabilistic Hough voting [4], and an example is shown in Fig. 5. In this case, a part in the scene activates a word representing a tabletop. Each element in A casts a vote for the center of the tabletop inside the point cloud. As a result of this voting step we obtain the following probability of finding an object of class o at position x:

$$p(o, x \mid A_i, l) = \begin{cases} \frac{1}{\#(A_i, o)} & \text{if } x = x_v; \\ 0.0 & \text{otherwise,} \end{cases} \quad (4)$$

where $\#(A_i, o)$ indicates the number of elements a in word A_i that pertain to object o. $x_v = l + \vec{d}$, where l is the position of the part S_i found in the point cloud, and $\vec{d}$ is the relation vector of the element's shape model. For the sake of the brevity we refer the reader to the original paper [1] for the exact probability derivation.

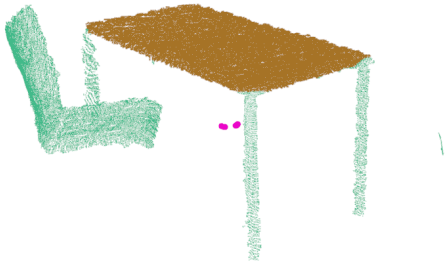


Fig. 5: Votes casted by tabletop for table center.

C. Hypotheses Selection

The different object hypotheses are found as local maxima in the voting space using a search window whose size corresponds to the half of the width of the particular object class we are looking for. We project the votes onto the (x,y)-plane in order to simplify this search. After obtaining the local maxima, we apply a threshold to select the best hypotheses (see Fig. 6).

D. Pose Estimation through Verification

Having detected positions where a certain object type is likely to be found, we verify the detection and select the best model for the object, along with its pose. To be able to do

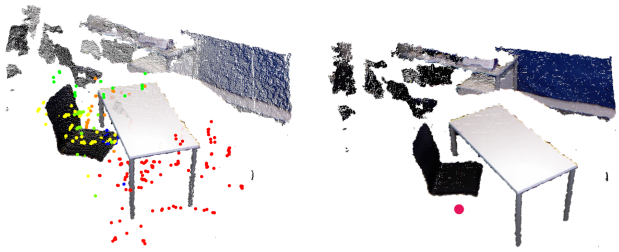


Fig. 6: Votes casted for chairs and local maxima projected to the floor.

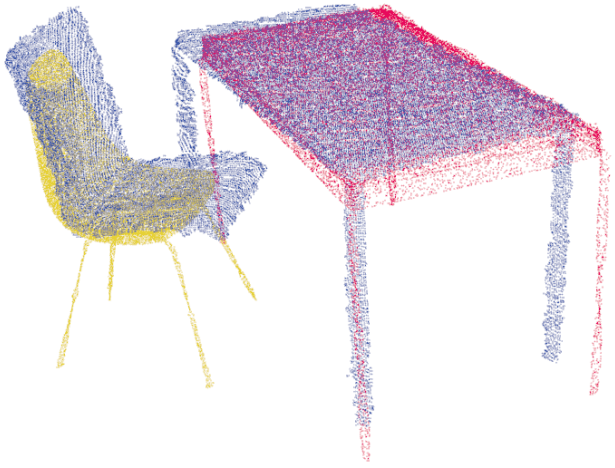


Fig. 7: Objects fitted into the scene using RANSAC.

this, we first need to retrieve the most important parts that voted for the location. Some of these parts are incorrect but their weights are low.

Our task is now to select the model and its pose from all virtually scanned models that explains most of the detected object parts. To achive this we fit the scanned models to the scene point clouds using use a 3DOF RANSAC-based algorithm in an exhaustive manner. We seek for the x and y position and orientation around the z axis (this approach assumes that objects are positioned up-right). In RANSAC algorithm We sample two points which have the same height in the model and in the scene cloud and select the ones that would result in a transformation that covers most of the scene points.

To filter out the outlier poses we apply a series of additional checks. One of them is the visibility score which is based on the 3D occupancy grid of the scene and the object inserted at the estimated pose in the scene. In the occupancy grid each voxel is labeled as free, occupied or unknown [26] and we reject models that have large parts in free space. The visibility score is defined as follows

$$vs = \frac{occupied * w + occluded}{occupied * w + occluded + free}. \quad (5)$$

Further, when the scene point has a normal vector which is close to being parallel to the upright axis, the rotation of the model can not be accurately estimated. In these cases, we use a random rotation and since we check a high percentage of points, the models are correctly identified. An example of the 2 correctly fitted object models is shown in Fig. 7.

E. Speed-up through Part Subset Classification

Since the geometric model fitting presented above that is used to reject false positive detections and to obtain the object's pose needs to iterate over the possible CAD models and fit them to the detected parts, increasing the number of used models would linearly increase the necessary runtime. Therefore we propose an additional classification step, that can be used to select the right CAD model, or at least score them in order to reduce the number of models to be tested.

The detection process described above identifies some of the object parts in the point cloud, based on which a descriptor can be computed for scoring the possible CAD models, and exclude from fitting those that have a score below a threshold. As a first experiment, presented here as a proof of concept, we computed a VFH descriptor for the detected object parts, and scored the possible CAD models using K Nearest Neighbors with an L2 distance metric. The number of identified parts is typically around 3, so in order to obtain the training data for classification, we generated all subsets of 1 to 3 parts from the simulated scans of the CAD models, resulting in around 25000 point clouds for the 8 CAD models we have currently. In each experiment, we used 80% of the data as training and 20% as testing data, and repeated the experiments 10 times. Here we report the mean classification success rates and their standard deviation (STD).

In identifying the correct CAD models, the average true positive rate was 75.36% with a STD of 3.92, while identifying the correct furniture category the average was 85.31% with a STD of 5.62. The full confusion matrix holding the results for the approximately 45000 tests can be seen in Table I. These results encourage us to pursue this approach further, as the correct CAD model is most likely obtaining a high score, even if not the best one, therefore not all of them need to be fit using the RANSAC procedure. However, further tests on labeled examples (and negative examples) obtained from complete scans would be required, something that we are actively working on.

Armc.	**88.9%**	6.0%	1.3%	0.2%	2.6%	0.3%	0.2%	0.4%
Chr1	3.0%	**86.1%**	**7.7%**	**1.0%**	0.3%	0.2%	0.5%	1.3%
Chr2	3.5%	**25.6%**	**67.5%**	**1.5%**	0.4%	0.4%	0.2%	1.0%
Chr3	1.8%	**21.2%**	**5.9%**	**68.5%**	0.2%	1.1%	0.6%	0.7%
Side.	23.5%	5.2%	4.3%	0.7%	**60.8%**	2.2%	2.1%	1.4%
Tbl1	0.2%	1.6%	0.4%	0.0%	0.0%	**81.7%**	**15.0%**	**1.2%**
Tbl2	0.2%	5.1%	1.4%	0.1%	0.5%	**29.4%**	**62.3%**	**1.1%**
Tbl3	5.2%	17.4%	4.9%	0.7%	0.3%	**1.7%**	**1.5%**	**68.4%**

TABLE I: Confusion matrix for the different CAD models (armchair, chairs, sideboard and tables). Successful category identification marked in bold.

V. Quantitative Results and Demonstration

Although we ran the algorithm on real data as demonstrated in Fig. 4, for this precision and recall based evaluation we used a dataset of generated point clouds from the 8 CAD models. The point clouds in the dataset were generated using the same aproach that was used for generating training data. We used around 200 scans for testing. As we know the ground truth centroid of the object position in every scan we can easily evaluate the algorithm's performance. If we have a local maxima in the 5cm window around the real object's centroid position we count it as a $TruePositive$. If we have a local maxima outside of the window or in a scene where no object of this class is present we count it as a $FalsePositive$. If there is no local maxima in the 5cm

window around the object we count it as a $FalseNegative$. Following these definitions, we compute the precision as:

$$\frac{TruePositive}{TruePositive + FalsePositive} \tag{6}$$

and the recall as:

$$\frac{TruePositive}{TruePositive + FalseNegative}. \tag{7}$$

By varying the threshold we build the precision-recall curve. We use a relative threshold which can vary from 0 to 1. The real threshold is computed for every scene using the following formula

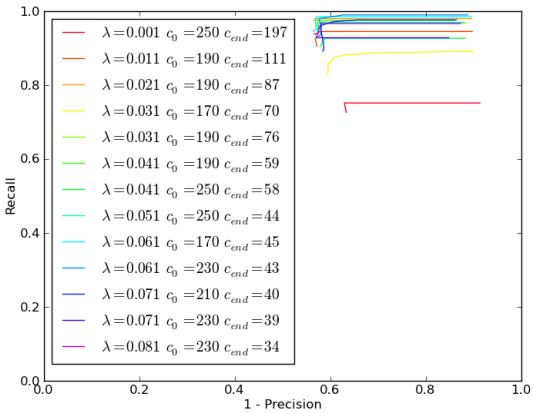
$$(MaxCellWeight - MinCellWeight) * RelativeThreshold + MinCellWeight. \tag{8}$$

The results depicted in Fig. 8 show clearly, that the k-means initialized RIM model consistently gives better results than k-means and allows for completely automatic high-quality vocabulary learning in this scenario. The preliminary experiments with PCA initialization show a slight improvement in repeatability over k-means initialization, but clearly more experiments are required to conclusively show a statistically significant improvement.

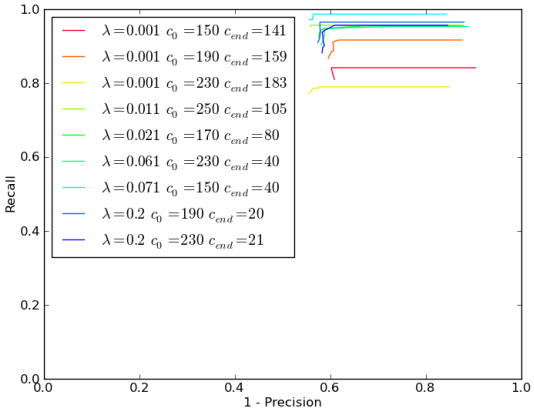
We provide the open-source software and the documentation[2]. **The system will be demostrated live during the workshop.**

VI. Conclusions and Future Work

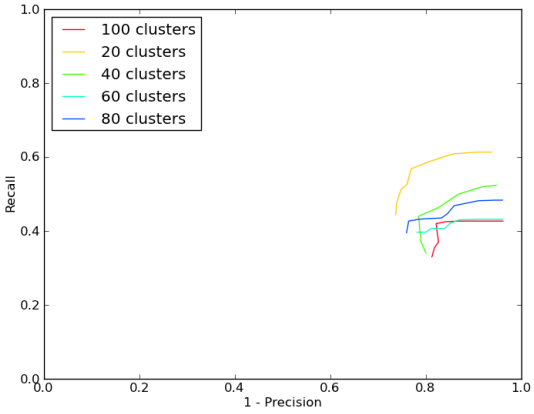
In this paper, we have revisited and implemented an approach by Mozos et al. [1] that allows a robot to learn models of office objects downloaded from the Web and to use them to detect and localize instances of these types of objects in new scenes represented by 3D point clouds. The original system was upgraded in that it now works with the Kinect sensor, it provides a better unsupervised clustering algorithm based on Regularized Information Maximization, and finally, we made a first step in making the system scalable for the large number of models by introducing an additional fast part-object classification step before the pose fitting step. In the future we will rigorously evaluate this latter step and in parallel also consider an implementation of this algorithm on the GPU or in the cloud. Finally we plan to extend the formalism behind the Semantic Objects Maps [5] to support semi-static and dynamic objects and subsequently integrate the detection and localization of herein investigated furniture articles.



(a) RIM, PCA initialization



(b) RIM, k-means initialization c_0 = #clusters after initialzation c_{end} = #clusters after optimization



(c) k-means

Fig. 8: Precision-recall curves for the clustering variants.

References

[1] O. M. Mozos, Z. C. Marton, and M. Beetz, "Furniture Models Learned from the WWW – Using Web Catalogs to Locate and Categorize Unknown Furniture Pieces in 3D Laser Scans," *Robotics & Automation Magazine*, vol. 18, no. 2, pp. 22–32, June 2011.

[2] R. Gomes, A. Krause, and P. Perona, "Discriminative clustering by regularized information maximization," in *Advances in Neural Information Processing Systems 23*, J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds., 2010, pp. 775–783.

[3] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3d recognition and pose using the viewpoint feature histogram," in *Proceedings of the 23rd IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, October 2010.

[4] B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," *International journal of*

[2] http://ros.org/wiki/furniture_classification

computer vision, vol. 77, no. 1-3, pp. 259–289, 2008.

[5] D. Pangercic, M. Tenorth, B. Pitzer, and M. Beetz, "Semantic object maps for robotic housework - representation, acquisition and use," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vilamoura, Portugal, October, 7–12 2012, accepted for publication.

[6] D. Huber, A. Kapuria, R. R. Donamukkala, and M. Hebert, "Parts-based 3D object classication," in *Conf. on Computer Vision and Pattern Recognition*, 2004.

[7] S. Ruiz-Correa, L. G. Shapiro, and M. Meila, "A new paradigm for recognizing 3-D object shapes from range data," in *Int. Conf. on Computer Vision*, 2003.

[8] F. Tombari and L. Di Stefano, "Object recognition in 3d scenes with occlusions and clutter by hough voting," in *Proceedings of the 2010 Fourth Pacific-Rim Symposium on Image and Video Technology*, ser. PSIVT '10, Washington, DC, USA, 2010, pp. 349–355.

[9] S. Agarwal and D. Roth, "Learning a sparse representation for object detection," in *Europ. Conference on Computer Vision*, 2002.

[10] M. Sun, B. Xu, G. Bradski, and S. Savarese, "Depth-encoded hough voting for joint object detection and shape recovery," in *Europ. Conference on Computer Vision*, 2010.

[11] Z. C. Marton, R. B. Rusu, D. Jain, U. Klank, and M. Beetz, "Probabilistic Categorization of Kitchen Objects in Table Settings with a Composite Sensor," in *Proceedings of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, USA, October 11-15 2009.

[12] R. Schnabel, R. Wahl, and R. Klein, "Efficient RANSAC for Point-Cloud Shape Detection," *Computer Graphics Forum*, vol. 26, no. 2, pp. 214–226, June 2007.

[13] Z.-C. Marton, D. Pangercic, N. Blodow, J. Kleinehellefort, and M. Beetz, "General 3D Modelling of Novel Objects from a Single View," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Taiwan, 2010.

[14] A. S. Mian, M. Bennamoun, and R. A. Owens, "Matching tensors for automatic correspondence and registration," in *Europ. Conf. on Computer Vision*, 2004.

[15] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *IEEE Int. Conf. on Robotics and Automation*, Japan, 2009.

[16] H.-P. Chiu, L. Kaelbling, and T. Lozano-Perez, "Virtual training for multi-view object class regnition," in *IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition*, 2007.

[17] A. Toshev, A. Makadia, and K. Daniilidis, "Shape-based object recognition in videos using 3d synthetic object models," in *IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition*, 2009.

[18] K. Lai and D. Fox, "3D laser scan classification using web data and domain adaptation," in *Robotics: Science and Systems*, USA, 2009.

[19] *Google 3D Warehouse*. [Online]. Available: http://sketchup.google.com/3dwarehouse/

[20] *Vitra Furnishnet*. [Online]. Available: http://www.vitra.com

[21] *EasternGraphics*. [Online]. Available: http://portal.pcon-catalog.com

[22] Z. C. Marton, R. B. Rusu, and M. Beetz, "On Fast Surface Reconstruction Methods for Large and Noisy Datasets," in *IEEE Int. Conf. on Robotics and Automation*, 2009.

[23] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3D Point Cloud Based Object Maps for Household Environments," *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 927–941, 2008.

[24] J. Nocedal, "Updating Quasi-Newton Matrices with Limited Storage," *Mathematics of Computation*, vol. 35, no. 151, pp. 773–782, 1980.

[25] T. Su and J. Dy, "In search of deterministic methods for initializing K-means and Gaussian mixture clustering," *Intelligent Data Analysis*, pp. 1–42, 2007. [Online]. Available: http://iospress.metapress.com/index/TN4334540U6L766T.pdf

[26] N. Blodow, R. B. Rusu, Z. C. Marton, and M. Beetz, "Partial View Modeling and Validation in 3D Laser Scans for Grasping," in *IEEE-RAS Int. Conf. on Humanoid Robots*, France, 2009.