

Learning to Discover Objects in RGB-D Images Using Correlation Clustering

Michael Firman

Diego Thomas

Simon Julier

Akihiro Sugimoto

Abstract—We introduce a method to discover objects from RGB-D image collections which does not require a user to specify the number of objects expected to be found. We propose a probabilistic formulation to find pairwise similarity between image segments, using a classifier trained on labelled pairs from the recently released RGB-D Object Dataset. We then use a correlation clustering solver to both find the optimal clustering of all the segments in the collection and to recover the number of clusters. Unlike traditional supervised learning methods, our training data need not be of the same class or category as the objects we expect to discover. We show that this parameter-free supervised clustering method has superior performance to traditional clustering methods.

I. INTRODUCTION

Many tasks in robotics require high-level reasoning about the environment. This can only be achieved through recognising objects, typically from visual sensor data. However, most existing approaches are limited to what can be learned from hand-labelled training data. Training data is expensive to produce, subject to human bias and most importantly limits the number of objects that can be recognized to those present in the training set. For example, while the impressive RGB-D object dataset [1] contains views of 300 objects, there are around 12,000 distinct items in the current IKEA product range alone [2].

We therefore reason that there are too many different things in the world to apply semantic tags from hand-labelled training data with supervised learning. Instead, we focus on the problem of the unsupervised *discovery* of objects. A large fraction of objects within our world are visually identical. That is, for each instance of such an object there exists in the world other things that have the same size, shape and appearance. Given an image collection, we expect to be able to recover each repeated object’s properties such as its size, shape and color. In addition, we should be able to simultaneously find the location of each such object in each image (Fig. 1). Representative views of these objects can then be presented to a human user for labelling, or a robotic system can attach learned information to each discovered item without ever knowing its human-readable name.

We follow recent works such as [3], [4], [5] in focusing on object *instance* discovery rather than class. That is, we consider objects which are visually identical to be similar,

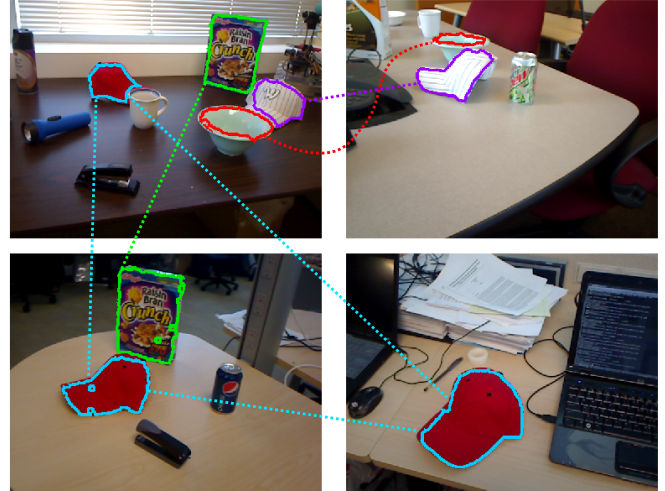


Fig. 1. An example of our object discovery results on four of the 36 images in our test set \mathcal{D}_2 . The colored boundaries and the dashed lines between segments indicate found object types. Note that matches are found in spite of large changes in viewpoint and poor segmentation at object boundaries. Our algorithm performs clustering of segments without user parameters such as the number of objects to be found.

rather than objects with the same semantic label. For example, we consider a tall red mug to be different from a short, blue one.

Many recent object discovery algorithms operate by first hypothesising segmentations of a scene, before grouping segments together using a clustering algorithm (e.g. [4], [6]). However, most such algorithms require a human user to specify parameters to control this grouping, typically the number of objects to be found. We consider it to be unreasonable to expect an autonomous system to require this level of supervision. Instead, we present a principled method for finding matching groups of segments, where we use a small set of *pairs* of hand-labeled training images to learn a probability measure that two segments should belong in the same cluster. While we use a supervised classifier, the training data *need not be of the same type or class* as the objects to be discovered, as the training data is only used to learn a measure of similarity. Furthermore, just a small selection of training examples can be used to discover many different types of objects. We then use *correlation clustering* [7], [8] to recover the correct segment grouping and automatically find the number of clusters, without relying on user parameters. We have released code of work presented in this paper, available at www.cs.ucl.ac.uk/staff/M.Firman.

For our work, we exploit the increasing prevalence of

M. Firman and S. Julier are with the Department of Computer Science, University College London. {m.firman, s.julier}@cs.ucl.ac.uk

D. Thomas and A. Sugimoto are with the National Institute of Informatics, Tokyo. {diego.thomas, sugimoto}@nii.ac.jp

M. Firman was partially supported by an International Internship Grant from the National Institute of Informatics, Tokyo.

depth cameras, and make use of the Microsoft Kinect scanner to capture static RGB-D frames.

II. PROBLEM STATEMENT

We reason that many objects found in the world can be considered as instantiations of abstract object *templates*. Each template can be thought of as a mold which holds information on the three-dimensional shape and size of the object, its color, texture, material and other physical properties. We define a *scene* as being a set of one or more object instantiations together with background environment (e.g. walls and floors), and further properties such as lighting and background color. Each scene is then captured in a single static RGB-D image \mathcal{D} .

The aim of our object discovery system is to recover the number of object templates and each of their appearance models, given a set of input RGB-D images $\{\mathcal{D}_{1..N}\}$. The secondary aim of object discovery is to recover the set of pixels in each image corresponding to each object.

III. RELATED WORK

Most object discovery algorithms in the vision and robotics communities operate in a similar way; by first segmenting the scenes into regions, then matching together similar regions. For example, Shin et al. [3] recover object templates by finding matches in feature space and confirming via alignment of the parts in 3D space. Endres et al. [6] use topic modelling to assign each segment from a scan to a separate cluster. While their results are impressive, this approach relies on a user-specified number of classes and is not robust to the inclusion of ‘noisy’ segments which do not correspond to any objects. Kang et al. [5] use multiple alternative divisions of each of their images to increase the chance of finding well-matching segmentations. They present impressive results on 2D images, but rely on some user thresholds to determine similarity measures. They perform object discovery on the network of pairwise matches using a graph-based community discovery algorithm.

Herbst et al. [4] present a novel segmentation concept, using the difference between two views of the same scene to accurately segment objects which have been moved. Their clustering algorithm is spectral clustering [9], which relies on a clean dataset; they manually remove noisy segments before applying their clustering algorithm. Similarly, affinity propagation [10] is a clustering algorithm used by Triebel et al. [11] to find objects which occur multiple times in indoor laser scans.

All of these algorithms reduce the more difficult scene understanding problem down to some form of clustering problem. The goal of clustering is to assign a label y_i to each item in a set of M distinct items, such that similar items have identical labels while dissimilar items have different labels. This stated objective for clustering raises the obvious question of what we mean by the terms *similar* and *differing*. Looking at different aspects of objects’ properties may give different clusterings, and for each property there are various degrees of ‘similarity’. In fact, many clustering methods

avoid ever formally defining these measures, and instead rely on good input data and one or more user specified parameters in order to constrain the problem. These parameters may be the number of clusters required (e.g. k -means [12] or spectral clustering [9]) or another measure (e.g. datapoint ‘self-similarity’ for affinity propagation [10]).

Furthermore, clustering in Euclidean feature space is highly sensitive to the scaling of features; e.g. a feature in the range $[0, 200]$ would have a far larger influence than one in the range $[0, 0.1]$. Even after suitable scaling of features, most clustering algorithms treat each feature as being equally important, when in fact it is likely that there are some features which are far more important than others for the task in hand. This makes distances in Euclidean space poorly suited to matching feature vectors. Finally, most clustering algorithms are very sensitive to the inclusion of singleton data points, which do not belong in any clusters. Many object discovery algorithms assume such ‘noisy’ segments are removed before the clustering stage (e.g. [6], [4]).

Our method for object discovery is inspired by recent work by Vicente et al. [13], who find accurate segmentations of objects in pairs of images by learning a similarity measure from pairs of ground truth segmentations. However, they cannot perform object discovery as they are limited to only one class of object at a time, and exactly one object instance is assumed per image. Our use of a correlation clustering solver [8], [7] means we do not face this limitation, and its combination with a Random Forest classifier [14] allows us to avoid the problems we have outlined associated with typical clustering methods.

Concurrently with our work, Collet et al [15] tackled the problem of object discovery using pairwise measures between image regions. In contrast to their graph-based approach, we use probabilistic methods to find our pairwise relationships and final clustering.

IV. METHODOLOGY

Our algorithm comprises of four stages. Firstly, we segment our images into regions, before computing a feature representation of each region. We then find the probability that each pair of regions depicts the *same* object. Finally, we use these pairwise probabilities to group mutually consistent segments (Fig. 2).

A. Segmentation

We desire an object segmentation algorithm which will accurately delineate the boundaries of individual objects within each image \mathcal{D} , i.e. to propose a set of regions $\{\mathcal{R}_{1..M}\}$, where $\mathcal{R}_i \subset \mathcal{D}$, such that each object’s projection in the image space will have a large overlap with exactly one region.

We first detect and remove large planar regions from our scenes with the RANSAC implementation of [16]. To form complete object segmentations, we then use a simple top-down segmentation, placing pixels in the same region if they are neighbors in image space and the 3D spatial distance between them is less than a threshold t_d . For our data we

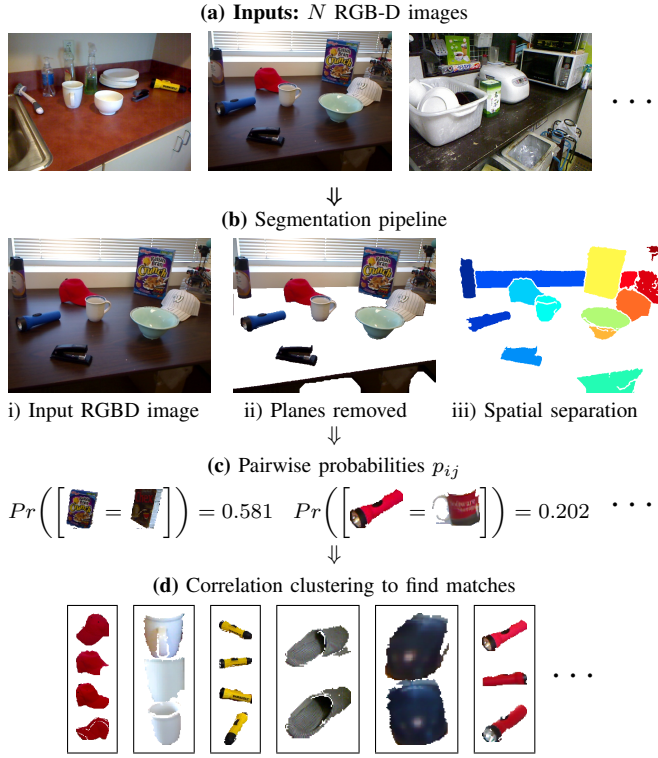


Fig. 2. Pictorial overview of our algorithm

found $t_d = 0.04$ m works well. Finally, regions for which the majority of their boundary is shared with the image border are removed. An example segmentation pipeline is shown in Fig. 2(b). Notice that some of the regions correspond to actual objects, while other, ‘clutter’ regions form parts of the wall or small sub-parts of objects in the scene. This initial segmentation of our RGB-D scenes is not a main focus of this work, and for more complex and cluttered scenes we expect that a different segmentation method will be required.

B. Feature vector representation of each region

We map each region \mathcal{R}_i to a feature vector \mathbf{x}_i , with the aim of removing some of the non-discriminative information contained in \mathcal{R}_i (such as pose), while retaining the information relevant to our final objective (such as size and shape). We use a selection of standard features from vision and robotics to achieve this; these are listed in Table I.

C. Pairwise scores between regions

Given a pair of image regions $(\mathcal{R}_i, \mathcal{R}_j)$, we want to know the probability p_{ij} that the two regions should be placed in the same cluster, i.e. that they depict the same object. As many of the regions in our test set will be background clutter (see Fig. 2(b)), we assert that p_{ij} is the probability that the two regions should take on the same label *and* both represent valid objects:

$$p_{ij} = Pr(y_{ij} \wedge \xi_i \wedge \xi_j), \quad (1)$$

where y_{ij} indicates whether both regions take on the same label or not (i.e. $y_{ij} \equiv [y_i = y_j]$, where $[\bullet]$ is the Iverson bracket) and ξ_i denotes the event that region \mathcal{R}_i actually

TABLE I
THE FEATURES USED IN OUR EXPERIMENTS

Feature name	Type	Dimension
Bounding box size ²	Size	3
Spin Image histogram [17]	Shape	50
RGB histogram of gradients	Texture	9
Mean RGB values	Color	3
Shape distribution [18]	Shape	50
Histogram over RGB values	Color	64

² Aligned to principal axes of 3D points. $\Sigma = 179$

depicts a single object as opposed to background clutter or a part of one or more objects. We use the definition of conditional probability to reformulate (1) like so:

$$Pr(y_{ij} \wedge \xi_i \wedge \xi_j) = Pr(y_{ij} | \xi_i \wedge \xi_j) Pr(\xi_i \wedge \xi_j) \quad (2)$$

We now describe how we compute each of the two quantities on the R.H.S of (2).

1) *Computing $Pr(y_{ij} | \xi_i \wedge \xi_j)$* : In our work we seek to directly *learn* $Pr(y_{ij} | \xi_i \wedge \xi_j)$, using a small set of segmented, labeled regions as training data. For each pair of regions we create a pairwise feature vector

$$\mathbf{x}_{ij} = |\mathbf{x}_i - \mathbf{x}_j|. \quad (3)$$

Each \mathbf{x}_{ij} is therefore of the same dimension as \mathbf{x}_i ; each element of \mathbf{x}_{ij} captures a difference in one feature dimension.

We then use a supervised learning algorithm, supplied with training data in the form of pairs of masked object regions. Each masked region in the training set is a complete view of a single object (i.e. $Pr(\xi_i) = 1$). Each *pair* of training objects thus constitutes a training example $(\mathbf{x}_{ij}, y_{ij})$. The training dataset we use is described in section V-A.

In our work we make use of the Random Forest classifier [14]. A Random Forest is a set of decision trees, each of which is trained on a different subset of the training data. At test time, each tree votes for a class (in our case, the classes correspond to $(y_{ij} | \xi_i \wedge \xi_j)$ and $(\neg y_{ij} | \xi_i \wedge \xi_j)$), and the fraction of votes for each class can be used to approximate a probability of class membership [19]. Random Forests are well suited to our purpose, as they automatically select the features most suitable for separating the classes and they examine each dimension separately, making them robust to different scaling of each feature (unlike e.g. SVMs).

2) *Computing $Pr(\xi_i \wedge \xi_j)$* : To compute the probability that both regions are valid objects, we exploit their independence, i.e. $Pr(\xi_i \wedge \xi_j) = Pr(\xi_i) Pr(\xi_j)$. Each unary probability $Pr(\xi_i)$ is computed using the Neural Network formulation of Silberman et al. [16], trained on their NYU dataset. Each region in their dataset is hand-labeled as being either ‘furniture’, ‘wall’, ‘ceiling’ or moveable ‘props’ such as pillows or bottles, and they train a Neural Network to give a class membership probability for any new segment. We find that using the probability for being a member of the ‘prop’ class or the ‘furniture’ helps to capture the property of being an object. We use the sum of these two class outputs

as our estimation for $Pr(\xi_i)$. An analysis of the performance of this classifier is given in section VI-B.

D. Finding matching groups of segments

It is tempting at this stage to simply place pairs of segments which have a high p_{ij} value in the same cluster. However, our probabilistic interpretation of p_{ij} (i.e. as the parameter for a Bernoulli distribution) implies that there is a probability $1-p_{ij}$ that regions $(\mathcal{R}_i, \mathcal{R}_j)$ should *not* be placed in the same cluster, and it only takes a few false positive connections to cause every \mathcal{R} to be assigned the same label. We therefore need a way of using all our inferred values of p_{ij} to find a clustering robust to false positives. For this task, we use *correlation clustering*.

Correlation clustering describes a family of clustering algorithms which rely on pairwise similarities between data points to both find optimal groupings and to estimate the number of clusters [8]. Given an affinity matrix W , where $W_{ij} \in [-\infty, \infty]$ represents a degree of attraction (+ve values) or repulsion (-ve values) between data points i and j , a correlation clustering algorithm finds a labelling of the data points $Y = \{y_1, \dots, y_M\}$ which minimizes the energy function

$$E(Y) = - \sum_{i=1}^M \sum_{j=1}^M W_{ij} [y_i = y_j]. \quad (4)$$

We follow [7] in interpreting W_{ij} as the log-odds ratio between the probability distribution over positive pairs and negative pairs; i.e. $W_{ij} = \log \left(\frac{p_{ij}}{1-p_{ij}} \right)$.

We use the Adaptive-Label ICM solver proposed by Bagon and Galun [7]. Based on earlier work by Besag [20], AL-ICM is a greedy iterative solver which changes the label of each y_i in turn to the label which minimises the energy $E(Y)$ the most. In addition to the labels already existing in Y , y_i can also be assigned to a new singleton label. This continues until the energy cannot be lowered any further. We add a novel extension whereby the solver is run multiple times with different random initial labellings; the solution with the lowest overall energy is then chosen. This helps to avoid local minima.

A benefit of our formulation is that it will automatically handle outliers. An outlying region \mathcal{R}_k is one which should not be placed in any cluster, either due to not representing an object, or because there are no other objects of that class in the dataset. Such segments will have a low value of p_{ik} for all i , and hence the energy in (4) will be minimised by assigning them unique, singleton labels.

E. Recovering the object templates

Each region in each resulting cluster forms part of the template model for that object. These could be used to discover new instances of this object in frames which did not form part of the original test set, or be presented to a human user for semantic labelling. We defer such tasks for further work.

V. TRAINING AND TEST DATA

In this section we introduce the training and test datasets which we use to evaluate our algorithm.

A. Training dataset

Our training set is formed using a total of 3100 views of 155 randomly-chosen unique objects from the RGB-D dataset [1]. Each masked region from this training dataset represents a full (i.e. well-segmented and unoccluded) view of an object, and has an associated instance label y_i such as `apple_3` or `food_jar_1`. A random 37,500 positive pairs ($y_{ij} = 1$) and 37,500 negative pairs ($y_{ij} = 0$) are used to train the Random Forest classifier.

B. Test datasets

We use two separate test datasets, one consisting of presegmented objects, and the other consisting of RGB-D images of real scenes. We want to demonstrate the generalizability of our system in grouping object types which did not form part of the training set. To show this effectively, we ensure that neither of our test sets contain any overlap with the training set at either the instance or category level. For example, a ‘notebook’ instance is included in the training set; therefore no notebooks are present in either test set.

Fully segmented dataset \mathcal{D}_1 : This dataset contains 100 randomly selected masked regions representing 10 different object instances, and is taken from the same dataset as the training data. For this well-segmented dataset we set $Pr(\xi_i) = 1$ for all values of i .

Real-world scene dataset \mathcal{D}_2 : This comprises a subset of 21 frames from the 8 video sequences accompanying the RGB-D object dataset [1]. We augment these images with 15 new RGB-D images, representing more views of household objects in indoor scenarios (see Fig. 1 for examples). The frames are mostly non-overlapping, although some view the same environment but from highly differing angles.

After the segmentation stage of our algorithm on \mathcal{D}_2 , we assign ground truth labelling. Each \mathcal{R}_i is given a label k iff it has a $> 70\%$ overlap with a ground truth region with label k . Regions which are left unassigned to any class are given a unique, singleton label for the purposes of clustering evaluation, and are separately denoted as being ‘clutter’ for evaluation of the unary classifier.

VI. EVALUATION

To evaluate the output of the clustering algorithms, we use the Adjusted Rand Index (ARI) [21], a continuous measure of similarity of partitionings. $ARI = 1$ indicates a partitioning of the data equal to the ground truth, while $ARI = 0$ indicates a partitioning no better than chance.

A. Clustering evaluation on dataset \mathcal{D}_1

We compare our correlation clustering algorithm with more commonly used clustering algorithms; in particular, we compare with k -means due to its ubiquity, and spectral clustering [9] and affinity propagation [10] due to their recent

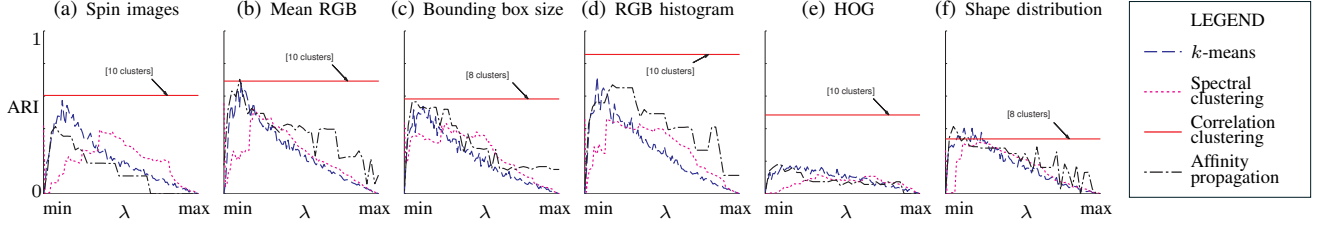


Fig. 3. Comparison of our correlation clustering implementation with other clustering methods, over a range of their parameter settings (see section VI-A for details of λ settings). Results are shown for dataset \mathcal{D}_1 ; each plot (a)-(f) corresponds to a different feature. The number of clusters recovered by our method (red line) is indicated. The ground truth labelling has 10 clusters. Higher values for adjusted Rand index are better.

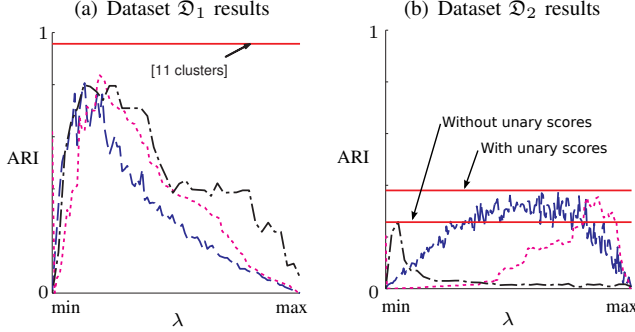


Fig. 4. Comparison of clustering algorithms on both datasets, using all features. See section VI-A for details of λ settings. Line types are as specified in Fig. 3.

use in object discovery algorithms. Algorithms which operate in Euclidean feature space, such as k -means, are strongly affected by the relative scaling of features. To provide a fair comparison, we perform some experiments using only single features. When using all features we scale each feature (e.g. ‘spin image histogram’ or ‘mean RGB’) to sum to one.

Each of our comparison algorithms accepts a single parameter λ . For k -means and spectral clustering, where λ is the number of classes, we test each possible value (i.e. $\lambda = [1, 2, \dots, M]$). λ for affinity propagation is a measure of ‘self-similarity’—for this we linearly interpolate 50 values between 0.01 and 5.

Results using single features to cluster segmented objects in \mathcal{D}_1 are shown in Fig. 3, and a confusion matrix for this clustering is presented in Fig. 5(a). Using single features, our method is only outperformed when ‘mean RGB’ and ‘shape distributions’ are used, and even then only with a small range of λ values. On \mathcal{D}_1 , the RGB histogram appears to be the feature with the most discriminatory power. This is most likely due to the highly controlled conditions under which this data was captured [1]. We then combine all features and repeat the experiment (Fig. 4(a)). Our algorithm maintains a superior performance over competing clustering algorithms.

B. Segmentation and unary terms on scenes dataset \mathcal{D}_2

Of the 105 instances of objects in dataset \mathcal{D}_2 , 85 are segmented well enough to be given an object label, according to section V-B. Of these 85 regions, 9 have unique object labels not shared with any other region. In addition to these 85 ‘object’ regions, there were 94 ‘clutter’ regions found

which did not share a good enough overlap with a ground truth object to be given a label. We assessed the quality of the $Pr(\xi)$ classifier (section IV-C.2) at discriminating between ‘object’ and ‘clutter’; the area under the receiver operating characteristic curve was found to be 0.754.

C. Evaluation on scenes dataset \mathcal{D}_2

We then evaluate our clustering algorithm on \mathcal{D}_2 , using the same method as for \mathcal{D}_1 . Results are presented in Fig. 4(b). As with \mathcal{D}_1 , we outperform our comparison clustering methods, managing to successfully recover many object types. We present a confusion matrix comparing the clusters we recovered with the ground truth clustering in Fig. 5(b). Some objects successfully recovered are presented in Figs. 6(a-c) and 2(d), and some results in the context of the original RGB-D images are delineated in Fig. 1.

Failures can occur for a number of reasons. False negatives in the pairwise matching can occur when two regions depict the same object but with very different segmentations, or viewed under different lighting conditions (e.g. Fig. 7(c)). Pairwise false positives most frequently occur when the unary probabilities $Pr(\xi_i)$ are incorrectly high, causing pairs of background clutter to be incorrectly given a high p_{ij} value—see Fig. 7(b) for an example of this.

Enough false positives cause errors in the final clustering. The cluster shown in Fig. 6(d), for example, is formed of background segments which score highly on both the conditional pairwise term $Pr(y_{ij} | \xi_i \wedge \xi_j)$ and the unary terms $Pr(\xi_i)$ and $Pr(\xi_j)$. In this instance, forming a better estimate of $Pr(\xi_i)$, perhaps trained on scenes more similar to our test data, could help prevent these kinds of errors.

D. Timings and complexity

Our segmentation and feature computation routines were coded in MATLAB and took on average 1.59s and 10.8s per image respectively. The $O(M^2)$ Random Forest classification and correlation clustering took 1.26s and 10.8s respectively when run on dataset \mathcal{D}_2 . All experiments were performed on a machine with a 2.4 GHz Intel i5 processor, with 8 GB of RAM.

VII. CONCLUSIONS AND FURTHER WORK

In this paper we have developed and demonstrated a method for object discovery in RGB-D data. Unlike previous supervised learning methods, which are trained on individual

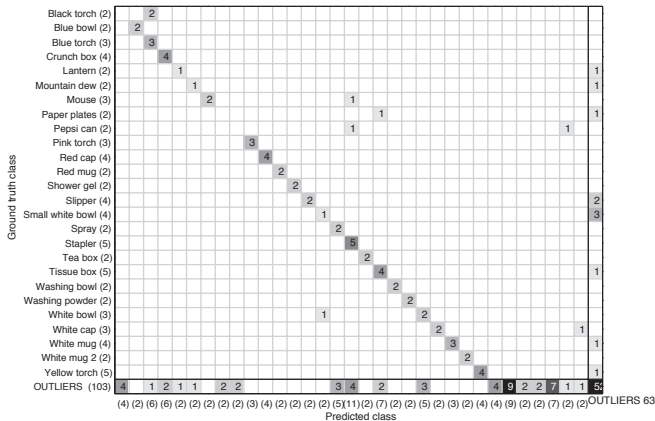
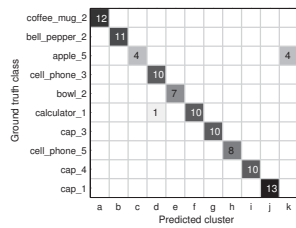
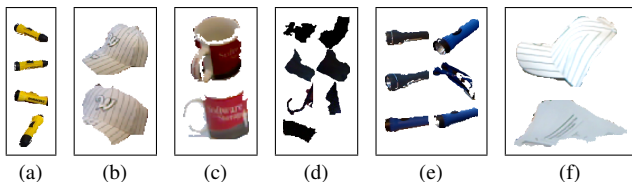


Fig. 5. Confusion matrices for the object discovery performed on our data, using all features. Each row represents a ground truth class, each column a found cluster; numbers in the grid represent assignment of items to ground truth and found clusters. A perfect result would be a square matrix with only on-diagonal entries. For \mathcal{D}_2 , outliers (i.e. singleton segments) are combined into the final row (ground truth) and final column (predicted)—otherwise, the ordering of rows and columns is arbitrary.



classes of a known fixed number, we merely learn the information about whether pairs of objects are the same or not. This means that the number of classes is not fixed, and novel classes can be discovered at test time. Unlike many other scene understanding methods, we do not need to specify the number of classes present.

For future work, we would like to investigate the improvements and extensions we suggest in sections VI-C and IV-E, such as improving the quality of the unary scoring. In addition to these areas, we would like to increase the number of test images used to explore the possibility of object discovery on a very large scale dataset. We are also interested in the possibilities for online learning using this algorithm, discovering objects ‘on-the-fly’ as a series of images are presented into the system.





	$y_{ij} = 1$	$y_{ij} = 0$
$p_{ij} > 0.5$	(a) True positive (75) 	(b) False positive (638) 
$p_{ij} < 0.5$	(c) False negative (13) 	(d) True negative (15,205) 

Fig. 7. Pairwise results examples on regions from \mathfrak{D}_2 comparing predicted outputs with ground truth. Numbers in brackets indicate the number of pairs falling into each category.

Acknowledgements: The authors would like to thank Yannick Verdie, Sara Vicente, and Gabriel Brostow and his group at UCL, for their extremely valuable discussions and contributions to the paper.

REFERENCES

- [1] K. Lai, L. Bo, X. Ren, and D. Fox, “A large-scale hierarchical multi-view RGB-D object dataset,” in *International Conference on Robotics and Automation (ICRA)*, 2011.
- [2] “IKEA FAQ,” http://www.ikea.com/ms/en_AU/customer_service/need_help/faq.html#0008, 2012, accessed: 20/02/2013.
- [3] J. Shin, R. Triebel, and R. Siegwart, “Unsupervised discovery of repetitive objects,” in *International Conference on Robotics and Automation (ICRA)*, 2010.
- [4] E. Herbst, X. Ren, and D. Fox, “RGB-D object discovery via multi-scene analysis,” in *Intelligent Robots and Systems (IROS)*, 2011.
- [5] H. Kang, M. Hebert, and T. Kanade, “Discovering object instances from scenes of daily living,” in *International Conference on Computer Vision (ICCV)*, 2011.
- [6] F. Endres, C. Plagemann, C. Stachniss, and W. Burgard, “Unsupervised discovery of object classes from range data using latent Dirichlet allocation,” in *Robotics Science and Systems (RSS)*, 2009.
- [7] S. Bagon and M. Galun, “Large scale correlation clustering optimization,” *Arxiv preprint arXiv:1112.2903*, vol. abs/1112.2903, 2011.
- [8] N. Bansal, A. Blum, and S. Chawla, “Correlation clustering,” in *Machine Learning*, 2002.
- [9] A. Y. Ng, M. I. Jordan, and Y. Weiss, “On Spectral Clustering: Analysis and an algorithm,” in *Neural Information Processing (NIPS)*, 2001.
- [10] B. J. Frey and D. Dueck, “Clustering by passing messages between data points,” *Science*, vol. 315, p. 2007, 2007.
- [11] R. Triebel, J. Shin, and R. Siegwart, “Segmentation and unsupervised part-based discovery of repetitive objects,” in *Robotics Science and Systems (RSS)*, 2010.
- [12] J. B. MacQueen, “Some methods for classification and analysis of multivariate observations,” in *Berkeley Symposium on Mathematical Statistics and Probability*, vol. 233, 1967.
- [13] S. Vicente, C. Rother, and V. Kolmogorov, “Object cosegmentation,” in *Computer Vision and Pattern Recognition (CVPR)*, 2011.
- [14] L. Breiman, “Random Forests,” *Machine learning*, vol. 45, no. 1, 2001.
- [15] A. Collet, B. Xiong, C. Gurau, M. Hebert, and S. S. Srinivasa, “Exploiting domain knowledge for object discovery,” in *International Conference on Robotics and Automation (ICRA)*, 2013.
- [16] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, “Indoor segmentation and support inference from RGBD images,” in *European Conference on Computer Vision (ECCV)*, 2012.
- [17] A. Johnson and M. Hebert, “Using spin images for efficient object recognition in cluttered 3D scenes,” *Pattern Analysis and Machine Intelligence (PAMI)*, vol. 21, no. 5, 1999.
- [18] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, “Matching 3D models with shape distributions,” in *Shape Modeling International*, 2001.
- [19] H. Bostrom, “Estimating class probabilities in Random Forests,” in *Conference on Machine Learning and Applications*, 2007.
- [20] J. Besag, “On the statistical analysis of dirty pictures,” pp. 259–302, 1986.
- [21] L. Hubert and P. Arabie, “Comparing partitions,” *Journal of Classification*, vol. 2, 1985.