2011 IEEE/RSJ International Conference on
Intelligent Robots and Systems
September 25-30, 2011. San Francisco, CA, USA

RGB-D Object Discovery via Multi-Scene Analysis

Evan Herbst Xiaofeng Ren Dieter Fox

Abstract— **We introduce an algorithm for object discovery from RGB-D (color plus depth) data, building on recent progress in using RGB-D cameras for 3-D reconstruction. A set of 3-D maps are built from multiple visits to the same scene. We introduce a multi-scene MRF model to detect objects that moved between visits, combining shape, visibility, and color cues. We measure similarities between candidate objects using both 2-D and 3-D matching, and apply spectral clustering to infer object clusters from noisy links. Our approach can robustly detect objects and their motion between scenes even when objects are textureless or have the same shape as other objects.**

I. Introduction

Our goal is to enable robots to learn about objects in an unsupervised way using changes in the environment. For instance, a mobile robot that moves around a domestic or office environment often will see a large number of different configurations of the same movable objects at various times. Over time, such a robot should be able to discover and geometrically reconstruct all movable objects in the environment along with structural components such as walls. In addition to thus decreasing the human supervision involved in semantic mapping, unsupervised object discovery can improve the performance of supervised vision tasks that use the notion of objects (e.g. [20], [24], [18]). Object detection, recognition, segmentation, and hierarchy modeling all require labeled object models, the process of creating which can be at least partially automated.

As a first step toward this goal, we have introduced [14] a technique that uses RGB-D cameras to detect changes between two visits to a scene. It does this by building a 3-D map from the video of each visit, aligning two of these maps, and computing for each surface element in one scene the probability that it is also detected in the other scene. We generate hypothesized moved objects by segmentation of a graph over surface elements. In this work we go beyond [14] to develop a scalable multi-scene model that jointly and robustly reasons about moved objects. Detected object segments are compared using a combination of 3-D and 2-D matching techniques using an RGB-D version of ICP error metric as well as shape and visual descriptors. We use spectral clustering to group detected segments together from noisy pairwise similarities and discover objects as clusters. Combining color and depth, our approach can robustly detect objects and their motion between scenes even when objects are textureless or have non-distinctive shapes.

After discussing related work, we develop the steps of our algorithm in section III. Experimental results are provided in section IV, followed by conclusions.

II. Related Work

Previous work on representing the static and dynamic parts of maps separately has been restricted to 2-D, whether using color or depth sensors, and has generally avoided the problem of segmentation. Biswas et al. [3], working with planar laser scans, assume a known number of objects and segment objects by connected components of occupied grid cells. Haehnel et al. [12] run mapping with 2-D laser scans, then approximate each 3-D object as the set of 3-D measurements within a 2-D bounding box. Konolige et al. [17] represent a map as a set of views (images) from a color camera, and cluster views by location. Using their methods, it is possible to represent objects indirectly by partitioning each location's set of views according to similarity, but they don't explicitly model objects or segment images, as object-level representation is not one of their goals.

Usually computer vision work using objects acts at one of two levels: instance and category. Object instances are typically modeled with point features and their geometric consistency (enforced with RANSAC [10]). Another approach is cosegmentation, which jointly finds regions in multiple images corresponding to the same object. Joulin et al. [15] cosegment by maximizing the margin between object and non-object pixels over a set of images. They have particular trouble when images have similar backgrounds. We avoid this issue by using motion for segmentation; we *require* that different scenes have the same background.

By contrast, there has been a wealth of work on object *category* discovery. Grauman and Darrell [11] assign each image a category by performing spectral clustering with a spatially aware kernel over sets of local features. Russell et al. [23] extract a "soup of segments", model each as a histogram of visual words, and perform topic discovery on the set of segments. Graph mining techniques have also been used: for example, Kim et al. [16] create a graph among local features, with edges specified by feature matching.

Working with depth data, Ruhnke et al. [22] align object views by using correspondences between local depth-map features, without color information. Their keypoints are corners in the depth map; they assume the shape of the edges of the 2.5-D object views is enough to distinguish both the object identity and the location of each point on the object. They too ignore the problem of segmentation, using

E. Herbst and D. Fox are with the University of Washington, Department of Computer Science & Engineering, Seattle, WA 98195. X. Ren and D. Fox are with Intel Labs Seattle, Seattle, WA 98105.

This work was funded in part by an Intel grant, by the NSF under contract IIS-0812671, and through the Robotics Consortium sponsored by the U.S. Army Research Laboratory under Cooperative Agreement W911NF-10-2-0016.

connected components on point clouds to produce candidate objects. Shin et al. [25] instead do bottom-up segmentation in a single scene. They discover objects using depth but not color, using ICP initialized by matching spin images. They do not demonstrate discovery of small-scale objects such as our dense scene reconstruction allows us to find. In [27] they extend their method to use co-occurrences of matching segments to discover multi-segment objects.

There has also been work on instance discovery from video. Liu and Chen [19] combine visual topic modeling with a motion model for the foreground segment to track a small moving object. Southey and Little [26] find multiple objects and don't assume they appear in all frames. They get sparse depth points from passive stereo and approximate motion from sparse optical flow. Rigid motions are fit to each frame pair using local features. They oversegment each frame and have segments vote for motions. In both cases, motion within a single video is used for segmentation; we use motion between multiple videos for segmentation and use the frames of each video to provide spatial consistency.

At a high level, the algorithms of [22] and this paper are both generate-and-test algorithms of the sort also used for 3-D alignment (e.g. [1]), and so consist of the same four steps as outlined in table I. We use scene differencing as much as possible, and avoid assuming there are enough distinctive point features for feature matching. Additional differences are our use of color information, our multiscene optimization for segmentation, and our fuller experiments.

Step	Ruhnke et al. detail	This paper detail
1. Segmentation	connected components	scene differencing
2. Hypothesis generation	create 2-D views; 2-D feature matching	ICP with random restarts
3. Hypothesis scoring	ICP error	scene differencing
4. Clustering	repeated Olson [21]	n-cuts

TABLE I: High-level structure of this paper and [22].

III. Multi-Scene Analysis

In sec. III-A, we describe how to reconstruct 3-D scenes from video. In sec. III-B, we show how to use motion cues to find possible objects. We calculate pairwise similarities, and use them for clustering 3-D segments, in sec. III-C.

A. 3-D Reconstruction of Single Scenes

Given a set of scene visits in the form of RGB-D video sequences, we run a modification of RGB-D Mapping [13] on each visit to produce the set of camera poses corresponding to all video frames. RGB-D Mapping consists of 1) visual odometry using SIFT features assigned to 3-D points, 2) loop closure detection, also using SIFT with 3-D locations, 3) pose-graph optimization over contiguous-frame and loop-closure edges, and 4) bundle adjustment over the whole map. The RANSAC variant we use in steps 1 and 2 optimizes reprojection error rather than 3-D error as in [13]. Another change we have made to the algorithm is to run bundle adjustment over each contiguous-frame pair following RANSAC in the visual odometry step.

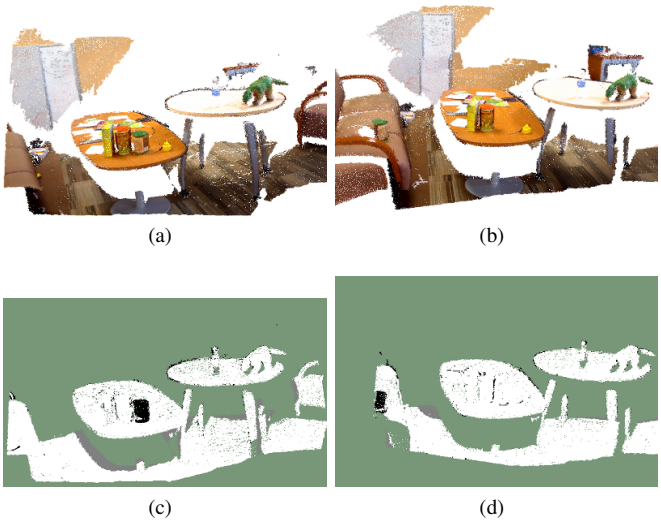


Fig. 1: (a), (b) two scenes and (c), (d) the result of differencing each against the other. Color gives pointwise probability of change, with white at zero and black at one. We assign high change probability to the object removed from the table in the left scene and the new object placed on the sofa in the right scene.

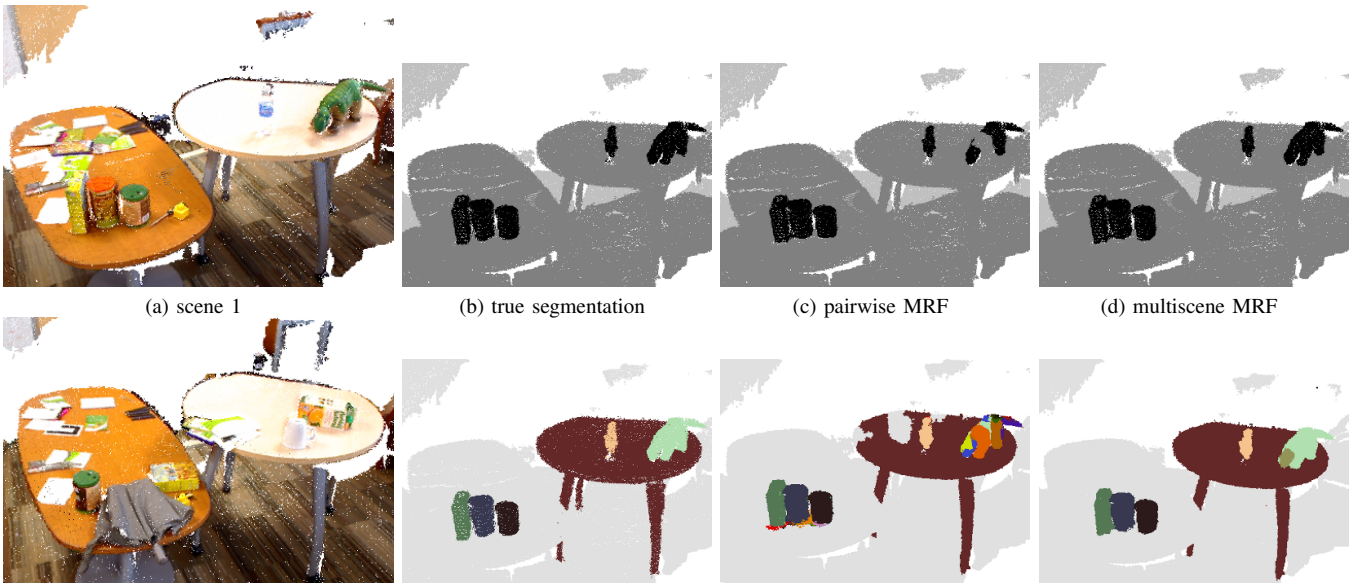
As in [13], we represent surfaces using *surfels*, small planar patches with color and orientation information. We make use of uncertainty in sensor readings by fitting Gaussian models for the position, color and orientation at each surfel, as we find that this improves the performance of scene differencing. We globally align all scenes using visual point feature matching followed by ICP, as in [14].

B. Multi-Scene Segmentation

Pairwise segmentation: In a previous work [14], we have introduced an algorithm to use RGB-D sensor readings to probabilistically decide which surfels in a scene are unchanged in a second scene. We run inference in a Markov random field (MRF) over these probabilistic judgments to create a binary (moved vs. not-moved) segmentation of each scene. We use this pairwise scene differencing algorithm to segment each visit to a scene. Fig. 1 shows a typical result of the probabilistic stage.

Multi-scene MRF: A straightforward extension of this approach to multiple scenes would be to compute a segmentation for each pair of scenes. As a result, we would get a so-called soup of segments for each scene, since the pairwise segmentations are not necessarily consistent, due largely to sensor noise effects differing widely between scenes. Furthermore, since we want to use our segments to reconstruct 3-D objects, we prefer to have only one segment per object per scene to simplify the decision of which surfels to include in each reconstructed object.

To obtain a single, maximally consistent segmentation per scene, we combine information from differencing between all scene pairs into a single labeling process. Instead of labeling a surfel as 0 or 1 (moved or not moved) for each pair of scenes, we generate a joint label vector $l \in [0,1]^{n-1}$



(a) scene 1 (b) true segmentation (c) pairwise MRF (d) multiscene MRF

(e) scene 2 (f) true object labels (g) pairwise MRF (h) multiscene MRF

Fig. 2: (a), (e) two scenes. (b) Ground truth for which surfels in the first scene move with respect to the second (black = moved; medium gray = not moved; light gray = not visible in (e)). Results of (c) pairwise and (d) multiscene differencing MRFs, showing which surfels of (a) are estimated moved/not moved in (e). The multiscene MRF uses information from a total of 10 scenes, including the two shown here. The pairwise MRF misses the shoulders of the dino (middle right of scene); the multiscene MRF does not. (f) The ground-truth object labelings for scene (a), with each object a different color. (g) Scene segmentation obtained by assigning each surfel a bit-vector label put together from the results of the two-scene MRF of [14]. (h) Scene segmentation output by our multiscene MRF, which is much more accurate than that in (g). The table is discovered as an object due to moving in one of our scenes. This figure (in particular the last row) is much better viewed in color.

for each surfel in scene i. The jth bit of this label vector represents whether the surfel changed between scenes i and j. To jointly reason about all surfel labels in a scene, we perform spatial regularization using a Markov random field (MRF). Each node corresponds to a surfel and is a random variable ranging over the values of that surfel's label vector. Connections between neighbors increase spatial consistency of label vectors by providing soft constraints on the transitions between labels.

Our data energy for label l at surfel s of scene i is

$$D(s,l) = \sum_{j \neq i} \begin{cases} p(s \text{ moved in } j), & v_j(s) \text{ and } l_j = 0 \\ 1 - p(s \text{ moved in } j), & v_j(s) \text{ and } l_j = 1 \\ 0 & \neg v_j(s) \end{cases},$$

where j is a scene, l_j is the jth bit of l, and $v_j(s)$ is true iff surfel s was visible in any frame observed in scene j (v stands for "visible"; surfels in i that are never seen in j contain no evidence for having moved or not, represented by a zero data energy term, or cost). $p(s \text{ moved in } j)$ is the results of pairwise scene differencing.

The MRF smoothness energy imposed on neighboring surfels is the Potts model (energy is 1 if labels differ, 0 if not). This smoothness term on the joint label l links together all the scenes: neighboring surfels typically correspond to the same object and thus should move consistently w.r.t. to any other scene. This energy is weighted by the local 3-D curvature of the scene to discourage segment boundaries from being placed in low-curvature areas. While such a weighting is not perfect (for example, papers on a table can move around without causing high-curvature edges), it gives good results in our test scenes.

We use graph cuts to perform MAP inference in the MRF. We then take each connected component of surfels with the same label to be a possible object. For a set of n scenes, one run of inference in this multiscene MRF makes use of the same information as n runs of inference in two-scene MRFs. An example can be found in Fig. 2.

While in this work we use only motion as a cue for segmentation, other cues such as shape and color priors could be included in our framework. In principle, our multiscene MRF can combine any number of binary segmentations, probabilistic or not, by including them as entries in the bit vector that we use as a label.

Efficient Implementation: The label set for each surfel in a scene has size exponential in the number of scenes. For 10 scenes of moderate size we are able to run inference in under four hours. However, a good way to speed inference is to reduce the label set size, as graph-cut inference (using the expansion-move algorithm of [6]) has complexity linear in the number of labels. Fortunately, the label vectors we create from pairwise differencing results provide a good estimate for which surfel moves we expect between a scene and any other. Suppose only two objects in scene 5 move, one in scenes 7 and 13 and the other in scenes 3, 7 and 10. Then for scene 5 we expect to see many label vectors with 1s in the 3rd, 7th, 10th, and/or 13th components and 0s everywhere

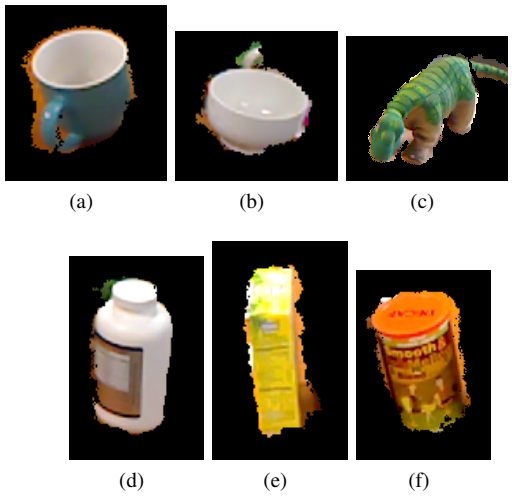


Fig. 3: examples of video frames masked by the projections of 3-D segments into camera coordinates. These are not entire frames, but cutouts of width about 100 pixels. We compute 2-D features over image regions like these.

else. Therefore we discard label vectors that occur rarely in pairwise results. Specifically, we sort label vectors of all surfels in a scene by their frequency (estimated from pairwise differencing) and only instantiate the most frequent labels such that at least 95% of surfels are represented. Fig. 4 shows the dramatic speed improvement we get via this heuristic. Empirically we see virtually no difference between the results with full and approximate label sets.

Scalability might still be a problem for very large scene sets. However, we believe that object discovery will still be possible by subsampling which scenes are considered jointly or by maintaining a canonical "background" scene that contains only static objects, and differencing each new scene only against the current background scene.

C. Object Discovery

In the computer vision community, object discovery is the process of partitioning a set of views (in our case a view is a segment extracted from a single scene) such that those in the same group represent the same object.

One unique aspect of our work is that we have access to both 2-D appearance (one RGB-D view in each video frame) and 3-D shape (reconstructed by merging a video sequence). We can thus match observations (segments from the multi-scene MRF) using both 2-D view-based matching and 3-D alignment-based matching.

2-D Matching: Fig. 3 shows examples of 2-D *views* extracted from the segmented 3-D scenes. To compare 3-D segments using these views, we extract features over entire segments, rather than matching point features calculated at interest points (such as in [8]). We have two reasons for this: the image regions corresponding to objects are generally small and have few features and even fewer reliable ones (due to region-edge effects), and some of our objects are textureless and have no reliable point features.

We use efficient match kernel (EMK) features [5] calculated over sets of kernel descriptors [4] to represent our segments. Kernel descriptors are a generalization of histogram-like features such as SIFT and achieve leading accuracies on image category recognition[4]. We have both color and depth information available, so we calculate gradient kernel descriptors over both color and depth, and color kernel descriptors over color. (These three descriptors had the highest learned importance in our preliminary experiments.) We calculate features over regions defined by projecting 3-D segments into each camera's frame rather than over the whole frame. For EMK we use a three-level pyramid and five hundred visual words.

For two 3-D segments S_1 and S_2, we determine for each of S_1's frames the closest frame in S_2 using logistic regression, with L_1 distances between EMK vectors as features. We combine these frame-to-frame match scores into a single score by taking the 90th-percentile score. This is essentially the best score over all frame pairs, but allowing for a few high-scoring matches due to chance.

3-D Matching: We also consider matching the 3-D segments we get from the segmentation step. We run an RGB-D variant of iterated closest points (ICP) n times for each pair of point clouds, initializing by setting the centroids equal and choosing a rotation uniformly at random. For speed we use point-to-point error [2] rather than the point-to-plane error of [7]. Color information is used by requiring each point to correspond to a point whose color is in the same hue-saturation bin (we use 4 bins in hue, 4 in saturation). We can score the result of each ICP run by taking the average 3-D error over points if at least half the points have correspondences, or a very large number if not. (If ICP is unable to match a majority of points, its error will not be a reliable measure of similarity.) Another error measure, designed to be more accurate, is an adaptation of the output of scene differencing [14]. We compute the probability of change at each point x in S_1 and associate each $x \in S_1$ with a point $c(x) \in S_2$ as in ICP, such that some points have no correspondences. The score $J(X)$ for the alignment X is then the sum of the pointwise change probabilities weighted depending on correspondence:

$$J(X) = \alpha \sum_{x|c(x)\in S_2} p(x) + \sum_{x|c(x)=\perp} p(x),$$

where $\alpha = 10$ works well. Using the scene differencing score significantly improves the reliability of the ICP matching.

Spectral Clustering: The pairwise similarities computed from 2-D and 3-D matching can be very noisy and inconsistent, partly due to low-resolution data, partial occlusions of objects, and errors in differencing and segmentation. Clustering is needed to enforce transitivity in the same-object relationship between segments.

We use *multiclass spectral clustering* [28], which minimizes the K-class Normalized Cuts criterion

$$ncuts_K(\{V_k\}) = \frac{1}{K}\sum_{t=1}^{K} \frac{\text{link}(V_t, V \backslash V_t)}{\text{degree}(V_t)},$$

where the total set V is partitioned into K subsets $\{V_k\}$, link is the sum of edge weights between two sets, and degree is

the set of edge weights within a set. A discrete solution to the K-class Normalized Cuts can be found efficiently by finding rotations in the eigenvector space using singular value decomposition [28]. We find that spectral clustering overcomes many sources of noise in the pairwise links and discovers challenging objects with large occlusions and without distinctive texture or shape (see examples in Figure 7).

IV. Evaluation

The camera was carried by a human. We test our algorithm on two datasets, one (A) containing 10 scenes and one (B) containing 9. The two have different sets of moved objects: B has more textureless objects than A, and more objects overall (15 versus 8). The average scene length is 140 frames for A and 230 frames for B, at 10 Hz. To show the power of our approach, we included similarly colored and shaped objects. For example, B contains three mugs and four bowls, none with much texture and all with white/blue color schemes.

A. Segmentation

To evaluate our segmentation algorithm, we manually labeled objects in each scene and marked which objects moved between each pair of scenes. We labeled only dataset A due to time constraints. We compare the two-scene "pairwise" MRF of [14] to our "multiscene" technique. For space reasons table II includes only aggregate statistics over surfels in the 90 scene pairs in dataset A. Most surfels in any scene pair do not move; the "always-not-moved" classifier attains over 97% accuracy. As can be seen, multiscene reduces the error by over 25% and increases both precision and recall (calculated w.r.t. surfels that truly moved) noticeably. (Over our dataset, this change in precision translates to 320,000 false positives avoided.) An example of the improvement we see from the multiscene MRF is given in Fig. 2; five other scene pairs in dataset A show similar improvement, and for no scene pair is multiscene numerically or visually significantly worse than pairwise.

algorithm	# surfels	accuracy	precision	recall	FP	FN
pairwise	38.7M	0.991	0.971	0.901	1.13M	3.84M
multiscene	38.7M	0.993	0.979	0.927	0.81M	2.83M

TABLE II: performance statistics for pairwise and multiscene differencing MRFs over the set of all scene pairs in a dataset. FP and FN are false-positive and false-negative counts. We reduce FP by 28% and FN by 26%.

We also demonstrate the scalability of our multi-scene formulation. The graph-cut inference we use does not require that potentials be stored in memory, which means that inference can run in a small amount of memory even with exponentially many labels, and that the time to set up inference varies very little as a function of the number of labels. Fig. 4 shows the decrease in label set size we get via the heuristic discussed in sec. III-B as well as the resulting inference runtime for scene 1 of dataset A. Inference setup time is about 21 seconds (not included in the plot).

After the segmentation step, we manually removed segments that were not "real" objects, such as table legs and pieces of floor, as well as segments containing (parts of) multiple "real" objects, in order to get better differentiation among methods on relatively clean data. This left us with 48 and 64 segments for datasets A and B, respectively, out of 116 and 183 segments total.

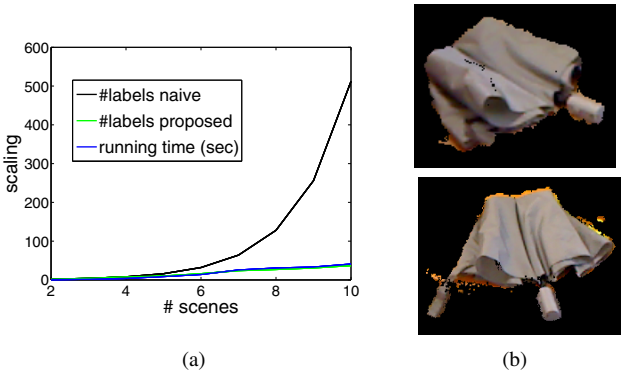


Fig. 4: (a) scalability of multiscene differencing with number of scenes. The black curve gives the number of labels used by the naive algorithm. The green curve is the number of labels we use. The blue curve (right on top of the green) gives runtime in seconds. All have the same y-axis scale. (b) An example of a segment pair that 2-D features have trouble matching. The gradient distributions of the two 2-D segments are very different because the views are from different orientations. The object is an umbrella. These are crops of widths 140 and 228 pixels.

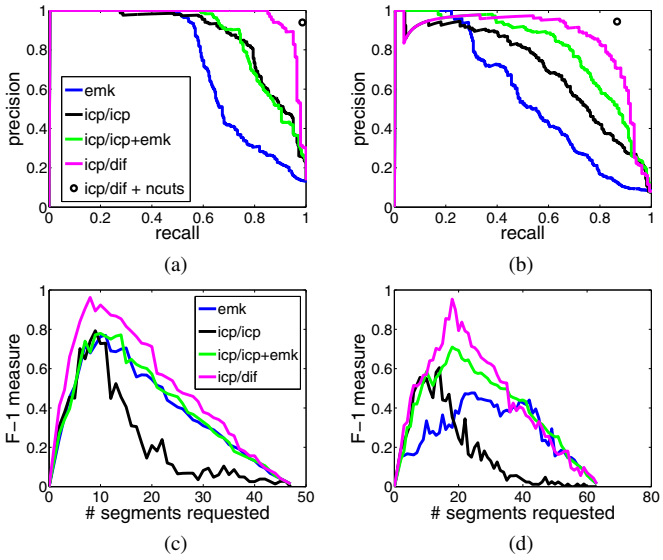


Fig. 5: (a), (b) precision-recall curves for pairwise matching on datasets A and B respectively. Blue: 2-D visual-feature-based matching; black: ICP with ICP-error scoring; green: combined visual features and ICP; magenta: ICP with differencing-based scoring; black circle: best result of spectral clustering on differencing-based scores. (c), (d) performance of spectral clustering on the two datasets (same color scheme). The black circles in (a), (b) come from the magenta curves here. F-measure is the harmonic mean of precision and recall. True numbers of clusters are 8 and 15.

(a) (b) (c)

Fig. 6: segments from different objects (two white bowls) that differencing thinks very likely to be the same object: (a), (b) images of the bowl segments; (c) the point cloud alignment of the two segments (upside down w.r.t. the other images).

B. Pairwise Segment Matching

Although our goal is clustering segments, we can also evaluate the intermediate step of pairwise segment matching. Each of our matching methods gives us a score for each pair of segments. We evaluate by taking scores over a threshold to specify true matches. We can thus calculate precision and recall of a score over the set of all (ordered) segment pairs in a dataset for each of a range of thresholds, and we do so in Fig. 5. Since the 2-D and 3-D alignment methods described in sec. III-C use very different sources of information, we expect the combination of their scores to perform better than either alone. As Fig. 5 shows, this is the case for both of our datasets when we use ICP-based scoring as the 3-D component. (To get a combined score for a segment pair, we normalize both score distributions by mean and standard deviation, then take a weighted average. For Fig. 5 we weight the ICP score by 30.) Combining the differencing-based score with the EMK score does not improve on differencing-only scoring, so we don't show it.

When we developed the differencing model we hoped to build the whole object discovery pipeline on it: segmentation, registration and matching. The fact that differencing-based transform scoring performs so well suggests that we can further improve matching performance by incorporating differencing in the registration step. We would like to optimize a differencing-based objective rather than the ICP error measure, and in a way that correctly deals with occlusion, rather than in a suboptimal way such as simply changing the objective in the optimization stage of each ICP iteration.

2-D and 3-D scoring methods make many of the same mistakes: false negatives happen for segment pairs with very different colors or that cover different parts of an object, and false positives happen for textureless objects of similar shape and for textured objects of similar color. Visual features make mistakes that ICP doesn't on segments containing different parts of an object—e.g. the two umbrellas in Fig. 4—because of the difference in shape. ICP makes mistakes that EMKs don't on examples that have different 3-D shape but similar 2-D shape. Differencing makes many of the same mistakes ICP scoring does, but fewer. This makes sense: differencing error is more or less ICP error with better handling of color, surface orientation and occlusion. An example of a false positive is shown in Fig. 6.

For the 64 segments in dataset B, it takes about 150 minutes to compute kernel descriptors and EMK features and 195 minutes to run matching (much of which is reading feature vectors from disk). ICP matching runs about 150 minutes (which depends linearly on the number of random ICP restarts), and scene differencing is another 75.

C. Object Discovery

Since all the similarities we calculate in sec. III-C are nonsymmetric, we have a choice of how to combine them when creating a similarity matrix. Given the scores for segment s against segment t and for t against s, we can set the similarity matrix entry to (the exponential of), for example, the mean, min or max of the two. These kernels all perform similarly when used with Normalized Cuts for our MRF inference, suggesting that while our similarities are not strictly symmetric, they are very close. In Fig. 5 we compare the performance of Normalized Cuts using the score from 2-D feature matching alone, the score from ICP alone and the combined score. Across the entire range of possible cluster counts for both datasets, combining the scores never decreases performance. Using the differencing-based score further improves clustering quality across all cluster counts. In Fig. 7 we show the clusters produced for dataset B with number of desired clusters set to 20 (results for other cluster numbers are similar). Even most of the textureless segments are correctly clustered, which could not have been accomplished using matching of 2-D features only. Furthermore, the approach is able to distinguish the different coffee mugs based on only slight color and shape differences.

In Table III we show the comparison of two leading spectral clustering algorithms, the Multiclass Normalized Cuts [28] and the Self-tuning Spectral Clustering [29]. We use the 48 segments in dataset A and manually label all pairs of segments as being the same object or not. Algorithms are evaluated using precision and recall of the binary labels on segment pairs. We vary the number of clusters from 2 to 48 and report the results with the best F-measures.

algorithm	precision	recall	F-measure
multiclass Ncuts [28]	0.939	0.986	0.962
self-tuning spectral clustering [29]	0.538	0.837	0.655

TABLE III: comparison of spectral clustering algorithms. Precision-recall values and F-measures are computed using ground-truth same-object and different-object relations between segments. Normalized Cuts performs the best in all our experiments.

V. Conclusions

We have introduced an RGB-D object discovery system using multiple visits to a scene and shown that 3-D shape and motion information improve both segmentation and matching of 3-D segments. Our approach jointly analyzes multiple scenes to consistently segment objects that move between scenes. Segments are then combined into objects using spectral clustering. Our experiments show that by combining shape and color descriptors, our approach can robustly detect objects and their motion between scenes, even when objects are textureless or have virtually the same shape as other objects, such as bowls or coffee mugs.

Fig. 7: Object clusters produced for dataset B with number of desired clusters set to 20. Incorrect cluster members are outlined in red, and object IDs are given in small text. Clusters of size > 1 are left of the line; singletons are to the right. (All three singletons are segments of the same object, a water bottle that the depth camera has trouble seeing due to translucency and refraction. The approach is able to distinguish between objects of very similar shape and appearance: coffee mugs 3,8, and 10, and bowls 9, 12, and 14. While the EMK features are crucial to distinguish between individual mugs and bowls, 3D shape information via ICP is important to distinguish across these two classes. Some of the objects (2,3) are oversegmented into multiple clusters.

REFERENCES

[1] D. Aiger, N. Mitra, and D. Cohen-Or. 4-points congruent sets for robust pairwise surface registration. In *ACM SIGGRAPH*, 2008.

[2] P. Besl and N. McKay. A method for registration of 3-d shapes. *IEEE PAMI*, 1992.

[3] R. Biswas, B. Limketkai, S. Sanner, and S. Thrun. Towards object mapping in dynamic environments with mobile robots. In *IEEE IROS*, 2002.

[4] L. Bo, X. Ren, and D. Fox. Kernel descriptors for visual recognition. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

[5] L. Bo and C. Sminchisescu. Efficient match kernels between sets of features for visual recognition. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

[6] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE PAMI*, 2001.

[7] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. *Image and Vision Computing*, 1992.

[8] A. Collet Romea, D. Berenson, S. Srinivasa, and D. Ferguson. Object recognition and full pose registration from a single image for robotic manipulation. In *IEEE ICRA*, 2009.

[9] I. Dhillon, Y. Guan, and B. Kulis. Kernel k-means, spectral clustering and normalized cuts. In *ACM SIGKDD*, 2004.

[10] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Communications of the ACM*, 1981.

[11] K. Grauman and T. Darrell. Unsupervised learning of categories from sets of partially matching image features. In *IEEE CVPR*, 2006.

[12] D. Haehnel, R. Triebel, W. Burgard, and S. Thrun. Map building with mobile robots in dynamic environments. In *IEEE ICRA*, 2003.

[13] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgb-d mapping: Using depth cameras for dense 3-d modeling of indoor environments. In *International Symposium on Experimental Robotics (ISER)*, 2010.

[14] E. Herbst, P. Henry, X. Ren, and D. Fox. Toward object discovery and modeling via 3-d scene comparison. In *IEEE ICRA*, 2011.

[15] A. Joulin, F. Bach, and J. Ponce. Discriminative clustering for image co-segmentation. In *IEEE CVPR*, 2010.

[16] G. Kim, C. Faloutsos, and M. Hebert. Unsupervised modeling of object categories using link analysis techniques. In *IEEE CVPR*, 2008.

[17] K. Konolige and J. Bowman. Towards life-long visual maps. In *IEEE IROS*, 2009.

[18] K. Lai and D. Fox. Object recognition in 3D point clouds using web data and domain adaptation. *International Journal of Robotics Research (IJRR)*, 29(8), 2010.

[19] D. Liu and T. Chen. A topic-motion model for unsupervised video object discovery. In *IEEE CVPR*, 2007.

[20] T. Malisiewicz and A. Efros. Improving spatial support for objects via multiple segmentations. In *Proc. of the British Machine Vision Conference*, 2007.

[21] E. Olson, M. Walter, S. Teller, and J. Leonard. Single-cluster spectral graph partitioning for robotics applications. In *Robotics: Science and Systems (RSS)*, 2005.

[22] M. Ruhnke, B. Steder, G. Grisetti, and W. Burgard. Unsupervised learning of 3d object models from partial views. In *IEEE ICRA*, 2009.

[23] B. Russell, A. Efros, J. Sivic, W. Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *IEEE CVPR*, 2006.

[24] B. Sapp, A. Saxena, and A. Ng. A fast data collection and augmentation procedure for object recognition. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, 2008.

[25] J. Shin, R. Triebel, and R. Siegwart. Unsupervised discovery of repetitive objects. In *IEEE ICRA*, 2010.

[26] T. Southey and J. Little. Object discovery through motion, appearance and shape. In *AAAI Workshop on Cognitive Robotics*, 2006.

[27] R. Triebel, J. Shin, and R. Siegwart. Segmentation and unsupervised part-based discovery of repetitive objects. In *Robotics: Science and Systems (RSS)*, 2010.

[28] Stella X. Yu and Jianbo Shi. Multiclass spectral clustering. In *International Conference on Computer Vision (ICCV)*, 2003.

[29] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems (NIPS)*, 2004.