Autoscanning for Coupled Scene Reconstruction and Proactive Object Analysis

Kai Xu[1,2] Hui Huang[1*] Yifei Shi[2] Hao Li[3] Pinxin Long[1] Jianong Caichen[2,1] Wei Sun[1] Baoquan Chen[3*]
[1]Shenzhen VisuCA Key Lab / SIAT [2]HPCL, National University of Defense Technology [3]Shandong University

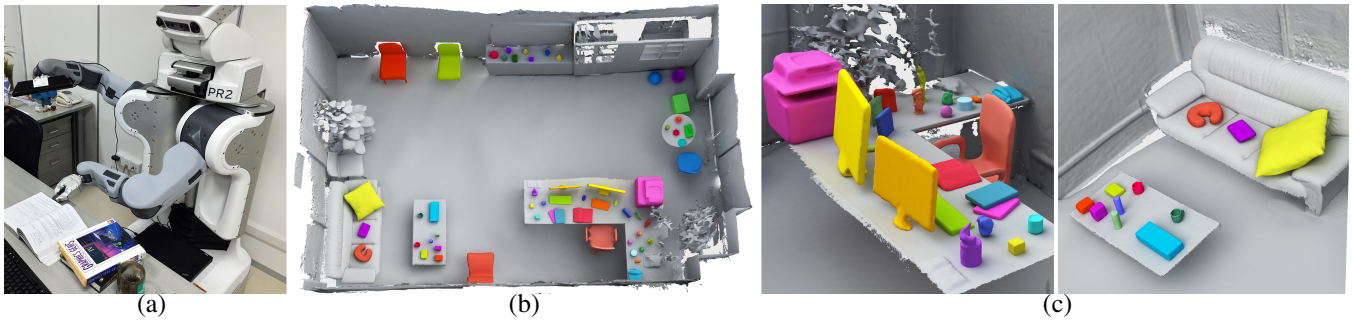


Figure 1: *Autonomous scene scanning and reconstruction with object analysis aided by robot pushing. (a): A PR2 robot with one arm equipped with a depth camera interacts with a cluttered table-top scene, to scan and extract the objects on top of it. (b): The reconstructed scene with extracted individual objects shown with distinct colors. (c): Zoomed-in views of the room corners.*

Abstract

Detailed scanning of indoor scenes is tedious for humans. We propose autonomous scene scanning by a robot to relieve humans from such a laborious task. In an autonomous setting, detailed scene acquisition is inevitably *coupled* with scene analysis at the required level of detail. We develop a framework for object-level scene reconstruction coupled with object-centric scene analysis. As a result, the autoscanning and reconstruction will be *object-aware*, guided by the object analysis. The analysis is, in turn, gradually improved with progressively increased object-wise data fidelity. In realizing such a framework, we drive the robot to execute an iterative *analyze-and-validate* algorithm which interleaves between object analysis and guided validations.

The object analysis incorporates online learning into a robust graph-cut based segmentation framework, achieving a global update of object-level segmentation based on the knowledge gained from robot-operated local validation. Based on the current analysis, the robot performs *proactive* validation over the scene with physical push and scan refinement, aiming at reducing the uncertainty of both object-level segmentation and object-wise reconstruction. We propose a joint entropy to measure such uncertainty based on segmentation confidence and reconstruction quality, and formulate the selection of validation actions as a maximum information gain problem. The output of our system is a reconstructed scene with both object extraction and object-wise geometry fidelity.

Links: DL PDF WEB VIDEO CODE

*Corresponding authors: {hhzhiyan,baoquan.chen}@gmail.com

1 Introduction

With an increasing demand for digitized large-scale indoor scenes, considerable research effort has been directed at improving the scalability and accuracy of scene reconstruction algorithms. In typical scene acquisition scenarios, a human operator walks through a scene with a hand-held depth camera to capture scene geometry. The operator's scanning effort is assisted by instant visual feedback from realtime reconstruction [Newcombe et al. 2011]. However, detailed scene scanning by humans is laborious, especially for large indoor scenes containing numerous objects. An attractive substitution is autonomous scanning, or *autoscanning*, where a mobile robot holding a depth camera replaces humans in the laborious scanning task. Consequently, the scanning process becomes more iterative and exploratory, while under higher-precision controls.

While enjoying the mobility, accuracy, and endurance of a robot, a significant challenge in autoscanning is how to make the robot exercise intelligence in executing its task. To enable this, some level of understanding of the scene being acquired is indispensable. Thus, scene acquisition is inevitably tied to *scene analysis*. For example, if the robot is to focus its acquisition effort on delineating objects, it needs to have some awareness of the object composition in the scene. Such understanding offers essential guidance to a well-targeted and efficient autoscanning. In return, online analysis will benefit from a more complete and accurate reconstruction from higher-quality scanning [Zhang et al. 2014]. Existing work usually solves scene reconstruction and analysis as separate problems. To achieve an intelligent autoscanning for quality and detailed scene reconstruction, we propose solving the two problems in a coupled manner, with an iterative feedback loop between the two processes.

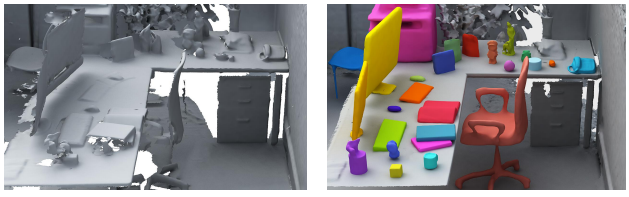


Figure 2: *Visual comparison of robot-operated scene reconstruction without (left) and with (right) our object-aware framework.*

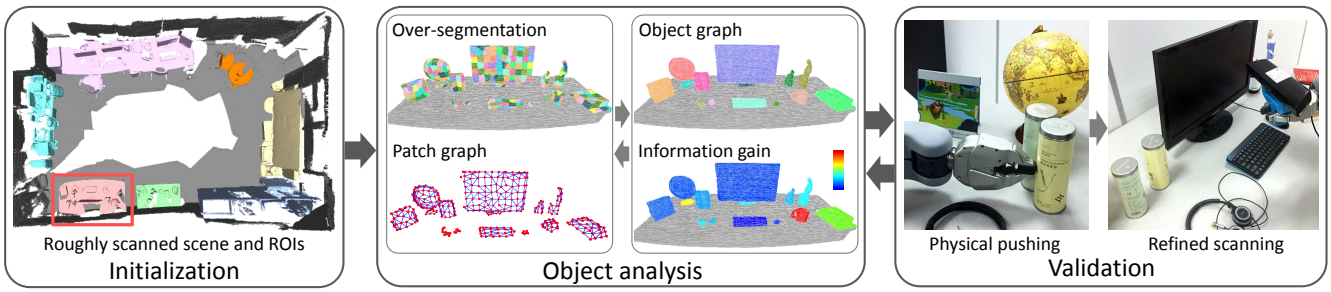


Figure 3: *An overview of the system pipeline.*

In developing our framework for coupled indoor scene reconstruction and analysis, we focus on object extraction, arguably the most fundamental scene analysis task. The autoscanning and reconstruction are guided by object-centric analysis. Meanwhile, the analysis is gradually improved with progressively increased data completeness. The final result is not a holistic reconstruction, as a single 3D model, but *object-aware scene reconstruction* encompassing both object-level segmentation and object-wise quality reconstruction; see Figure 1, as well as a visual comparison in Figure 2. To enable such an approach, the robot executes an iterative *analyze-and-validate* algorithm. The algorithm interleaves between scene analysis for extracting objects and robot conducted validation for improving the segmentation and object-aware reconstruction.

The basic system setup is a mobile robot holding a depth camera performing real-time scene reconstruction [Nießner et al. 2013]. With the current reconstruction, we perform object analysis to segment the scene into a set of hypothetical objects. We then estimate the joint uncertainty in both object-level segmentation and object-wise reconstruction, which is used to guide the robot in validating the object-aware reconstruction through physical push and scan refinement. Physical push is intended to verify the local segmentation, and assist data acquisition through separating close-by objects to reduce occlusion (Figure 1(a)). Scan refinement is then devoted to cast more scans around the moved objects to improve their data completeness. Both validations aim to reduce the joint uncertainty as much as possible. The gained knowledge about segmentation, as well as the newly acquired depth data, is incorporated into the current object-aware reconstruction, which would in turn improve the analysis in the next round. The whole process repeats until the overall uncertainty does not decrease significantly.

In summary, our main contributions include:

- A framework for coupled scene reconstruction and object analysis based on an analyze-and-validate algorithm, leading to object-aware scene reconstruction.
- An object analysis method integrating a robust graph-cut based segmentation with online cut cost learning, enabling a global update of the segmentation based on the knowledge gained from local proactive validation.
- A joint entropy measure of both object-level segmentation and object-wise reconstruction, which is, to the best of our knowledge, the first attempt to integrate segmentation confidence and reconstruction quality within a unified entropy-based formulation, for robot action selection.

Our work focuses on household objects which are rigidly movable over a supporting plane and separable against its surroundings under a moderate movement. Our method is invasive so it does not reconstruct the original scene with exact object positions. This is acceptable for real-life indoor scenes since the *exact* positioning of movable objects is usually quite casual and accidental. In this sense, our method produces a good approximation to the ground truth scene since the small amount of movement of objects would not change their mutual relations characteristically.

2 Related work

Scene reconstruction. With the maturation of the techniques of surface reconstruction from the point cloud of a single object [Berger et al. 2014], the interest on 3D scanning and reconstruction has been shifting toward scenes, especially indoor scenes. The emergence of commodity depth cameras, such as Microsoft Kinect and Asus Xtion, together with the recent progress on the depth map SLAM (Simultaneous Localization and Mapping) technique [Curless and Levoy 1996; Newcombe et al. 2011], have made real-time scene scanning and reconstruction very popular (e.g., [Roth and Vona 2012; Whelan et al. 2012; Chen et al. 2013a; Nießner et al. 2013]). Common to these methods is the holistic reconstruction which results in a single 3D model for the entire scene.

An indoor scene is characterized by the geometry of all objects contained in the scene, as well as their spatial relations [Fisher et al. 2011]. The raw reconstruction results above, however, lose the structural information and are hardly usable in subsequent applications, such as retrieval [Fisher et al. 2011] and synthesis [Fisher et al. 2012]. Therefore, a more useful scene reconstruction should extract and reconstruct the individual objects and infer their mutual relations, to reveal the composition and structure of the scene.

Scene analysis. Indoor scenes are ubiquitous in graphics and vision applications such as virtual reality, gaming, and robot exploration. Scene understanding based on images is a long-standing problem in computer vision (e.g., [Hedau et al. 2010]). With the availability of depth cameras, depth maps can greatly enhance scene analysis [Silberman et al. 2012; Gupta et al. 2013]. In computer graphics, most existing scene analysis techniques use off-the-shelf 3D models with object level segmentation and/or semantic tags, where the emphasis has been on the structural and contextual relationship between objects [Fisher et al. 2011; Liu et al. 2014].

Our work is more related to scene analysis with a *given* scanned point cloud, for which a widely adopted approach is to utilize a 3D model database to assist object extraction and recognition [Nan et al. 2012; Shao et al. 2012; Chen et al. 2014; Li et al. 2015]. Some works exploit the object repetition cues in indoor scenes [Kim et al. 2012; Mattausch et al. 2014]. Zhou and Koltun [2013] perform offline analysis of recorded sensor trajectories to globally distribute registration error in SLAM. Another notable line of research is to facilitate semantic scene analysis using humans action data [Jiang and Saxena 2013; Savva et al. 2014]. Since post-scanning analysis can not acquire additional information on scene structure, offline scene analysis has to rely on prior knowledge (prescribed by human or learned from data) or extra information (e.g. sensor trajectory) recorded during scanning. Our *online analysis* is enhanced by powerful *proactive validation* operated by a robot.

Online analysis during capturing. For real-time scene reconstruction, structure analysis has been exploited during scanning to recover structural information. To realize object extraction, existing methods are either data-driven [Salas-Moreno et al. 2012], or human-assisted where the user helps in object extraction and conducts object scanning [Zhang et al. 2014; Herbst et al. 2014]. In these works, the analysis would benefit the reconstruction in terms of camera tracking and depth data fusion in SLAM [Zhang et al. 2014], but does not provide guidance for scene scanning which is key to autonomous reconstruction. Our method complements to these works in that our scene analysis is tightly coupled with scanning and reconstruction, as a feedback loop in an autonomous system, where no model database or human input is required.

Autonomous scanning by robot. The idea of autonomous 3D scanning by a robot is not new; many prior works have been conducted for *single object* scanning aiming at complete scanning with minimal effort [Khalfaoui et al. 2013] and active vision [Chen et al. 2011]. Recently, Wu et al. [2014] propose an autoscanning system for single objects driven by a Poisson-based reconstruction quality measure. We employ this measure in evaluating reconstruction uncertainty, as well as their Next-Best-View (NBV) estimation for object-targeted scan refinement. Autonomous scene scanning is also well-studied, but mainly from the robotics point of view, with the goal of environment capturing and exploration (e.g. [Callieri et al. 2004; Foster et al. 2011; Wagner et al. 2013]), but not full reconstruction, especially, object-aware scene reconstruction.

Scene segmentation by robot interaction. There is an immense body of previous works on unknown object extraction from a scene with robot interaction (e.g. [Allen 1988; Bersch et al. 2012; Hausman et al. 2013]). Our *key difference* from the existing works is that our robot interaction is driven by the uncertainty accounting for not only segmentation confidence but also reconstruction quality. To the best of our knowledge, our work is the first to integrate robot interaction into the problem of full reconstruction of indoor scenes, with the goal being not only object extraction but more importantly, object-wise quality reconstruction.

3 Overview

Problem statement. The basic setting of our problem is realtime scene reconstruction using a depth camera. The input is captured raw depth images of an indoor scene and the output is a full reconstruction of the scene as a combination of fixed room wall/furniture and extracted individual objects with relatively high geometry fidelity. The core problem is *object-aware reconstruction* concerning how to extract unknown objects from the scene being reconstructed and at the same time, utilize the object analysis result to guide the robot validation for better object-targeted scanning.

System overview. Our system contains two main components, a 3D scanning system performing realtime 3D acquisition and reconstruction and a physical interaction mechanism enabling proactive validation. Both of the two components are delegated to a PR2 from Willow Garage, a mobile robot with two hands. We mount a Kinect sensor on one of its hands and drive the other one to physically interact with the scene. The system pipeline is shown in Figure 3.

Given an indoor room, the robot first navigates the room while scanning for a rough reconstruction of the entire room space. To facilitate efficient processing, we take a divide-and-conquer scheme by decomposing the scene into several regions of interest (ROIs) and processing them one by one. For each ROI, the system interleaves object analysis and guided validation.

To perform object analysis, we first over-segment the currently reconstructed ROI and represent it as a patch graph. Then we generate a set of object hypotheses by partitioning the patch graph using graph-cut, and build an object graph by contracting the patch graph. Based on the two graphs, we estimate the uncertainty of object-aware reconstruction, to guide the proactive validation, including physical push and object-targeted scan refinement. We adopt *horizontal push* to verify the local segmentation, as well as to assist data acquisition by separating close-by objects apart to reduce occlusion. Scan refinement is targeted at the moved objects by push, to improve their data completeness. The validation results are incorporated to update both segmentation and reconstruction, thus reducing the corresponding uncertainty. Such an interleaving process repeats until the uncertainty stops decreasing.

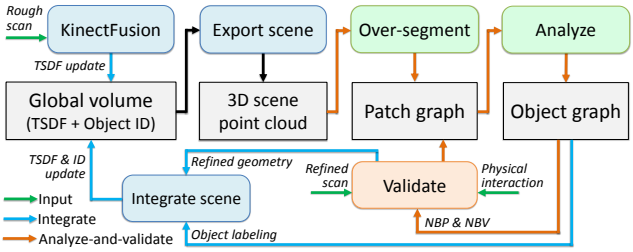


Figure 4: *Our framework of reconstruction and analysis.*

4 Prerequisite and initialization

Online reconstruction and analysis framework. Our system is built upon the voxel hashing based KinectFusion [Prisacariu et al. 2014; Nießner et al. 2013]. Figure 4 shows our basic framework of online reconstruction and analysis. KinectFusion maintains a global volumetric representation, i.e., Truncated Signed Distance Field (TSDF), for the geometry of the whole scene. It receives new frames of depth images captured in realtime and fuses them into the global volume through updating the TSDF. To accommodate object level segmentation, we associate each voxel with an object ID, similar to [Zhang et al. 2014]. To enable object analysis, we export all zero-crossing points of the TSDF of the current ROI. The analysis and validation are iteratively performed based on this 3D point cloud. In each iteration, new object labels and scans are fused into the global volume, by updating object ID and TSDF, respectively.

Scene representation. Our system maintains four scene representations including global volume, 3D point cloud, patch graph and object graph. While the global volume is maintained for the entire scene, the latter three are constructed for only the current ROI, whose geometry and labeling information can be integrated into the global volume for visualization and output purposes (Figure 4).

Our object analysis is performed over the 3D point cloud, in contrast to the method in [Zhang et al. 2014] where object detection is based on region growing in ray-casted 2D depth images. While 2D space analysis facilitates instant visual feedback, which is suitable for human-operated interactive reconstruction, the extraction of 3D objects is especially useful in the estimation of robot actions.

The analyze-and-validate process is centered around the patch and object graph. Specifically, object analysis produces an object graph through partitioning and contracting the patch graph. The two graphs are then used to estimate the next best push actions for proactive validation. Segmentation validation updates the patch graph in both graph topology (local update) and edge cut cost (global update). In summary, our method interleaves between the two graphs, to gradually improve the object-aware reconstruction (see the loop formed by the orange arrows in Figure 4).

Indoor space exploration and decomposition. Entering an indoor room, the PR2 first explores the entire scene while scanning it to obtain the overall geometry and a rough spatial layout. To drive the robot to explore the room, we employ the PR2 navigation package implemented in ROS (Robot Operating System) [ROS 2014]. We decompose the roughly reconstructed room into several regions of interest (ROIs) so that the robot can process them one by one. An ROI is a relatively isolated subscene with intensive object presence demanding detailed scanning (e.g., standalone tables, cubicles, wall corners, etc.); see Figure 3. Such isolated regions can be detected based on spatial isolation or planar surface separation. The output of the initialization stage is a set of ROIs connected by their interconnecting path, similar to [Zhou and Koltun 2013].

5 Object analysis

Analysis overview. We first over-segment the scene point cloud, using the method described in [Papon et al. 2013]. We then build an adjacency graph for the over-segmented patches, denoted with $G_{\mathrm{p}} = (\mathcal{V}_{\mathrm{p}}, \mathcal{E}_{\mathrm{p}})$, with nodes representing patches and edges indicating patch adjacency relation. Based on the patch graph, we compute a set of candidate object hypotheses, using binary graph cuts with the foreground corresponding to a candidate hypothesis [Golovinskiy et al. 2009]. We then select the most prominent hypotheses using a voting-based scheme in a multi-class segmentation.

Our method has several advantages. First, the binary segmentation is performed many times, using every patch as foreground seed, while the final set of objects are extracted using a hypothesis selection mechanism. This avoids the use of any heuristic seeding scheme. Second, our method produces a coherent segmentation of the input scene, instead of outputting a set of hypotheses which may be in conflict with each other. Finally, by dealing with patches, instead of points, local geometric entities can be robustly estimated. All these features make our method especially robust.

To enable the robot to accumulate knowledge about scene composition and improve the segmentation as the proactive reconstruction proceeds, we propose integrating online learning into the graph-cut segmentation framework, through learning the graph-cut cost with on-the-fly obtained training examples.

Object hypothesis generation. In generating object hypotheses using binary graph cuts, we select one patch as foreground seed, but do not prescribe any seed for background. This is achieved by introducing a background penalty for each non-seed patch. Specifically, we select one patch, denoted by P_s, labeling it as foreground $x_s = 1$, and minimize over binary patch labels $X = [x_1, \ldots, x_n], x_i \in \{0, 1\}$, with n being the number of patches, the following parametric energy function:

$$E^{\lambda}(X) = \sum_{u \in \mathcal{V}_{\mathrm{p}}} E_d^{\lambda}(x_u) + \sum_{(u,v) \in \mathcal{E}_{\mathrm{p}}} E_s(x_u, x_v), \qquad (1)$$

where the data term is defined as:

$$E_d^{\lambda}(x_u) = \begin{cases} \infty, & \text{if } x_u = 0 \text{ and } u = s, \\ f_u, & \text{if } x_u = 1 \text{ and } u \neq s, \\ 0, & \text{otherwise.} \end{cases}$$

$$f_u = \begin{cases} k(d(P_s, P_u) - \lambda), & \text{if } d(P_s, P_u) > \lambda, \\ 0, & \text{otherwise.} \end{cases}$$

f_u is the background penalty which penalizes a non-seed patch that is distant from the seed being labeled as foreground. $d(P_s, P_u)$ is the Euclidean distance between the patch centers of P_s and P_u. We use $k = 2.0$ for a steep penalty to quickly prevent those patches, whose distance to P_s is larger than λ, from being labeled as foreground. The parameter λ controls the range, centered around the seed patch, within which we seek for foreground patches. Instead of using a hard threshold on this range, we slide λ from 0 to ℓ_{d} (the diagonal length of the bounding box of the entire scene) and find the first point where the total cut cost drops significantly (up to 50%) and take the cuts as the segmentation result.

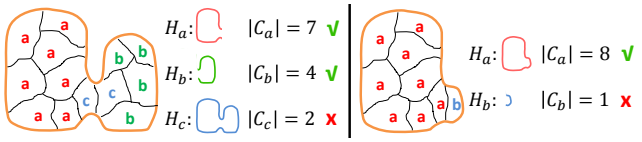


Figure 5: *Our hypothesis selection can overcome false segmentation through consensus voting from hypothesis clusters. The labels on each patch corresponds to the hypothesis generated by seeding with that patch. Left: The hypothesis H_c, representing a false under-segmentation, is generated by seeding with a patch in the middle. It will not be selected since its corresponding cluster $\mathcal{C}_c$ is much smaller than the other two. Right: Suppose a single patch hypothesis H_b is generated by noisy cuts, indicating a false over-segmentation. It will also be discarded due to its small cluster size.*

The smooth term, or cut cost, is defined as the probability of two adjacent patches belonging to the same object, which will be learned based on the results of robot proactive validation. The learning of cut cost is discussed at the end of this section.

The remaining issue is how to select foreground seeds. Instead of relying on heuristic rules, we opt to use every patch as seed and perform binary graph cuts multiple times, leading to many candidate foregrounds with redundancy. Next, we cluster the foreground segments using mean-shift. The similarity between two segments, e.g., S and T, is measured by the Jaccard index [Levandowsky and Winter 1971], $s(S,T) = |S \cap T|/|S \cup T|$. For each foreground cluster, we select the cluster center as the representative object hypothesis for that cluster. This results in a pool of k hypothetical objects, $\mathcal{H} = \{H_i\}_{i=1}^{k}$, each corresponding to a cluster.

Object hypothesis selection. The hypothesis set is not necessarily a partition of the patch graph. Some hypotheses would overlap with each other, making the labeling of patches within the overlapping regions ambiguous. Existing methods either filter the hypotheses based on some heuristic rules, or rank them with a model learned from training data. To select good hypotheses without relying on any heuristics, we propose to let the hypotheses compete with each other in producing a *partition* of the patch graph. This is formulated as a multi-class Markov random field (MRF) segmentation with *object label selection*. Specifically, we minimize the following energy function:

$$E(L) = \sum_{u \in \mathcal{V}_{\mathrm{p}}} E_d(l_u; P_u) + \sum_{(u,v) \in \mathcal{E}_{\mathrm{p}}} E_s(l_u, l_v), \qquad (2)$$

over the labeling for all patches: $L = [l_1, \ldots, l_n], l_u \in \{1, \ldots, k\}$.

The data term $E_d(l_u; P_u)$ is defined as the log-likelihood that the patch P_u belongs to a particular object hypothesis. In particular, for patch P_u and the i-th object hypothesis H_i, the data term is defined based on the ratio between the times of P_u being covered by the members in the foreground cluster corresponding to H_i, denoted by $\mathcal{C}_i$, over the times it is covered by all clusters:

$$E_d(l_u = i; P_u) = -\ln\left(t(P_u, \mathcal{C}_i) / \textstyle\sum_j t(P_u, \mathcal{C}_j)\right), \qquad (3)$$

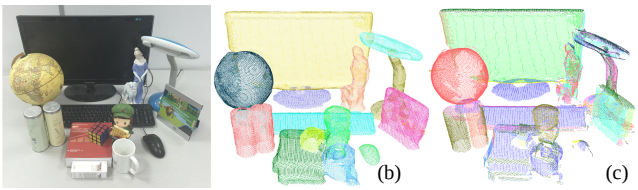


Figure 6: *Object extraction from the point cloud of a highly cluttered scene (a) by using our method (b) and RANSAC based primitive fitting (c). In the input data, close-by objects merge together due to low scan resolution and the back view is not scanned, making the segmentation quite challenging. Our method can segment out most objects accurately. In (c), some points are discarded by the original algorithm due to large fitting error. For both methods, the table plane has been detected and removed, via plane fitting.*

where $t(P_u, \mathcal{C}_i) = |\{P_u \subset H_j | H_j \in \mathcal{C}_i\}|$ is the presence times of patch P_u in cluster C_i. The smoothness term is the same as the one used in the binary graph cuts.

In essence, the data term selects a label for each patch based on a consensus voting from all foreground clusters. The rationale of this voting scheme is that the larger a foreground *cluster* is, the more probable its corresponding object hypothesis represents an independent object, since the object has been proposed by the binary segmentations seeded from many patches. Figure 5 illustrates how our method can discriminate false object hypotheses, in two typical cases, i.e., under- and over-segmentation. Figure 6 demonstrates the segmentation results over a highly cluttered scene. A quantitative evaluation against ground-truth data is presented in Section 7.2.

Online learning of the cut cost. 3D scene segmentation is affected by many factors, such as geometry, color, texture, and even high level structural information. It is difficult to combine all of these factors into a cut cost due to the weighting difficulty. Therefore, we opt to learn the cut cost from the robot proactive validation as the probability that two adjacent patches are labeled differently:

$$E_s(l_u, l_v) = 1 - p(l_u \neq l_v | \mathbf{x}(P_u, P_v)), \tag{4}$$

where $\mathbf{x}(\cdot) \in R^n$ is the feature vector extracted for a pair of patches. Specifically, we train a Support Vector Machine (SVM) prediction function over the feature vector:

$$p(l_u \neq l_v | \mathbf{x}(P_u, P_v)) = g(f(\mathbf{x})) \triangleq p_c(e_{uv}), \tag{5}$$

where f is the prediction function returning a positive value if patch P_u and P_v are labeled differently (the patch graph edge e_{uv} connecting them is cut), and negative otherwise. $g(t) = 1/(1 + e^{-t})$ is the logistic sigmoid function used to convert the prediction value into a probability, which we refer to as the *cut probability* of an edge e_{uv}, denoted by $p_c(e_{uv})$. To train the SVM, we collect a set of examples from physical validation: $\{\mathbf{x}(P_i, P_j), y_{ij}\}$, with $y_{ij} \in \{-1, +1\}$ indicating whether edge e_{ij} is cut. The way that physical validation provides training examples will be discussed in Section 6.3. An important benefit of cut cost learning is that the learned cost can be used to improve the segmentation of the entire patch graph in a global manner, instead of locally updating only the subgraph affected by robot validation.

A major issue with such learning, however, is that the features extracted for different factors are heterogeneous and have different notions of similarity, thus requiring different kernels in SVM training. To this end, we employ the Multiple Kernel Learning (MKL) method [Bach et al. 2004], which learns an optimal kernel for each type of feature from a predefined pool of base kernels. Another concern is that since the training examples arrive sequentially as the process of analyze-and-validate proceeds, it is more efficient to achieve *online learning*, to incrementally update the predictor with progressively gathered examples. To realize online learning with MKL, we adapt the passive-aggressive algorithm [Crammer et al. 2006], which is originally designed for SVM, to the MKL setting. The details about online cut cost learning, along with the kernels and features that we employ, can be found in the Appendix in the supplemental material.

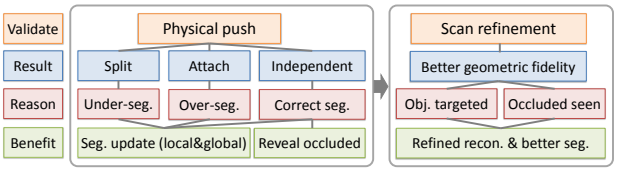


Figure 7: *An overview of the two robot-conducted validations: physical interaction and scan refinement. For each validation, we list the possible consequences, underlying causes, as well as benefits, associated by grey links.*

In the initial stage, when no training data is available, we bootstrap the segmentation simply by defining the cut probability based on local geometric concavity:

$$p_c(e_{uv}) = \eta(1 - \cos\theta_{uv}), \tag{6}$$

where θ_{uv} is the angle between the average normals of patch P_u and P_v. For η, we take 0.01 (a small value) if the two adjacent patches form a convex dihedral angle and 1 otherwise, to encourage cuts around a concave region [Katz and Tal 2003].

6 Entropy-based proactive validation

Validation overview. To improve the object-level segmentation and object-wise reconstruction, the robot performs proactive validation over the current ROI using physical push and scan refinement, driven by the next best push (NBP) and next best view (NBV), respectively. Both NBP and NBV are estimated based on the result of the current object analysis. Physically interacting with the scene can verify and/or correct the segmentation with the gained knowledge from moved objects. Different from existing works, our proactive push is also designed for spatially separating close-by objects and resolving their mutual occlusion, to better the condition for object-targeted scan refinement.

A physical push may cause three consequences for the moved hypothetical objects: (1) An object may move independently, suggesting a correct segmentation of the object; (2) An object may split into multiple ones, indicating an under-segmentation of multiple objects; (3) An object may be attached with its adjacent objects during moving, which means an over-segmentation of a single object. While the latter two lead to corrections over the segmentation, the first could unveil the unobserved regions of the object occluded by its surrounding objects, by separating it clear against them. Figure 7 summarizes the motivation and benefit of the validations.

6.1 Information gain maximization for NBP selection

Based on segmentation, the patch graph G^p is contracted into an object graph, denoted as $G^o = (\mathcal{V}^o, \mathcal{E}^o)$. Our NBP selection is based on the two graphs (Figure 8). Since our system aims for both object-level segmentation and object-wise reconstruction, the NBP should account for both aspects. Firstly, the push should maximally reduce the uncertainty in scene segmentation. On the other hand,

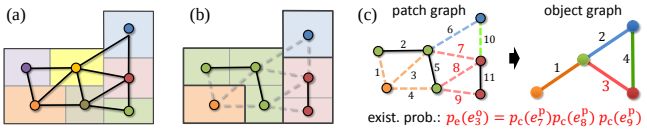


Figure 8: *(a): Patches (shaded in distinct colors) and patch graph. (b): Objects (in distinct colors) extracted by partitioning the patch graph. (c): The patch graph is contracted into an object graph. The existence probability of object graph edges is calculated as the joint cut probability of the corresponding contracted patch graph edges.*

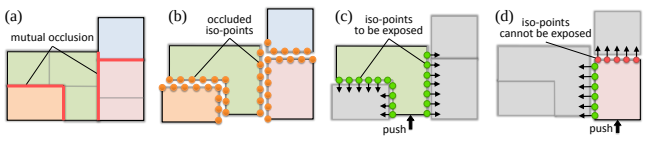


Figure 9: *(a): Mutual occlusion (red interface) given the segmentation in Figure 8. (b): Iso-points (orange dots) sampled on the Poisson iso-surface reconstructed at the occluded regions on every object (objects are slightly moved apart for better visualization). (c): The push on the green object can potentially expose the green iso-points on the object. (d): The push on the red object cannot expose the red iso-points (with normal along the pushing direction).*

to increase the data completeness, the robot should act to *expose* the possibly unobserved region as much as possible, via *separating mutually occluding objects*. While segmentation uncertainty is measured based on the cut probability defined in Equation (5), the uncertainty of reconstruction is defined based on the Poisson-based reconstruction quality measure proposed in [Wu et al. 2014].

Joint entropy of segmentation and reconstruction. We use the *Shannon entropy* [Cover and Thomas 1991] to jointly measure the uncertainty in both segmentation and reconstruction:

$$H = H(S, R), \tag{7}$$

where S and R are the random variables describing the possible segmentation and reconstruction of the current ROI, respectively. This *joint entropy* measures the uncertainty, or amount of information, possessed in the random variables. Since we have $H(S, R) = H(S) + H(R|S)$, the estimation of the joint entropy $H(S, R)$ breaks down into two parts: the segmentation entropy, $H(S)$, and the conditional entropy of reconstruction over segmentation, $H(R|S)$. Next, we explain the estimation of both in detail.

For segmentation, we discretize S with the set of possible partitions of the patch graph G^{p}, denoted as $\mathcal{S}(G^{\mathrm{p}})$. Thus, a segmentation $S_i \in \mathcal{S}(G^{\mathrm{p}})$ can be expressed as the joint cut probability of all edges of G^{p}. Assuming that the cuts of different edges are mutually independent, we have $p(S_i) = \prod_{e \notin S_i} p_{\mathrm{c}}(e) \prod_{e \in S_i} (1 - p_{\mathrm{c}}(e))$, where $p_{\mathrm{c}}(e)$ is the cut probability defined in (5). Therefore, we can estimate H(S) as the entropy in partitioning the patch graph:

$$H(S) = -\sum_{S_i \in \mathcal{S}(G^{\mathrm{p}})} p(S_i) \log p(S_i) = -2 \sum_{e \in \mathcal{E}^{\mathrm{p}}} p_{\mathrm{c}}(e) \log p_{\mathrm{c}}(e) \triangleq H_S(\mathcal{E}^{\mathrm{p}}). \tag{8}$$

To estimate the uncertainty, or data fidelity, in reconstruction, we first compute a Poisson field *for each object* in the ROI using its point cloud and extract the zero-crossing iso-surface. The iso-surface is uniformly sampled into a set of iso-points, denoted by Ω. According to [Wu et al. 2014], the local data fidelity at an iso-point $s \in \Omega$ can be measured based on the Poisson field gradient:

$$c(s) = \Gamma(s) \cdot \mathbf{n}_s, \tag{9}$$

where $\Gamma(s)$ is the gradient of the Poisson field at s with its normal being $\mathbf{n}_s$. Based on this measure, we evaluate the entropy of object-aware reconstruction using the iso-points of all objects:

$$H(R) = -\sum_{s \in \Omega} g(c(s)) \log g(c(s)), \tag{10}$$

where g is the logistic sigmoid function, the same as before.

Self and mutual occlusion are two main factors affecting the scanning visibility, and hence the data completeness for reconstruction. While self-occlusion depends only on the geometry of an object itself and can mostly be resolved by moving the scanner according to estimated NBVs [Khalfaoui et al. 2013], *mutual occlusion*, on the other hand, is neighborhood sensitive and can only be sorted out by moving the objects apart. Therefore, mutual occlusion is directly related to the reconstruction uncertainty that could be reduced by NBPs. Given a scene where the objects are unknown, the object-level segmentation is the only information we have for finding potential mutual occlusion; see Figure 9(a). Thus, it is natural to estimate the reconstruction entropy conditioned on segmentation, through parameterizing the entropy over the edges of the object graph $\mathcal{E}^{\mathrm{o}}(S)$ corresponding to segmentation S:

$$H(R|S) = -\sum_{e \in \mathcal{E}^{\mathrm{o}}(S)} p_{\mathrm{e}}(e) \sum_{s \in \bar{\Omega}(e)} g(c(s)) \log g(c(s)) \triangleq H_{R|S}(\mathcal{E}^{\mathrm{o}}(S)), \tag{11}$$

where the existence probability p_{e} of an edge in $\mathcal{E}^{\mathrm{o}}(S)$ can be estimated as the joint cut probability of the related edges in $\mathcal{E}^{\mathrm{p}}$; see Figure 8(c). $\bar{\Omega}(e)$ is the set of iso-points on the interface between the two objects connected by e (Figure 9(b)). Since these iso-points lie in the occluded region, they introduce significant uncertainty in reconstruction. This conditional entropy measures the uncertainty of reconstruction that is due to the mutual occlusion between neighboring objects extracted by the segmentation. Note that we do not consider the occlusion between an object and its supporting plane, which can not be resolved by horizontal push.

Maximum information gain. An informative push action should aim to maximally reduce the joint uncertainty in segmentation and reconstruction, thus gaining the maximum amount of information for both aspects. Sampling the reconstructed surface of the current ROI, we obtain a set of push candidates, $\mathcal{P} = \{\langle \mathbf{p}_u, \mathbf{d}_u \rangle\}_u$, each characterized by its position and anti-normal direction. The information gain of a push u is measured as the Kullback-Leibler divergence of the joint entropy before and after u is performed:

$$I(S, R|\langle \mathbf{p}_u, \mathbf{d}_u \rangle) = H(S, R) - H'(S, R|\langle \mathbf{p}_u, \mathbf{d}_u \rangle), \tag{12}$$

where $H'(S, R|\langle \mathbf{p}_u, \mathbf{d}_u \rangle)$ is the posterior entropy given push u, whose estimation is discussed below. The NPB is selected from the candidate set as the one which maximizes the information gain:

$$u^* = \arg\max_u I(S, R|\langle \mathbf{p}_u, \mathbf{d}_u \rangle). \tag{13}$$

Posterior entropy. To estimate the posterior entropy caused by a push action, we need to compute the posterior probability of segmentation and reconstruction after the push. This requires reasoning about the objects affected by the push, which are unknown before the push action is taken. To simplify the estimation, we make the two *key assumptions*, based on the current object extraction result. Given a push u, let us denote O_u as the object containing the push point $\mathbf{p}_u$ and $\mathcal{E}^{\mathrm{o}}(O_u)$ the set of object graph edges incident to O_p. *First*, we assume that the cut status of the edges in

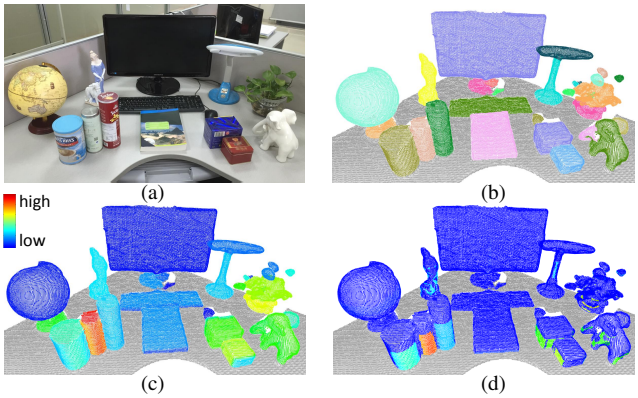


Figure 10: *Plot of push information gain for all points (c) scanned for the input scene (a), based on the segmentation in (b). In (d), physically infeasible pushing points are filtered out, by zeroing their information gain values. The NBP can be selected from the remaining non-zero points.*

$\mathcal{E}^{\mathrm{o}}(O_u)$ will be determined by push u. Thus, the cut uncertainty of these edges will become zero. *Second*, it is assumed that the interface between O_u and its adjacent object O_j will be exposed, if the averaged normal of the interface, $\mathbf{n}_{uj}$, does not agree with the push direction ($\mathbf{n}_{uj} \cdot \mathbf{d}_u < 0.8$); see Figure 9(c,d). Denote by $\mathcal{E}^{\mathrm{o}}(O_u, \mathbf{d}_u) \subset \mathcal{E}^{\mathrm{o}}(O_u)$ the set of object graph edges whose corresponding interface satisfies the above requirement. Consequently, the reconstruction uncertainty in these occluded regions can be eliminated by the push, supposing that sufficient scans will be devoted afterwards (by scan refinement in Section 6.3).

The rationale of these assumptions is that, the current object analysis provides the only, yet informative, cues for reasoning about the unknown configuration of the scene after push. It is reasonable to minimize the entropy as much as possible, while assuming the object extracting is trustable. This is a standard scheme in interleaving optimization. Based on the assumptions, the posterior entropy, after a push u is executed, can be estimated by zeroing the uncertainty corresponding to the determined edges and the exposed iso-points due to u. Thus, the information gain, before and after u is performed, is simply the joint entropy over these edges and iso-points:

$$I(S, R|\langle \mathbf{p}_u, \mathbf{d}_u \rangle) = H_S(\mathcal{E}^{\mathrm{p}}(O_u)) + H_{R|S}(\mathcal{E}^{\mathrm{o}}(O_u, \mathbf{d}_u)). \quad (14)$$

See the denotation of $H_S(\cdot)$ and $H_{R|S}(\cdot)$ in Equation (8) and (11), respectively. Figure 10(c) plots the values of information gain for all points in a table-top scene.

6.2 Physically feasible NBP

The NPB selected based on information gain could be physically infeasible due to various physical limitations. Therefore, we need to filter out those infeasible push candidates to select a physically feasible NBP. Designing a sophisticated method for efficient filtering while handling various physical constraints is out of the scope of this work. We instead propose a heuristic method to select a feasible yet effective pushing action.

For efficiency, we down-sample the point cloud and take the samples as candidate pushing points, among which we first filter out the physically infeasible ones using a series of heuristic rules. These filtering rules are object-aware, meaning that they are designed with respect to the object that the push point lies in, based on the current object extraction result. Figure 10(d) shows the result of pushing

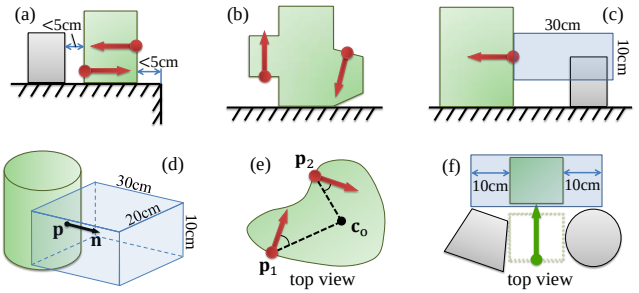


Figure 11: *(a-e): Invalid push candidates (red arrows) or clearance regions (blue boxes) illustrated for the filtering rules. (f): Clearance region used for detecting neighborhood separation.*

point filtering. From the remaining push candidates, we select the NBP by comparing their information gain.

The set of five filtering rules are meant to quickly reject those push candidates which either may have a damaging consequence, such as falling or dropping (**R1**-**R3**), or can not be executed by the robot (**R4**); see Figure 11 for illustrations. We also prefer a translational move (**R5**) of the pushed object to ease the movement detection (Section 6.3). For each candidate, the five filters are applied in order based on their computational cost. Figure 19 shows a sequence of selected NBPs for a virtual scene.

R1 (Lower push): We filter out the pushes whose pushing point is above the $2/3$ height of the object being pushed, to reduce the possibility of falling.

R2 (Block & boundary): We reject the pushes along whose direction the object is too close (less than 5cm) to other objects, or the boundary of its supporting plane (Figure 11(a)).

R3 (Horizontal push): We remove the pushes whose direction points toward or away from the supporting plane, i.e., $|\mathbf{d}_k \cdot \mathbf{n}| > 0.2$, with $\mathbf{n}$ being the normal of the supporting plane, to ensure horizontal pushing over the supporting plane (11(b)).

R4 (Accessibility): We remove those pushes whose pushing point can not be accessed by the robot hand. We make a cuboid clearance region around each pushing point (11(d)). If there is any other object (including the supporting plane) intersecting with the clearance region, the pushing point is marked as inaccessible (11(c)).

R5 (Translational move): To reduce rotational motion of the pushed object, we avoid the pushes whose direction deflects significantly from the vertical line passing through the object's center of mass (11(e)): $\mathbf{d}_k \cdot \mathbf{p} < 0.8$, where $\mathbf{p}$ is the normalized projection of vector $\mathbf{q} = \mathbf{c}_{\mathrm{o}} - \mathbf{p}_k$ onto the horizontal plane, with $\mathbf{c}_{\mathrm{o}}$ being the center of mass of the object being pushed.

Once the NBP is selected, we drive the robot to touch the pushing point and slowly increase the pushing force along the pushing direction. The pushing distance is determined by two factors. First, along the pushing direction, if there are other objects ahead of the object being pushed, we should avoid touching those objects. Otherwise, if there are neighboring objects beside it, we push the object as far as to separate it clear from the neighbors, given enough space ahead. Such separation can be determined with the help of a clearance region (Figure 11(f)). These cases can be easily detected with simple geometric test. If none of the above cases applies, we push for a small distance (e.g., 5cm) which is sufficient for movement detection (see details below).

Note that our scheme for physically feasible push selection is very preliminary and many situations are not handled, especially for ob-

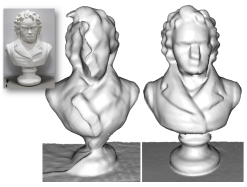
jects with very complex shape and scenes with complicated object clutter. More advanced methods can be employed. Currently, the rules are designed to be as conservative as possible, trying to avoid dangerous actions while accounting for feasibility and effectiveness. Moreover, our selection scheme is prioritized for easily executed pushes. For a given push candidate, if any condition can not be evaluated due to occlusion, we simply filter it out. Therefore, if an object is severely occluded by its surroundings, the push against it would be deferred until its neighbors have been pushed.

Movement tracking. During push, we track the movement of the affected objects using a method combining the algorithms for both textured [Bersch et al. 2012] and textureless [Hausman et al. 2013] object tracking. For textured tracking, visual features are extracted from the RGB images captured by Kinect and tracked by optical flow. For textureless cases, geometric features are extracted from depth images and tracked using the particle filtering technique. Finally, we cluster the tracked trajectories of all the feature points and then assign the features back to their corresponding objects in the original configuration before push (cf. [Hausman et al. 2013]). We have tested that such a combination leads to rather robust tracking and meets our requirement for segmentation validation.

Based on the tracking result, we can easily identify the three cases for segmentation verification (see Figure 12): (1) If the tracked feature points, originally from a single hypothetical object according to the assignment, move jointly according to their trajectories, the object moves independently, implying that it was correctly segmented; (2) If the feature points from a single hypothetical object are clustered into multiple trajectory clusters, then the hypothetical object was under-segmented; (3) If the feature points from multiple hypothetical objects move jointly, it indicates the hypothetical objects together constitute an over-segmentation of a single object. Note that some objects may fall over unexpectedly due to push where our system may lose the tracking of falling objects. In such cases, we simply discard the corresponding segments being tracked and continue to capture the new configuration with following depth frames.

6.3 Scan refinement and validation incorporation

Object-targeted scan refinement. To improve the data completeness of the moved objects by push, we compute a serial of NBVs for the Kinect sensor based on the union of their point clouds, using the method in [Wu et al. 2014]. To adapt the method for our setting, we remove inaccessible NBVs, similar to the accessibility filtering **R4**, with a clearance region suitable for the Kinect sensor. The computed NBVs can reduce tracking drift in KinectFusion significantly, since the NBV transformations can serve as a good initial guess for the ICP-based camera pose estimation. This leads to high quality object-targeted scanning with loop closure, which is difficult for KinectFusion. The inserted figure shows a comparison of loop closure scanning without (left) and with (right) NBV-assisted depth fusion. Note that after every push, KinectFusion has to rebuild the TDSF volume for the moved objects. Therefore, scan refinement can not reuse the previous scans. This is fine for our problem setting since every object is supposed to be pushed at most once. We leave the dynamic update of TDSF for the future.



Incorporating validation results. This involves the incorporation of the information on both segmentation and reconstruction gained from validation. The new data from scan refinement is fused into the global volume by KinectFusion. For segmentation, based on the movement tracking results, we support both a local update

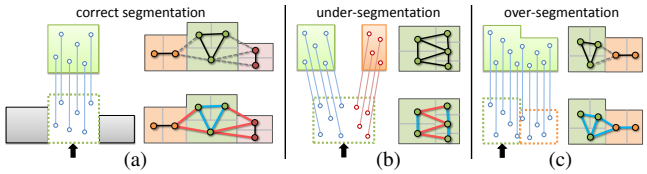


Figure 12: *Three cases in segmentation validation by push. In each case, we show the case identification by movement tracking (left), the hypothetical segmentation (top-right), and the positive/negative example edges (red/blue) offered by the validation (bottom-right).*

to the patch graph and a global update of the graph-cut cost. For a correctly segmented object, we simply merge its patches into a single one and update the patch graph by contracting the corresponding nodes. For wrongly segmented objects, since scan refinement will update their surface reconstruction afterwards, we choose to redo the patch-level segmentation for the moved objects after scan refinement, and then plug the new patches into the patch graph, replacing those corresponding to the original object hypotheses.

A push action, verifying the object-level segmentation, would consequently determine the cut of the related patch graph edges, which provides both positive and negative cut examples for online training of the graph-cut cost. For example, a correctly segmented object will contribute positive cut examples, corresponding to its outgoing patch graph edges, as well as negative ones with its internal edges. Figure 12 illustrates the three cases of segmentation verification and the corresponding training examples extracted.

Termination conditions. The iterative process of analyze-and-validate terminates when the information gain of the next best push is below a pre-defined threshold. However, our method can not guarantee a convergence after finite iterations. This is because our proactive validation by physical push does not guarantee to reduce the uncertainty of both segmentation and reconstruction. For example, a push may sometimes introduce new object clutter, for some complicated object configurations (see the failure cases summarized in Section 7). The scan refinement, on the other hand, can always reduce the uncertainty in reconstruction with improved data completeness. Therefore, our termination conditions are threefold: (1) The maximum information gain of the NBP is less than 5% of the initial joint entropy: $I(S, R|\langle \mathbf{p}_{u^*}, \mathbf{d}_{u^*} \rangle) < 0.05 H^0(S, R)$, (2) There is no more feasible push that can be executed and (3) The maximum number (30) of pushes per ROI has been reached.

7 Results and evaluation

We test our system in scanning several real-life scenes. To quantitatively evaluate our method, we build several test scenes with ground truth data for both segmentation and reconstruction, and propose a quality measure for object-aware reconstruction. We also evaluate the performance of the various algorithmic components in object analysis and proactive validation. Finally, we compare our object-aware physical validation with two alternative methods.

Experimental setting. Our system runs with a PR2 robot. PR2 has a built-in computer running ROS system which provides tools for realizing standard robot behavior such as navigation and hand actions. A Microsoft Kinect is mounted on its left hand, leaving its right hand (optionally holding a rod) for physical push. The depth fusion and analysis algorithms run on a laptop PC with an Intel I7-3740QM CPU (dual core, 2.7GHz), 16GB RAM, and an Nvidia Quadro K2000M graphics card. The laptop is carried by PR2. The Kinect sensor is powered by the carried-on battery of PR2. Thus,

Scene	#ROI	#Obj.	#Push	Recall	Precision	t (min.)
Office	9	~80	40	79%	90%	43
Meeting	1	28	12	66%	95%	16
Kitchen	2	24	13	80%	92%	29
Lab	7	~60	23	68%	88%	51
Cafe	6	~140	71	63%	91%	87
Apart.	9	~120	78	50%	83%	96

Table 1: *Statistics and timings over six real scenes. For each scene, we give the number of ROIs (#ROI), the number of movable objects (#Obj.) in the real scene, the total number of pushes/iterations required (#Push). We also report recall and precision rates of object extraction, as well as the total time cost for scanning and reconstructing the whole scene.*

the whole system is self-contained and cableless, making it flexible for moving. Note that we do not use the pre-installed Kinect or camera of PR2. See the picture of our system in Figure 1(a).

Parameters and complexity. Most of the parameters have been given along with the algorithms. Here we provide the settings for the remaining ones: patch size (diameter): ~6cm; Poisson iso-point sampling density (point spacing): ~2cm; point sampling rate (spacing) for NBP selection: ~2cm; number of scan refinement NBVs per iteration: 4. We use same parameter settings for all our experiments. We give the complexity of two key algorithmic components. The complexity is $O(n^4)$ for object analysis, with n being the number of patches, and $O(pmk)$ for entropy estimation, where p is the number of sample points for NBP selection, m the average number of neighbors per object, and k the average number of iso-points in the interface between two adjacent objects.

7.1 Results on real-life indoor scenes

We run our system to scan and reconstruct six real-life indoor scenes: an office, a meeting room, a kitchen, a computer lab, a cafe, as well as an apartment with three rooms. These real-life scenes contain a variety of object clutters so that a holistic scanning and reconstruction without object delineation can not express the rich details of the content and structure. Figures 1 and 20 demonstrate the object-aware reconstruction results for the six scenes. See the accompanying video for our system at work.

Table 1 reports timing and some related statistics of our system running over different scenes. For each scene, we report the number of pushes required, the recall/precision rate of object extraction, as

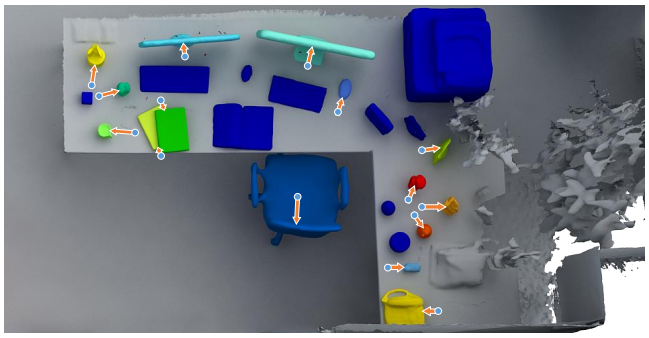


Figure 13: *Push vectors (direction and displacement) of all pushed objects on the table (a subscene of the office scene in Figure 1) are depicted by arrows. This figure shows the final configuration of the objects. Color-coding of objects indicates pushing order.*

Subscene	#Point	#Patch	#Obj.	t_o	t_e	t_p
S4	11K	101	9	1.9	5.2	0.3
Fig. 3	66K	376	11	2.1	7.8	0.3
Fig. 10	98K	572	14	3.5	9.3	0.6

Table 2: *Running time (in sec.) of the various components of our algorithm over three subscenes. For each subscene, we report the number of points, segmented patches and extracted objects, as well as the average computation time of object extraction (t_o), entropy computation (t_e) and push selection (t_p), per iteration.*

well as the total time required to finish reconstructing the entire scene. Table 2 lists the computation time for the key components of our algorithm over three different subscenes (ROIs). Based on the experiments, we make the following conclusions:

- *Efficiency bottleneck*: The major portion of time was spent on robot actions, including navigation, pushing and scanning, while the various computational steps are relatively efficient. 90% of the time on entropy estimation was due to the computation of Poisson iso-points.
- *Recall/precision rate of object extraction*: For the six scenes, our method correctly extracts about 50%~80% (recall rate) of the visible and movable objects in the scenes, depending on spatial accessibility and object clutteredness, etc. The percentages of correctly extracted objects over all extracted ones range from 83%~95% (precision rate). The correctness of the extracted objects is assessed by humans after reconstruction.
- *Push efficiency*: Over all scenes, our system extracts 1.4 objects per push on average. There are two cases when an object will not be pushed: (1) It is unpushable and (2) It is isolated (no mutual occlusion) with low segmentation uncertainty so that the information gain is too small (see termination condition (1)). Figure 13 shows the pushing directions (arrows) for all pushed objects over the office table in Figure 1, where the color-coding of objects indicates the order of pushes.
- *Termination*: Among all 34 ROIs the robot has processed, 14 was terminated by the condition (1), and the rest by (2); see the conditions in Section 6.3. The majority terminating status is no push executable (condition (2)), which is mainly due to our conservative push selection. For the scenes we have tested, we did not encounter a case where the maximum number of pushes per ROI has to be executed (condition (3)).

7.2 Quantitative evaluation and comparison

Test scenes with pseudo-ground truth. To quantitatively evaluate our system, we physically construct several test scenes and build for each a ground truth reconstruction with object-level segmentation and per-object geometry fidelity. To do so, we first reconstruct a ground-truth 3D model for each of the objects used in building the test scenes, via detailed scanning. For each test scene, we create a *pseudo-ground truth* via manually placing the object models in 3D, to reproduce their spatial layout in the real scene. Since our reconstruction is invasive, such ground truth should be built for the final scene configuration after running the test.

Using a set of nine objects with the ground-truth model, we construct six table-top scenes (S1~S6) with an increasing degree of object clutter (Figure 15). Clutteredness is measured by object compactness, i.e., the ratio of the sum of bounding box volume for all objects over that of the whole set of objects. For each scene, after running a test over it, we put markers on the table, aligning to the markers on the bottom of the objects, to record object placement

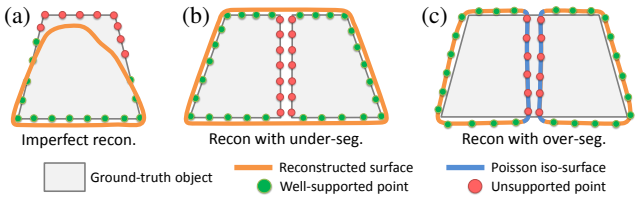


Figure 14: *Quality measure of object-aware reconstruction. (a): The reconstruction is imperfect; the red points on the ground truth can not find support from the reconstructed surface due to severe geometric deviation. (b): The reconstructed surface represents an under-segmentation of the ground truth; many ground truth points are not covered by the reconstructed surface. (c): The ground truth model is over-segmented, resulting in two reconstructed surfaces. In both reconstructions, the Poisson iso-surface will be estimated to fill the "missing regions" inside the ground truth. The iso-points on such regions are not supported by the ground truth.*

(position and rotation). We then remove the objects and scan the table-top along with the markers. Finally, over the reconstructed table-top, we manually place the ground-truth 3D models based on their respective markers. See Figure 18(a) for the test scene S4, as well as the corresponding ground truth built for its initial configuration. All test scenes and associated ground-truth data can be found in the supplemental material.

A measure for object-aware reconstruction. To evaluate the quality of object-aware reconstruction against ground truth, we propose a *unified* measure for both object extraction and object-wise reconstruction, via computing the *mutual data support* between the reconstructed and ground-truth scene. Suppose $T = \{(p_j, \mathbf{n}_j)\}$ is the set of surface points (with normal) of the ground-truth scene and $S = \{(p_i, \mathbf{n}_i)\}$ that of Poisson iso-points of the reconstructed scene, we measure the mutual support between the two point sets:

$$\Pi(S,T) = \frac{\sum_{(p_i,\mathbf{n}_i)\in T} \pi(p_i,\mathbf{n}_i,S)}{|T|} + \frac{\sum_{(p_j,\mathbf{n}_j)\in S} \pi(p_j,\mathbf{n}_j,T)}{|S|}, \tag{15}$$

where $\pi(p_i, \mathbf{n}_i, S)$ measures how well a point p_i in T is supported by the points in S, and vice versa. The data support is defined with a normalized bilateral weighted sum:

$$\pi(p_i,\mathbf{n}_i,S) = \left(\sum_{(q_k,\mathbf{n}_k)\in\Omega_i^T} e^{-\frac{\|p_i-q_k\|^2}{(\sigma/2)^2}} e^{-\frac{\|1-\mathbf{n}_i\cdot\mathbf{n}_k\|^2}{(\theta/2)^2}} \right) / |\Omega_i^T|,$$

where $\Omega_i^T \subset T$ is point p_i's K-nearest neighbor points on T ($K = 100$). $\sigma = 0.5$cm is the support radius for distance and $\theta = 0.2$ for normal difference. $\|\cdot\|$ denotes ℓ_2-norm. By definition, a higher value of $\pi(p_i, \mathbf{n}_i, S)$ indicates p_i in T receives more support from the points in S which are close to p_i in terms of both Euclidean distance and normal direction. $\pi(p_j, \mathbf{n}_j, T)$ is defined similarly.

A notable feature of such a mutual support measure is that it accounts for not only reconstruction quality, by penalizing missing data in reconstruction with respect to the ground truth (the first term in Equation (15)), but also object extraction accuracy, by penalizing both over- (the second term) and under-segmentation (the first one). See Figure 14 for a 2D illustrative explanation.

Performance of object-aware reconstruction. With the quantitative measure, we evaluate the quality of object-aware reconstruction of our system. We run our system (with both analysis and validation) on the test scenes S1~S6 and output the final point clouds,

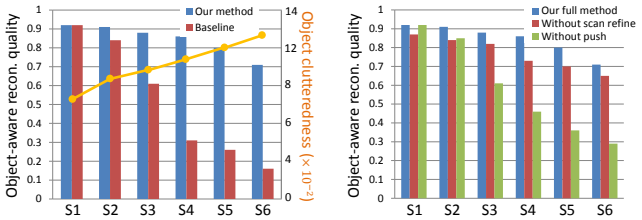


Figure 15: *Left: Comparison of reconstruction quality between our method and a baseline method adapted from [Zhang et al. 2014], over the six test scenes with varying clutteredness (indicated by the yellow curve). Right: Results by turning off each of the two validations, in comparison with our full method.*

with the extracted objects represented as different segments. Figure 15(left) plots the quality measure for the point clouds w.r.t. their corresponding ground truth data. For comparison, we also plot the reconstruction results of a baseline method adapted from [Zhang et al. 2014], where the supporting planes are first identified and objects are detected as the isolated non-planar regions over the planes. For this method, we run autoscanning over the non-planar objects to acquire as complete as possible point cloud data, but do not further segment them. As shown in the plots, when objects are not cluttered, both methods obtain reasonably good reconstruction quality. When object clutter becomes severe, however, the data quality by the baseline method decreases significantly due to occlusion and mis-segmentation. In contrast, our method produces good per-object reconstruction even for highly cluttered scenes.

Performance of object analysis *without* learning. To test the robustness of our voting-based graph-cut segmentation algorithm, we run a method with the fixed cut cost in Equation 6, on scenes with varying data completeness and object clutteredness. For test scene S4, we run KinectFusion with NBVs [Wu et al. 2014] estimated for the entire scene and output six point clouds with increasing number of NBVs (1~6). Then for each scene of S1~S6, we scan it with 6 NBVs, generating another six point clouds. For each point cloud, we estimate the ground truth segmentation through transferring the object labels from the corresponding ground truth scene to that point cloud, using closest point matching. Figure 16 plots the Rand Index [Chen et al. 2013b] of our segmentation over the different data configurations. As a comparison, we also plot the results by RANSAC primitive fitting for point cloud segmentation [Schnabel et al. 2007]. Our segmentation method is robust against data incompleteness and object clutter, making it particularly effective for generating object hypotheses in our problem setting.

Performance of object analysis *with* online learning. The performance of our segmentation method can be significantly enhanced by online learning. Figure 17(top) plots the segmentation accuracy over the scene in Figure 10 with an increasing number of pushes. As the proactive validation progresses, our system gains training examples from the pushes in a sequential manner, to incrementally improve the segmentation model by updating the cut cost therein. The plots show that our online learning with MKL not only improves the segmentation over our method without learning, but also supersedes SVM-based online learning. In Figure 17(bottom), we show how the learned segmentation model can benefit the subsequent analysis of a similar subscene (another cubicle in the same lab), where our method achieves higher accuracy and faster convergence. Such accurate segmentation greatly improves the efficiency in scanning a large scene containing many objects, especially replicated ones. We have also tested the scanning of the computer lab

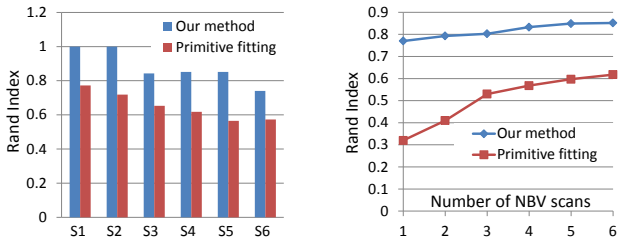
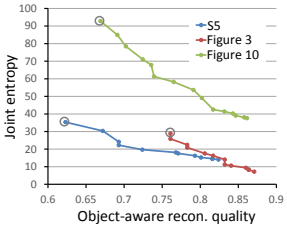


Figure 16: *Comparison of segmentation accuracy (Rand Index) between our method and RANSAC based primitive fitting [Schnabel et al. 2007] with increasing object clutteredness (left, with test scenes S1~S6) and data completeness (right, on test scene S4).*

and the apartment scenes, for with and without online segmentation learning. With online learning, our system saves 30% and 20% pushes for the two scenes, respectively, over without learning.

Effect of validations. To assess the effect of physical push and object-targeted scan refinement, we evaluate our method with each one of them turned off. Using the test scenes S1~S6, we evaluate the quality of object-aware reconstruction for our method without physical push, where scanning is driven by the NBVs estimated for the entire scene. This basically boils down to an extension of the autoscanning method by Wu et al. [2014], from object- to scene-oriented. Over the same set of scenes, we also evaluate our method without *objected-targeted* scan refinement. Figure 15(right) plots the quality measure for the two methods. The autoscanning method in [Wu et al. 2014] can not achieve quality reconstruction for cluttered scenes due to the severe mutual occlusions between objects, which needs to be addressed by physical interaction.

To demonstrate the actual effect of the NBPs selected in each iteration step, we plot in the inserted figure the change of joint entropy and object-aware reconstruction quality over the number of iterations executed, over three scenes: test scene S5 and the two scenes from Figure 3 and Figure 10, respectively. The initial point (0-th iteration) is marked with a circle. From the plots, it is clear that the selected NBPs are effective in reducing the joint entropy, as well as in improving the reconstruction quality. The curves also confirm that our object-aware reconstruction quality measure is consistent with our measure of joint uncertainty of segmentation and reconstruction.



Comparison of NBP selection. We compare our push selection with two alternative methods for textured [Bersch et al. 2012] and textureless [Hausman et al. 2013] objects, respectively. In these methods, the pushes are directed to corner points detected from a top-view of the scene. The goal is to discern independent objects rather than to separate them out of clutter for better scanning. Both methods stop when either a prescribed number of objects are detected or the maximum number of pushes is reached. Over test scene S4, we run each method for 10 pushes and measure the quality of object-aware reconstruction after each push. To conduct a fair comparison against their methods, we run scan refinement after each push, targeted at the pushed object, with the same number of NBVs as we use. The plots in Figure 18 demonstrate that our method is more effective in achieving object-aware reconstruction, since our physical interaction is designed not only for segmenting objects, but also for exposing occluded regions to assist scanning.

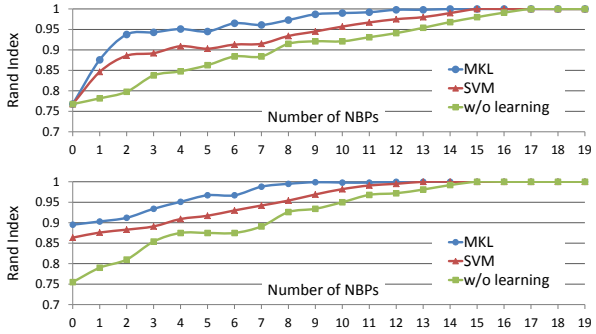


Figure 17: *Top: We plot the segmentation accuracy over a scene with MKL-based online learning of the segmentation model from proactive validation. As a comparison, the accuracy by SVM-based online learning and the non-learning method are also plotted. Bottom: The trained models (both by MKL and SVM) are then applied to segment another similar scene, achieving good accuracy even from the beginning, with fast convergence.*

Simulation on virtual scenes. As a complementary approach to quantitative evaluation, we have implemented a simulation of our system with virtual scanning and pushing, and tested it over 3D scene models. The advantage of virtual scanning is that the accurate ground truth is available. However, a full-fledged simulation is extremely difficult especially for physical pushing. In our simulation, push is simply treated as translation over a supporting plane without considering rigid body dynamics. Figure 19 shows the progressive reconstructions produced by our simulated system in all iteration steps. The final result demonstrates good quality object-aware reconstruction, in contrast to the initial reconstruction.

7.3 Failure cases and limitations

Failure cases. The failure cases of our system stem mainly from the proactive validation. We summarize a few typical cases below:

- *Object stack*: If two vertically stacked objects are recognized as a single object due to under-segmentation, our method may fail to separate them, since they will move together if the pushing point is selected on the bottom object.
- *Interlocking object placement*: Some highly complicated object configurations, such as interlocking objects, may require more sophisticated scheduling of pushes, or even extra robot actions such as grasping, to resolve.
- *Non-rigid objects*: Due to the rigid object assumption, our method can not correctly extract non-rigid or articulated objects, such as a piece of cloth placed on a table.

Limitations. The main limitations of our method include:

- *Extraction rate*: Our current system extracts only a limited portion of the large number of objects in a real-life scene. This is due to many factors, such as the accessibility of room space, flexibility of the robot, the capability of pushing action, and the scanning resolution of Kinect, among others.
- *Under-segmentation penalization*: Unlike over-segmentation, which is penalized by entropy of both segmentation (Equation (8)) and reconstruction (Equation (11)), under-segmentation cannot be penalized by the latter since there is no segmentation interface. In an extreme case, when two identical cubes are tightly huddled and perfectly aligned so that they can easily be treated as a single object, the uncertainty of this under-

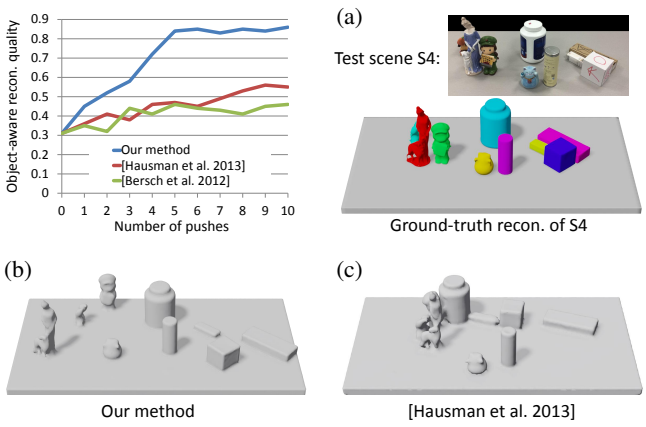


Figure 18: *The plots compare object-aware reconstruction between our method and those with the pushing scheme in [Hausman et al. 2013] and [Bersch et al. 2012], respectively, over test scene S4. Their methods do not aim at object-wise reconstruction, so their reconstruction quality does not increase as prominently as ours with more pushes exerted. (a): Test scene S4 with ground truth data. (b) and (c): The reconstruction results of the final object configurations for our method and [Hausman et al. 2013], respectively.*

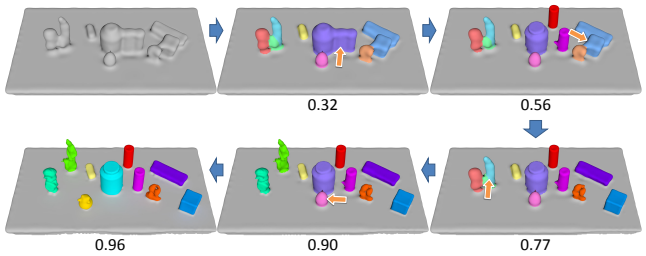


Figure 19: *Simulation of our system with virtual scanning and pushing. Starting from an initial reconstruction (top-left), the simulation iteratively performs object analysis and validation. Object to be validated is marked with pushing direction (orange arrow). The numbers are the quality values of object-aware reconstruction at each step, measured w.r.t. the respective ground truth scenes.*

segmentation can be very low (Equation (8) also fails). In this case, our method would fail to identify and correct this case, unless some random pushes are issued intermittently.

- *Vertical support*: Our physical validation supports only horizontal push which cannot deal with vertical support, such as objects hanging on the wall (see Figure 20). Our segmentation method can extract some of these objects, over which scan refinement can be still performed. Currently, there is no push validation for these objects, since they not are pushable according to our NBP selection method.

- *Robot platform*: The efficiency bottleneck of our system is mainly on the moving and acting speed of the robot. Thus, it takes quite a long time for our system to scan a large scene. Besides that, the current system is quite costly due to the PR2 robot. Nevertheless, our core algorithms are general and can be easily extended to other robot platforms.

8 Conclusion and future work

We develop a framework for autonomous scanning and reconstruction of indoor scenes, through closing the loop of reconstruction and analysis. The idea is that the scanning, guided by the on-the-fly analysis, would progressively improve data fidelity, to benefit reconstruction and in turn, further analysis. Specifically, we employ a robot to perform both data acquisition and analysis, allowing the analysis to provide immediate feedback to the acquisition process and to obtain data on demand, while the scene is being reconstructed. Moreover, we leverage the mobility of the robot to physically interact with the scene, to make the scene more discernible, thus improving the condition for scanning and reconstruction.

In realizing such a framework, we propose an analyze-and-validate approach covering two major components: object analysis with online learning and entropy-guided proactive validation. They are built on top of a flexible scene representation, i.e., the interleaving patch and object graph, making the analysis particularly robust and the validation results easily incorporated. The output is object-aware scene reconstruction, including both object-level segmentation and object-wise quality reconstruction.

Future work. Our coupled reconstruction and analysis framework has exemplified a general closed loop approach that would facilitate active object analysis for scene reconstruction. With the ever increasing popularity of robots and their maneuvering capability, we plan to resort to more aggressive probing and thus more active analysis, to handle scenes with significant clutter, where individual objects are still indiscernible by simple prodding. We also plan to harness 3D shape databases and data-driven shape analysis to endow the robot with higher level intelligence, with structural or semantic analysis, to support more powerful autonomous scanning and reconstruction. As a natural follow-up, we are interested in having multiple robots working together in a collaborative manner, for indoor scene reconstruction. Another interesting direction is to exploit flying robots, which are more agile, in the exploration and reconstruction of large scale scenes. All these directions pose challenging computer graphics problems for future research.

Acknowledgments

We thank all the reviewers for their valuable comments and constructive suggestions. We are especially grateful to Hao Zhang and Daniel Cohen-Or for their help and discussion. We would also like to acknowledge our research grants: NSFC (61232011, 61572507, 61202333, 61272327), National 973 Program (2015CB352500, 2015CB352501), Guangdong Science and Technology Program (2015A030312015, 2014B050502009, 2014TX01X033), Shenzhen VisuCA Key Lab (CXB201104220029A), and SIAT Innovation Program for Excellent Young Researchers (201402).

References

ALLEN, P. K. 1988. Integrating vision and touch for object recognition tasks. *Int. J. Robotics Research 7*, 6, 1533.

BACH, F., LANCKRIET, G., AND JORDAN, M. 2004. Multiple kernel learning, conic duality, and the smo algorithm. In *Proc. ICML*, 1–6.

BERGER, M., TAGLIASACCHI, A., SEVERSKY, L. M., ALLIEZ, P., LEVINE, J. A., SHARF, A., AND SILVA, C. 2014. State of the art in surface reconstruction from point clouds. *Eurographics STAR*, 165–185.

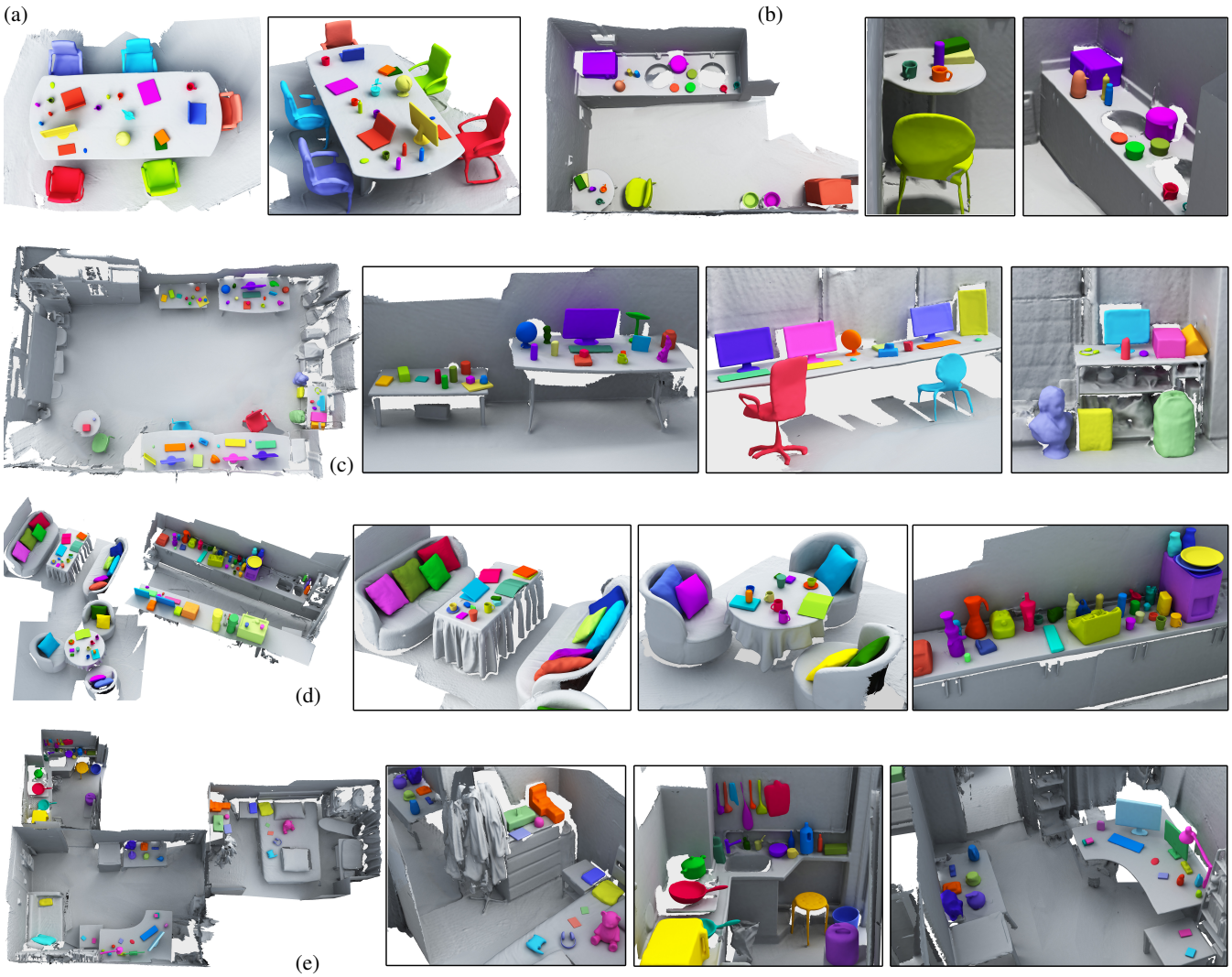


Figure 20: *A gallery of object-aware reconstruction results for five real-life indoor scenes, including a meeting room (a), a kitchen (b), a computer lab (c), a cafe (d), and a three-room apartment (e). Our system extracts* 18, 19, 41, 93 *and* 60 *objects for these scenes, respectively, while achieving object-wise quality reconstruction. The reconstruction of large scale room structure (e.g., ground, walls and large furnitures) is incomplete, which is not the focus of this work. Note in the kitchen of the apartment (the second zoom-in view in (e)), the hanging kitchen wares are extracted by segmentation without physical validation.*

BERSCH, C., PANGERCIC, D., OSENTOSKI, S., HAUSMAN, K., MARTON, Z.-C., UEDA, R., OKADA, K., AND BEETZ, M. 2012. Segmentation of cluttered scenes through interactive perception. In *RSS Workshop on Robots in Clutter: Manipulation, Perception and Navigation in Human Environments*.

CALLIERI, M., FASANO, A., IMPOCO, G., CIGNONI, P., SCOPIGNO, R., PARRINI, G., AND BIAGINI, G. 2004. Roboscan: an automatic system for accurate and unattended 3D scanning. In *Proc. of 3DPVT*, 805–812.

CHEN, S., LI, Y., AND KWOK, N. M. 2011. Active vision in robotic systems: A survey of recent developments. *Int. J. Robotics Research 30*, 11, 1343–1377.

CHEN, J., BAUTEMBACH, D., AND IZADI, S. 2013. Scalable real-time volumetric surface reconstruction. *ACM Trans. on Graph. (SIGGRAPH) 32*, 4, 113:1–113:16.

CHEN, X., GOLOVINSKIY, A., AND FUNKHOUSER, T. 2013. A benchmark for 3D mesh segmentation. *ACM Trans. on Graph. (SIGGRAPH) 28*, 3, 73:1–73:12.

CHEN, K., LAI, Y.-K., WU, Y.-X., MARTIN, R., AND HU, S.-M. 2014. Automatic semantic modeling of indoor scenes from low-quality rgb-d data using contextual information. *ACM Trans. on Graph. (SIGGRAPH Asia) 33*, 6, 208:1–208:15.

COVER, T., AND THOMAS, J. 1991. *Elements of Information Theory*. Wiley.

CRAMMER, K., DEKEL, O., KESHET, J., SHALEV-SHWARTZ, S., AND SINGER, Y. 2006. Online passive-aggressive algorithms. *J. Mach. Learn. Res. 7* (Dec.), 551–585.

CURLESS, B., AND LEVOY, M. 1996. A volumetric method for building complex models from range images. In *Proc. of SIGGRAPH*, 303–312.

FISHER, M., SAVVA, M., AND HANRAHAN, P. 2011. Characterizing structural relationships in scenes using graph kernels. *ACM Trans. on Graph. (SIGGRAPH) 30*, 4, 34:1–34:11.

FISHER, M., RITCHIE, D., SAVVA, M., FUNKHOUSER, T., AND HANRAHAN, P. 2012. Example-based synthesis of 3D object arrangements. *ACM Trans. on Graph. (SIGGRAPH Asia) 31*, 6, 135:1–135:11.

FOSTER, R. B., WANG, R., AND GRUPEN, R. 2011. A mobile robot for autonomous scene capture and rendering. *UMass Technical Report UM-CS-2011-019*.

GOLOVINSKIY, A., KIM, V. G., AND FUNKHOUSER, T. A. 2009. Shape-based recognition of 3D point clouds in urban environments. In *Proc. ICCV*, 2154–2161.

GUPTA, S., ARBELAEZ, P., AND MALIK, J. 2013. Perceptual organization and recognition of indoor scenes from RGB-D images. In *Proc. CVPR*, 564–571.

HAUSMAN, K., BALINT-BENCZEDI, F., PANGERCIC, D., MARTON, Z.-C., UEDA, R., OKADA, K., AND BEETZ, M. 2013. Tracking-based interactive segmentation of textureless objects. In *Proc. ICRA*, 1122–1129.

HEDAU, V., HOIEM, D., AND FORSYTH, D. 2010. Thinking inside the box: Using appearance models and context based on room geometry. In *Proc. ECCV*. 224–237.

HERBST, E., HENRY, P., AND FOX, D. 2014. Toward online 3-D object segmentation and mapping. In *Proc. ICRA*, 3193–3200.

JIANG, Y., AND SAXENA, A. 2013. Hallucinating humans for learning robotic placement of objects. In *Proc. Experimental Robotics*, 921–937.

KATZ, S., AND TAL, A. 2003. Hierarchical mesh decomposition using fuzzy clustering and cuts. *ACM Trans. on Graph. (SIGGRAPH) 22*, 3, 954–961.

KHALFAOUI, S., SEULIN, R., FOUGEROLLE, Y., AND FOFI, D. 2013. An efficient method for fully automatic 3D digitization of unknown objects. *Computers in Industry 64*, 9, 1152–1160.

KIM, Y. M., MITRA, N. J., YAN, D.-M., AND GUIBAS, L. 2012. Acquiring 3D indoor environments with variability and repetition. *ACM Trans. on Graph. (SIGGRAPH Asia) 31*, 6, 138:1–138:11.

LEVANDOWSKY, M., AND WINTER, D. 1971. Distance between sets. *Nature 234*, 5, 34–35.

LI, Y., DAI, A., GUIBAS, L., AND NIESSNER, M. 2015. Database-assisted object retrieval for real-time 3D reconstruction. *Computer Graphics Forum (Eurographics) 34*, 2.

LIU, T., CHAUDHURI, S., KIM, V. G., HUANG, Q., MITRA, N. J., AND FUNKHOUSER, T. 2014. Creating consistent scene graphs using a probabilistic grammar. *ACM Trans. on Graph. (SIGGRAPH Asia) 33*, 6, 211:1–211:12.

MATTAUSCH, O., PANOZZO, D., MURA, C., SORKINE-HORNUNG, O., AND PAJAROLA, R. 2014. Object detection and classification from large-scale cluttered indoor scans. *Computer Graphics Forum (Eurographics) 33*, 2.

NAN, L., XIE, K., AND SHARF, A. 2012. A search-classify approach for cluttered indoor scene understanding. *ACM Trans. on Graph. (SIGGRAPH Asia) 31*, 6, 137:1–137:10.

NEWCOMBE, R. A., DAVISON, A. J., IZADI, S., KOHLI, P., HILLIGES, O., SHOTTON, J., MOLYNEAUX, D., HODGES, S., KIM, D., AND FITZGIBBON, A. 2011. KinectFusion: Real-time dense surface mapping and tracking. In *Proc. IEEE Int. Symp. on Mixed and Augmented Reality*, 127–136.

NIESSNER, M., ZOLLHÖFER, M., IZADI, S., AND STAMMINGER, M. 2013. Real-time 3D reconstruction at scale using voxel hashing. *ACM Trans. on Graph. (SIGGRAPH Asia) 32*, 6, 169:1–169:11.

PAPON, J., ABRAMOV, A., SCHOELER, M., AND WÖRGÖTTER, F. 2013. Voxel cloud connectivity segmentation - supervoxels for point clouds. In *Proc. CVPR*, 2027–2034.

PRISACARIU, V. A., KÄHLER, O., CHENG, M. M., VALENTIN, J., TORR, P. H. S., REID, I. D., AND MURRAY, D. W. 2014. A framework for the volumetric integration of depth images. *ArXiv e-prints*, 1410.0925.

ROS, 2014. ROS Wiki. `http://wiki.ros.org/`.

ROTH, H., AND VONA, M. 2012. Moving volume KinectFusion. In *Proc. BMVC*, 112:1–112:11.

SALAS-MORENO, R. F., NEWCOMBE, R. A., STRASDAT, H., KELLY, P. H. J., AND DAVISON, A. J. 2012. SLAM++: Simultaneous localisation and mapping at the level of objects. In *CVPR*, 1352–1359.

SAVVA, M., CHANG, A. X., HANRAHAN, P., FISHER, M., AND NIESSNER, M. 2014. Scenegrok: Inferring action maps in 3D environments. *ACM Trans. on Graph. (SIGGRAPH Asia) 33*, 6.

SCHNABEL, R., WAHL, R., AND KLEIN, R. 2007. Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum 26*, 2, 214–226.

SHAO, T., XU, W., ZHOU, K., WANG, J., LI, D., AND GUO, B. 2012. An interactive approach to semantic modeling of indoor scenes with an RGBD camera. *ACM Trans. on Graph. (SIGGRAPH Asia) 31*, 6, 136:1–136:11.

SILBERMAN, N., KOHLI, P., HOIEM, D., AND FERGUS, R. 2012. Indoor segmentation and support inference from RGBD images. In *Proc. ECCV*, 746–760.

WAGNER, R., FRESE, U., AND BUML, B. 2013. Real-time dense multi-scale workspace modeling on a humanoid robot. In *Proc. IROS*, 5164–5171.

WHELAN, T., KAESS, M., FALLON, M., JOHANNSSON, H., LEONARD, J., AND MCDONALD, J. 2012. Kintinuous: Spatially extended KinectFusion. In *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*.

WU, S., SUN, W., LONG, P., HUANG, H., COHEN-OR, D., GONG, M., DEUSSEN, O., AND CHEN, B. 2014. Quality-driven poisson-guided autoscanning. *ACM Trans. on Graph. (SIGGRAPH Asia) 33*, 6, 203:1–203:12.

ZHANG, Y., XU, W., TONG, Y., AND ZHOU, K. 2014. Online structure analysis for real-time indoor scene reconstruction. *ACM Trans. on Graph.*.

ZHOU, Q.-Y., AND KOLTUN, V. 2013. Dense scene reconstruction with points of interest. *ACM Trans. on Graph. (SIGGRAPH) 32*, 4, 112:1–112:8.