A Linear Approach to Matching Cuboids in RGBD Images

Hao Jiang
Boston College
hjiang@cs.bc.edu

Jianxiong Xiao
Massachusetts Institute of Technology
jxiao@csail.mit.edu

Abstract

We propose a novel linear method to match cuboids in indoor scenes using RGBD images from Kinect. Beyond depth maps, these cuboids reveal important structures of a scene. Instead of directly fitting cuboids to 3D data, we first construct cuboid candidates using superpixel pairs on a RGBD image, and then we optimize the configuration of the cuboids to satisfy the global structure constraints. The optimal configuration has low local matching costs, small object intersection and occlusion, and the cuboids tend to project to a large region in the image; the number of cuboids is optimized simultaneously. We formulate the multiple cuboid matching problem as a mixed integer linear program and solve the optimization efficiently with a branch and bound method. The optimization guarantees the global optimal solution. Our experiments on the Kinect RGBD images of a variety of indoor scenes show that our proposed method is efficient, accurate and robust against object appearance variations, occlusions and strong clutter.

1. Introduction

Finding three-dimensional structures and shapes from images is a key task in computer vision. Nowadays, we can obtain reliable depth map using low cost RGBD cameras from the digital consumer market, *e.g*. Microsoft Kinect, Asus Xtion and Primesense. Just like digital cameras that capture raw RGB data, these devices capture raw depth maps along with RGB color images. RGBD images provide a pointwise representation of a 3D space. We would like to extract structures from such data.

Recently, there are a few heroic efforts in extracting structures in RGBD images, *e.g*. [14, 13]. However, most of these approaches group pixels into surface segments, *i.e*. the counterpart of image segmentation for RGB images. Although some noteworthy studies [10] infer support relations in scenes, there is still very little volumetric reasoning used in the 3D space, which should be even more important as the depth is available, than pure image information with which volumetric reasoning is well studied [12, 6, 15, 3, 2].

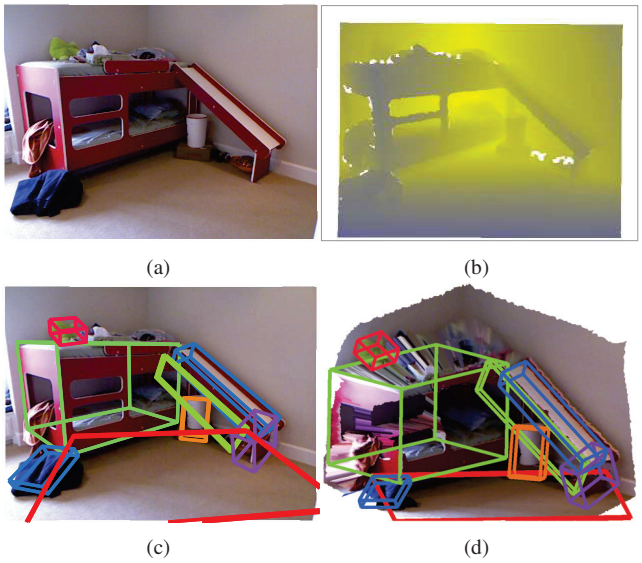


Figure 1. Given a color image and depth map we match cuboid-shaped objects in the scene. (a) and (b): The color image and aligned depth map from Kinect. (c): The cuboids detected by the proposed method and projected onto the color image. (d): The cuboids in the scene viewed from another perspective. These cuboids reveal important structures of the scene.

In this paper, we design an efficient algorithm to match cuboid structures in an indoor scene using the RGBD images, as illustrated in Fig. 1. A cuboid detector has many important applications. It is a key technique to enable a robot to manipulate box objects [5]. Cuboids also often appear in man-made structures [15]. A cuboid detector thus facilitates finding these structures. Detecting cuboids from RGBD images is challenging due to heavy object occlusion, missing data and strong clutter. We propose an efficient and reliable linear method to make a first step towards solving this problem.

Even though matching planes, spheres, cylinders and cones in point clouds has been intensively studied [4], there have been few methods that are able to match multiple cuboids simultaneously in 3D data. RANSAC has been combined with extra constraints to reconstruct geometrical primitives on industry parts from relatively clean range data [1]; this method assumes perfect geometric primitives. To

1063-6919/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPR.2013.282

recover cuboid-like objects in cluttered scenes, we need a different method.

Local approaches have been proposed for fitting cuboids to point clouds. In [7], shapes are modeled as superquadratics and detected in clutter-free range data. A gradient descent method has been proposed in [5] to find multiple cuboids. Due to high complexity, this method has been used to find cuboids in simple scenes with clutter removed. In contrast to these local methods, our proposed method is able to work on cluttered scenes, does not need initialization and guarantees globally optimal result. Our method is also much more efficient.

Cuboid detection has been intensively studied with 2D images as the input. In [6], a method is proposed to reliably extract cuboids in 2D images of indoor scenes. This method assumes that all the cuboids are aligned to three dominant orientations. In [12], spatial reasoning is used to understand outdoor scenes in 2D images; object interactions are modeled with a small set of 3D configurations. Sampling method has been proposed in [18] to extract 3D layout in 2D indoor images. Recently, a method [15] is proposed to detect cuboids with flexible poses in 2D images. Finding 3D cuboids in 2D images requires different domain knowledge to achieve reliable results. In contrast, our method directly works on RGBD images and there is no restriction on the cuboid configuration: cuboids may have arbitrary pose and they can interact in complex ways. By using a branch and bound global optimization, our method is able to give more reliable results than 2D approaches.

The proposed method is also related to 3D point cloud segmentation. In [13] and [14], points in color point clouds are classified into a small number of categories. Supporting relations between patches are further extracted in [10]. These point cloud or RGBD data segmentation and classification methods do not explicitly extract "volumes" of objects. In [8], range data are pre-segmented and local search is proposed to fit shape primitives. In [9], a greedy scheme is proposed for spatial reasoning and segmenting a 3D scene from stereo into object proposals enclosed in bounding boxes. This method is currently applied to simple scenes and the result is dependent on the quality of object level segmentation. In this paper, instead of trying to segment a 3D scene into regions, we match cuboids to the scene. Our method constructs reliable cuboid candidates by using pairs of planar patches and globally optimizes the cuboid configuration in a novel linear framework.

Finding cuboids in cluttered RGBD images is still unsolved. No previous methods are able to globally optimize the cuboid configuration when there is no restriction on the poses and interactions among objects. In this paper, we propose a linear method that works on generic scenes. The proposed method first partitions the 3D point cloud into groups of piecewise linear patches using the graph method [11]. These patches are then used to generate a set of cuboid candidates, each of which has a cost. We globally optimize the selection of the cuboids so that they have small total cost and satisfy the global constraints. The optimal cuboid configuration has small intersection, and we prefer a large coverage of the cuboids. At the same time, we make sure the cuboids satisfy the occlusion conditions. The optimization is formulated as a mixed integer linear program and efficiently solved by a branch and bound method.

Our contribution is a novel linear approach that efficiently optimizes multiple cuboid matching in RGBD images. The proposed method works on cluttered scenes in unconstrained settings. Our experiments on thousands of images in the NYU Kinect dataset [10] and other images show that the proposed method is efficient and reliable.

2. Method

2.1. Overview

We optimize the matching of multiple cuboids in a RGBD image from Kinect. Our goal is to find a set of cuboids that match the RGBD image and at the same time satisfy the spatial interaction constraint. We construct a set of cuboid candidates and select the optimal subset. The cuboid configuration is denoted by $\mathbf{x}$. We formulate cuboid matching into the following optimization problem,

$$\min_{\mathbf{x}}\{U(\mathbf{x}) + \lambda P(\mathbf{x}) + \mu N(\mathbf{x}) - \gamma A(\mathbf{x}) + \xi O(\mathbf{x})\} \quad (1)$$

s.t. Cuboid configuration $\mathbf{x}$ satisfies global constraints.

Here $U(\mathbf{x})$ is the unary term that quantifies the local matching costs of the cuboids, $P(\mathbf{x})$ is a pairwise term that quantifies the intersection between pairs of cuboids, $N(\mathbf{x})$ is the number of matched cuboids in the scene, $A(\mathbf{x})$ quantifies the covered area of the projected cuboids on the image plane, and O penalizes the occlusions among the cuboids. λ, μ, γ and ξ control the weight among different terms. In this paper, $\mu = 0.1$, $\lambda = \xi = 0.02$ and $\gamma = 1$. By minimizing the objective, we prefer to find the multiple cuboid matching that has low local matching cost, small object intersection and occlusion, and covers a large area in the image with a small number of cuboids. Besides the soft constraints specified by the objective function, we further enforce that the optimal cuboid configuration $\mathbf{x}$ satisfies hard constraints on cuboid intersection and occlusion. This optimization is a combinatorial search problem. In the following, we propose an efficient linear solution.

2.2. Cuboid candidates

We first construct a set of cuboid candidates using pairs of superpixels in the RGBD image. Finding high quality cuboid candidates is critical; we propose a new method as follows.

Partition 3D points into groups: We first use the graph method in [11] to find superpixels on the RGBD image.

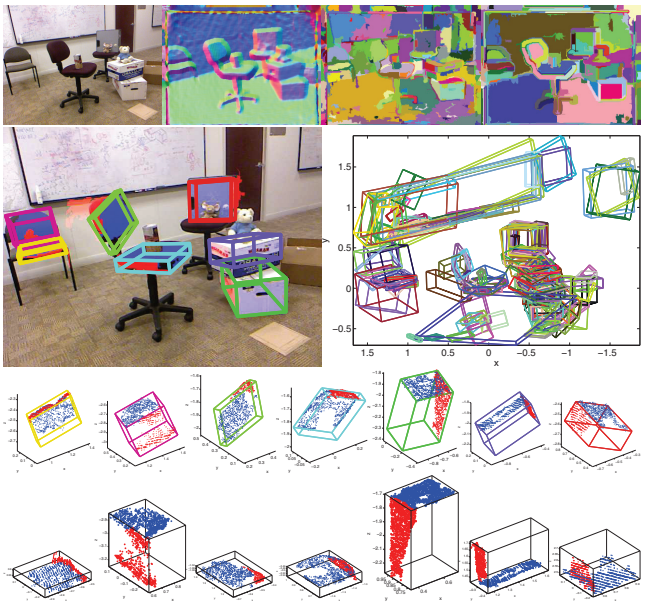


Figure 2. Row 1: From left to right are the color image, normal image with the three channels containing the x, y and z components, the superpixels by using both the color and normal images, and the superpixels by using the normal image only. The two superpixel maps are extracted with fixed parameters in this paper. Row 2: Left shows the cuboids constructed using neighboring planar patches and projected on the image; right shows the top 200 cuboid candidates. Row 3: 3D view of the cuboids in the color image in row 2. The red and blue dots are the points from the neighboring surface patches. Row 4: The normalized poses of these cuboids with three edges parallel to the xyz axises.

With both color and surface normal images, we partition the depth map into roughly piecewise planar patches. As shown in row one of Fig. 2, we also use the superpixels from the normal image itself; this helps find textured planar patches.

Constructing cuboids: We use each pair of neighboring 3D surface patches to construct a cuboid candidate. We define that two surface patches are neighbors if their corresponding superpixels in the color or normal image have a distance less than a small threshold, *e.g.* 20 pixels. The distance of two superpixels is defined as the shortest distance between their boundaries. To remove outliers, we fit a plane to each 3D patch by RANSAC. In the following process, we use only the inlier points.

We are ready to construct cuboid candidates from pairs of patches. We select one of the two neighboring patches and rotate the 3D points in them so that the normal vector of the chosen one is aligned with the z axis. We then rotate the 3D points again so that the projected normal vector of the second 3D patch on the xy plane is aligned with y axis. We then find two rectangles parallel to the xy and xz plane to fit the points on the two 3D patches. The sizes of the two rectangles can be obtained by finding the truncated boundaries of the point histograms in each axis direction. For the first plane the z coordinate is the mean z of the points in the 3D patch, and for the second plane its y is the mean y of the points.

The cuboid is the smallest one that encloses both of the rectangles as shown in Fig. 2 and Fig. 4 (a). Fig. 2 rows 2-4 illustrate some of the cuboids reconstructed from neighboring superpixels. We change the red and blue channels of the color image to show the neighboring superpixels. The projection of these cuboids in Fig. 2 row 2 shows that the 3D cuboid estimation is accurate. Each candidate cuboid is represented by the lower and upper bounds of x, y and z coordinates in the normalized pose and a matrix T that transforms the cuboid back to the original pose. We also keep the inverse of T as F. Such a representation facilitates the computation of cuboid space occupancy and intersection.

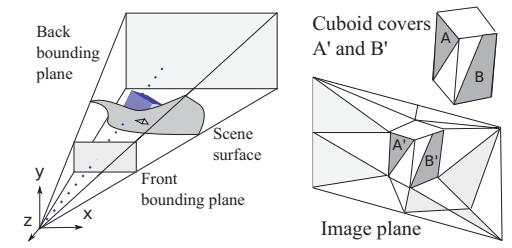
Local matching costs: The quality of the matching of a cuboid to the 3D data is determined by three factors.

The first factor is the coverage area of the points in the two contact cuboid faces. To simplify the computation, the coverage area of the surface points on a cuboid face is determined by the tightest bounding box as shown in Fig. 4 (a). We compute the ratio r of the bounding box area to the area of the corresponding cuboid face. The smaller one of the two ratios are used to quantify the cuboid local matching. A perfect cuboid matching has the ratio r of 1.

The second factor is the solidness. We require that cuboids should mostly be behind the 3D scene surface. To measure the solidness of cuboids, as shown in Fig. 3 (a), we quantize the space into voxels, whose centers are located on the rays starting from the origin point and passing through each image pixel. The space of interest is bounded in 1 to 10 meters from the camera. In the bounded space, 200 points are uniformly selected along each ray. The points behind the scene surface have z coordinates less than the surface z coordinates. To compute the solidness of cuboid i, we transform the points using the cuboid matrix F_i defined before to bring the cuboid to the normalized position. We do not need to transform all the points but only the points inside the bounding box of the cuboid in the original pose; other points are irrelevant. The solidness is approximated by n_s/n_a, where n_s is the number of solid space points in the cuboid and n_a is the number of all the transformed points falling in the cuboid. We keep only the cuboid candidates whose solidness is greater than 0.5.

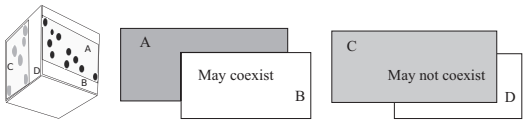
The third factor is the cuboid boundary matching cost. When we project each cuboid candidate to the target image, the candidate's projection silhouette should match the image edges. We find the average distance between the projection silhouette boundary and the edge pixels, which can be computed efficiently using the distance transform of the image edge map. We keep only the cuboid candidates whose silhouettes to edge average distance is less than 10 pixels.

We choose the top M cuboids ranked by the surface matching ratio r with the solidness and average boundary error in specific ranges, *e.g.*, solidness greater than 0.5 and



(a) (b)

Figure 3. (a): We compute the solidness of a cuboid by discretizing the space in front of and behind the scene surface into voxels. (b): We encourage cuboids to cover large area of superpixels on an image.



(a) (b) (c)

Figure 4. (a): The cuboid matching ratio r is $\min(A/B, C/D)$, where A, B, C, D are the areas of the rectangular regions. (b) and (c): The projection of cuboids and their depth order determine the occlusion. Cuboid A and B may coexist, but C and D cannot.

boundary error less than 10 pixels in this paper. The costs of cuboids are $c_i = 1 - r_i, i = 1..M$. In this paper, we keep at most top 200 cuboid candidates.

2.3. Formulation

Local matching is not enough to determine the optimal set of cuboids. Since their matching costs are nonnegative, we would obtain a trivial all-zero solution if we simply minimize the unary term. One method is to specify the number of objects to be included in the scene. However, in practice, the number of cuboids is usually unknown. Another method is to train an SVM cuboid classifier based on the features and the classification result would have positive and negative values. Our experiment shows that the cuboid classifier has quite low classification rate and the top 200 candidates are almost always classified into the same category. We need to incorporate more global constraints and estimate the number of the cuboid objects and their poses at the same time. We propose a linear formulation of the optimization in Eq. (1).

2.3.1 Unary term

We define a binary variable x_i to indicate whether cuboid candidate i is selected. If candidate i is selected, $x_i = 1$, and otherwise $x_i = 0$. The unary term U is the overall local cuboid matching cost, $U = \sum_i c_i x_i$, where nonnegative coefficient c_i is the cost of choosing cuboid candidate i; c_i is defined in section 2.2. There is a guarantee that all the cuboid candidates are at least 50% solid and have projection silhouettes with the average distance of less than 10 pixels to the image edges. Directly minimizing U would give trivial all-zero results. We need extra constraints.

2.3.2 Volume exclusion

Since each cuboid is solid, they tend to occupy non-overlapping space in the scene. However, completely prohibiting the cuboid intersection is too strong a condition due to the unavoidable errors in candidate pose estimation. We set a tolerance value t for the cuboid intersection and in this paper $t = 0.1$, which means cuboids may have up to 10% intersection. Here the intersection ratio of two cuboids is defined as the ratio of the volume intersection to the volume of the smaller cuboid. If one cuboid contains the other, the ratio is 1. The intersection ratio from cuboid i to j is computed by projecting the regularly sampled points in cuboid i in the normalized pose back to the original pose using cuboid matrix T_i and then projecting to cuboid j's normalized pose by using matrix F_j and finally computing the ratio of the inside points to all these projected points. The intersection ratio between cuboid i and j is denoted as $e_{i,j}$, which is the larger one of the two possible intersection ratios.

The volume exclusion term has two parts, the first part is soft and the other is a hard constraint. If two cuboid candidates have intersection less than t, we have the soft term in the objective function, $P = \sum_{\{\{i,j\}: 0<e_{i,j}<t\}} e_{i,j} x_i x_j$. When optimizing the objective function, we try to minimize the intersection between cuboids. We linearize the quadratic term $x_i x_j$ by introducing an auxiliary variable $z_{i,j}$ and letting $z_{i,j} \leq x_i$, $z_{i,j} \leq x_j$, $z_{i,j} \geq 0$, and $z_{i,j} \geq x_i + x_j - 1$. It can be verified that $z_{i,j}$ is indeed the product of x_i and x_j.

Apart from the soft term, we prohibit any two cuboids from having intersection ratio that is greater than or equals t. The hard constraint is thus $x_i + x_j \leq 1$ for each pair of cuboid candidates i and j that have volume intersection $e_{i,j} \geq t$. The hard constraint ensures that cuboid candidates whose intersection ratio is at least t do not coexist; the corresponding soft intersection penalty in the objective function is zero.

2.3.3 Surface coverage

The volume exclusion constraint ensures the correct space occupancy of the selected cuboids. However, it still does not solve the trivial all-zero solution problem, *i.e.*, if we simply minimize the unary and pairwise terms no cuboids will be selected. To solve the problem, we introduce a surface coverage term to encourage the selected cuboids to cover large surface area. We define that a cuboid covers the surface patches by which we construct the cuboid. And, we say a cuboid covers a superpixel in an image, if it covers the corresponding 3D scene patch, as illustrated in Fig. 3 (b). We define a variable y_k, which is 1 if superpixel k is covered by a selected cuboid, and otherwise 0. The surface coverage term is $A = \sum_k a_k y_k$, where a_k is the area of superpixel

k. As we minimize the objective function, a large coverage area is preferred.

Superpixel indicator variable y_k and the cuboid indicator variable are correlated by

$$y_k \leq \sum_{\text{cuboid } i \text{ covers superpixel } k} x_i, \quad 0 \leq y_k \leq 1.$$

If all the cuboids that cover superpixel k are not selected, $y_k = 0$. If at least one cuboid that covers superpixel k is selected, $y_k = 1$ and the term A is maximized. Therefore, when the objective is optimized, A is the covered area of the selected cuboids in the image.

Since cuboids may come from color-normal superpixels or normal superpixels, the superpixels may overlap. We construct fine-grained superpixels, each of which is a subregion of a superpixel from the two superpixel sets. Variables y_k correspond to these fine-grained superpixels. The surface coverage term thus encourages the selection of a set of cuboids that not only have low local matching cost but also cover a large region in an image.

2.3.4 Smaller number of cuboids

We prefer to use a small number of cuboids to explain the scene. The number of cuboids, $N = \sum_i x_i$, is introduced into the objective function. With the surface coverage term, the small number heuristic indicates that we tend to choose few large cuboids instead of many small ones. The cuboid number term and the unary term are merged in the objective function in Eq. (2).

2.3.5 Occlusion constraints

The selected cuboids should have small occlusion among each other from the camera view. If a cuboid is completely occluded by another cuboid from the camera view, one of them has to be a false detection. Apart from complete occlusion, partial occlusion may indeed happen. We therefore introduce a soft term O to penalize the partial occlusion among the chosen cuboids, and we use hard constraints to prohibit the complete occlusion.

Occlusion happens if two cuboids' projection regions on the image plane overlap. The occlusion can thus be defined as $\mathcal{A}(H \cap Q)/\mathcal{A}(Q)$, where H is the projected region of the closer cuboid and Q is the projected region of the farther cuboid, $H \cap Q$ is the intersection of the two projection regions, and $\mathcal{A}(.)$ extracts the area of a region. To test which cuboid is closer, we first find the cuboid surface points corresponding to the common projected region of both cuboids and then we compare their average distances to the camera center. To count for the estimation errors, we define that there is a partial occlusion if the overlap ratio is less than u, *e.g.* 0.75 in this paper, and otherwise there is a complete occlusion. The occlusion ratio between cuboid i and j is denoted as $q_{i,j}$. The occlusion reasoning is illustrated in Fig. 4 (b) and (c).

We introduce pairwise variable $w_{i,j}$ for cuboid candidates i and j that are partially occluded. We penalize the partial occlusion in the objective function, $O = \sum_{\{\{i,j\}:\, 0<q_{i,j}<u\}} q_{i,j} w_{i,j}$, where $q_{i,j}$ is the occlusion ratio. Similarly to the pairwise cuboid intersection penalty term, we let $w_{i,j} = x_i x_j$. In an equivalent linear format, $w_{i,j} \leq x_i$, $w_{i,j} \leq x_j$, $w_{i,j} \geq 0$, and $w_{i,j} \geq x_i + x_j - 1$. If there is a complete occlusion between cuboid candidates i and j, they cannot coexist. We thus let $x_i + x_j \leq 1$, if $q_{i,j} \geq u$.

2.4. Optimization

By combining these terms, we have a complete mixed integer linear optimization:

$$\begin{aligned}
&\min\{\sum_i (c_i + \mu) x_i + \lambda \sum_{\{i,j\}} e_{i,j} z_{i,j} - \gamma \sum_k a_k y_k + \\
&\quad \xi \sum_{\{i,j\}} q_{i,j} w_{i,j}\} \qquad (2)\\
&\text{s.t.} \quad z_{i,j} \leq x_i, z_{i,j} \leq x_j, z_{i,j} \geq x_i + x_j - 1,\\
&\quad \forall \{i,j\} \text{ that } 0 < e_{i,j} < t\\
&\quad y_k \leq \sum_{\text{cuboid } i \text{ covers superpixel } k} x_i,\ y_k \leq 1,\ \forall \text{ superpixel } k\\
&\quad w_{i,j} \leq x_i, w_{i,j} \leq x_j, w_{i,j} \geq x_i + x_j - 1,\\
&\quad \forall \{i,j\} \text{ that } 0 < q_{i,j} < u\\
&\quad x_i + x_i \leq 1, \forall \{i,j\} \text{ that } e_{i,j} \geq t \text{ or } q_{i,j} \geq u\\
&\quad x_i = 0 \text{ or } 1. \text{ All variables are nonnegative.}
\end{aligned}$$

We solve the mixed integer linear program efficiently by branch and bound. We use the linear program relaxation that discards the integer constraints to obtain the lower bound. The upper bound is initialized by sequentially picking up low cost cuboids that do not have space or occlusion conflict with previous selections. A branch and bound search tree is expanded at the branch with the lowest active lower bound on variable x whose value is the closest to 0.5. In the two new branches, we set the chosen x to 0 and 1 respectively. We update the lowest upper bound and the lowest active lower bound in each step. A branch is pruned if there is no solution or if the lower bound by linear programming is greater than the current upper bound. The lower bound can be efficiently computed using dual simplex method since only one variable changes the value. The procedure converges quickly. With 200 cuboid candidates and a tolerance gap of 0.01, the optimization takes less than 1 second in a 2.8GHz machine. There is in fact no need to set a tight tolerance gap between the upper and lower bounds; a large gap such as 0.5 gives little performance degradation comparing to a small one such as 0.01 and enables even faster convergence.

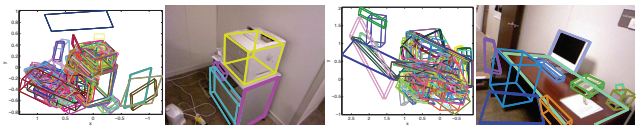


Figure 5. Top 200 cuboid candidates and the cuboids matched by the proposed method and projected on the color images.

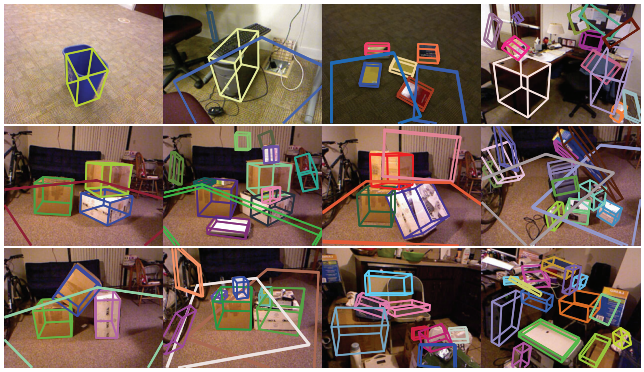


Figure 6. Sample cuboid matching results with the proposed method on our captured data.

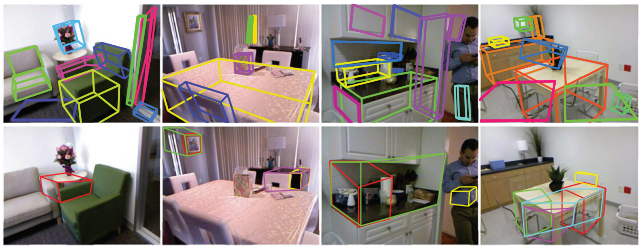


Figure 7. Comparison with 2D cuboid detection method [15]. The proposed method's result (Row 1) is more reliable than the result from 2D images only (Row 2).

3. Experiments

3.1. Qualitative evaluation

Fig. 5 shows two examples of the cuboid matching results from the proposed method. More results on our captured data are shown in Fig. 6. We further apply the proposed method to finding cuboids in the 1449 RGBD images from the NYU Kinect dataset [10]. Sample results of the proposed method on the NYU dataset are shown in Fig. 9. Comparing with a method [15] that uses only color images for cuboid detection, as shown in Fig. 7, the proposed method gives more reliable results. The proposed method is able to handle concave objects, objects with or without texture, large objects such as tables and small ones such as books. It works well on cluttered scenes.

3.2. Quantitative evaluation

We compare the proposed method with its variation and competing methods using the NYU dataset. We labeled 215 images from the 1449 images. These images are from different scene categories. Sample images with the ground truth projection overlaid are shown in Fig. 8. We extract

Figure 8. Sample images with ground truth cuboid labeling. The saliency of the labeled cuboids are denoted by the color: the warmer the color, the more salient the object is. There are 215 ground truth images and totally 476 labeled 3D cuboids.

the 3D corner coordinates of the cuboids in images using a semi-automatic method. We choose two vertical planes of a cuboid by marking the regions on the color image. The two-plane cuboid reconstruction method is used to extract the 8 corner coordinates of the cuboid. Note that we do not need to mark a complete rectangular region in each cuboid face. Instead, we just need a patch that extends to the face's width and height. The two-plane method reconstructs the cuboid automatically. Such a process sometimes needs to iterate for a few times to obtain a satisfactory 3D labeling. During the labeling, we also specify the sequence of the saliency of labeled cuboids. The object with label one is the most salient and usually it is the biggest cuboid object in the scene. Ties of saliency are broken randomly. Note that even though ground truth 2D segmentations are available for the NYU Kinect dataset, they cannot be used directly for the evaluation because our task is to match cuboids in the images.

Comparison with greedy method: We compare the proposed method with a greedy method that chooses the top 20 cuboids with the smallest local matching costs. Our method detects less than 20 objects and 6 on average per image in the ground truth test. To evaluate the matching performance, we compute the average corner distance from each ground truth cuboid to all the detections and find the minimum distance. There is no predefined order for the 8 corners; we use bipartite matching to compute the best matching configuration for each pair of cuboids. We further normalize the average distance by the longest diagonal of the ground truth cuboid. We use the detection rate curve to quantify the performance. The curves are shown in Fig. 10, which illustrate the detection rate for top saliency, top two, top three and top ten objects in each image. The proposed method has much higher detection rate than the greedy approach. Local matching costs themselves are not reliable. We need to globally optimize all the cuboid matching together to achieve a reliable result.

Comparison with local search method [5] and MCMC [16]: These local search methods need to initialize the cuboid location, size and orientation. We randomly initialize these parameters. For each test image, we detect 50 cuboids with these competing methods. Color is not used by these methods. As shown in Fig. 10, the proposed method has much higher detection rate than these local methods. The complexity of the proposed method is also lower than

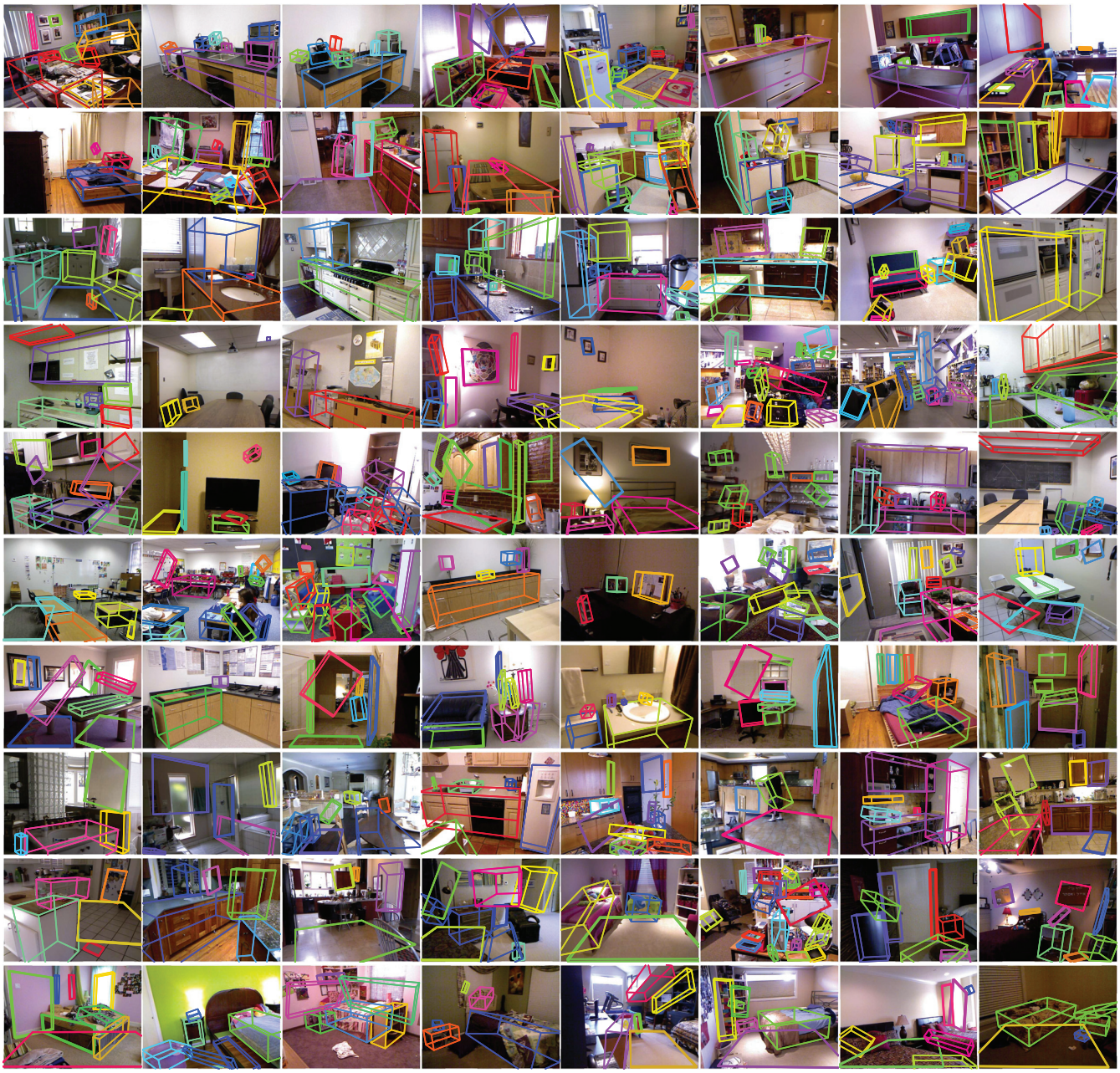


Figure 9. Randomly sampled results of the proposed method on the 1449 images from the NYU Kinect dataset [10]. We show the projected cuboids in the images. Our method gives reliable result in cluttered scenes.

these competing methods.

Comparing with bottom-up approaches: The above competing methods are top-down. We would like to compare with potential bottom-up competing methods. The segmentation method in [9] also generates object bounding boxes and thus can potentially be used to find cuboids in RGBD images. This method relies on hierarchical superpixels. If the object level segment is not available, the corresponding object cannot be detected. Assuming that we can recover a cuboid perfectly from a superpixel, the chance that the segmentation forms a complete object sets an upper bound that a bottom-up method can achieve when finding cuboids. We compare the proposed method with this upper bound.

Here we use the region overlap ratio to quantify the quality of a detection. To obtain a cuboid region in the target image for the proposed method, we project the cuboid's corners to the image plane and find the convex hull. We use the method in [9] to obtain a large set of superpixels using different parameters followed by merging neighbors progressively based on the color similarity and coplanar degree. We check how the superpixels and our cuboid projection over-

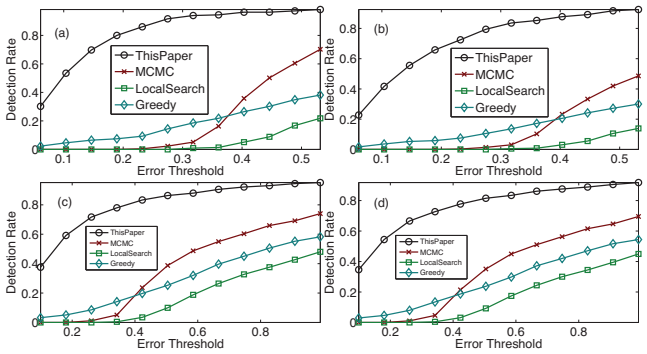


Figure 10. Comparing with greedy, local search [5] and MCMC [16] methods. We show the rate of finding objects when the distance error threshold changes for (a) top one, (b) top two, (c) top three and (d) top ten salient cuboids in each image.

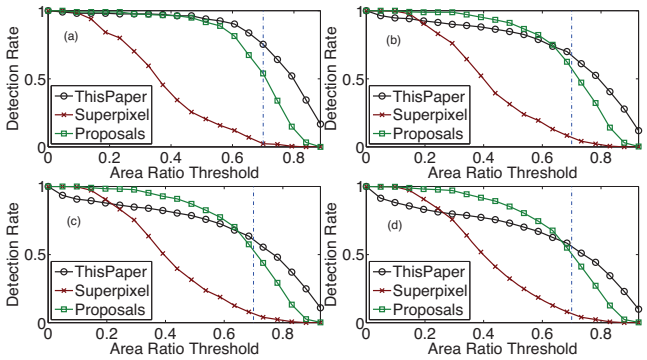


Figure 11. Detection rate comparison with bottom-up methods. We detect (a) the most salient object, (b) the top two, (c) the top three, and (d) the top ten objects in each image. The 0.7 threshold, above which the detection is visually correct, is marked with a vertical blue line.

lap with the ground truth object foreground. The region overlap is quantified by the ratio of the region intersection to the region union. We plot the overlap ratio threshold *vs.* detection rate curve in Fig. 11. The proposed method gives better results than the superpixel based approach. This is not a surprise. The hierarchical superpixel method is not always able to merge two faces of a cuboid if they have very different color and texture.

Another method to generate object level regions is by object class independent proposals [17]. The original region proposal method is for color images only. For fair comparison, we include the 3D point coordinates into the superpixel detection. We adjust the weight between the rgb and xyz vectors to achieve the best performance. The proposal regions are more likely to match the cuboid objects than the superpixels as shown in Fig. 11. However, the region proposal method also has a hard time to find many cuboid objects especially large objects with complex textures. Above the region overlap threshold 70%, the detection rate of the proposed method is always higher than that of the region proposal method. The object independent proposal method has higher detection rate if the region overlap threshold is less than 0.6 because it uses many more region detections – several thousand, comparing to less than 20 detections of the proposed method. In fact, the low threshold detection rate is not important because a detection that has less than 60% overlap with the ground truth is most likely wrong and it will be hard to recover the 3D cuboid object using such segments. The bottom-up region based method is also more complex. There are often 5000 region candidates and it takes 5-10 minutes to find them in a 2.8GHz machine. Our proposed method uses many fewer cuboid candidates; it is more efficient. The optimization takes less than one second with the same machine. The whole process that includes extracting candidates, computing intersection and occlusion takes less than one minute per image.

4. Conclusion

We propose a novel method to match cuboids in RGBD images. The proposed linear method guarantees the global optimization of cuboid matching. It also automatically determines the number of cuboids in the scene. Our experiments on thousands of RGBD images of a variety of scenes show that the proposed method is accurate and it is robust against strong clutter. We believe that this method is a useful component to help 3D scene understanding.

Acknowledgment: This research is supported by the U.S. NSF funding 1018641. Xiao is supported by Google U.S./Canada Ph.D. Fellowship.

References

[1] Y. Li, X. Wu, T. Chrysathou, A. Sharf, D. Cohen-Or, N.J. Mitra, "GlobFit: Consistently Fitting Primitives by Discovering Global Relations", Siggraph'11.
[2] V. Hedau, D. Hoiem, D. Forsyth, "Recovering Free Space of Indoor Scenes from a Single Image", CVPR'12.
[3] Y. Zhao, S.C. Zhu, "Image Parsing via Stochastic Scene Grammar", NIPS'11.
[4] R. Schnabel, R. Wessel, R. Wahl, and R. Klein, "Shape Recognition in 3D Point-Clouds", WSCG'08.
[5] R. Truax, R. Platt, J. Leonard, "Using Prioritized Relaxations to Locate Objects in Points Clouds for Manipulation", ICRA'11.
[6] D.C. Lee, A. Gupta, M. Hebert, and T. Kanade. "Estimating Spatial Layout of Rooms using Volumetric Reasoning about Objects and Surfaces", NIPS'10.
[7] A Leonardis, A. Jaklic, and F. Solina, "Superquadrics for Segmenting and Modeling Range Data", TPAMI, vol.19, no.11, 1997.
[8] S.J. Dickinson, D. Metaxas, A. Pentland, "The Role of Model-Based Segmentation in the Recovery of Volumetric Parts From Range Data", TPAMI, vol.19, no.3, 1997.
[9] M. Bleyer, C. Rhemann, C. Rother, "Extracting 3D Scene-Consistent Object Proposals and Depth from Stereo Images", ECCV'12.
[10] N. Silberman, D. Hoiem, P. Kohli and R. Fergus, "Indoor Segmentation and Support Inference from RGBD Images", ECCV'12.
[11] P.F. Felzenszwalb and D.P. Huttenlocher, "Efficient Graph-Based Image Segmentation", IJCV, vol.59, no.2, 2004.
[12] A. Gupta and A.A. Efros and M. Hebert, "Blocks World Revisited: Image Understanding Using Qualitative Geometry and Mechanics", ECCV'10.
[13] H.S. Koppula, A. Anand, T. Joachims, A. Saxena, "Semantic Labeling of 3D Point Clouds for Indoor Scenes", NIPS'11.
[14] N. Silberman and R. Fergus, "Indoor Scene Segmentation using a Structured Light Sensor", ICCV Workshop on 3D Representation and Recognition, 2011.
[15] J. Xiao, B.C. Russell, and A. Torralba, "Localizing 3D Cuboids in Single-view Images", NIPS'12.
[16] H. Huang, C. Brenner, M. Sester, "3D Building Roof Reconstruction from Point Clouds via Generative Models". GIS'11.
[17] I. Endres, D. Hoiem, "Category Independent Object Proposals", ECCV'10.
[18] L.D. Pero, J. Guan, E. Brau, J. Schlecht, K. Barnard, "Sampling Bedrooms", CVPR'11.