Data-Driven Indoor Scene Modeling from a Single Color Image with Iterative Object Segmentation and Model Retrieval

Mingming Liu, Kexin Zhang, Jie Zhu, Jun Wang, Jie Guo, and Yanwen Guo

Abstract—We propose a new method for modeling the indoor scene from a single color image. With our system, the user only needs to drag a few semantic bounding boxes surrounding the objects of interest. Our system then automatically finds the most similar 3D models from the ShapeNet model repository and aligns them with the corresponding objects of interest. To achieve this, each 3D model is represented as a group of view-dependent representations generated from a set of synthesized views. We iteratively conduct object segmentation and 3D model retrieval, based on the observation that good segmentation of the objects of interest can significantly improve the accuracy of model retrieval and make it robust to cluttered background and occlusions, and in turn, the retrieved 3D models can be used to assist with object segmentation. Segmentation of all objects of interest is achieved simultaneously under a unified multi-labeling framework which fully utilizes the correspondences between the objects of interest and retrieved model images. Besides, we propose a new method to estimate the scene layout of the input image with the segmentation masks, which helps compose the resulting scene and further improves the modeling result remarkably. We verify the effectiveness of our approach through experimenting with a variety of indoor images and comparing against the relevant methods.

Index Terms—Indoor scene modeling, data-driven, object segmentation, model retrieval, layout estimation

1 Introduction

Indoor scene modeling is a long-studied problem in the Graphics community, due to its wide applications to 3D interior design, game development, and robot navigation. Compared with outdoor scenes, indoor scenes are often more constrained since they behave mostly as Manhattan Worlds and are comprised of a limited variety of basic elements such as walls, windows, doors, and furniture objects including chairs, tables, beds, cabinets and sofas, etc.

Recently, several methods [1], [2], [3], [4] have been proposed to model indoor scenes with a depth camera (e.g., Microsoft Kinect). Additionally, indoor scene modeling can be achieved in a data-driven manner with the aid of a 3D model database such as the Sketchup 3D Warehouse or the ShapeNet model repository [5], [6], [7]. Despite these efforts, recovering 3D geometries of the scene from a single image is still known to be an enormously difficult and unsolved problem. The challenge lies in that the problem is intrinsically ill-posed, since the 2D image cannot embody the 3D information about the scenes and object geometries. The cluttered background and object occlusion further complicate this problem.

Our goal in this paper is to reconstruct the 3D indoor scene given only a single color image. Due to scene ambiguity and information loss, it is usually impossible to accurately recover the complete scene with great details. Therefore, we attempt to make the reconstructed scene geometries of selected objects of interest (OOIs) as close as possible compared to the original image.

Our method is data-driven with the assistance of the ShapeNet 3D model repository [8]. Given an indoor scene image, we try to find the most similar 3D models from the database as compared to the corresponding OOIs and align them properly. To accomplish this, it is essential to figure out what objects are in this image and where they are located. Though object detection and semantic segmentation have been well studied, the results are not always reliable, especially for cluttered scenes with severe occlusions. To circumvent this, we ask the user to simply drag a few semantic bounding boxes of these objects to indicate their locations and categories. This is the only interaction required by our system.

Even with known object categories and locations, modeling the scene automatically and accurately still faces two difficulties, i.e., how to find the most similar model for each object effectively and how to accurately estimate its spatial location and pose. We make the problem tractable by taking the matching between the OOI with view-dependent representations of 3D models in the same category as the

- *M. Liu, K. Zhang, J. Zhu, J. Guo and Y. Guo are with the National Key Lab for Novel Software Technology, Nanjing University, Nanjing, Jiangsu 210023, China. Y. Guo is also with the Science and Technology on Information Systems Engineering Laboratory, China. E-mail: ieliuming@163.com, claireikxiu@gmail.com, magickuang@126.com, {guojie, ywguo}@nju.edu.cn.*
- *J. Wang is with the College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu 210016, China. E-mail: wjun@nuaa.edu.cn.*

Manuscript received 29 Jan. 2018; revised 16 Oct. 2018; accepted 4 Nov. 2018. Date of publication 12 Nov. 2018; date of current version 4 Mar. 2020. (Corresponding author: Yanwen Guo.) Recommended for acceptance by T. Ju. Digital Object Identifier no. 10.1109/TVCG.2018.2880737

objective, where model retrieval and pose estimation are optimized simultaneously.

We find that better object segmentation facilitates model retrieval, since the influence from other objects and background can be eliminated. On the other hand, more accurate retrieval benefits segmentation as the appearance of retrieved models provides importance guidance for image labelling. Based on these observations, we perform object segmentation and 3D model retrieval alternatively. In the step of object segmentation, we achieve the segmentation of all the OOIs simultaneously under a Markov Random Field (MRF) based unified multi-labeling framework which fully exploits the correspondences between the OOIs and retrieved model images. In the step of model retrieval, we search for optimal models by matching only each object mask with the models in the same category. This makes our method robust to the challenges such as cluttered background, occlusion, and so on. The two steps iterate until no changes on the results are made. Finally, we estimate the layout of the scene and refine the pose for each OOI with the obtained segmentation. This further improves the scene modeling result remarkably.

Contributions. To summarize, this paper makes the following contributions:

- A new method for indoor scene modeling from a single image.
- An iterative optimization strategy for object segmentation and model retrieval.
- A layout estimation approach with pose refinement for accurately composing the scene.

2 Related Work

2.1 Image-Based Scene Modeling

The problem of scene modeling from a single image or multiple images has received considerable attention. The image structure can be obtained by significant amount of user interactions [9], or by exploring the learned relationships between image features and geometry [10], [11], [12], [13], [14], [15], [16], [17]. Part-based correspondences between 3D CAD models and real photos can be established by framing the task as a 2D-to-3D alignment problem via resorting to the mid-level visual elements [18]. More recent efforts have exploited the depth for each image pixel can be learned from special cues [19] or plenty of examples [20]. Liu et al. [21], [22] present an interactive approach for semantically modeling the indoor scenes by utilizing estimated normal maps, which is only applicable to the object with mutually orthogonal vanishing lines. Surface normal is also leveraged by Bansal et al. [23] to retrieve 3D models from a large CAD object library. Massa et at. [24] present an end-to-end convolutional neural network (CNN) for 2D-3D exemplar detection, which shows state-of-the-art exemplar detection performance. Compared with these approaches, the core to our method is to iterate between 3D model retrieval and object segmentation. Segmentation of multiple OOIs plays a key role in retrieving the best matched models and in improving layout estimation. Our method is intrinsically more robust to complicated scenes with cluttered background and occlusions.

2.2 Modeling from RGBD Images

With the popularity of consumer-level RGBD cameras, indoor scenes modeling from RGBD images has received more and more interests [1], [2], [3], [4]. However, it remains a challenging problem because of the complex structure of interior objects and poor qualities of RGBD data acquired by consumer-level sensors [25]. To achieve semantic modeling of indoor scenes, Shao et al. [5] present an interactive approach by first segmenting the captured RGBD images into regions with object labels, and then replacing the segmented objects with matched models in the database. Another approach that discovers object repetitions and exploits them to speed up large scale indoor acquisition towards high-level scene understanding is given by Kim et al. [26]. Other efforts include the search-classify approach which interleaves segmentation and classification in an iterative manner [6]. Chen et al. [7] propose an automatic semantic modeling approach by using the contextual relationships learned from the database of indoor scenes to constrain reconstruction. Gupta et al. [27] align 3D models to RGBD images of cluttered scenes using a CNN. Choi et al. [28] present an approach to indoor scene reconstruction from RGBD videos by combining geometric registration of scene fragments with robust global optimization based on line processes. Compared with these approaches, our method requires only a single color image as input, thus being applicable to those existing images for which the scenes are not available to scan.

2.3 Indoor Scene Understanding

Our goal is to reconstruct the indoor scenes. There has been considerable research on indoor scene understanding, whose goal is to infer 3D properties of object components, annotate them with a list of tags [29], and infer scene structures [30], [31], [32], [33]. Given an image, we now ask the users to drag a few semantic bounding boxes to indicate object locations and the corresponding categories. As a pre-processing step, semantic segmentation [34], [35] with the aim of realizing image segmentation and object annotation in a unified framework is applicable to our approach.

3 Overview

This section summarizes the main steps in our system. Fig. 1 shows the overall pipeline of our system.

In the training stage (Section 4.1), we render each model under a group of virtual viewpoints sampled from the viewing sphere. For each viewpoint, the rendered image is represented as a set of mid-level visual elements, serving as a view-dependent representation of this model.

Given a single indoor color image, the user first drags a few semantic bounding boxes indicating the locations of OOIs. The depth order of these OOIs can be automatically obtained with the method of [36].

We then conduct object segmentation and 3D model retrieval iteratively (Section 4.2). In the step of model retrieval, we use HOG based mid-level visual elements to retrieve the similar models with the object mask obtained by segmentation. In the step of object segmentation, we build a MRF model to characterize the probabilities of certain labels for each image pixel, with the assistance of retrieved rendered models. Object segmentation and 3D model retrieval promote each

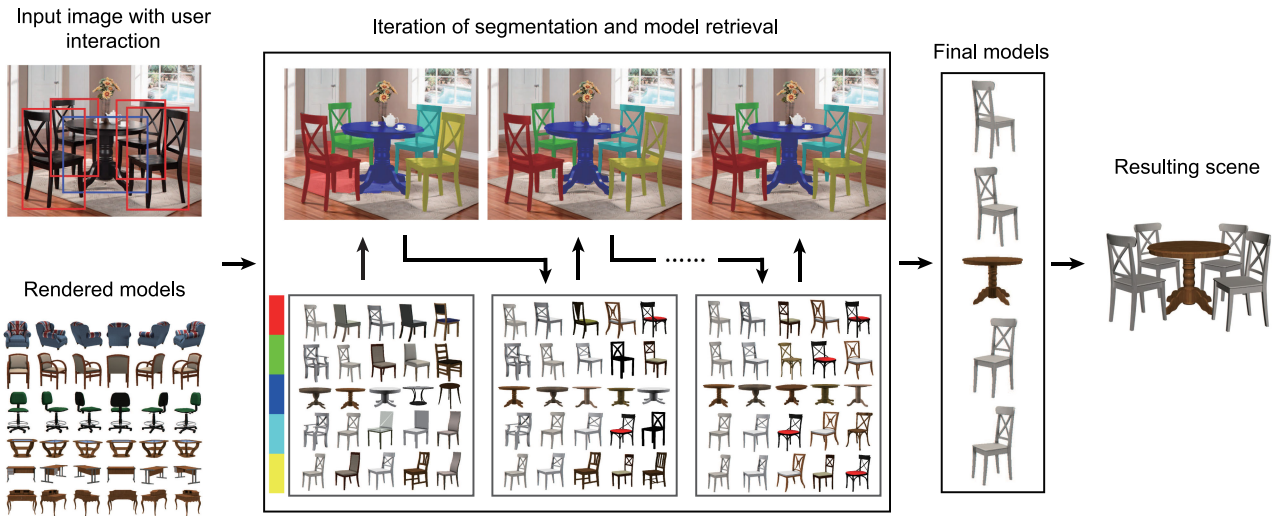


Fig. 1. The pipeline of our framework. The user first drags a few semantic boxes surrounding the objects of interest . We then conduct object segmentation and 3D model retrieval iteratively, yielding the optimal models with rough poses. Finally, we estimate the layout of the scene and refine the poses of all the objects under a unified framework. The 3D models are composed together to obtain the resulting scene.

other effectively. The iteration stops when the retrieved top K models do not change.

In order to reconstruct an accurate and harmonious scene for the input image, we further estimate the layout of the scene (Section 4.3). We compute the 3D locations and poses of the objects with the aid of segmentation masks. The result of scene modeling is further refined in this way.

4 The Proposed Method

4.1 View-Dependent Model Representation

The ShapeNet repository has more than 3,000,000 indoor and outdoor models from which we select a subset with about 20,000 furniture models for testing. This subset contains several classes of common furniture objects in indoor scenes, including chair, sofa, table, bed, cabinet, bookshelf and ashcan. We further collected 3D models for furniture categories not contained in the ShapeNet repository, including bench, ottoman and TV stand. Table 1 shows the statistical data of the models we have used. For each 3D model, we render it on a white background under 72 viewpoints sampled over the upper half of the viewing sphere centered on it. Specifically, the selected latitudes of each model category are listed in Table 1 while the longitudes are uniformly distributed with a 10° interval, resulting in total 72 samples and hence 72 synthesized images. Just as [18], the latitudes are selected by considering the viewpoints under which human sees the objects of different categories the most often. For each synthesized image, we extract a set of HOG-based mid-level visual elements which is used as a view-dependent representation of this model [18]. In this way, a 3D model has 72 view-dependent representations corresponding to the 72 viewpoints. For each user-specified bounding box, we will use such representations to find a group of models that match well with the OOI in it, associated with the corresponding poses.

TABLE 1
Statistical Data of Selected 3D Models

Category	Number of models	Rendering latitudes
Chair	6778	10°, 30°
Sofa	3173	10°, 30°
Table	8436	10°, 30°
Bed	233	0°, 20°
Cabinet	1571	10°, 30°
Bookshelf	452	0°, 10°
Ashcan	342	10°, 30°
Bench	25	10°, 30°
Ottoman	56	10°, 30°
TV stand	64	10°, 30°

4.2 Iterative Object Segmentation and Model Retrieval

The core to our method is an iterative method for OOI segmentation and 3D model retrieval. The two procedures promote each other greatly. On one hand, the retrieved 3D models help segment out all the OOIs by solving a Markov Random Field (MRF) based multi-labeling problem which fully exploits the correspondences between the OOI and retrieved model images. On the other hand, more accurate segmentation in turn facilitates the selection of optimal models. This procedure normally converges after 3 to 4 iterations, when the top K retrieved 3D models for each object keep unchanged. The iteration of segmentation and model retrieval in Fig. 1 illustrates the process of the iterative method.

4.2.1 *Object Segmentation*

Traditional image segmentation methods [37], [38], [39] mainly rely on color appearance models. In comparison, our method achieves better segmentation results thanks to the additional information brought in by those synthesized images of matched models.

With the help of matched images, we aim to segment all the OOIs simultaneously under a unified framework. Such treatment is better than handling each OOI independently, since the latter strategy would fail if an OOI is occluded by its neighbors with similar appearance. By framing segmentation of all objects as a unified segmentation problem, objects would compete with each other. Occlusions among objects can be naturally handled.

Fig. 2. An example of matching between an input patch and a synthetic overlapping patch. Frow left to right: Input image, synthetic scene, the object label image and contour image of the synthetic scene. The rectangles indicate the patch regions.

The segmentation is formulated as a multi-labeling problem in which each pixel is assigned to one of the OOIs. We exploit the pixel-level correspondences between the top K matched synthesized images and the corresponding rectangular object region in input image to assist segmentation. We reformulate the foreground labeling problem as a MRF. Each node in the MRF represents an image pixel. A foreground status $F \in [0, N]$ is defined on these nodes, where N denotes the number of OOIs. It takes 0 for the background and a non-zero integer for the corresponding OOI. We use U to represent the union of image pixels and those matched model images. Following the Bayes' theorem, the joint posterior probability over F given U can be factorized as

$$P(F|U) = \frac{P(U|F)P(F)}{P(U)}. \tag{1}$$

The likelihood term is defined as

$$P(U|F) = \prod_{s \in S} exp(-\phi(s, f_s, U)), \tag{2}$$

where S denotes the set of image pixels. The evidence function ϕ is defined as

$$\phi(s, f_s, U) = L_C(s, f_s, U) + \mu L_R(s, f_s, U), \tag{3}$$

measuring the compatibility between the foreground status f_s and pixel s. This formula contains two cost functions L_C and L_R, which will be explained in details later, measuring dissimilarity of colors and the features obtained from model images respectively, as well as a weight μ computed as

$$\mu = \eta \max(1 - d_c - u_s, 0.1), \tag{4}$$

where d_c is the discrimination of color which is defined as the ratio of pixels inside the corresponding bounding box region possessing the color not appeared outside the region, to the overall pixels inside the bounding box. u_s is the uncertain degree of object label obtained from the matched model images, which is defined as the ratio of pixels inside the corresponding bounding box region possessing the label confidence ranged from 0.2 to 0.6, to the overall pixels inside the bounding box.

The first cost function L_C is defined as

$$L_C(s, f_s, U) = \begin{cases} -log\, \mathcal{G}_b(s) & f_s = 0, m_{0,s} = 0 \\ -log\, \mathcal{G}_{f_s}(s) & f_s > 0, m_{f_s,s} = 1 \\ +\infty & f_s > 0, m_{f_s,s} = 0 \\ 0 & otherwise \end{cases} \tag{5}$$

where $\mathcal{G}_{f_s}$ and $\mathcal{G}_b$ are GMMs built for OOIs and background based on the current foreground segmentation, respectively.

Fig. 3. An example of object label and contour transfer. The input image is shown on the left, and the transferred label probabilities are shown in the middle. The right is the transferred contours.

Binary variable $m_{f,s} \in M$ takes 1 if the pixel s is in the fth box and 0 otherwise, in which M represents the union of user-specified bounding boxes.

The second cost function L_R is defined with the help of the matched synthesized model images. We observe that object labels and contours can be effectively transferred from the matched model images to the input image by SIFT flow. Specifically, the input image is divided into overlapped patches. For each patch, we train an Exemplar-SVM [40]. On the other hand, we put the top K matched images as well as their label images onto the locations where the corresponding user-specified bounding boxes locate, according to the estimated depth sequence (see Fig. 2). In this way, the scene context of the input image can be simulated. Each object corresponds to K model images, resulting in a large number of synthetic overlapping images. For each overlapped label image, we also detect its contours. For each patch in the input image, we match the trained Exemplar-SVM with the patches of all possible synthetic overlapping images in its neighboring regions and select the one with the highest score as its matched patch. The label and contour information of the matched patch is also recorded. SIFT flow is used to accomplish pixel-wise matching between the input patch and its matched synthetic overlapped patch. Through pixel-wise matching, label and contour information of the synthetic overlapped patch is easily transferred to the input patch. The energies on SIFT flow matching are used to calculate the probabilities for both object label and contour. From Fig. 3 (middle), it is clear that the labels of pixels with similar colors are quite different in which case the color GMM would fail. And the border between the two chairs in the red rectangle of the input image is obscure, while the edge is successfully detected in the right image by contour transfer. By contrast, an edge in the green rectangle of the input image exists, while no contour is detected as shown in the right image.

With C and B representing object label confidence and contour confidence, respectively, we define L_R as

$$L_R(s, f_s, U) = \begin{cases} 1 - c_{0,s} & f_s = 0, m_{0,s} = 0 \\ 1 - c_{f_s,s} & f_s > 0, m_{f_s,s} = 1 \\ +\infty & f_s > 0, m_{f_s,s} = 0 \\ 0 & otherwise \end{cases} \tag{6}$$

where $c_{f_s,s}$ is the object confidence of label f_s at pixel s.

The conditional probability of a node in the MRF depends only on its neighboring nodes. Let E denote the edge set in the 4-connected image graph. The prior $P(F)$ is calculated as

Fig. 4. Segmentation improves model retrieval results. Each row represents a group. (a): The objects of interest. (b): The segmentation information in different forms. (c): The retrieved models, in which the top row is the results obtained without segmentation and the bottom row with segmentation information.

$$P(F) = \prod_{(s,t)\in N} exp(-\lambda(\varrho(s,t) + \mu b(s,t))\delta(f_s, f_t)), \quad (7)$$

where s and t are two neighboring pixels in the input image, with f_s and f_t being their foreground statuses. λ is the weight used to balance prior and likelihood. $\varrho(s,t)$ is the color difference between s and t. $b(s,t)$ is the maximum of b_s and b_t. We compute $\varrho(s,t)$ as [39]. $\delta(i,j)$ takes 1 if $i \neq j$ and 0 otherwise.

We optimize $P(F|U)$ in an iterated manner as [39]. In each iteration, we update the foreground GMMs $\mathcal{G}_{f_s}$ and background GMM $\mathcal{G}_b$, and compute the likelihood and prior. Graph cut [41] is used to compute the optimal labels of foreground. The iteration terminates until the changing of foreground objects between two iterations is subtle.

4.2.2 Model Retrieval

To retrieve the most similar 3D models, we match HOG features extracted from the corresponding rectangular object region to the view-dependent model representations. As no segmentation information is obtained in the beginning, we conduct 2D-3D matching with the source image as [18]. Unlike Aubry et al. [18], the sliding window manner is not required since the bounding box of each object is already provided by the user. We simply compute HOG features at the same locations and scales with the view-dependent model representations. This saves much computation time. In fact, we traverse all 3D models only once in the first round of model retrieval and select N_{cands} ($N_{cands} = 1000$) rendered model images having the highest matching scores as the candidates. With the detected vanishing points for the input scene and these candidate 3D models, the depth order of all OOIs can be automatically estimated using the method of [36] in this step. We then select the most similar model for each object from this limited set.

Fig. 5. Segmentation information facilitates conducting forward matching.

In the succedent model retrieval procedure, segmentation is used to assist 3D model retrieval. The benefit of the segmentation information is four-fold.

First, segmentation eliminates the negative influence of cluttered background as evidenced in the top row of Fig. 4. Taking the whole rectangular region as input, all the retrieved chair models are under the wrong poses to match with the table in the background. Instead, we search for the models with the segmented region ((b) of top row in Fig. 4). The retrieved chair models have the similar poses and shapes with the input.

Second, segmentation reduces the influence brought by severe occlusion. As shown in the second row of Fig. 4, the chair inside the red box is severely occluded by the table and another chair, leading to inaccurate retrieval result. Without the aid of segmentation, the most similar model in this case is a club chair whose shape deviates greatly from the input chair. In (b), we segment the chair from the image and set the gradients to 0 at the occluded edges (marked green). Only the segmented foreground is used for matching, resulting in more accurate poses and more reasonable shapes.

Third, mask information helps to improve the retrieval. Since HOG features are used to match with the view-dependent representations and neglect mask information, the pixel occupancies of the input object and retrieved model images may differ a lot. In the bottom row of Fig. 4, the top five matched models (in the top row of (c)) are retrieved without using the mask information. By comparison, in the bottom row of (c), mask information is added to re-order the candidates. The results are refined.

Last, segmentation helps to accomplish bi-directional matching between the input object and the candidate models. Generally, HOG features are extracted from the OOI and matched with the mid-level visual elements extracted from rendered models. We call this step backward matching. Obviously, such unidirectional matching is less reliable than bi-directional matching. To realize bi-directional matching, we also extract the mid-level visual elements from the OOI and match them with HOG features of the rendered models. While we find extracting visual elements from the whole rectangular region always brings undesirable models (see the first row of Fig. 5), segmentation is useful to resolve this issue. More specifically, for the target object, we get the regions occupied by other objects who occlude it. As can be seen from Fig. 5a, two sofas occluding the table are extracted. We then overlay the segmented object with the extracted regions, from which we can obtain view-dependent representations (see Fig. 5b). Similarly, we overlay the candidate model images with the extracted

regions and compute HOG features for matching. By this, the occlusion information shared by the model images improves the accuracy of model retrieval in the presence of complex occlusions. We call this procedure forward matching.

Algorithm 1. 3D Model Retrieval

Input:
The segmentation $\mathcal{M}_{\mathcal{O}}$ of the OOI $\mathcal{O}$ and the matched candidate model images $\mathcal{C}_{\mathcal{O}}$;
Output:
The top K matched model images, $\mathcal{R}_{\mathcal{O}}$;
1: Compute the mask similarity S_M between each $\mathcal{C}_{\mathcal{O}}$ and $\mathcal{O}$;
2: Compute the maximum S_M^m and the average S_M^a of S_M;
3: Compute the forward matching score S_F between $\mathcal{C}_{\mathcal{O}}$ and $\mathcal{O}$ with $\mathcal{M}_{\mathcal{O}}$;
4: Compute the backward matching score S_B between $\mathcal{C}_{\mathcal{O}}$ and $\mathcal{O}$ with $\mathcal{M}_{\mathcal{O}}$;
5: Compute the overall score S_O according to Equation (8);
6: Select $\mathcal{R}_{\mathcal{O}}$ from $\mathcal{C}_{\mathcal{O}}$ with the top K scores.
7: **return** $\mathcal{R}_{\mathcal{O}}$;

In practice, our model retrieval procedure is described in Algorithm 1. In line 1, we compute the similarity between model mask and the OOI mask obtained by segmentation. The AABB of the object mask and its corresponding model mask are uniformly divided into overlapped cells. We then overlay the object mask with the corresponding model mask. For each cell in the model mask, we compute the 0-1 overlapping ratios with the neighborhood cells of its corresponding location in the object mask and select the maximum as its score. The mask similarity is computed as the average of all its cell scores. In the forward matching step (line 1), we extract HOG features on the segmented foreground instead of the full rectangular region, and the gradients are set to 0 at the occluded edges of the segmented object to reduce the influence brought by severe occlusion. We compute the overall score S_O as

$$S_O = (S_F + S_B) exp(\frac{-\rho(1 - S_M)^2}{(S_M^m - S_M^a)^2}), \tag{8}$$

in which S_F and S_B are the forward and backward matching scores respectively, S_M is the mask similarity, S_M^m and S_M^a are the maximum and the average of the mask similarity, ρ is used to weight the mask similarity and HOG matching score. We use $S_M^m - S_M^a$ instead of a constant in Equation (8) to adapt to different distributions of mask similarities.

Fig. 6 shows the matching scores for different model images. For each group, the matching scores of the model images become smaller from left to right. This is coincident with the visual similarity between the model images and the input object, proving the effectiveness and reasonability of Equation (8).

4.3 Modeling Refinement and Layout Estimation

When the iteration of object segmentation and model retrieval terminates, for each OOI an optimal model with an initially estimated pose is retrieved. The initial pose, yielded by the 72 sampled viewpoints, is not accurate. On the other hand, we render each model by assuming that its center is

Fig. 6. Matching scores for different model images. For each group (row), four model images with the matching scores are illustrated.

consistent with the focusing point of the view. However, this assumption is easily violated for those scenes whose OOIs are not located at the focusing point. We aim to reconstruct the whole scene as accurately as possible. To achieve this goal, we need to refine the poses of all the OOIs jointly with which the 3D bounding boxes of all OOIs can be obtained. We will show this will bring us the scene layout naturally.

Similar to [42], [43], [44], we are able to obtain the sizes and poses of the OOIs by exploiting the correspondences between the vertices of 3D bounding boxes of OOIs and their 2D projections. The relationship between a 3D point P and its projection p in the image is given by

$$p \simeq MP, \tag{9}$$

where $\simeq$ means equality up to a scale. M is the projection matrix, expressed as:

$$M_{3\times4} = K_c \cdot [R|t], \tag{10}$$

where K_c is the camera intrinsic matrix, R and t represent the rotation and translation of the camera system relative to the world coordinate system. K_c contains three unknown parameters, since we vary u, v, and f. Both R and t contain three unknowns. However, we can easily get K_c and R_{scene} with the detected vanishing points of the indoor scene [42], with R_{scene} being the rotation between the camera and the scene. As we assume that all the OOIs stand on the ground, they have only a different rotation around the gravity axis with respect to R_{scene}. As a result, for each OOI R only contains one unknown parameter. For an OOI $\mathcal{O}_i$, let B_i denote its 3D bounding box with l_i, w_i, and h_i representing its length, width, and height, separately. Now, its state is determined solely by the translation t_i^x, t_i^z (t_i^y can be expressed with h_i since all the OOIs are standing on the ground) and rotation R_i^y around the gravity axis. Each bounding box B_i is then associated with six parameters (l_i; w_i; h_i; t_i^x; t_i^z; R_i^y). Similar to [44], we assume $l_1 = 1$. For the first OOI, there

Fig. 7. Left: The interest object. Middle: Detected line segments. Right: The line segments along with the contour of the table but do not correspond to the Y-direction vanishing point.

Fig. 8. An example illustrates how to use matching to select the 3D bounding box when no dominant line segments are detected. Left: The interest object. Top right: The detected 3D bounding boxes for the densely sampled vanishing points. Bottom right: The rendered models under the poses determined by the corresponding 3D bounding boxes in the same column. By matching the rendered models with the input sofa, the most similar model together with the 3D bounding box is selected (shown with the green box).

exist only five parameters. Therefore, all the unknowns can be computed from the correspondences between the vertices of the 3D bounding boxes and their 2D projections.

With the segmentation information and vanishing points for an OOI, a 3D bounding box can be detected automatically [45]. The method in [45] on traffic understanding assumes that the vehicles are moving from/towards the first vanishing point, leading to a single 3D bounding box. In our implementation, the orders of the three vanishing points are not constrained, and up to six 3D bounding boxes can be obtained. We just select the smallest one encircling the object region. However, we find that the 3D bounding boxes automatically obtained with the segmentation information and estimated vanishing points are not accurate, a new method to optimize the 3D bounding boxes is thus required.

It should be noted that the vanishing points used to detect the 3D bounding box are those of the OOI itself, instead of the vanishing points of the scene image. In the following, we first describe how to estimate the vanishing points for each OOI. We then optimize all the 3D bounding boxes in an iterative manner, yielding the final scene layout.

4.3.1 Estimation of Vanishing Points

Indoor scenes behave mostly as Manhattan Worlds, i.e., most edges in the image are associated with parallel lines defined in terms of three dominant vanishing points which are orthogonal to each other. Based on this observation, we estimate the vanishing points of each OOI by utilizing detected line segments. Since each OOI has different rotation R_y around the gravity axis of the scene, we only need to evaluate R_y for the OOI. After obtaining R_y, the vanishing points vp for OOI can be calculated as [42]

$$vp \simeq K_c \cdot Rodrigues(R_{scene}) \cdot Rodrigues(0, R_y, 0), \qquad (11)$$

where K_c and $Rodrigues$ represent the camera matrix and the Rodrigues rotation formula, separately. To estimate the vanishing points $vp_{\mathcal{O}}$ for an OOI $\mathcal{O}$, we densely sample vanishing points around the gravity axis. The line segments along with the contour of $\mathcal{O}$, excluding the Y-direction, are then obtained. Fig. 7 shows an example. We just select the group of line segments L_{max} supporting a certain group of vanishing points with the maximum length. $vp_{\mathcal{O}}$ can be computed as

$$vp_{\mathcal{O}} = \arg\max_{vp} \sum_{l \in L_{max}} c(l, vp), \qquad (12)$$

where $c(l, vp)$ is the consistency measure that evaluates the degree of l being consistent with vp used in [46]. $c(l, vp)$ has the simple expression $c(l, vp) = \ell(l)cos(alv(l, vp))$, where ℓ denotes the length of a line segment, and $alv(l, vp)$ represents the angle between l and the line from the vanishing point vp to the middle point of l. According to (11) and (12), $vp_{\mathcal{O}}$ can be easily obtained since there is only one unknown R_y.

However, not all furniture objects have enough valid line segments to compute the vanishing points, for instance a swivel chair. For such an object, we select $vp_{\mathcal{O}}$ from the densely sampled vanishing points with the highest matching score. It should be noted that we sample 360 rotations around the gravity axis of the scene instead of directly sampling the vanishing points and use Formula (11) to compute the corresponding vanishing points for each sampled rotation. Specifically, for each group of sampled vanishing points, we first detect the 3D bounding box of the corresponding OOI. The matched 3D model under the pose obtained with the detected 3D bounding box is further rendered. Matching between the rendered model and the imaged object is then conducted, and the matching score is computed with Equation (8). The group of vanishing points for the rendered model having the highest matching score is selected. Fig. 8 shows an example.

4.3.2 3D Bounding Box Optimization

With the detected 3D bounding boxes, we can easily compute the poses for all the OOIs according to Equations (9) and (10) [43], [44]. The method we used to detect the 3D bounding box is intuitive, but may incur errors, especially for the object with a curved contour. In fact, the 2D bounding box of the rendered model could be very different from the specified 2D semantic bounding box of the object. The first row of Fig. 9 shows such an example.

To remedy this, a straightforward solution is to re-calculate the sizes and locations of the OOIs by minimizing the pixel error between the specified 2D semantic bounding boxes and the 2D bounding boxes of the rendered models, which can be formulated as:

$$\mathcal{P}^* = \arg\min_{\mathcal{P}} \|ub - p(\mathcal{P})\|^2, \qquad (13)$$

where $\mathcal{P}$ denotes the parameters of the OOI, including l, w, h, t^x and t^z. ub represents the user-specified 2D bounding box of the corresponding OOI. $p(\mathcal{P})$ is the 2D bounding box of the snapshot of the retrieved 3D model rendered according to $\mathcal{P}$.

Unfortunately, it is an ill-posed problem since for each OOI the number of constraints (4 for the 2D bounding box) is

Fig. 9. Comparison of 3D bounding boxes and projected 3D models before and after bounding box optimization. Top right: The 2D bounding box (dashed) of the table model projected with the rendering parameters obtained from the detected 3D bounding box differs much from the specified bounding box (solid).

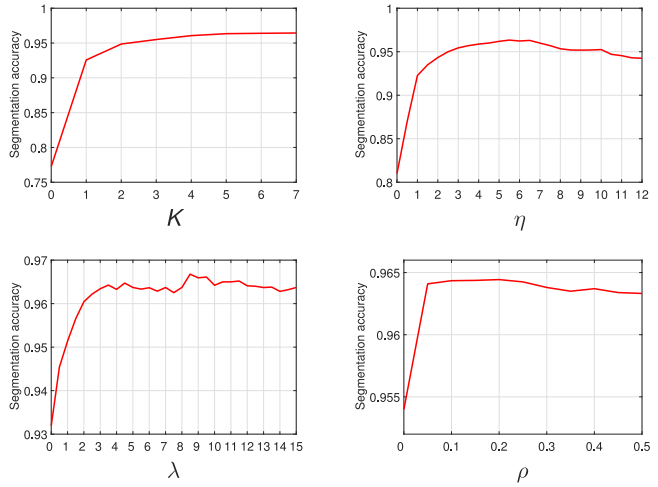


Fig. 10. Variation of segmentation accuracy with respect to different parameters.

less than the number of unknowns (l, w, h, t^x, t^z). To solve this problem, we propose a new method to refine the 3D locations and sizes of the OOIs. The method intrinsically follows the principle of expectation maximization, optimizing the sizes and locations of 3D bounding boxes alternately. In other words, we optimize the sizes by assuming that the locations are fixed and optimize the locations by assuming that the sizes do not change in each iteration. The second row of Fig. 9 displays the result of 3D bounding box optimization. After optimization, the projected models match well with the input objects, and the 2D bounding box of projected table model is consistent with the specified bounding box.

5 Experiments

In this section, we first study the effects of some important parameters used in our proposed method. We then conduct intensive experiments to verify the effectiveness of the key steps of our scene modeling method which include object segmentation, 3D model retrieval, layout estimation and pose refinement. A user study on the results of image-guided model retrieval is finally conducted to compare our method with other relevant methods.

5.1 Dataset

The dataset used for experiments consists of three parts: a subset of the PASCAL3D+ dataset including 22 indoor images, 10 indoor images we rendered, and 200 images searched from the web. Please refer to the supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2018.2880737 for them. Although the PASCAL3D+ dataset comprises thousands of indoor images, most of them contain only one furniture object or objects severely occluded by other objects such as the lamps and human bodies not supported by our current implementation. Excluding such images, we select 22 indoor images following two rules. First, the objects in the image belong to the categories we handle and are not occluded severely by other objects not belonging to the categories our system now supports. Second, each image contains more than one object, and the objects in the image occlude each other to guarantee scene complexity. The 10 rendered indoor images are mainly used to quantitatively measure the results of layout estimation and pose refinement. For each one, we record the 3D geometry, rendering parameters, and segmentation information of OOIs. The web images are from the Bing image search engine with the keywords "furniture sets", and the top 200 are used for experiments. Fig. 6 in the supplementary material, available online shows the screenshots of the website, and the thumbnails of the images. For each of the image from the first and third parts, we further manually label the ground truth of segmentation for measuring the segmentation accuracy.

5.2 Parameter Setting

Through experiments, we study the effects of the four parameters used in our proposed method. They are K representing the number of model images used to assist segmentation, η adjusting the weight of transferred labels and contours, λ balancing prior and likelihood, and ρ denoting the weight of mask similarity in Equation (8).

By fixing three out of the four parameters each time, we vary the rest one in turn to see the influence of this parameter on segmentation. Fig. 10 shows the curves of segmentation performance on these parameters. As can be seen, the segmentation accuracy without model images as the assistance ($K = 0$) is much lower than the results with model images ($K > 0$), revealing that the model images used improve segmentation significantly. In general, the more model images we use, the more accuracy our segmentation is. However, too many model images lead to higher computational cost. From the segmentation curve, we set $K = 5$ as a tradeoff between segmentation accuracy and efficiency. η is another important parameter weighting the transferred information and color information. As is observed, segmentation accuracy is stable when η is set to 6.0. The influence of λ and ρ on segmentation accuracy is slight. Good performance can be achieved for a wide range of λ and ρ. We set them to 8.5 and 0.2, separately.

Fig. 11. Comparison of segmentation results. (a): Input images. (b): PSPNet [47]. (c): GrabCut [39]. (d): Our method. (e): Ground truth.

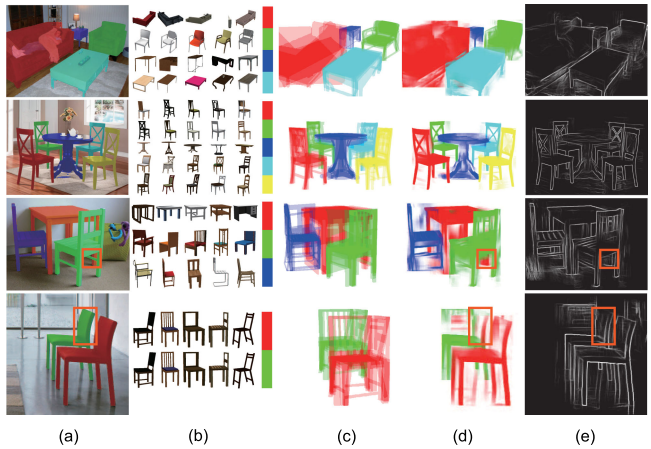


Fig. 12. Segmentation with dissimilar 3D models. (a): Segmentation results. (b): Randomly selected 3D models from top 1000 candidates. (c): Masks by directly stacking the models at the corresponding locations. (d): Transferred label probabilities. (e): Transferred contours.

5.3 Image Segmentation

We compare our segmentation results with the PSPNet [47] and GrabCut [39] methods in Fig. 11. Similar to our method, the GrabCut method also needs use-specified bounding boxes around the interest objects. Therefore, the same boxes are initialized for both methods. Apparently, our segmentation method fully exploiting the correspondences between OOIs and retrieved models outperforms PSPNet and GrabCut in general. As can be observed from Fig. 11b, PSPNet works well for those non-hollow objects, but may fail for hollow objects, leading to inaccurate results. Besides, as a semantic segmentation method, PSPNet only considers semantic information for each pixel. Hence, different objects in the same category would obtain the same label. GrabCut only considers color information, so it works poorly for those adjacent objects with similar colors. Please refer to the supplementary material, available online for more results.

Our segmentation is assisted with the matched 3D models, so intuitively the similarity between the OOIs and matched models are critical for segmentation performance. As the ShapeNet model repository has a large number of 3D models classified into different categories, we could retrieve similar 3D models in most cases. Nevertheless, to evaluate our segmentation performance in the case that similar models are not available, we randomly select K 3D models from top 1000 candidates for each OOI for experiment.

Fig. 12 shows four groups of examples. Obviously, most selected 3D models differ much from the imaged objects, even with quite different poses. The segmentation results as shown in (a) verify our effectiveness. From the experiments, we have two observations. First, compared with the masks shown in column (c), the transferred labels in (d) are more accurate. This means that the performance of label and contour transfer does not get worse evidently when the retrieved models have dissimilar shapes. Our method is robust to this. The reason could be attributed to two aspects. On one hand, the OOIs are match with model images at patch-level to get the most similar pairs. For a patch in an OOI, only one similar patch is required from a group of synthetic overlapped model images. On the other hand, we further conduct pixel-level matching with SIFT flow which allows for robust matching across different object-model appearance. Second, color information and transferred labels and contours complement with each other. Color helps to obtain good segmentation, even though the transferred labels and contours are not accurate. For example, in Fig. 12, the transferred labels and contours of the chair spindles in the third group and the back in the last group marked with the orange boxes are inaccurate. However, by the aid of color information, good segmentation can be achieved.

5.4 3D Model Retrieval

We compare our model retrieval method with direct HOG matching [48] and Aubry et al.'s method [18]. The results are shown in Fig. 13. For HOG matching, we extract HOG features for both the OOI and the rendered model image and use cosine similarity as the distance metric. As can be seen from Fig. 13b, direct HOG matching generates the worst results. With proper weight for each pixel after training [18], the retrieval results are improved significantly as shown in Fig. 13c. Obviously, our retrieved models look more similar to the OOIs than the results of HOG matching and [18] due to the mutual promotion of object segmentation and model retrieval. Moreover, our method is more robust to occlusion and cluttered background compared to other methods. For more results, please see the supplementary material, available online.

Additionally, we compare the top five retrieved models obtained by [18] and our method in Fig. 14. Augmented with the extra segmentation information, our retrieved models match well with the OOIs in the aspect of both shapes and internal structures.

We also compare our method with the image-shape joint embedding method [17] on chairs, using the trained CNN model for the chair category of the ShapeNet repository. This method builds a joint embedding space for both 2D images and 3D shapes and can thus be used for cross-search of images and models. Fig. 15 compares the results

Fig. 13. Comparison of model retrieval and scene reconstruction. (a): Input images. (b): HOG results. (c): Results by [18]. (d): Our results without layout estimation. (e): Our results with layout estimation. (f): The top views of (e). The columns (b), (c) and (d) are generated by directly stacking the retrieved model images on the corresponding regions, while each sample in (e) and (f) is rendered as a whole scene since the layout of the scene has been estimated. Note that the results in (d) and (e) may not have the same appearances since we use different rendering engines.

produced by [17], [18] and our method. Note that the joint embedding method [17] cannot estimate object poses. For the sake of fairness, we render the models retrieved by [17] with the same poses as our models. Again, our method obtains more similar chair models than the other two methods. From Fig. 15b, we can see that the retrieval results of [17] differ much from the input chairs. This is probably due to the negative influences of occlusion and cluttered background on search performance.

A user study is designed to compare the performance of our method, the joint embedding space method [17] and Aubry et al.'s method [18] on model retrieval. There are totally 15 groups of results used in the study. Each group

Fig. 14. Top five retrieved models obtained by [18] and our method. Left: The input images with specified target objects. Right: The top five retrieved models, in which for each group the first row is the results by [18], and the second is ours.

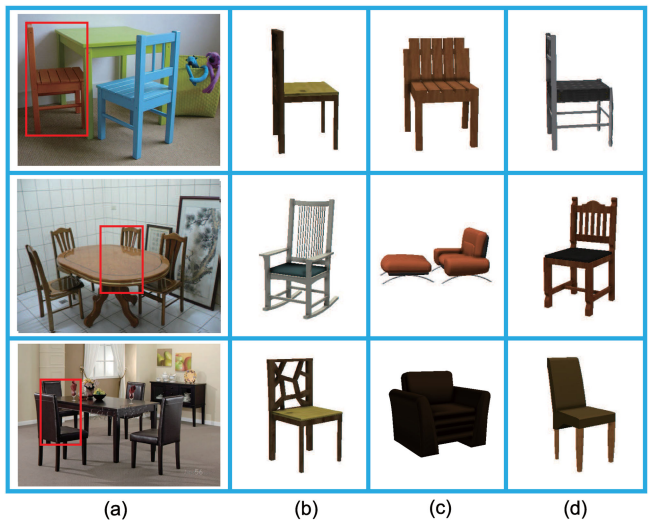


Fig. 15. Comparison of model retrieval on the chair category. (a): Input chairs. (b): The retrieval results obtained by joint embedding space [17]. (c): The results by [18]. (d): Our results.

contains an input image with the specified chair of interest, and the models retrieved by the three methods. These target chairs belong to different sub-categories, such as the club chair, swivel chair, straight chair and arm chair, etc. The input images may have cluttered background, with chairs occluded by other objects. Fig. 15 shows three groups used in the study, with each row corresponding to a group.

We have recruited 64 participants who were not involved in this work for the task. Most participants are the undergraduate or graduate students majored in Computer Science. During the study, each participant viewed all these 15 groups. For each group, we shuffled the order of models retrieved by the three methods and asked every participant to first rate each model image with a score ranging from 1 (dissimilar) to 5 (very similar), and second, select the most similar model in his/her opinion.

Fig. 16 (left) shows the average score on each group by different methods. It has been shown that our approach obviously outperforms the other two on 10 groups. We also calculate the average score on all the models retrieved by each method. Again, our method gets the highest average score 3.53, while the scores of [18] and [17] are 2.66 and 2.99, respectively. Fig. 16 (right) shows for each group the percentage that the participants selected the results by different methods as the most similar models. Obviously, our method surpasses the other two on most groups. We further compute the average percentage on all the 15 groups for each method. Our method has the highest percentage

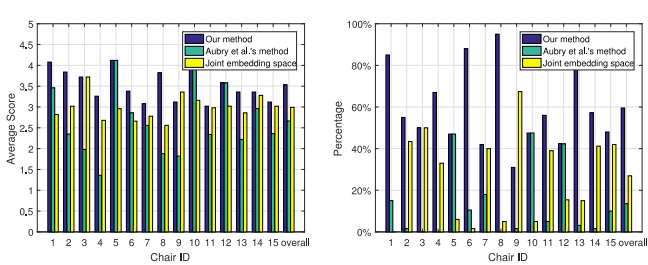


Fig. 16. The average scores and normalized votes for the most similar models on the 15 groups.

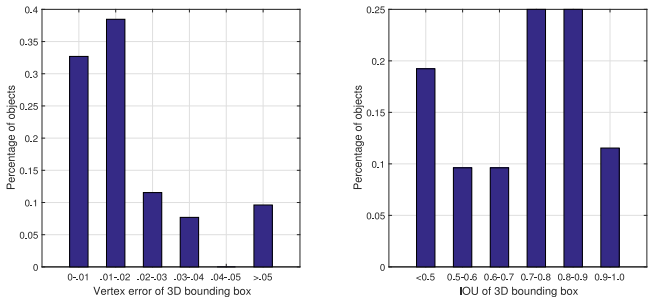


Fig. 17. Comparison of layout estimation results.

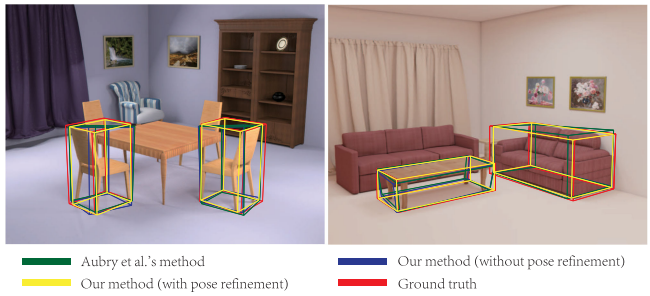


Fig. 19. Comparison of 3D bounding boxes.

59.5 percent, which is much higher than [18] 13.6 percent and [17] 26.9 percent. In conclusion, the user study shows that our method outperforms the other two for model retrieval on the chair category.

5.5 Layout Estimation

To quantitatively measure the results of layout estimation and pose refinement, we render 10 indoor scenes containing 52 objects and record the ground truth pose for each object in the scenes. Two of the rendered scenes are shown in Fig. 19. We conduct experiments on these rendered indoor images.

We use two metrics to measure the layout estimation results: vertex error and the intersection of union (IOU) of 3D bounding boxes. For the former, we compute the ratio of euclidean distance between the vertex of our estimated 3D bounding box and the ground truth to the size of the scene. Fig. 17 shows the layout estimation results measured with the two metrics. As we can see, for more than 70 percent of the objects, the vertex error of 3D bounding box is less than 0.02, and for more than 80 percent of the objects, the IOU of 3D bounding box with ground truth is larger than 0.5, both of which imply the effectiveness and practicability of our layout estimation method. Moreover, we show the reconstructed scenes from the top views in Fig. 13f, which again validates that the estimated layouts are reasonable.

Our layout estimation method also refines the pose for each OOI. We use another two metrics to measure the pose refinement results: the pixel error of 3D bounding box and vanishing point angle. For the former, we calculate the average pixel distance between the projected 3D bounding box and the ground truth. The vanishing point angel $avv(vp1, vp2)$ for $\mathcal{O}$ is defined as the angle between the lines from the vanishing points $vp1$ and $vp2$ to the center of specified bounding box of $\mathcal{O}$. Fig. 18 shows the 3D bounding box pixel error and vanishing point angle. From Fig. 18, we can see that the poses of our retrieved 3D models (before pose refinement) are better than that of [18]. The reason to this is that our method facilitates retrieving similar models in terms of not only the model shape, but also its pose. Furthermore, it is obvious that after pose refinement the results improve greatly. Our pose refinement returns a vanishing point angle within 1 degree of the ground truth for about 80 percent of the examples, compared with less than 10 percent without pose refinement. Fig. 19 shows two indoor scenes with colored 3D bounding boxes. In Fig. 13e, we show more pose refinement results qualitatively for the retrieved models. Compared with the initial poses shown in Fig. 13d, the refined poses coincide with the OOIs. More importantly, since we optimize the poses for all the scene objects in a unified framework, the refined poses are more accurate.

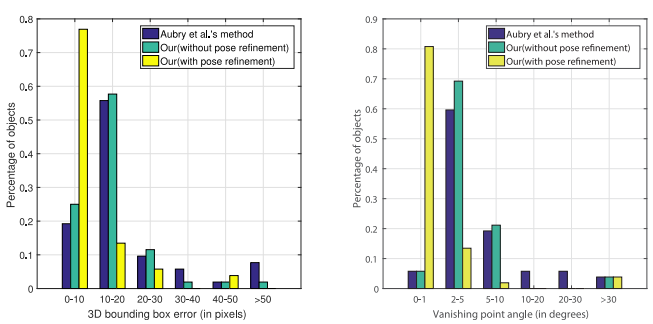


Fig. 18. Comparison of pose results.

5.6 Further Discussion

In order to study the relationship between the number of available 3D models and results and to investigate the results when 3D models similar to the input objects are hard to find, we also test the 200 web images on the subsets of the total 3D models (S_{total}). For each category the subsets contain 1, 10, 100, and a quarter of randomly selected 3D models from S_{total}, which are represented as S_1, S_{10}, S_{100}, and $S_{quarter}$ respectively.

The experimental results on the 200 images are all shown in the supplementary material, available online, in which Figs. 10– 27 and Figs. 28-47 compare the results of model retrieval and segmentation of our method using S_1, S_{10}, S_{100}, $S_{quarter}$, S_{total}, respectively, and Fig. 9 illustrates the segmentation accuracy of our method with different subsets of the 3D model repository. From these experiments, we have two observations. For modeling, on one hand, we retrieve the most similar 3D models from the model repository to model the input scene. The modeling depends heavily on the retrieved 3D models as they are. If similar 3D models are not available, our method would not return satisfactory modeling results. On the other hand, with the number of 3D models growing, the retrieved models are more similar with the input objects. This is reasonable since more similar models can be retrieved from more 3D models. The improvements are significant from S_1 to S_{10} and from S_{10} to S_{100}. However, the improvements on modeling results using more 3D models are less noticeable. This is probably due to that the model number is no longer the key for retrieval since it seems that the 3D models are already enough. Even though the model number increases dramatically, the models still can not cover all the shapes that are

Fig. 20. A failure case of segmentation due to similar appearance between background and the retrieved models. Influenced by the background, all the retrieved models have side stretchers and the strip is labeled as a part of the table in the segmentation step.

similar with the input OOIs for the 200 images. For segmentation, similar with the results of 3D model retrieval, with the number of available 3D models growing, the segmentation accuracy improves. And the influence of model number on segmentation accuracy is subtle once the available 3D models exceed a certain amount. For example, the total number of 3D models in S_{total} is about 20 times of S_{100}, while the segmentation accuracy with S_{total} improves less than 1 percent compared with S_{100}. In other words, even if all the 3D models differ much with the input OOIs, our method returns good segmentation results in most cases. Similar conclusion has been reached with Fig. 12, and the reason for this has been discussed in the last paragraph of Section 5.3.

5.7 Running Times

We implemented our system on a PC with a Core i7 3.60GHz CPU and a NVIDIA GTX-1080ti GPU. The off-line pre-processing stage, including rendering all the 3D models in the database and extracting the view-dependent representations, takes about 52 hours CPU time. We have 20,643 models in total for the 6 categories our system now supports. At run-time, 3D model retrieval is conducted on GPU, and we use CUDA to accelerate the processing. In the first iteration, model retrieval takes about 45s for each OOI. Thereafter, it takes about 7s for each OOI during each iteration since we do not need to traverse all the 3D models as in the first iteration. Object segmentation and layout estimation are implemented by CPU. Segmentation takes most time, 3∼5 minutes in each iteration, since computation of SIFT flow is time-consuming. Layout estimation only requires 5∼10s. Normally, for a scene consisting of five models, we need 3∼4 iterations. Overall, the timing for scene reconstruction for such an image takes about 18 minutes. We plan to further accelerate the computation by Cloud computing in future.

6 Limitations

Our system has several limitations. First, in our current implementation, the user needs to drag semantic rectangles indicating the objects of interest. This step can be automated by using the most recent object detection technique. Since this is not our focus, and object detection is not robust enough to always generate satisfactory results, we now resort to user interaction to achieve this. We intend to integrate object detection into our framework to automate our method. Second, our system depends on the 3D model repository. If only a small repository is available and all the 3D models differ much from the target objects in the input image, our system would not generate a satisfactory result. Third, the performance could be influenced by image background and severe occlusions, if there exist 3D models that look similar with occlusions. A failure case is shown in Fig. 20. Last, 3D bounding boxes are used to estimate the 3D locations and sizes of OOIs and collision detection between these bounding boxes is not supported in our current implementation. As a result, though rarely encountered, the 3D models may penetrate its neighbors in some cases.

7 Conclusion

We have proposed a novel framework for modeling of the indoor scene, given only a single color image as input. With very simple user interaction, the scene is populated with 3D models similar to the objects of interest in a data-driven manner. The core to our framework is to conduct object segmentation and 3D model retrieval iteratively such that the two aspects promote each other. Moreover, we carefully estimate the 3D locations and sizes of the objects with the inferred 3D bounding boxes, which further improves the result and leads to the scene layout naturally. Our experiments show that we can reliably model objects of interest even for complex and cluttered indoor scenes.

Acknowledgments

The authors would like to thank the reviewers for their constructive comments which helped improve this paper greatly. This work was supported in part by the National Natural Science Foundation of China under Grants 61772257 and 61672279, the Natural Science Foundation of Jiangsu Province under Grants BK20150016.

References

[1] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments," in *Experimental Robotics*. Berlin, Germany: Springer, 2014, pp. 477–491.

[2] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, et al., "Kinectfusion: Real-time 3D reconstruction and interaction using a moving depth camera," in *Proc. 24th Annu. ACM Symp. User Interface Softw. Technol.*, 2011, pp. 559–568.

[3] J. Wang, Q. Xie, Y. Xu, L. Zhou, and N. Ye, "Cluttered indoor scene modeling via functional part-guided graph matching," *Comput. Aided Geometric Des.*, vol. 43, pp. 82–94, 2016.

[4] K. Xu, Y. Shi, L. Zheng, J. Zhang, M. Liu, H. Huang, H. Su, D. Cohen-Or, and B. Chen, "3D attention-driven depth acquisition for object identification," *ACM Trans. Graph.*, vol. 35, no. 6, 2016, Art. no. 238.

[5] T. Shao, W. Xu, K. Zhou, J. Wang, D. Li, and B. Guo, "An interactive approach to semantic modeling of indoor scenes with an RGBD camera," *ACM Trans. Graph.*, vol. 31, no. 6, 2012, Art. no. 136.

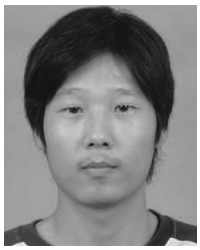
[6] L. Nan, K. Xie, and A. Sharf, "A search-classify approach for cluttered indoor scene understanding," *ACM Trans. Graph.*, vol. 31, no. 6, 2012, Art. no. 137.

[7] K. Chen, Y. Lai, Y.-X. Wu, R. R. Martin, and S.-M. Hu, "Automatic semantic modeling of indoor scenes from low-quality RGB-D data using contextual information," *ACM Trans. Graph.*, vol. 33, no. 6, 2014, Art. no. 208.

[8] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "Shapenet: An information-rich 3D model repository," arXiv:1512.03012 [cs.GR], 2015.

[9] B. M. Oh, M. Chen, J. Dorsey, and F. Durand, "Image-based modeling and photo editing," in *Proc. 28th Annu. Conf. Comput. Graph. Interactive Techn.*, 2001, pp. 433–442.

[10] D. Hoiem, A. A. Efros, and M. Hebert, "Automatic photo pop-up," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 577–584, 2005.
[11] A. Saxena, M. Sun, and A. Y. Ng, "Make3D: Learning 3D scene structure from a single still image," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 5, pp. 824–840, May 2009.
[12] K. Xu, H. Zheng, H. Zhang, D. Cohen-Or, L. Liu, and Y. Xiong, "Photo-inspired model-driven 3D object modeling," *ACM Trans. Graph.*, vol. 30, no. 4, 2011, Art. no. 80.
[13] S. Satkin, J. Lin, and M. Hebert, "Data-driven scene understanding from 3D models," in *Proc. 23rd Brit. Mach. Vis. Conf.*, 2012, pp. 58–64.
[14] K. Karsch, K. Sunkavalli, S. Hadap, N. Carr, H. Jin, R. Fonte, M. Sittig, and D. Forsyth, "Automatic scene inference for 3D object compositing," *ACM Trans. Graph.*, vol. 33, no. 3, 2014, Art. no. 32.
[15] H. Su, Q. Huang, N. J. Mitra, Y. Li, and L. J. Guibas, "Estimating image depth using shape collections," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 37–1, 2014.
[16] Q. Huang, H. Wang, and V. Koltun, "Single-view reconstruction via joint analysis of image and shape collections," *ACM Trans. Graph.*, vol. 34, no. 4, 2015, Art. no. 87.
[17] Y. Li, H. Su, C. R. Qi, N. Fish, D. Cohen-Or, and L. J. Guibas, "Joint embeddings of shapes and images via CNN image purification," *ACM Trans. Graph.*, vol. 34, no. 6, pp. 234:1–234:12, 2015.
[18] M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic, "Seeing 3d chairs: exemplar part-based 2D-3D alignment using a large dataset of cad models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 3762–3769.
[19] J. Shi, X. Tao, L. Xu, and J. Jia, "Break ames room illusion: Depth from general single images," *ACM Trans. Graph.*, vol. 34, no. 6, 2015, Art. no. 225.
[20] K. Karsch, C. Liu, and S. B. Kang, "Depth extraction from video using non-parametric sampling," in *Proc. 12th Eur. Conf. Comput. Vis.*, 2012, pp. 775–788.
[21] M. Liu, Y. Guo, and J. Wang, "Normal guided data-driven semantic modeling from a single indoor image," in *Proc. Int. Conf. Cyberworlds*, 2016, pp. 111–118.
[22] M. Liu, Y. Guo, and J. Wang, "Indoor scene modeling from a single image using normal inference and edge features," *Visual Comput.*, vol. 33, pp. 1227–1240, 2017.
[23] A. Bansal, B. Russell, and A. Gupta, "Marr revisited: 2D-3D alignment via surface normal prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 5965–5974.
[24] F. Massa, B. C. Russell, and M. Aubry, "Deep exemplar 2D-3D detection by adapting from real to rendered views," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 6024–6033.
[25] K. Chen, Y.-K. Lai, and S.-M. Hu, "3D indoor scene modeling from RGB-D data: A survey," *Comput. Visual Media*, vol. 1, no. 4, pp. 267–278, 2015.
[26] Y. M. Kim, N. J. Mitra, D.-M. Yan, and L. Guibas, "Acquiring 3D indoor environments with variability and repetition," *ACM Trans. Graph.*, vol. 31, no. 6, 2012, Art. no. 138.
[27] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik, "Aligning 3D models to RGB-D images of cluttered scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 4731–4740.
[28] S. Choi, Q.-Y. Zhou, and V. Koltun, "Robust reconstruction of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 5556–5565.
[29] M.-M. Cheng, S. Zheng, W.-Y. Lin, V. Vineet, P. Sturgess, N. Crook, N. J. Mitra, and P. Torr, "Imagespirit: Verbal guided image parsing," *ACM Trans. Graph.*, vol. 34, no. 1, 2014, Art. no. 3.
[30] M. Fisher, M. Savva, and P. Hanrahan, "Characterizing structural relationships in scenes using graph kernels," *ACM Trans. Graph.*, vol. 30, no. 4, pp. 34:1–34:12, Jul. 2011.
[31] V. Hedau, D. Hoiem, and D. Forsyth, "Recovering free space of indoor scenes from a single image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 2807–2814.
[32] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 746–760.
[33] T. Shao, A. Monszpart, Y. Zheng, B. Koo, W. Xu, K. Zhou, and N. J. Mitra, "Imagining the unseen: Stability-based cuboid arrangements for scene understanding," *ACM Trans. Graph.*, vol. 33, no. 6, pp. 1–11, 2014.
[34] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 580–587.
[35] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3431–3440.
[36] M. Liu, Y. Zhang, J. He, J. Guo, and Y. Guo, "Image-based 3D model retrieval for indoor scenes by simulating scene context," in *Proc. IEEE Int. Conf. Image Process.*, 2018, pp. 3658–3662.
[37] Y. Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in nd images," in *Proc. 8th IEEE Int. Conf. Comput. Vis.*, 2001, vol. 1, pp. 105–112.
[38] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, "Lazy snapping," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 303–308, 2004.
[39] C. Rother, V. Kolmogorov, and A. Blake, "GrabCut: Interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 23, no. 3, 2004, pp. 309–314.
[40] T. Malisiewicz, A. Gupta, and A. A. Efros, "Ensemble of exemplar-svms for object detection and beyond," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 89–96.
[41] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, Nov. 2001.
[42] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.
[43] N. Jiang, P. Tan, and L.-F. Cheong, "Symmetric architecture modeling with a single image," *ACM Trans. Graph.*, vol. 28, no. 5, 2009, Art. no. 113.
[44] Y. Zheng, X. Chen, M.-M. Cheng, K. Zhou, S.-M. Hu, and N. J. Mitra, "Interactive images: Cuboid proxies for smart image manipulation," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 99–1, 2012.
[45] M. Dubská, J. Sochor, and A. Herout, "Automatic camera calibration for traffic understanding," in *Proc. British Mach. Vis. Conf.*, 2014, pp. 1–10.
[46] P. Denis, J. H. Elder, and F. J. Estrada, "Efficient edge-based methods for estimating manhattan frames in urban imagery," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 197–210.
[47] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. 2017 IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 6230–6239, doi: 10.1109/CVPR.2017.660.
[48] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2005, vol. 1, pp. 886–893.

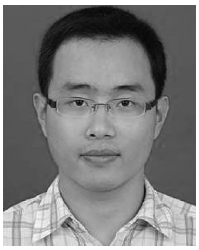


Mingming Liu received the BS degree in computer science from Zhengzhou University, China in 2011. He is working toward the PhD degree in the Department of Computer Science and Technology, Nanjing University. His research interests include computer vision and graphics, digital image processing and pattern recognition.

Kexin Zhang received the BS degree in computer science from the Hebei University of Economics and Business, China, in 2015. She is working toward the master's degree in the Computer Science and Technology Department of Nanjing university. She majors in computer vision and image processing.

Jie Zhu received the BE degree in computer science from Jiangnan University, Wuxi, China, in 2012. He is working toward the PhD degree in the Department of Computer Science and Technology, Nanjing University. His research interests include computer vision and computer graphics.

Jun Wang received the PhD degree in CAD from the Nanjing University of Aeronautics and Astronautics (NUAA) in 2007. From 2008 to 2010, he conducted research as a postdoctoral scholar with the University of California, Davis and the University of Wisconsin, Milwaukee. From 2010 to 2013, he was a senior engineer at Leica Geosystems Inc.. In October 2013, he joined NUAA as a professor. His research areas include geometry processing and 3D laser scanning.



Jie Guo received the bachelors and PhD degrees in computer science from Nanjing University, in 2008 and 2013, respectively. Currently, he is an assistant researcher with State Key Laboratory for Novel Software Technology, Nanjing University. He was a research intern with Microsoft Research Asia. His research interests include appearance modeling, real-time rendering, and virtual reality.

Yanwen Guo received the PhD degree in applied mathematics from the State Key Lab of CAD&CG, Zhejiang University, China, in 2006. He is currently a professor with the National Key Lab for Novel Software Technology, Department of Computer Science and Technology, Nanjing University, Jiangsu, China. He worked as a visiting professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, in 2006 and 2009, respectively, and the Department of Computer Science, The University of Hong Kong, in 2008, 2012, and 2013, respectively. He was a visiting scholar in the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, from 2013 to 2015. His research interests include image and video processing, vision, and computer graphics. He is the corresponding author of this paper.