IM2CAD

Hamid Izadinia
University of Washington

Qi Shan
Zillow Group

Steven M. Seitz
University of Washington

Figure 1: IM2CAD takes a single photo of a real scene (left), and automatically reconstructs its 3D CAD model (right).

Abstract

Given a single photo of a room and a large database of furniture CAD models, our goal is to reconstruct a scene that is as similar as possible to the scene depicted in the photograph, and composed of objects drawn from the database. We present a completely automatic system to address this IM2CAD problem that produces high quality results on challenging imagery from interior home design and remodeling websites. Our approach iteratively optimizes the placement and scale of objects in the room to best match scene renderings to the input photo, using image comparison metrics trained via deep convolutional neural nets. By operating jointly on the full scene at once, we account for inter-object occlusions. We also show the applicability of our method in standard scene understanding benchmarks where we obtain significant improvement.

1. Introduction

In his 1963 Ph.D. thesis, Lawrence Roberts [34] demonstrated a system that infers a 3D scene from a single photo (Figure 2). Leveraging a database of known 3D objects, his system analyzed edges in the image to infer the locations and orientations of these objects in the scene. Unlike the vast majority of modern 3D reconstruction techniques, which capture only visible surfaces, Robert's method was capable of inferring back-facing and occluded surfaces, object segments, and recognized which objects are present.

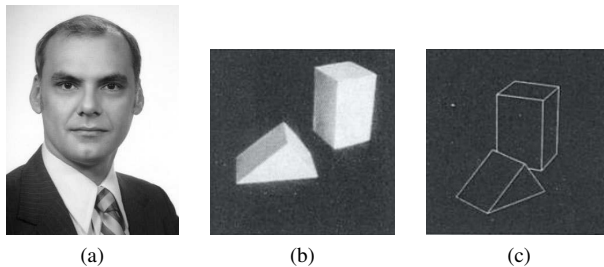While Robert's method was visionary, more than a half

Figure 2: Lawrence Roberts's (a) 1963 system took an input photo (b) and computed a 3D scene, rendered to a novel viewpoint (c).

century of subsequent research in computer vision has still not yet led to practical extensions of his approach that work reliably on realistic images and scenes. One major limitation is the need for an accurate, a priori 3D model of each object in the scene. While a chair model, e.g., is not hard to come by, obtaining exact 3D models of every chair in the world is not presently feasible. A further challenge is the need to reliably match between features in photographs and CAD models, particularly when the model does not exactly match the object photographed.

We therefore introduce a variant of Robert's original problem, that we call *IM2CAD*, in which the goal is to reconstruct a scene that is **as similar as possible** to the scene depicted in a photograph, where the reconstruction is composed of objects drawn from a database of available 3D object models. For example, the bed in Fig. 1 resembles but does not exactly match the one in the input photograph at
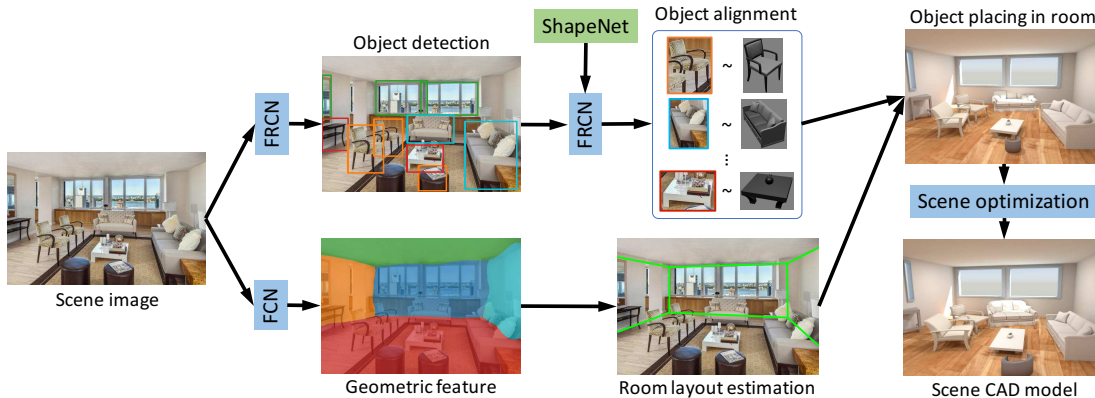
Figure 3: System overview: an input image (left) is processed through a series of steps to produce a scene CAD model (bottom right).

left, as we did not have that specific bed in the database. While our results are not perfect, they represent a significant step forward to achieving Robert's vision on real-world imagery. Producing CAD models of real scenes has applications for virtual reality (VR), augmented reality (AR), robotics, games, and education.

Our work builds on a number of recent advances in the computer vision and graphics research community. First, we leverage ShapeNet [6], which contains millions of 3D models of objects, including thousands of different chairs, tables, and other household items. This dataset is a game-changer for 3D scene understanding research, and was key to enabling our work. Second, we use state-of-the-art object recognition algorithms [32] to identify common objects like chairs, tables, windows, etc.; these methods work impressively well in practice. Third, we leverage deep features trained by convolutional neural nets (CNNs) [21] to reliably match between photographs and CAD renderings [3, 20, 36, 18]. Finally, we build on recent research on room reconstruction [15, 22, 27].

Our main contribution is a fully automatic system that produces full-scene CAD models (room + furniture) from a single photo. While many of the technical ingredients of our system draw heavily from prior work (as detailed in the previous paragraph), we also contribute noteworthy technical advances on room modeling and scene optimization. Our room modeling approach produces significant improvement on standard benchmarks. Our novel full-scene optimization approach iteratively adjusts the placement and scale of objects to best align rendered photos with input images, operating jointly on the full scene at once, and accounting for inter-object occlusions. Our models include semantics (e.g. "table", "chair") segmented into objects, and take only a few bytes to represent, encoded as a collection of ShapeNet object IDs and transformations that define position, orientation and scale. We evaluate our performance on scene understanding using the datasets of [15], LSUN [1], SUN RGB-D [42] and 3DGP [8]. We show significant improvements in the 2D and 3D room layout estimation as well as

3D object location using only single RGB images.

2. Related Work

The last decade has seen renewed interest in single-image 3D modeling, following the work of Hoiem et al., [16] and Saxena et al., [2]. Single-image modeling of indoor scenes has enjoyed significant recent progress, with a series of papers on room-shape estimation (floor, walls, ceiling), e.g., [15, 22, 27, 9, 33] that yield increasingly good results. Our approach for room shape estimation obtains competitive results. More recently, researchers have moved beyond walls, and toward approximating furniture in the room using *cuboids* [46, 49, 8, 14, 29, 38]. While the cuboid based approach avoids the need for object databases, the resulting models are primitive and do not accurately depict scene appearance.

Another closely related line of research is 3D object and pose recognition of chairs and other objects [3, 20, 36, 23, 18, 43, 4, 44]. These methods can produce very accurate alignment of a single object to a photograph or depth image. Our work leverages similar 3D object recognition techniques, combined with room shape estimation, to jointly solve for all of the objects in the room in a way that accounts for inter-object occlusions. Our work also builds upon recent advancements of research on object detection from single images [12, 32].

Researchers have explored a variety of techniques to automatically compute CAD scene models using non-photographic means, e.g., using example based approaches [11], utilizing text descriptions [7], and optimizing for furniture arrangements in a given space [47, 28]. These approaches rely on analyzing location and pose correlations between furniture types, based on analyzing databases of scene models. Collecting such data is a challenge, and therefore these approaches can greatly benefit from our solution which generates more comprehensive and plausible indoor models in a fully automatic fashion.

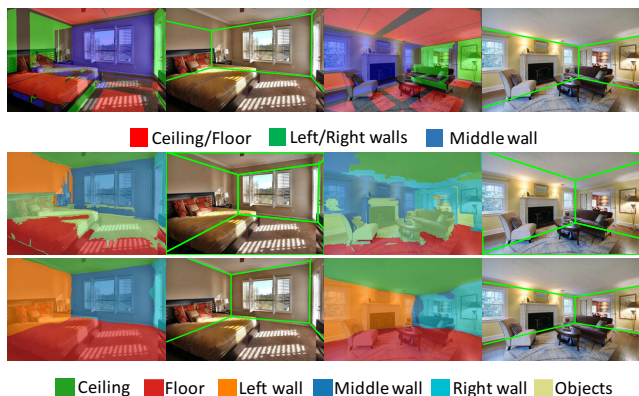The closest works to ours are [37] and [25] which find

Figure 4: Geometric feature and room layout estimation. Results from (Row1) [22] and (Row2) [15]. Bottom row: our results.

the best matching 3D scene model to a given image. Our system is a significant advance in a number of ways. In particular, [37] requires a complete scene in the database that matches each image. Hence, their approach can be thought of as "3D scene retrieval," whereas we reconstruct each scene from scratch, using a database of furniture (not scene) models. The latter allows for a much broader range of reconstructable scenes. While [25] reconstructs the scene by placing individual pieces of furniture, they make a number of limiting assumptions (axis aligned furniture, no walls, easy-to-segment objects), operate on a much smaller database (180 models), and do not demonstrate as broad a range of results. Both of [37] and [25] use hand-crafted features, while our proposed method uses CNN feature which is learned end-to-end on just image data. Guo et al. [13] render a synthesized model of the scene using RGBD (depth) images while our method only uses RGB information. The synthesized rooms produced by [13] have low fidelity in terms of object details while we retrieve the detailed ShapeNet CAD model for each object.

In the context of 3D prediction, several previous approaches estimate the depth and surface normals of visible surfaces from a single image [10, 48, 4]. In contrast, our approach does not require dense surface normal estimation but is capable of estimating both visible and invisible surfaces through joint estimation of room and object CAD models.

3. Algorithm

Our approach to reconstructing CAD models from an image (see Figure 3) is based on recognizing objects in the scene, inferring room geometry, and optimizing 3D object poses and sizes in the room to best match synthetic renderings to the input photo.

The proposed approach involves several steps, as follows. We first fit room geometry, by classifying pixels as being on walls, floor, or ceiling, and fitting a box shape to the result. In parallel, we detect all of the chairs, tables, sofas, bookshelves, beds, night tables, chest, and windows in the scene using state of the art object detection techniques.

Wherever an object, e.g., a bed, is detected with high confidence, we estimate its 3D pose, by comparing its appearance with renderings of hundreds of beds from many different angles, using a deep convolutional distance metric, trained for this purpose. Finally, we optimize for the placement of all objects in the reconstructed room by optimizing the difference between the rendered room and the photograph. Our optimization approach operates on all objects jointly, and thus accounts for inter-object occlusions.

In the remainder of the section, we describe these technical components in detail: room geometry estimation, object detection, object alignment, and scene optimization.

3.1. Room Geometry Estimation

Humans are adept at interpreting the shape of a room (i.e., positions of walls, ceiling, and floor), even in the presence of significant clutter. Computer vision algorithms have also become increasingly good at this task in the last few years by following a paradigm introduced by Hedau et al. [15] and Lee et al. [22] in which a set of room shapes are hypothesized (typically 3D boxes), and evaluated using features in the image.

We improve upon previous approaches to room geometry estimation, by adopting an alternative approach for ranking the room 3D box hypothesis using deep convolutional features. Specifically, we train a network that estimates per-pixel surface labels (ceiling, floor, left, middle, and right walls). These features are analogous to the context geometric feature ("support", "vertical", and "sky") of [17].

Unlike [17] that learns the geometry features from hand-designed low level descriptors (e.g., color, texture, and other perspective cues) over superpixels, our method uses an end-to-end deep Fully Convolutional Network (FCN) [26], using VGG [41] and converting each fully connected layer into a convolutional layer with a kernel covering the entire input region. Finally, the weights are fine-tuned for the pixel-level labeling task. In this work, we produce the output dense score map of size $41 \times 41 \times 5$ given an input image of 321×321. We then use upsampling to produce a probability map with the same size of the input image. We trained the FCN network on the annotated indoor scenes in the LSUN dataset [1].

A key advantage of the FCN-based architecture is that it integrates contextual information over the entire image. Whereas most methods use a "distractors" class [15, 27] to remove furniture from consideration, the FCN is able to use furniture as additional context, e.g., using the presence of a bookshelf or bed to infer the likely presence of an adjacent wall. We note that [27] also used a convolutional network, but rather than classifying surface orientations directly as we do, they estimate informative edges in the scene, and employ a second stage to iteratively re-label room surfaces and rank room box estimates.

Figure 5: Object detection result on sample images. Each object category is shown with a different color. The numbers attached to boxes show the probabilities assigned to each detection.

3.2. Object Detection

The first step in our furniture modeling pipeline is to detect the presence of objects of interest in the image, and their 2D bounding boxes. While any number of object detectors can be trained, we focused specifically on the following: chair, table, sofa, bookshelf, bed, night table, chest, and window.

Object detection is an area that has seen explosive progress in the last several years, and existing methods work impressively well. In particular, we use the state-of-the-art Faster-RCNN [32] deep network for detection. This network performs two steps to detect objects. First it produces object region proposals, and then it computes the likelihood of each category for the proposed objects using deep convolutional layers. The region proposal layer produces bounding boxes of different scales and aspect ratios. This network is initialized with pre-trained models from large scale object recognition tasks (ILSVRC2012) [19]. The network weights are then fine-tuned for the object proposal and object detection tasks by minimizing an objective function for a multi-task loss on bounding box regression and object misclassification. The trained network is then able to produce bounding boxes with object categories for any image. The network output also includes an object score which shows the probability of that particular object in the bounding box. Greedy non-maximum suppression (NMS) is used to obtain a single peak detection for each object, remove low scoring detections that overlap with higher scoring object bounding boxes.

Our Faster-RCNN implementation uses VGG16 [41] architecture. We further fine-tune the weights of this network for the object detection task on our eight furniture categories using three publicly available datasets, namely SUN2012 detection dataset [45], ImageNet detection challenge dataset [35], and the window category of Rent3D dataset [24]. We show detection results on a sample of our images in Figure 5.

3.3. CAD Model Alignment

The object detection results from Section 3.2 identify the presence of a "chair" (e.g.,) in a certain region of the image with high probability. Now we wish to determine what *kind* of chair it is, its shape, and approximate 3D pose.

Inspired by [3], we solve this retrieval problem by searching for 3D models that are most similar in appearance to the detected objects in the image. Specifically, we consider all 3D models in the ShapeNet repository [6] associated with our object categories of interest, i.e., chair, table, sofa, bookshelf, bed, night table, chest, yielding 9193 models in total. Each 3D model is rendered to 32 viewpoints, consisting of 16 uniformly sampled azimuth angles and two elevation angles (15 and 30 degrees above horizontal).

Robust comparison of photos with CAD model renderings is not straightforward; simple norms like L2 do not work well in practice, due to differences in shape, appearance, shading, and the presence of occluders. We achieve good results, once again, by using convolutional nets (see Figure 6); we compute deep features for each of the rendered images and the detected image bounding boxes and use cosine similarity as our distance metric. More specifically, we use the convolution filter response in the ROI-pooling layer of the fine-tuned Faster-RCNN network [32] described in Section 3.2. A benefit of using the ROI-pooling layer is that the length of its feature vector does not depend on the size and the aspect ratio of the bounding box, thus avoiding the need for non-uniform rescaling (a source of artifacts in general). Choosing the rendering that best matches each image object detection yields an estimate both for the best-matching CAD model and its approximate 3DOF orientation.

3.4. Object Placement in the Scene

Equipped with a set of CAD models and their approximate orientations, we now wish to place them in the reconstructed room. This placement need not be exact, as we will further optimize it in a subsequent step, but should be a reasonable initial estimate. To this end, we first estimate the intrinsic camera parameters (K) and camera rotation (R) with respect to the room space using three orthogonal vanishing points [15], and choose one of the visible room corners as the origin of the world coordinate system. If none of the corners are visible, we choose the origin to be the intersection of the visible wall edge with the floor.

The ShapeNet 3D models are normalized with a bounding box corresponding to a unit cube. Based on the alignment procedure from Section 3.3, we can determine the input photo pixel locations corresponding to each of the eight corners of this cube. We can find the object location and scale in the x and y (parallel to ground plane) directions by intersecting the ground plane with the ray casted from the camera center through the input image pixels corresponding to the bottom four corners of the aligned CAD model cube.
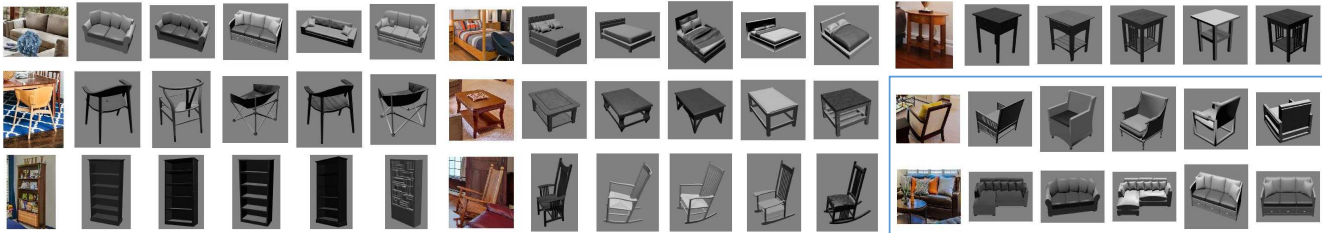
Figure 6: Results of the top five aligned CAD models retrieved for the given object detection bounding box. The retrieved models have similar style and pose with the given object. Last two rows on the right column show failure cases: (Row1) visual feature confusion between different poses of the chair, and (Row2) heavy occlusion of sofa by table has made it visually similar to an L-shaped sofa.

To compute the object scale along the z axis, we compute the ratio between the length of the four vertical edges of the projected cube and the length of those edges from the ground plane to the intersection of those lines with the horizontal vanishing line. Note that the height of the vanishing line is equal to the camera height.

We treat windows as a special case, as they are attached to walls instead of floor. To place windows, we find the intersection of the window bounding box from object detection with each of the walls and assign the window to the wall with which it has the largest overlap. The window's detected bounding box in the image back-projects to a quadrilateral on the assigned wall. The pose and location of window is determined by the largest axis-aligned rectangle on the wall plane contained within that quadrilateral.

3.5. Scene Optimization via Render and Match

The placement procedure in Section 3.4 is sensitive to several sources of error including the quantization of object orientations, ground plane misregistration, occlusions, and other factors, which can lead to erroneous estimates of object pose and scale. We therefore introduce an optimization in which the configurations of all objects in the scene are jointly aligned. A benefit of this procedure is that it properly accounts for inter-object occlusions, and yields more accurate estimates for object location, scale, and orientation.

After estimating the 3D room geometry and the initial placement of the objects in the scene, we refine our object placements by optimizing the visual similarity of the rendered scene with that of the input image. To this end, we solve an optimization problem where the variables are the 3D object configurations in the scene and the objective function is the minimization of the cosine distance between the convolutional features obtained from the camera view rendered scene and the input image.

More formally, suppose we detect objects $\{O_1, ..., O_k\}$ in the scene. The placement of each object O_i is represented by its (x, y, z) location, scale along the x, y and z axis as well as the rotation. The variables for all N objects are concatenated into a $7N$ parameter vector. Given a parameter vector, we can generate the rendered image of the scene, denoted I^*. The cost function used in our optimization tries

to minimize the cosine distance between I^* and the original input image I:

$$\min \Phi(I^*, I) = \frac{1}{|\mathcal{C}|} \sum_{C_i \in \mathcal{C}} 1 - \frac{C_i(I^*) \cdot C_i(I)}{\| C_i(I^*) \| \| C_i(I) \|} \quad (1)$$

We model the feature vector of an image by using the outputs of all convolutional layers [1]. In the above equation, $\mathcal{C}$ refers to the set of conv layers in the network and C_i is the feature vector obtained from the ith layer. The total cost function is the average similarity of all layers. The convolution filters in higher layers of the network provide abstract shape features while the details of the images such as edges and corners appear in the features obtained from the lower layers of the network. The features in higher levels have larger receptive fields, and can therefore cope with larger displacements, and help the optimization to converge in the first iterations when the initial estimates are far off. Similarly, the lower convolutional layers play a greater role in later iterations, to help the objects converge with more precision. In this way, the network provides a natural coarse-to-fine structure to the optimization.

Since our objective function is not differentiable we use COBYLA [30], a derivative free numerical optimization method, deployed in a Python optimization package. We found this procedure to work very well in practice. Figure 7 shows the convergence of the method for example scenes.

4. Coloring CAD models

We use a medoid color of each object in the input image for scene optimization (Section 3.5) and visualization. The process is as follows. First, we project the best aligned CAD model of an object onto its bounding box in the image. We then find the median value of each color channel separately, and take the closest color which appears within the mask. We also compute the medoid color for each wall of the room using a similar approach. We compute the mask of each wall through the room geometry, and exclude the bounding boxes from detected objects. This approach works well in

---

[1] We use conv1-1, conv1-2, conv2-1, conv2-2, conv3-1, conv3-2, conv3-3, conv4-1, conv4-2, conv4-3, conv5-1, conv5-2 and conv5-3 layers in the VGG network

| iter: 0 | 30 | 60 | 100 | iter: final | input photo | error convergence |

Figure 7: Results of the joint scene optimization step. (Column 1) The initial object placement in the scene. (Columns 2-5) Rendering of the scene in sample iterations during optimization. (Column 5) The last iteration of optimization. (Last column) The objective function error and the optimization convergence. The objective function minimizes dis-similarity between the real and the rendered image. Red dots show the sample iterations that are shown above.

Method	Pixel Error(%)
Lee et al. [22]	24.70
Hedau et al. [15]	21.20
Del Pero et al. [29]	16.30
Gupta et al. [14]	16.20
Zhao et al. [50]	14.50
Schwing et al. [39]	13.59
Ramalingam et al. [31]	13.34
Mallya et al. [27]	12.83
Dasgupta et al. [9]	9.73
Ren et al. [33]	8.67
IM2CAD	10.15

Table 1: Room layout pixel misclassification error on Hedau [15].

practice for scene optimization and creates visually pleasant renderings without falling into the uncanny valley [40] (see results in Figure 8).

5. Experimental Results

We evaluate our IM2CAD system both qualitatively and quantitatively on scene understanding benchmarks.

5.1. Qualitative Evaluation

We evaluated the proposed IM2CAD system with 100 real world indoor images collected from "Zillow Digs" [51]. These images are living room and bedroom shots as our training object categories are chair, table, sofa, bookshelf, bed, night table, chest, and window, i.e., typical bedroom and living room furniture. We cover a variety of room styles from traditional to modern with various furniture arrangement, complexity, and clutter that are representative of real world scenes. We also show example results on the SUN RGB-D dataset.

Our IM2CAD approach consistently produces reasonable results on most of the test images. Figure 8 is representative of the top 30% of our results, where most pieces of furniture are detected, represented using well-matched CAD models, and properly posed. Typical failure results are shown in Figure 9. Our failures are rarely catastrophic,

Method	Pixel Error(%)
Hedau et al. [15]	24.23
Mallya et al. [27]	16.71
Dasgupta et al. [9]	10.63
Ren et al. [33]	9.31
IM2CAD	10.04

Table 2: Room layout pixel misclassification error on LSUN [1].

and generally fall into the category of some furniture items being omitted or misplaced.

Object pose estimation can sometimes get stuck in local optimal. Notice that the foreground chair in Figure 9(a) is in an incorrect pose while the chair legs are aligned almost perfectly to the image. The last two rows of Figure 6 demonstrate cases where the visual similarity fails to retrieve appropriate CAD models. Heavily occluded objects impose additional challenges. Notice the missing chair and coffee table in Figure 9(a) and (b). If the room shape is not perfectly cubic (Figure 9(c)), room layout estimation can fail to recover the true room shape. The windows can be confused with paintings as they have very similar visual features (see Figure 8). Both windows and paintings typically appear as glassy and shiny rectangular shapes on a wall.

We use Caffe [19] to implement and train our deep networks. We learn the weights of our FCN network for the room geometry estimation using stochastic gradient descent with initial learning rate of 0.001 and weight decay of 5e-4. We train our network in 45 epochs where the learning rate decreases every 15 epochs. For the object detection we use same threshold for all object categories and only keep the detection boxes with scores higher that 0.5.

Object detection and geometric feature extraction are processed on a Titan X GPU, while room layout sampling and object pose estimation are computed on CPU. For a typical input image of size 300×500, the computational time is approximately 0.15 seconds for object detection, 0.3 seconds for geometric feature extraction, 8 seconds for room layout sampling and ranking, and 10 seconds for object placement. Scene optimization is an iterative process where each iteration takes about 1 second. We set the maximum
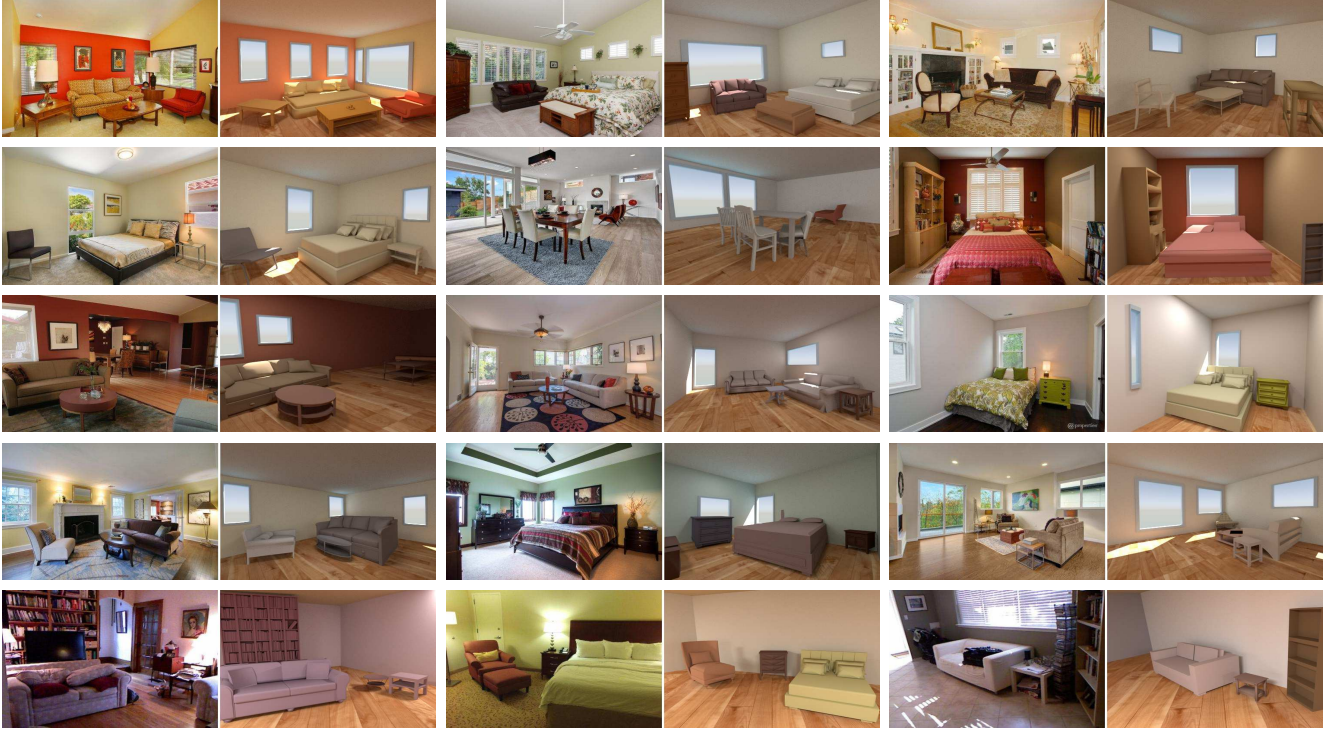
Figure 8: The reconstruction results. In each example the left image is the real input image and the right image is the rendered 3D CAD model produced by IM2CAD. Last row shows example results on the SUN RGB-D dataset.

Method	SUN RGB-D	3DGP
Hedau et al. [15]	49.4	47.3
IM2CAD	62.6	63.2

Table 3: 3D room estimation results using voxel IoU on SUN RGB-D [42] and 3DGP [8] datasets (higher is better).

Method	SUN RGB-D		3DGP	
	voxel IoU	mAP	voxel IoU	mAP
3DGP [8]	38.7	42.1	38.4	59.7
IM2CAD (w/o optim.)	46.1	74.7	53.5	86.6
IM2CAD (w/ optim.)	49.0	75.6	53.8	86.6

Table 4: 3D scene free space prediction (voxel IoU) and object localization (mAP) results on SUN RGB-D [42] and 3DGP [8] datasets (higher is better).

number of iterations to be 250. The overall CAD model creation process finishes within 5 minutes.

To produce final room renderings with global illumination, we use Blender Cycle Render Engine [5], with fixed lighting consisting of distant sunlight from the top right point and five area lights on the ceiling. The final rendering process takes about 15 minutes with global illumination.

5.2. 2D Room Layout Estimation

To evaluate the accuracy of room layout estimation, we compute the pixelwise difference between the predicted layout and the ground truth layout labels, averaged across all images as the evaluation metric. We evaluated on the test split of [15] dataset (we do not use their training split). Our FCN features (without 3D box estimation) achieve a 12.4% pixel misclassification error compared to 28.9% of [17] on the leading benchmark dataset [15] (see Figure 4). When combined with a box-fitting step of [15, 22], we achieve competitive result of 10.15% error compared with [9] and [33] as shown in Table 1. More specifically, we improve the reported result of [27] by 2.7%, [9] by 3.1%, and [33] by 4.2%. As an ablation study to evalu-

ate the effect of different room hypothesis estimation approaches, we tested our approach while being combined with either of [15] or [22] and we obtain an error of 11.02% and 11.13%, respectively.

We also evaluated performance on the task of room layout pixel misclassification using the LSUN dataset [1]. As summarized in Table 2, IM2CAD outperforms previous approaches [15, 27] significantly as well as [9] and obtains competitive results with recent approach of [33].

5.3. 3D Room Estimation and Scene Understanding

Our IM2CAD system is also applicable for 2D and 3D scene understanding as well as room layout estimation. For evaluating our performance in scene understanding tasks, we use the SUN RGB-D dataset [42]. This dataset contains images captured from different view points, some of the images have low field of view and a considerable number of them are captured from highly cluttered scenes. Note that, although the SUN RGB-D dataset contains the depth data

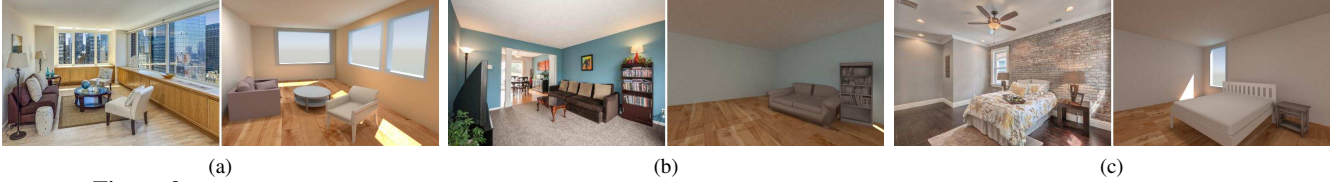| (a) | (b) | (c) |

Figure 9: Failure cases: inaccurate chair pose (a); mis-detection of a chair (a) and table (b); non-cubic room shape (c).

for all the images, we do not use the depth information at either train or test time, but estimate the 3D room geometry as well as object layout using only single 2D images. We use the test split for the bedroom and living room scene categories with a total of 484 images.

3D Room Layout Estimation   3D room layout estimation enables precise reasoning about free space versus spaces occupied by objects. In the absence of depth data, this task is challenging as it requires reasoning about room geometry from 2D images. Our 3D room layout estimation is evaluated by computing the intersection over union (IoU) between the predicted and the ground truth free spaces. Following [42], we assume empty rooms without objects and define a voxel grid of $0.1 \times 0.1 \times 0.1$ meter. The effective voxels are the ones that are located within 0.5 and 5.5 meters from the camera and are inside the field of view. We check whether each voxel is inside the 3D room polygon and compute the intersection and union computed by counting the 3D voxels. Table 3 summarizes our obtained results. Our method outperforms [15] by 13.2%.

Scene Understanding   The task of scene understanding integrates recognition and localization of all the objects as well as estimating the room structure. Compared to the task of 3D room estimation, this is a more challenging task as it requires detecting non-free spaces occupied by the objects. We compute the distance between the projection of the box centroid on the ground plane for all pairs of predicted and ground truth objects with the same label. We sort the distances in ascending order for each available pair and choose the pair with the shortest distance while the two boxes are marked as unavailable. We compute the precision and recall by varying the distance threshold and use the mean average precision as object localization metric.

Free space prediction is evaluated in a similar manner to the 3D room layout. The visible 3D voxels for the free space inside the room polygon but outside any object bounding box is computed and then the IoU between the free space prediction and the ground truth is computed. Table 4 shows the results of free space prediction and object localization on SUN RGB-D dataset. We compare the performance of our approach for scene understanding with [8]. IM2CAD obtains superior results compared with [8] in both metrics i.e., 33.5% boost in the mean AP and 11.7% in scene free space prediction. We compare our results before and after applying scene optimization (Section 3.5). Our scene optimization approach results in improved accuracy for the task of scene understanding.

We also report IM2CAD performance on the dataset presented in [8] which we call 3DGP. We use 372 images from the test split of living room, bedroom and dining room categories. However, we do not train our model on the 3DGP training set. To estimate the ground truth camera parameters, we compute the pseudo ground truth vanishing points by using the annotated ground truth edges corresponding to the three vanishing points following the experimental setting of [8] for 3D scene evaluation. We evaluate on the three tasks of 3D room layout, whole scene free space prediction, and object localization. These results are summarized in Tables 3 and 4. For the task of 3D room estimation, IM2CAD significantly outperforms [8] by 15.9%. In the free space prediction task, IM2CAD obtains significantly better results than 3DGP in both voxel IoU and mean AP criteria.

6. Conclusion

This paper presents a fully automatic system that reconstructs a 3D CAD model of an indoor scene from a single photograph, by utilizing a large database of 3D furniture models. It estimates room geometry, and detects and aligns objects in the image with accurate 3D poses. We introduce novel approaches for room modeling and scene optimization, that are keys to the success of our system. We evaluate on a wide range of living room and bedroom photographs with a variety of home styles. The results demonstrate the effectiveness of our approach in creating 3D CAD models that faithfully resemble the real scenes. With the abundance of indoor photos available online, our system is applicable to produce a large database of indoor scene models. Our approach obtains significant improvement on the 2D room layout estimation and 3D scene understanding benchmarks.

Our system does have limitations suggesting a number of areas for future work. We assume the room geometry in the image can be modeled with a cube. Working with complicated room geometry is an area of future improvement. Understandably, heavily occluded objects impose challenges. We assume objects are always on the ground plane (e.g., chairs and beds) or attached to walls (windows), posing a lamp on a table would require extension of our work. Incorporating more object types would lead to more general scenes and room types (e.g. kitchens and bathrooms).

Acknowledgements

References

[1] Lsun room layout estimation dataset. http://lsun.cs.princeton.edu/, 2015. 2, 3, 6, 7

[2] A. Y. N. Ashutosh Saxena, Sung H. Chung. Learning depth from single monocular images. In *NIPS*, 2005. 2

[3] 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 *CVPR*, 2014. 2, 4

[4] A. Bansal, B. Russell, and A. Gupta. Marr revisited: 2d-3d alignment via surface normal prediction. In *CVPR*, 2016. 2, 3

[5] Blender. Blender cycles render engine. https://www.blender.org/manual/en/render/cycles/index.html. 7

[6] 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. Technical Report arXiv:1512.03012 [cs.GR], 2015. 2, 4

[7] A. X. Chang, M. Savva, and C. D. Manning. Learning spatial knowledge for text to 3d scene generation. In *EMNLP*, 2014. 2

[8] W. Choi, Y.-W. Chao, C. Pantofaru, and S. Savarese. Indoor scene understanding with geometric and semantic contexts. *IJCV*, 2015. 2, 7, 8

[9] S. Dasgupta, K. Fang, K. Chen, and S. Savarese. Delay: Robust spatial layout estimation for cluttered indoor scenes. In *CVPR*, 2016. 2, 6, 7

[10] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015. 3

[11] M. Fisher, D. Ritchie, M. Savva, T. Funkhouser, and P. Hanrahan. Example-based synthesis of 3d object arrangements. *TOG*, 31(6):135, 2012. 2

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 2

[13] R. Guo, C. Zou, and D. Hoiem. Predicting complete 3d models of indoor scenes. *arXiv preprint arXiv:1504.02437*, 2015. 3

[14] A. Gupta, M. Hebert, T. Kanade, and D. M. Blei. Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. In *NIPS*, 2010. 2, 6

[15] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In *ICCV*, 2009. 2, 3, 4, 6, 7, 8

[16] D. Hoiem, A. A. Efros, and M. Hebert. Automatic photo pop-up. In *SIGGRAPH*, 2005. 2

[17] D. Hoiem, A. A. Efros, and M. Hebert. Recovering surface layout from an image. *IJCV*, 2007. 3, 7

[18] Q. Huang, H. Wang, and V. Koltun. Single-view reconstruction via joint analysis of image and shape collections. In *SIGGRAPH*, 2015. 2

[19] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*, 2014. 4, 6

[20] N. Kholgade, T. Simon, A. Efros, and Y. Sheikh. 3D object manipulation in a single photograph using stock 3d models. In *SIGGRAPH*, 2014. 2

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*. 2012. 2

[22] D. C. Lee, M. Hebert, and T. Kanade. Geometric reasoning for single image structure recovery. In *CVPR*, 2009. 2, 3, 6, 7

[23] J. J. Lim, A. Khosla, and A. Torralba. Fpm: Fine pose parts-based model with 3d cad models. In *ECCV*, 2014. 2

[24] C. Liu, A. G. Schwing, K. Kundu, R. Urtasun, and S. Fidler. Rent3d: Floor-plan priors for monocular layout estimation. In *CVPR*, 2015. 4

[25] Z. Liu, Y. Zhang, W. Wu, K. Liu, and Z. Sun. Model-driven indoor scenes modeling from a single image. In *Proceedings of the 41st Graphics Interface Conference*, 2015. 2, 3

[26] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 3

[27] A. Mallya and S. Lazebnik. Learning informative edge maps for indoor scene layout prediction. In *ICCV*, 2015. 2, 3, 6, 7

[28] P. Merrell, E. Schkufza, Z. Li, M. Agrawala, and V. Koltun. Interactive furniture layout using interior design guidelines. In *SIGGRAPH*, 2011. 2

[29] L. D. Pero, J. Bowdish, D. Fried, B. Kermgard, E. Hartley, and K. Barnard. Bayesian geometric modeling of indoor scenes. In *CVPR*, 2012. 2, 6

[30] M. J. Powell. A direct search optimization method that models the objective and constraint functions by linear interpolation. In *Advances in optimization and numerical analysis*. 1994. 5

[31] S. Ramalingam, J. Pillai, A. Jain, and Y. Taguchi. Manhattan junction catalogue for spatial reasoning of indoor scenes. In *CVPR*, 2013. 6

[32] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 2, 4

[33] Y. Ren, C. Chen, S. Li, and C.-C. J. Kuo. A coarse-to-fine indoor layout estimation (cfile) method. *arXiv preprint arXiv:1607.00598*, 2016. 2, 6, 7

[34] L. G. Roberts. *Machine perception of three-dimensional solids*. PhD thesis, Massachusetts Institute of Technology, 1963. 1

[35] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015. 4

[36] R. Salas-Moreno, R. Newcombe, H. Strasdat, P. Kelly, and A. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *CVPR*, 2013. 2

[37] S. Satkin, M. Rashid, J. Lin, and M. Hebert. 3dnn: 3d nearest neighbor. *IJCV*, 2015. 2, 3

[38] A. G. Schwing, S. Fidler, M. Pollefeys, and R. Urtasun. Box in the box: Joint 3d layout and object reasoning from single images. In *ICCV*, 2013. 2

[39] A. G. Schwing and R. Urtasun. Efficient exact inference for 3d indoor scene understanding. In *ECCV*. 2012. 6

[40] J. Seyama and R. S. Nagayama. The uncanny valley: Effect of realism on the impression of artificial human faces. *Presence*, 16(4):337–351, 2007. 6

[41] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3, 4

[42] S. Song, S. P. Lichtenberg, and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *CVPR*, 2015. 2, 7, 8

[43] S. Tulsiani and J. Malik. Viewpoints and keypoints. In *CVPR*, 2015. 2

[44] J. Wu, T. Xue, J. J. Lim, Y. Tian, J. B. Tenenbaum, A. Torralba, and W. T. Freeman. Single image 3d interpreter network. In *ECCV*, 2016. 2

[45] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010. 4

[46] J. Xiao, B. Russell, and A. Torralba. Localizing 3D cuboids in single-view images. In *NIPS*, 2012. 2

[47] L.-F. Yu, S.-K. Yeung, C.-K. Tang, D. Terzopoulos, T. F. Chan, and S. J. Osher. Make it home: automatic optimization of furniture arrangement. In *SIGGRAPH*, 2011. 2

[48] B. Zeisl, M. Pollefeys, et al. Discriminatively trained dense surface normal estimation. In *ECCV*, 2014. 3

[49] Y. Zhang, S. Song, P. Tan, and J. Xiao. PanoContext: A whole-room 3d context model for panoramic scene understanding. In *ECCV*, 2014. 2

[50] Y. Zhao and S.-C. Zhu. Scene parsing by integrating function, geometry and appearance models. In *CVPR*, 2013. 6

[51] Zillow. Zillow Digs. http://www.zillow.com/digs/. 6