Recovering the Spatial Layout of Cluttered Rooms

Varsha Hedau
Electical and Computer Engg. Department
University of Illinois at Urbana Champaign
vhedau2@uiuc.edu

Derek Hoiem, David Forsyth
Computer Science Department
University of Illinois at Urbana Champaign
{dhoiem,daf}@uiuc.edu

Abstract

In this paper, we consider the problem of recovering the spatial layout of indoor scenes from monocular images. The presence of clutter is a major problem for existing single-view 3D reconstruction algorithms, most of which rely on finding the ground-wall boundary. In most rooms, this boundary is partially or entirely occluded. We gain robustness to clutter by modeling the global room space with a parameteric 3D "box" and by iteratively localizing clutter and refitting the box. To fit the box, we introduce a structured learning algorithm that chooses the set of parameters to minimize error, based on global perspective cues. On a dataset of 308 images, we demonstrate the ability of our algorithm to recover spatial layout in cluttered rooms and show several examples of estimated free space.

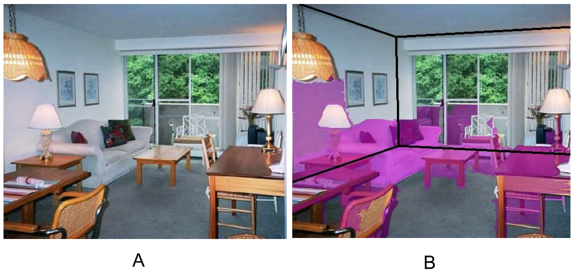


Figure 1. We model the space of a room with a 3D box layout of its entire extent (black lines) and surface labels that localize visible objects (pink regions). By estimating the joint box parameters using global perspective cues and explicitly accounting for objects, we are often able to recover the spatial layout in cluttered rooms.

1. Introduction

Look at the image in Fig. 1. From this single image, we humans can immediately grasp the spatial layout of the scene. Our understanding is not restricted to the immediately visible portions of the chairs, sofa, and walls, but also includes some estimate of the entire space. We want to provide the same spatial understanding to computers. Doing so will allow more precise reasoning about free space (e.g., where can I walk) and improved object reasoning. In this paper, we focus on indoor scenes because they require careful spatial reasoning and are full of object clutter making the spatial layout estimation difficult for existing algorithms.

Recovering Spatial Layout. To recover the spatial layout of an image, we first need to answer: how should we parameterize the scene space? Existing parametrizations include: a predefined set of prototype global scene geometries [17]; a gist [18] of a scene describing its spatial characteristics; a 3D box [11, 23] or collection of 3D polyhedrals [6, 15, 19]; boundaries between ground and walls [1, 4]; depth-ordered planes [26]; constrained arrangements of corners [13]; a pixel labeling of approximate local surface orientations [9], possibly with ordering constraints [14]; or depth estimates at each pixel [22]. Models with few parameters allow robust estimation at the risk of introducing bias. But even loosely parameterized models may not capture the full spatial layout. For instance, depth maps provide only local information about visible surfaces and may not be useful for path planning in densely occupied scenes. Our approach is to model the scene jointly in terms of a 3D *box layout* and *surface labels* of pixels. The box layout coarsely models the space of the room as if it were empty. The surface labels provide precise localization of the visible object, wall, floor, and ceiling surfaces. By modeling both of these together, we attain a more complete spatial layout estimate. We also gain robustness to clutter from the strongly parameterized box layout, without sacrificing the detail provided by the surface labels.

Once we decide how to parameterize the spatial layout, how do we estimate the parameters using image cues? Region-based local color, texture, and edge cues, often combined with segmentation or CRF inference have been used with success [9, 14, 22] for labeling pixels according to orientation or depth. Accordingly, we use Hoiem et al.'s algorithm [9] and color, texture, position, and perspective cues to estimate confidences for our surface labels. But these region-based approaches and local cues are sensitive to clutter and, therefore, not suitable for the 3D box estimation. For instance, estimating the box parameters by looking for the wall-floor, wall-wall, and wall-ceiling boundaries in

2009 IEEE 12th International Conference on Computer Vision (ICCV)
978-1-4244-4419-9/09/$25.00 ©2009 IEEE

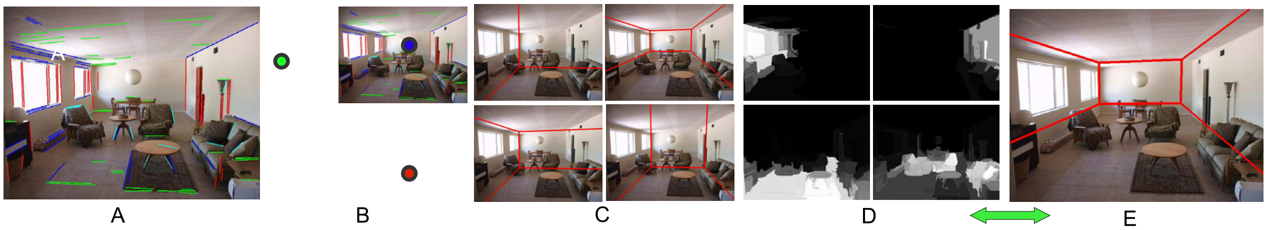


Figure 2. Our process takes the original image, identifies long line segments (A), and uses these to find vanishing points (B). The line segments in (A) are colored with the color of the vanishing point they vote for. This information, and other features, are used to produce a set of possible layouts, which are ranked by a function learned with structure learning (four are shown in C). The top ranked layouts produce maps of label probabilities (shown in D for "left wall","floor", "right wall" and "object"). In turn these maps are used to re-estimate features, and the re-estimated features are used to produce a second ranking. The top ranked layout for this image is in (E).

the image can be effective only when those boundaries are visible, which is usually not the case. We propose to estimate the box parameters in two steps. First, we find mutually orthogonal vanishing points in the image, specifying the box orientation. Then, we sample pairs of rays from two of the vanishing points, which specifies the translation of the walls. We propose a structured learning [25] approach to select the joint set of parameters that is most likely to maximize the similarity to the ground truth box based on global perspective cues.

The box layout and surface labels are difficult to estimate individually, but each provides cues that inform the other. For this reason, we iteratively estimate the box and surface labels, estimating parameters for one based, in part, on cues computed from the current estimate of the other. Our experiments show that this leads to roughly a one-third reduction in error for the box layout and surface label estimates. This integrative scene analysis is conceptually similar to recent work (e.g., [8, 10, 2, 16, 24]) on combining object, depth, viewpoint, occlusion boundary, scene category, and/or surface orientation estimates. However, our method is the first to integrate local surface estimates and global scene geometry, and we define appropriate techniques and interaction cues to take full advantage of their synergy.

Clutter. Most rooms are full of objects, presenting a major problem for existing spatial layout estimation methods. In some cases, the issue is the choice of scene space model. For instance, the assumption of a continuous ground-vertical boundary made by Delage et al. [4] and Barinova et al. [1] is not valid for cluttered scenes. Likewise, Hoiem et al. [9, 10] cannot estimate the depth of walls when their boundary is fully occluded. The method of Liu et al. [14] to label into floor, walls, and ceiling does not account for objects and has difficulty assigning the correct label to the occluded walls or floor. In other cases, such as Nedovic et al. [17], the texture and edges on objects obscure the scene shape. In all of these cases, clutter is an acknowledged problem (see Sec. 5.1 for experimental comparison).

Summary of Contributions. Our main contribution is an approach to recover spatial layout of indoor scenes in a way that is robust to clutter. We achieve robustness for three main reasons. First, our strongly parameteric 3D box model allows robustness to spurious edges and is well-suited to describe most room spaces. Second, we propose a new algorithm to jointly estimate the box parameters, using structured learning to predict the most likely solution based on global perspective cues. Third, we explicitly model clutter and propose an integrated approach to estimate clutter and 3D box, each aiding the other. The surface labels allow us to distinguish between lines that lie on objects and those that lie on walls (improving the box estimates), while the box estimates provide strong constraints on the surface labels. Our experiments on 308 images of indoor scenes show that our method can accurately estimate spatial layout in cluttered scenes and that the above innovations greatly contribute to its success. We also show that our recovered spatial layout can be used to estimate the free space of a scene by making some simple assumptions about the objects (Sec. 5.3). While we have focused on indoor scenes in this paper, we believe that many of the contributed models and techniques will be useful for estimating spatial layout in other types of scenes.

2. Overview of Approach

Our approach is illustrated in Fig. 2. We first find straight lines in the image (Fig. 2A) and group them into three mutually orthogonal vanishing points [20, 12, 3] (Fig. 2B). The vanishing points specify the orientation of the box, providing constraints on its layout. By sampling rays from these vanishing points, we generate many candidates for the box layout (Fig. 2C) and estimate the confidence for each using edge-based image features and learned models. We then estimate the surface labels given the most likely box candidate, providing a set of confidence maps from pixels to surfaces (Fig. 2D). The surface labels, in turn, allow more robust box layout estimation (Fig. 2E) by providing confidence maps for visible surfaces and distinguishing between edges that fall on objects and those that fall on walls, floor, or ceiling.

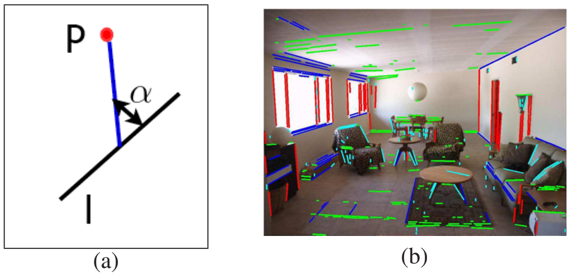


Figure 3. **(a) Angular distance** of a line segment to a vanishing point, computed as the angle between the line segment and the line joining the mid point of the segment to the vanishing point. **(b) Line memberships**: red, green and blue lines correspond to 3 vanishing points, and the outlier lines are shown in cyan.

3. Estimating the Box Layout

We generate candidate box layouts in two steps. First, we estimate three orthogonal vanishing points using a standard algorithm (Sec. 3.1) to get the box orientation. Next, by sampling pairs of rays from two of these vanishing points (Sec. 3.2), we specify set of wall translations and scalings that are consistent with the orientation provided by the vanishing points. Choosing the best candidate layout is difficult. We propose a criteria that measures the quality of a candidate as a whole and learn it with structured outputs (Sec. 3.3).

3.1. Estimating the Box Orientation

Under perspective projection, parallel lines in 3D intersect in the image plane at vanishing points. We assume that the room can be modelled by a box layout and that most surfaces inside the room are aligned with the room directions. We want to estimate a triplet of vanishing points corresponding to the three principal orthogonal directions of a room, which specifies the box layout orientation. Several works [3, 12, 20] address estimation of vanishing points from an image (see [5] for an excellent discussion). In our implementation, we modify Rother's algorithm [20] for finding mutually orthogonal vanishing points with more robust voting and search schemes. Rother ranks all triplets using a voting strategy, scoring angular deviation between the line and the point (see Fig. 3(a)) and using RANSAC driven search. Candidate points are chosen as intersection points of all detected lines, among which triplets are selected. We use an alternate greedy strategy. We first select the candidate point with the highest vote and then remove lines that cast high votes for this point. We quantize the remaining intersection points using variable bin sizes in the image plane, increasing as we go outwards from the image center. We use variable sizes because the position errors for vanishing point close to the image center are more critical to the estimation of room box rotation. The above operations drastically reduce the complexity of search space and works well for indoor scenes, where most lines lie along one of the principal directions.

We also extend the linear voting scheme used in [20] to a more robust exponential voting scheme. This makes the voting space more peaky, facilitating discrimination between good and bad vanishing point candidates. We define the vote of a line segment l for a candidate point p as,

$$\boldsymbol{v}(l, p) = |l| * \exp -(\frac{\boldsymbol{\alpha}}{2\sigma^2}) \quad (1)$$

α , where, is the angle between the line connecting p and midpoint of l, as shown in Fig. 3(a), and σ is the robustness threshold. In all our experiments, we set $\sigma = 0.1$. The straight lines of length greater than 30 pixels are used for estimation, resulting in 100-200 lines per image. Once the winning triplet is identified, each detected line in the image can be assigned to one of the vanishing points according to the vote it casts for these points, which we refer to as $line\ membership$. Fig. 3(b) shows lines corresponding to different vanishing points in different colors. The lines which cast votes below threshold are assumed to be outlier lines shown in cyan.

3.2. Getting the Box Translation

Knowledge of the box orientation imposes strict geometric constraints on the projections of corners of the box, as shown in Fig. 4 and listed below. At most 5 faces of the box,corresponding to 3 walls, floor and ceiling, can be visible in the image, each projecting as a polygon. The 3D corners of the box are denoted by *A*, *B*, *C*, and *D*, and their counterparts in the image are *a*, *b*, *c* and *d*. The vanishing points corresponding to three orthogonal directions in world are given by vp_1, vp_2, and vp_3.

1. Lines ab and cd should be colinear with one of the vanishing points, say vp_1,
2. Lines ad and bc should be colinear with the second vanishing point, vp_2, and,
3. The third vanishing point, vp_3, should lie inside the quadrilateral $abcd$.

To generate the candidate box layouts, we choose vp_1 and vp_2 as the two farthest vanishing points (see Fig. 4) and draw pairs of rays from these points on either side of vp_3. The intersections of these rays define the corners of the middle wall, $a - d$ in the image. The rest of the face polygons are generated by connecting points $a - d$ to vp_3. When fewer than 5 faces are visible, the corners will lie outside of the image, handled by a single dummy ray not passing through the image. An example box layout is overlaid in red in Fig. 4. In our experiments, we use 10 evenly spaced rays per vanishing point to generate the different candidate layouts for an image.

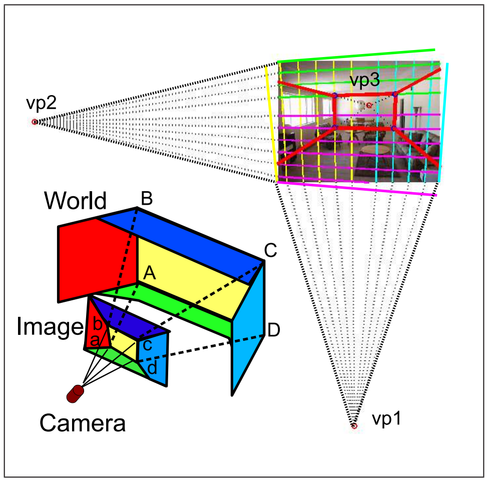


Figure 4. **Layout generation**: Once the vanishing points are known, we sample the space of translations. A layout is completely specified by two rays through each of two vanishing points, which give four corners and four edges, and the remaining edges of the box follow by casting rays through the third vanishing point and these corners.

3.3. Learning to Rank Box Layouts with Structured Outputs

We want to rank the box layouts according to how well they fit the ground truth layout. Given a set of indoor training images $\{x_1, x_2, ...x_n\} \in \mathbb{X}$ and their layouts $\{y_1, y_2, ...y_n\} \in \mathbb{Y}$ we wish to learn a mapping $f : \mathbb{X}, \mathbb{Y} \rightarrow \mathbb{R}$ which can be used to assign a score to the automatically generated candidate layouts for an image, as described in Sec. 3.2. Each layout here is parameterized by five *face* polygons, $y = \{F_1, F_2, F_3, F_4, F_5\}$. The mapping f should be such that $f(x_i, y)$ takes a high value for the correct combination of input image and layout, $y = y_i$, and its value reduces as the deviation of y from y_i increases. Thus, for a new test image x, the correct layout can be chosen as y^*, where,

$$y^* = \arg\max_y f(x, y; w) \tag{2}$$

The above is a structured regression problem, where the output is not a binary decision, but a layout which has a complex structure. To solve this, we use the structured learning framework described in [25], which models the relationships between different outputs within the output space to better utilize the available training data. We set, $f(x, y) = w^T\psi(x, y)$, where $\psi(x, y)$ is a vector of features. The mapping f is learned discriminatively by solving

$$\begin{gathered} \min_{w,\xi} \tfrac{1}{2}||w||^2 + C\textstyle\sum_i \xi_i \\ \text{s.t.} \quad \xi_i \geq 0 \; \forall i, \quad \text{and} \\ w^T\psi(x_i, y_i) - w^T\psi(x_i, y) \geq \Delta(y_i, y) - \xi_i, \\ \forall i, \forall y \in \mathbb{Y}/\; y_i \end{gathered} \tag{3}$$

where ξ_i's are slack variables, $\Delta(y_i, y)$ is the loss function quantifying the deviation between two layouts, C is a scaling constant, and, $\psi(x_i, y)$ is the set of features extracted for image layout pair (x_i, y). In our experiments, we choose $C = 1$. We define the loss function Δ with three terms: Δ_t penalizes the absence of a *face* F_k in one layout if it is present in the other one; Δ_c measures the shift in centroid c_k of the *faces* of the two layouts; and Δ_p is the sum of pixel errors of the corresponding *faces* of the two layouts, which is measured as their areas of overlap.

$$\begin{aligned} \Delta(y_i, y) &= \Delta_t(y_i, y) + \Delta_c(y_i, y) + \Delta_p(y_i, y) \\ \Delta_t(y_i, y) &= \Sigma_{k\in[1,5]}\delta(F_{ik}, F_k) \\ \Delta_c(y_i, y) &= \Sigma_{k\in[1,5]}||c_{ik} - c_k||^2 \\ \Delta_p(y_i, y) &= \Sigma_{k\in[1,5]}(1 - \frac{Area(F_{ik} \cap F_k)}{Area(F_{ik} \cup F_k)}) \end{aligned} \tag{4}$$

where $\delta(F_{ik}, F_k) = 1$ if $Area(F_{ik}) > 0$ and $Area(F_k) = 0$ or $Area(F_{ik}) = 0$ and $Area(F_k) > 0$; otherwise $\delta(F_{ik}, F_k) = 0$.

For a given choice of loss function and features the defined objective function is convex with linear constraints. However the a large number of constraints is usually a binding factor in solving such problems. In our case the number of constraints are manageable, due to the sampling of rays. However if a denser sampling is performed, one would need to resort to other approximate methods like those described in [25].

Features. $\psi(x_i, y)$ is the set of features of layout y. We use the line membership features that depend on detected straight lines in the image and their memberships to the three vanishing points, as defined in Sec. 3.1. For each face F_k, the unweighted line membership feature f_l is defined as,

$$f_l(F_k) = \frac{\sum_{l_j \in C_k} |l_j|}{\sum_{l_j \in L_k} |l_j|} \tag{5}$$

where L_k is the set of lines in F_k, C_k is the set of lines which belong to the two vanishing points for face F_k. We denote length of line by $|l|$.

Since each face is characterized by two vanishing points, most of the lines inside each face should belong to one of these two vanishing points. However this is not necessarily true for the lines on objects. For instance, in Fig. 2, the red lines on the sofa and table correspond to vertical vanishing point but fall inside the floor face. For this reason, we also compute a set of line memberships that are weighted by the confidence that a line is not inside an object region (estimated as part of the surface labels, described in Sec. 4). We also include the average pixel confidences for the surface labels within each face, which incorporates information from the color, texture, and perspective cues that influence the surface labels. Note that the box layout is first estimated without these surface-based features and then re-estimated using all features after estimating the surface labels.

4. Estimating Surface Labels

It is difficult to fit the box layout to general pictures of rooms because "clutter", such as tables, chairs, and sofas, obscures the boundaries between faces. Some of the features we use to identify the box may actually lie on this clutter, which may make the box estimate less accurate. If we have an estimate of where the clutter is, we should be able to get a more accurate box layout. Similarly, if we have a good box layout estimate, we know the position of its faces, which should allow us to better localize the clutter.

To estimate our surface labels, we use a modified version of Hoiem et al.'s surface layout algorithm [9]. The image is oversegmented into superpixels, which are then partitioned into multiple segmentations. Using color, texture, edge, and vanishing point cues computed over each segment, a boosted decision tree classifier estimates the likelihood that each segment is valid (contains only one type of label) and the likelihood of each possible label. These likelihoods are then integrated pixel-wise over the segmentations to provide label confidences for each superpixel. We modify the algorithm to estimate our floor, left/middle/right wall, ceiling, and object labels and add features from our box layout. As box layout features, we use the percentage area of a segment occupied by each face and the entropy of these overlaps, which especially helps to reduce confusion among the non-object room surfaces. In training, we use cross-validation to compute the box layout cues for the training set.

5. Experiments and Results

All experiments are performed on a dataset of 308 indoor images collected from the web and from LabelMe [21]. Images include a wide variety of rooms including, living rooms, bed rooms, corridors etc (see Fig. 6). We label these images into ground truth box layout *faces*: polygon boundaries of floor, left wall, middle wall, right wall, and ceiling. We also label ground truth surface labels: segmentation masks for object, left, middle, and right wall, ceiling, and floor regions. We randomly split the dataset into a training set of 204 and test set of 104 images.

We first evaluate the accuracy of our box layout and surface label estimates (Sec. 5.1). Our results indicate that we can recover layouts in cluttered rooms, that the integration of surface labels and box layout is helpful, and that our method compares favorably to existing methods. In Sec. 5.2, we then analyze the contribution of errors by vanishing point estimation and ray sampling in estimating box translation. Finally, we show several qualitative results for free space estimation in Sec. 5.3.

5.1. Evaluation of spatial layouts

We show several qualitative results in Fig. 6, and report our quantitative results in Tables 1, 2 and 3.

Method	Hoiem et al.	Ours (initial)	**Ours (final)**
Pixel error	28.9%	26.5%	**21.2%**
Corner error	–	7.4%	**6.3%**

Table 1. Error for box layout estimation. Our method achieves lower error than Hoiem et al.'s region labeling algorithm [9] and improves significantly further after re-estimation using cues from the surface label estimates (final). See text for details.

Method	Hoiem et al.	+Box Layout (ours)
Pixel error	26.9%	**18.3%**

Table 2. Pixel error for surface label estimation. Use of cues based on our box layout estimates significantly improves results from Hoiem et al.'s surface layout algorithm [9], which was also trained on our dataset.

Surface labels	Floor	Left	Middle	Right	Ceiling	Objects
Floor	74/68	0/0	0/1	0/1	0/0	24/30
Left	1/0	75/43	14/44	0/0	1/1	9/12
Middlle	1/0	5/2	76/82	4/6	2/1	13/9
Right	1/1	0/0	14/48	73/42	3/2	10/7
Ceiling	0/0	4/3	28/47	2/5	66/45	0/0
Objects	16/12	1/1	5/10	1/2	0/0	76/76

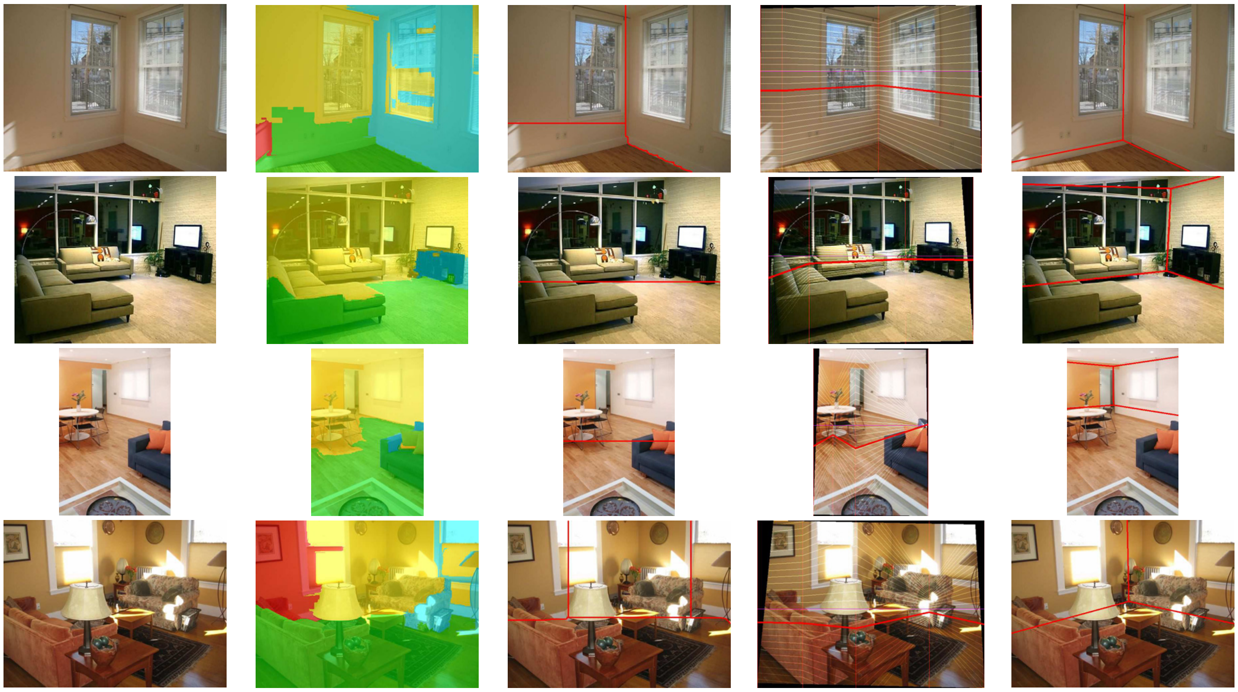
Table 3. Confusion matrix (Ours/Hoiem et al.) for surface label estimation. The (i, j)-th entry in a confusion matrix represents the percentage of pixels with ground truth label i which are estimated as label j, over all test images.

Box Layout. We evaluate the box layout using both pixel error, computed as the percentage of pixels on the box faces that disagree with ground truth, and the RMS error of corner placement as a percentage of the image diagonal, averaged over the test images. As shown in Table 1, we improve in both measures when re-estimating layouts using the surface label cues (pixel error: 26.5% to 21.2%; corner error: 7.4% to 6.3%). Our method also outperforms Hoiem et al.'s algorithm [9] (pixel error: 28.9%), which we trained on our dataset as a baseline. Note that we cannot compute RMS error for Hoiem et al.'s algorithm, since it may not provide an estimate that is consistent with any 3D box layout. In Fig. 5, we also qualitatively compare several methods [9, 14, 1]. Our method is better able to deal with clutter in indoor scenes, due to its joint estimation of box layout parameters (based on structured learning) and the explicit modeling of clutter using surface labels.

Surface Labels. Our surface label estimates improve considerably (by 8.6% average pixel error) due to inclusion of cues from the box layout, as shown in Table 2. The confusion matrix shown in Table 3 also exhibits considerable gains. Ceiling and wall estimates especially improve because they are difficult to classify with local color and texture cues and because an estimate of the box layout provides a strong prior.

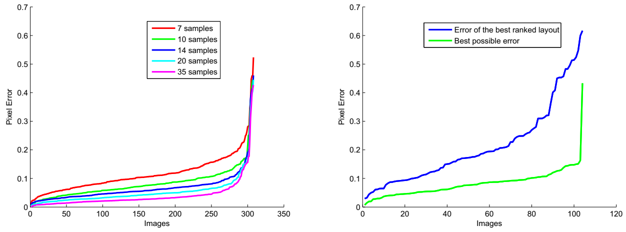
5.2. Error Analysis

Errors in vanishing point estimation and coarse sampling of rays when generating box layout candidates also con-



Input Image Hoiem *et al.* [9] Liu *et al.* [14] Barinova *et al.* [1] Our Algorithm

Figure 5. Qualitative comparison of box layout estimation for several methods. On the left, we show the input image. Then, from left to right, we compare four methods. Column 2: Most likely labels from Hoiem et al.'s algorithm [9] (floor=green; left wall=red; middle wall=yellow; right wall=cyan; ceiling=blue;). Column 3: Improved estimates using Liu et al.'s [14] ordering constraints, intialised by [9]. Column 4: Barinova et al.'s algorithm [1] recovers the ground vertical boundary as a continuous polyline (thick red) and indicates wall faces with a white spidery mesh and thin red lines. Column 5: Our algorithm. Note that Barinova et al.'s algorithm was trained on urban imagery, causing the boundary to shift upwards due to a prior for deeper scenes. Hoiem et al.'s method is sensitive to local contrast (row 1: reflection on the floor) and clutter (rows 2-4). The ordering constraints of Liu et al. improve results but cannot fix large errors. Barinova et al.'s algorithm is more robust but still has trouble with clutter, due to its assumption of a continuous ground-vertical polyline which is not true for cluttered rooms. Our box parameterization of scene space is similar to that of Barinova et al., but our method is more robust to clutter because we search the joint space of all the box parameters (learning with structured outputs), which contrasts with their greedy search strategy. Our approach of explicitly modeling clutter with the surface labels provides a further improvement. **Best viewed in color.**



(a) (b)

Figure 7. **(a)** Each curve shows the lowest pixel error between the ground truth layout and all generated layouts of an image for a given number of ray samples. Shown for all 308 training images. **(b)** For 10 ray samples per vanishing point, green and blue curves show the lowest possible achievable pixel error using any layout, and the pixel error of the layout estimated by our algorithm, respectively. Shown for all 104 test images.

tribute to our errors in spatial layout estimation. In Fig. 7(a), we compare the accuracy of the best possible box layout, while varying the granularity of ray sampling (see Sec. 3.2). The best possible layout is the one with the least pixel error w.r.t. to ground truth layout among all the candidates generated under the constraints imposed by the vanishing points (VPs). We can see that even with very fine sampling (35 rays per VP), the error is sometimes high, which is due to error in estimating the vanishing points. For 7, 10, 14, 20, and 35 sampled rays, the average minimum possible error (assuming a perfect layout ranker) is 11.6%, 8.3%, 6.7%, 5.3%, and 4.0%. Thus, about 4% of our error is likely due to our coarse sampling of 10 rays. In Fig. 7(b) we compare the top-ranked layout for our test images according to our learned parameters and the best possible layout for these images after sampling 10 rays. The average gap between the error of the layout predicted as best by our method and the best possible layout using this ray sampling is 13.0%, which is due to the errors in our ranking estimates (Sec. 3.3, 4).

As we show in Sec. 5.3, even small errors in the spatial layout can have large effects on 3D reasoning. For instance, placing the wall-floor boundary even slightly too high in the image can greatly exaggerate the depth of the room (Fig. 8,

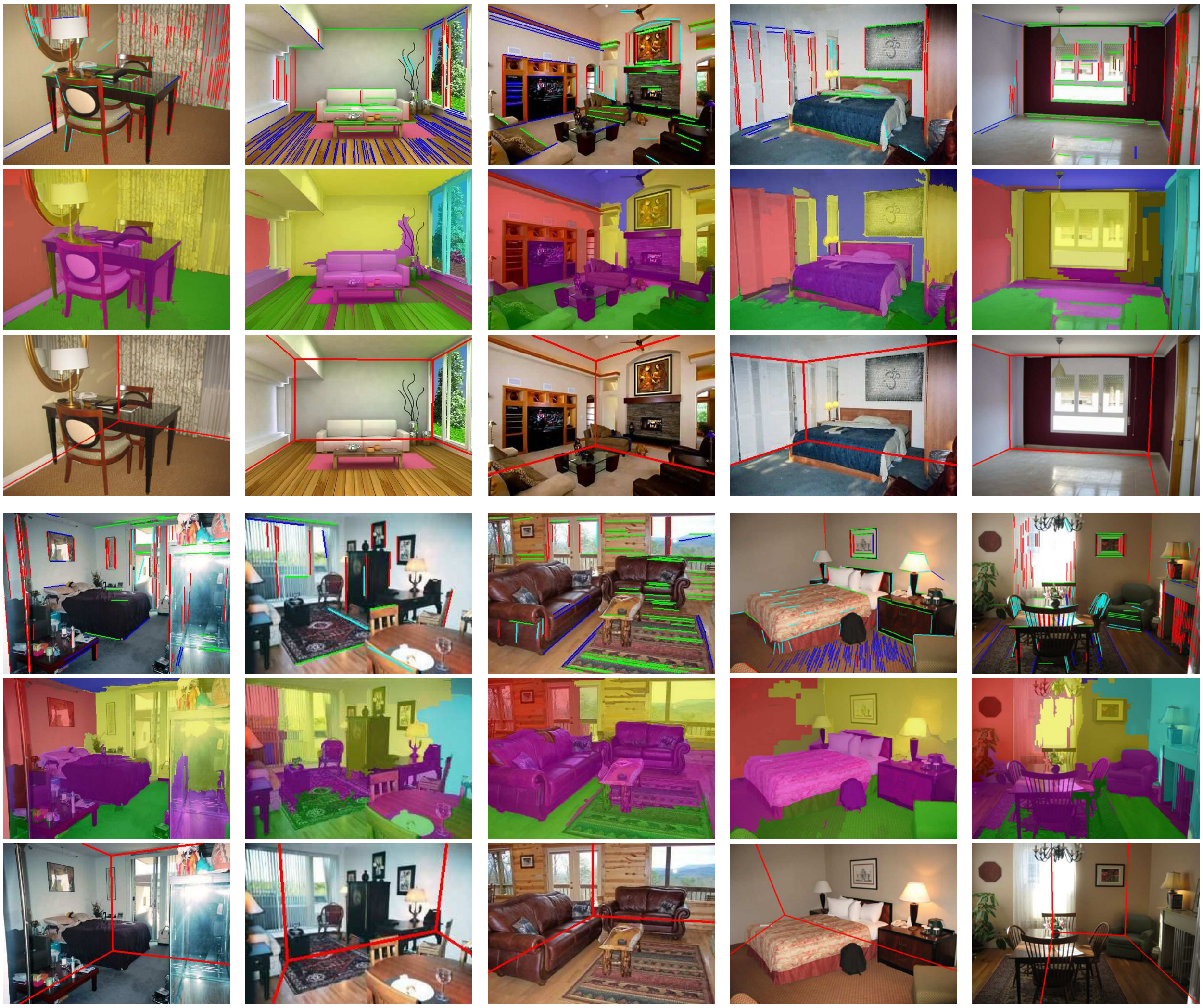


Figure 6. Qualitative test results for our spatial layout estimation. We show an even sampling from least to highest pixel error (left-to-right, top-to-bottom). For each image, original image with detected lines is shown in the top row, the detected surface labels in the middle row, and estimated box layout in the bottom row. Lines corresponding to the three vanishing points are shown with red, green and blue color and the outliers are shown in cyan. Each surface label is shown in different color (floor=green; left wall=red; middle wall=yellow; right wall=cyan; ceiling=blue; objects=pink) and the saturation of color is varied according to the confidence of that surface label. The box layout is shown with red lines. Notice that due to the accuracy of estimated vanishing points, most of the images have nearly all correct line-memberships. The estimates of box rotation suffer if large number lines features are not aligned in room directions (6th row, 4th column and 6th row, 5th column). The line membership features are not effective if the available line support in any particular direction is small (6th row, 2nd column). Note that these are the cases which account for highest pixel errors. **Best viewed in color.**

last row). Considering this, we believe that the improvement achieved in Tables 1, 2, 3 can make a significant difference when the layout is used to help understand the scene.

5.3. Free space estimation

We demonstrate the application of our estimated spatial layouts towards recovering the free space inside a room. As shown in Fig. 8, the surface labels provide object confidences at each pixel in the image which can be thresholded to localize object pixels. We need to find the 3D location of these object pixels to recover free space. The location of the vanishing points provides camera calibration and a 3D reconstruction of the box up to a scale [7, 20]. To obtain an absolute estimate of free space, we assume that the camera is 4.5 feet above the floor (at about chest height), giving a complete projection matrix; alternative possible sources of information include the tendency of beds, tables and other objects to be at fairly regular heights [8].

Using this projection matrix, we determine the *visual hull* corresponding to object pixels. Assuming that the objects are supported by floor and are cuboid shaped, the footprint provides us with a *vertical hull* for the object. To esti-

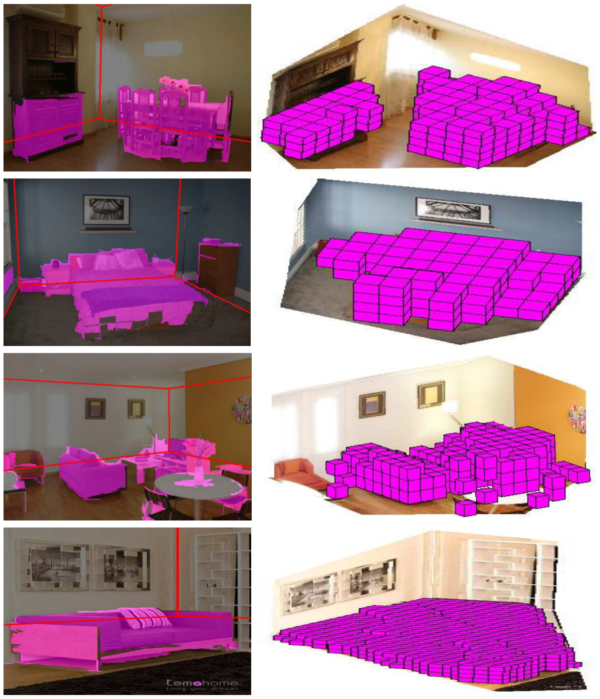


Figure 8. Qualitative results for free space estimation. For each row image in first column shows original image with object support,(obtained from our surface labels) shown in pink and the estimated box layout overlaid on the top in red. The second column shows the result of our free space estimation. Occupied voxels are shown in pink in the rendered 3D room. Note that free space estimation accuracy depends on the accuracy of box layouts and surface labels. First and second row show the result with accurate estimated box layout and surface labels, third row with bad box layout, and the fourth row with bad surface labels. Notice that small errors in box layout can result in considerable errors in 3D space. **Best viewed in color.**

mate this footprint, we project object pixels to floor face of the estimated box layout. The intersection of visual hull and vertical hull provides us with the occupied portion of the room. We estimate this occupied portion as voxels with 0.5 feet resolution Fig. 8 shows estimated free space for some examples with the occupied voxels shown in pink.

6. Conclusion

We have proposed an approach to recover spatial layout of cluttered indoor scenes by modeling them jointly in terms of a 3D box layout and surface labels of pixels. Our experiments show that the integrated reasoning of box layout and object clutter allows better estimation of each. Our spatial layout estimates are robust to clutter and can be used to estimate free space and more precisely reason about objects.

Acknowledgements. This work was supported in part by the Office of Naval Research under N00014-01-1-0890 as part of the MURI program and Beckman Fellowship for Derek Hoiem. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the Office of Naval Research.

References

[1] O. Barinova, V. Konushin, A. Yakubenko, K. Lee, H. Lim, and A. Konushin. Fast automatic single-view 3-d reconstruction of urban scenes. In *proc. ECCV*, pages II: 100–113, 2008.

[2] N. Cornelis, B. Leibe, K. Cornelis, and L. V. Gool. 3d urban scene modeling integrating recognition and reconstruction. *IJCV*, 78(2-3):121–141, July 2008.

[3] J. Coughlan and A. Yuille. Manhattan world: Compass direction from a single image by bayesian inference. In *proc. ICCV*, 1999.

[4] E. Delage, H. Lee, and A. Y. Ng. A dynamic Bayesian network model for autonomous 3D reconstruction from a single indoor image. In *proc. CVPR*, 2006.

[5] P. Denis, J. Elder, and F. Estrada. Efficient edge-based methods for estimating manhattan frames in urban imagery. In *proc. ECCV*, 2008.

[6] F. Han and S. Zhu. Bayesian reconstruction of 3D shapes and scenes from a single image. In *proc. HLK03 in 3D Modeling and Motion Anal.*, 2003.

[7] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2004.

[8] D. Hoiem, A. A. Efros, and M. Hebert. Putting objects in perspective. In *CVPR*, 2006.

[9] D. Hoiem, A. A. Efros, and M. Hebert. Recovering surface layout from an image. *IJCV*, 75(1), 2007.

[10] D. Hoiem, A. A. Efros, and M. Hebert. Closing the loop on scene interpretation. In *proc. CVPR*, June 2008.

[11] Y. Horry, K. Anjyo, and K. Arai. Tour into the picture: using a spidery mesh interface to make animation from a single image. In *ACM SIGGRAPH*, 1997.

[12] J. Kosecka and W. Zhang. Video compass. In *proc. ECCV*, 2002.

[13] D. Lee, M. Hebert, and T. Kanade. Geometric reasoning for single image structure recovery. In *proc. CVPR*, June 2009.

[14] X. Liu, O. Veksler, and J. Samarabandu. Graph cut with ordering constraints on labels and its applications. In *proc. CVPR*, 2008.

[15] D. Lowe and T. Binford. The recovery of three-dimensional structure from image curves. *PAMI*, 7(3), May 1985.

[16] K. Murphy, A. Torralba, and W. T. Freeman. Graphical model for recognizing scenes and objects. In *proc. NIPS*. 2003.

[17] V. Nedovic, A. W. M. Smeulders, A. Redert, and J. M. Geusebroek. Depth information by stage classification. In *proc. ICCV*, 2007.

[18] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42(3), 2001.

[19] P. Olivieri, M. Gatti, M. Straforini, and V. Torre. A method for the 3d reconstruction of indoor scenes from monocular images. In *proc. ECCV*, 1992.

[20] C. Rother. A new approach to vanishing point detection in architectural environments. *IVC*, 20, 2002.

[21] B. Russell, A. Torralba, K. Murphy, and W. Freeman. Labelme: A database and web-based tool for image annotation. *IJCV*, 77, 2008.

[22] A. Saxena, M. Sun, and A. Y. Ng. Make3d: Learning 3-d scene structure from a single still image. In *PAMI*, 2008.

[23] T. Shakunaga. 3-d corridor scene modeling from a single view under natural lighting conditions. *PAMI*, 14, 1992.

[24] E. Sudderth, A. Torralba, W. T. Freeman, and A. Wilsky. Depth from familiar objects: A hierarchical model for 3D scenes. In *proc. CVPR*, 2006.

[25] I. Tsochantaridis, T. Joachims, T. Hofmann, Y. Altun, and Y. Singer. Large margin methods for structured and interdependent output variables. *JMLR*, 2005.

[26] S. Yu, H. Zhang, and J. Malik. Inferring spatial layout from a single image via depth-ordered grouping. In *CVPR Workshop*, 2008.