This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the version available on IEEE Xplore.

DeLay: Robust Spatial Layout Estimation for Cluttered Indoor Scenes

Saumitro Dasgupta, Kuan Fang*, Kevin Chen*, Silvio Savarese
Stanford University
{sd, kuanfang, kchen92}@cs.stanford.edu, ssilvio@stanford.edu

Abstract

We consider the problem of estimating the spatial layout of an indoor scene from a monocular RGB image, modeled as the projection of a 3D cuboid. Existing solutions to this problem often rely strongly on hand-engineered features and vanishing point detection, which are prone to failure in the presence of clutter. In this paper, we present a method that uses a fully convolutional neural network (FCNN) in conjunction with a novel optimization framework for generating layout estimates. We demonstrate that our method is robust in the presence of clutter and handles a wide range of highly challenging scenes. We evaluate our method on two standard benchmarks and show that it achieves state of the art results, outperforming previous methods by a wide margin.

1. Introduction

Consider the task of estimating the spatial layout of a cluttered indoor scene (say, a messy classroom). Our goal is to delineate the boundaries of the walls, ground, and ceiling, as depicted in Fig. 1. These bounding surfaces are an important source of information. For instance, objects in the scene usually rest on the ground plane. Many objects, like furniture, are also usually aligned with the walls. As a consequence, these support surfaces are valuable for a wide range of tasks such as indoor navigation, object detection, and augmented reality. However, inferring the layout, particularly in the presence of a large amount of clutter, is a challenging task. Indoor scenes have a high degree of intra-class variance, and critical information required for inferring the layout, such as room corners, is often occluded and must be inferred indirectly.

There are works which approach the same problem given either depth information (e.g. an RGBD frame) or a sequence of monocular images from which depth can be inferred. For our work, we restrict the input to the most general case: a single RGB image. Given this image, our

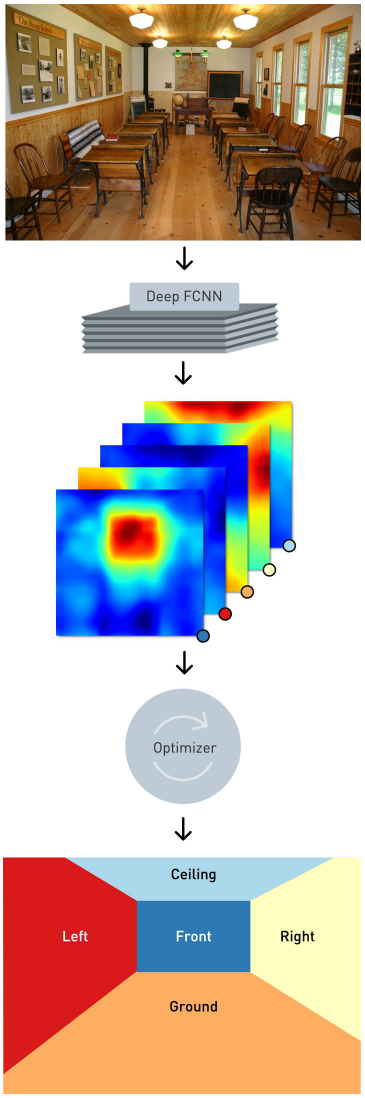


Figure 1: An overview of our layout estimation pipeline. Each heat map corresponds to one of the five layout labels shown in the final output. They are color coded correspondingly.

*indicates equal contribution.

framework outputs the following: i) a dense per-pixel labeling of the input image (as shown in Fig. 1), and ii) a set of corners that allows the layout to be approximated as the projection of a box. The classes in the dense labeling are drawn from the following set: {Left Wall, Front Wall, Right Wall, Ceiling, Ground}. The parameterization of the scene as a box is described in further detail in Sec. 3.2.

Prior approaches to this problem usually follow a two-stage process. First, a series of layout hypotheses are generated. Next, these are ranked to arrive at the final layout. The first stage is usually accomplished by detecting three orthogonal vanishing points in the scene, often guided by low-level features such as edges. For instance, the influential work by Hedau *et al*. [6] generates layout candidates by inferring vanishing points and then ranking them using a structured SVM. Unfortunately, this first stage is highly susceptible to clutter and often fails to produce a sufficiently accurate hypothesis. While subsequent works have proposed improvements to the second stage of this process (i.e., ranking the layouts), they are undermined by the fragility of the candidate generation.

Our method is motivated by the recent advances in semantic segmentation using fully convolutional neural networks [11, 2, 23], since one can consider layout estimation to be a special case of this problem. That said, constraints that are unique to layout estimation prevent a direct application of the existing general purpose semantic segmentation methods. For instance, the three potential wall classes do not possess any characteristic appearance. Multiple sub-objects may be contained within their boundaries, so color-consistency assumptions made by CRF methods are not valid. Furthermore, there is an inherent ambiguity with the semantic layout labels (described in further detail in Sec. 3.4.5). This is in contrast to traditional semantic segmentation problems where the labels are uniquely defined.

Our contributions are as follows:

- We demonstrate that fully convolutional neural networks can be effectively trained for generating a belief map over our layout semantic classes.
- The FCNN output alone is insufficient as it does not enforce geometric constraints and priors. We present a framework that uses the FCNN output to produce geometrically consistent results by optimizing over the space of plausible layouts.

Our approach is robust even when faced with a high degree of clutter. We demonstrate state of the art results on two datasets.

2. Related Work

The problem, as stated in Sec. 1, was introduced by Hedau *et al*. in [6]. Their method first estimates three orthogonal vanishing points by clustering line segments in the scene. These are then used for generating candidate box layouts that are ranked using a structured regressor. Unlike our approach, this method requires the clutter to be explicitly modeled. Earlier work in this area by Stella *et al*. [19] approached this problem by grouping edges into lines and quadrilaterals and finally depth ordered planes.

In [20], Wang *et al*. model cluttered scenes using latent variables, eliminating the need for labeled clutter. Extending upon this work in [17], Schwing *et al*. improve the efficiency of learning and inference by demonstrating the decomposition of higher-order potentials. In a subsequent work [16], Schwing *et al*. propose a branch and bound based method for jointly inferring both the layout and the objects present the scene. While they demonstrate that their method is guaranteed to retrieve the global optimum of the joint problem, their approach is not robust to occlusions.

Pero *et al*., in [14] and [13], investigate generative approaches for solving layout estimation. Inference is performed using Markov Chain Monte Carlo sampling. By incorporating the geometry of the objects in the scene, their method achieves competitive performance.

A number of works consider a restricted or special variant of this problem. For instance, Liu *et al*. [10] generate the room layout given the floor plan. Similarly, [7] assumes that multiple images of the scene are available, allowing them to recover structure from motion. This 3D information is then incorporated into an MRF-based inference framework. In [1], Chao *et al*. restrict themselves to scenes containing people. They use the people detected in the scene to reason about support surfaces and combine it with the vanishing point based approach of [6] to arrive at the room layout.

More recently, Mallya and Lazebnik [12] used FCNN for the task, similar to ours. However, while we use an FCNN for directly predicting per-pixel semantic labels, their method uses it solely for generating an intermediate feature they refer to as "informative edges". These informative edges are then integrated into a more conventional pipeline, where layout hypotheses are generated and ranked using a method similar to the one used in [6]. Their results do not improve significantly upon those achieved by Schwing *et al*.

3. Method

3.1. Overview

Given an RGB image $\mathbf{I}$ with dimensions $w \times h$, our framework produces two outputs:

1. $\mathbf{L}$, a $w \times h$ single channel image that maps each pixel in the input image, $\mathbf{I}_{ij}$, to a label in the output image $\mathbf{L}_{ij} \in \{\text{Left, Front, Right, Ceiling, Ground}\}$.

2. The box layout parameters, as described in Sec. 3.2.

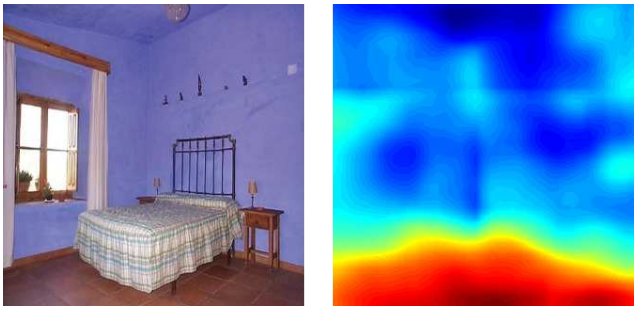


Figure 2: Given a $w \times h \times 3$ input RGB image (shown on the left), our neural network outputs a $w \times h \times 5$ belief map, where each of the 5 slices can be interpreted as a classification map for a specific label. For instance, the slice shown on the right corresponds to the ground plane label.

The pipeline is described broadly in Fig. 1. The estimation of $\mathbf{L}$ begins by feeding $\mathbf{I}$ into the fully convolutional neural network described in Sec. 3.3. The normalized output of this network is a $w \times h \times 5$ multidimensional array $\mathbf{T}$ which can be interpreted as:

$$\mathbf{T}^{(k)} = \Pr\left(\mathbf{L}_{ij} = k \mid \mathbf{I}\right) \qquad \forall k \in \{1, ..., 5\} \qquad (1)$$

where $\mathbf{T}^{(k)}$ is the k^{th} channel of the multidimensional array $\mathbf{T}$. This "belief map" is then used as the input to our optimization framework which searches for the maximum likelihood layout estimate that fits our box parameterization.

3.2. Modeling the Layout

Most works in this area [6, 17, 20], including ours, assume that the room conforms to the so-called "Manhattan world assumption" [3] that is based on the observation that man-made constructs tend to be composed of orthogonal and parallel planes. This naturally leads to representing indoor scenes by cuboids. The layout of an indoor scene in an image is then the projection of a cuboid.

Hedau [6] and Wang [20] describe how such a cuboid can be modeled using rays emanating from mutually orthogonal vanishing points. The projection of such a cuboid can be modeled using four rays and a vanishing point [20], as described in Fig. 3. Our parameterization of this model is $\tau = (l_1, l_2, l_3, l_4, v)$, where l_i is the equation of the i^{th} line and v is the vanishing point.

Given τ, we can partition an image into polygonal regions as follows:

- The intersections of the lines l_i give us four vertices p_i. The polygon described by these four vertices corresponds to one of the walls.
- The intersections of the rays starting at v passing through p_i with the bounds of the image give us four more vertices, e_i. We can now describe four additional polygons defined by $(p_i, e_i, e_{i+1}, p_{i+1})$ (where the index additions are modulo 4). These correspond to two additional walls, the ceiling, and the ground.

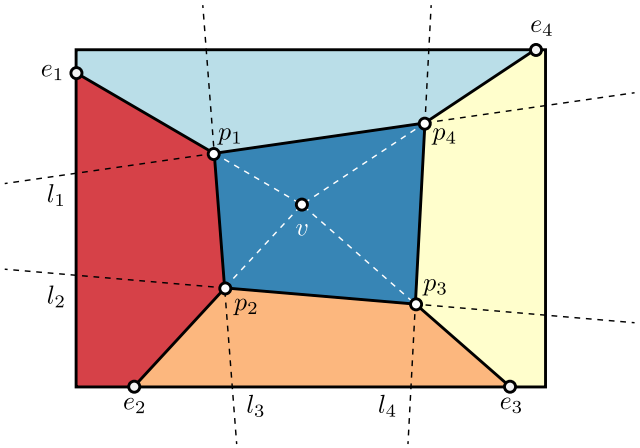


Figure 3: The layout parametrized using four lines, (l_1, l_2, l_3, l_4), and a vanishing point, v, as described in Sec. 3.2.

The vertices p_i and e_i may lie outside the bounds of the image, in which case the corresponding polygons are either clipped or absent entirely. We also define a deterministic labeling for these polygons. The top and bottom polygon are free from ambiguity and always labeled as "ceiling" and "ground". The polygons corresponding to the walls are labeled left to right as (left, front, right). If only two walls are visible, they are always labeled (left, right). If only a single wall is visible, it is labeled as "front".

3.3. Belief Map Estimation via FCNN

Deep convolutional neural networks (CNN) have achieved state-of-the-art performance for various vision tasks like image classification and object detection. Recently, they have been adapted for the task of semantic segmentation with great success. Nearly all top methods on the PASCAL VOC segmentation challenge [4] are now based on CNNs. The hierarchical and convolutional nature of these networks is particularly well suited for layout segmentation. Global context cues and low-level features learned from the data can be fused in a pipeline that is trained from end-to-end.

Our CNN uses the architecture proposed by Chen *et al.* in [2], which is a variant commonly referred to as a fully convolutional neural network (FCNN). Most common CNN architectures used for image classification, such as AlexNet [9] and its variants, incorporate fully-connected terminal layers that accept fixed-sized inputs and produce

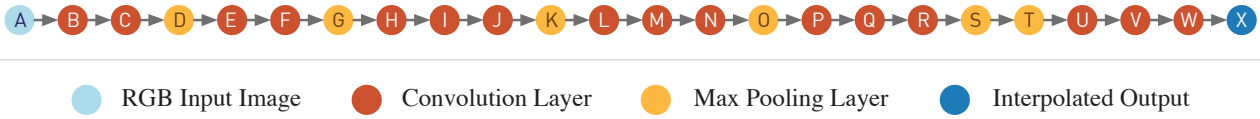


Figure 4: The fully convolutional network architecture used for our layout estimation. It is based on the "LargeFOV" variant described in [2], which in turn is based on the VGG16 architecture proposed by Simonyan and Zisserman in [18] .
Each convolution layer depicted in this figure is followed by a rectified linear unit (ReLU), excluding the final one (layer W). During training, dropout regularization is applied to layers U and V.

non-spatial outputs. In [11], Long *et al*. observe that these fully-connected layers can be viewed as convolutions with kernels that cover their entire input regions. Thus, they can be re-cast into convolutional layers which perform sliding-window style dense predictions. This conversion also removes the fixed-size constraint previously imposed by the fully connected layers, thereby allowing the new networks to operate on images of arbitrary dimensions.

One caveat here is that the initial network produces dense classification maps at a lower resolution than the original image. For instance, the network proposed by Long *et al*. produces an output that is subsampled by a factor of 32. They compensate for this by learning an upsampling filter, implemented as a deconvolutional layer in the network. In contrast, our network produces an output that is subsampled by only a factor of 8. As a result, simple bilinear interpolation can be used to efficiently upsample the classification map.

We finetune a model pretrained on the PASCAL VOC 2012 dataset. The weights for the 21-way PASCAL VOC classifier layer (corresponding to layer W shown in Fig. 4) are discarded and replaced with a randomly initialized 5-way classifier layer. Since our belief map before interpolation is subsampled by a factor of 8, the ground truth labels are similarly subsampled. The loss function is then formulated as the sum of cross entropy terms for each spatial position in this subsampled output. The network is trained using stochastic gradient descent with momentum for 8000 iterations. Chen *et al*. describe an efficient method for performing convolution with "holes" [2] - a technique adopted from the wavelet community. We use their implementation of this algorithm within the Caffe framework [8] to train our network.

3.4. Refinement

3.4.1 The Problem

Given the CNN output $\mathbf{T}$, a straightforward way to obtain a labeling is to simply pick the label with the highest score for each pixel:

$$\hat{\mathbf{L}}_{ij} = \arg\max_k \mathbf{T}_{ij}^{(k)} \quad \forall i \in [1, ..., w], j \in [1, ..., h] \quad (2)$$

However, note that there are no guarantees that this layout will be consistent with the model described in Sec. 3.2. Indeed, the wall/ground/ceiling intersections in $\hat{\mathbf{L}}$ are always "wavy" curves (rather than straight lines), and often contain multiple disjoint connected components per label. This is because our CNN does not enforce any smoothness or geometric constraints. A common solution used in general semantic segmentation is to refine the output using a CRF. These usually use the CNN output as the unary potentials and define a pairwise potential over color intensities [2]. However, we found these CRF based methods to perform poorly in the presence of clutter, where they tend to segment along the clutter boundaries that occlude the true wall/ground/ceiling intersection.

3.4.2 Overview of our Approach

Given the neural network output $\mathbf{T}$, we want to obtain the refined box layout, τ^*, and the corresponding label-map, $\mathbf{L}^*$. Let f be a function that maps a layout (parametrized as described in Sec. 3.2) to a label-map. Then, we have:

$$\mathbf{L}^* = f(\tau^*) \quad (3)$$

For any given layout, we define the following scoring metric:

$$S\left(\mathbf{L} = f(\tau) \mid \mathbf{T}\right) = \frac{1}{wh} \sum_{i,j} \mathbf{T}_{ij}^{(L_{ij})} \quad (4)$$

We now pose the refinement process as the following optimization problem:

$$\tau^* = \arg\max_{\tau} S\left(f(\tau) \mid \mathbf{T}\right) \quad (5)$$

This involves ten degrees of freedom: two for each of the four lines, and two for the vanishing point. While searching over the entire space of layouts is intractable, we can initialize the search very close to the solution using $\hat{\mathbf{L}}$. Furthermore, we can use geometric priors to aggressively prune the search space.

3.4.3 Preprocessing

We use $\hat{\mathbf{L}}$ (as defined in Eq. 2) for initialization and determining which planes are not visible in the scene. An issue

here is that $\hat{\mathbf{L}}$ may contain spurious regions. In particular, it often includes spurious front wall regions (not surprisingly, given the ambiguity described in Sec. 3.4.5). Furthermore, there may be multiple disjoint components per label.

We address the multiple disjoint components by pruning all but the largest connected component for each label. The presence of potentially spurious regions is addressed by considering two candidates in parallel: one with these regions pruned, and another with them preserved. The candidate with the highest score is selected at the end of the optimization. In practice, we found that it is sufficient to restrict this pruning to just the front wall. The "holes" in labeling created by pruning are filled in using a k-nearest neighbor classifier.

3.4.4 Initialization

Given a preprocessed $\hat{\mathbf{L}}$, we can produce an initial estimate of the four lines l_i in τ by detecting the wall/ceiling/ground boundaries. We do this by considering each relevant pair of labels (say, ground and front wall) and treating it as a binary classification problem. The corresponding line is then obtained using logistic regression.

3.4.5 Optimization

Algorithm 1: Layout Optimization

Input: $\mathbf{T}$ `// The output of our CNN`
(l_1, l_2, l_3, l_4) `// Initialization`
Output: Layout $\tau^* = (l_1, l_2, l_3, l_4, v)$
repeat
  foreach *Candidate vanishing point* p **do**
    evaluate $S\left(\tau = (l_1, l_2, l_3, l_4, p) \mid \mathbf{T}\right)$
    if *Score Improved* **then**
      $v := p$
    end
  end
  foreach $i \in (1...4)$ **do**
    foreach *Candidate line* l **do**
      evaluate $S\left(\tau = (l_1, ...l..., l_4, v) \mid \mathbf{T}\right)$
      if *Score Improved* **then**
        $l_i := l$
      end
    end
  end
until *Score did not improve*

Given an input image, $\mathbf{I}$, so far we have described how to obtain:

- A "belief map" $\mathbf{T}$ using our neural network
- A scoring function S that can be used for comparing layouts
- An initial layout estimate τ_0

To obtain the final layout, τ^*, we use an iterative refinement process. Our optimization algorithm, described in algorithm 1, is reminiscent of coordinate ascent. It greedily optimizes each parameter in τ sequentially, and repeats until no further improvements in the score can be obtained.

Sampling vanishing points: We start with the vanishing point, $v \in \tau$, since our initialization only provides us estimates for the four lines. While we could have used $\hat{\mathbf{L}}$ to provide an initial estimate for v, we found that directly using grid search works better in practice. The feasible region for v is the polygon described by the vertices (p_1, p_2, p_3, p_4) as shown in Fig. 3. We evenly sample a grid within this region to generate candidates for v. For each candidate vanishing point, we compute the score using S and update our parameters if the score improves.

Sampling lines: Next, each line, $l_i \in \tau$, is sequentially optimized. The search space for each line is the local neighborhood around the current estimate. Let (x_1, y_1) and (x_2, y_2) be the intersections of l_i with the image bounds. We evenly sample two sets of points centered about (x_1, y_1) and (x_2, y_2) along the image boundary. Our search space for l_i is then the cartesian product of these two sets.

Handling label ambiguity: There is an inherent ambiguity in the semantic layout labels as demonstrated in Fig. 5. Indeed, our network often has trouble emitting a consistent label when faced with such a scenario. In such cases, the probability is split between the labels "front" and either "left" or "right" as the case may be. Our existing scoring function does not take this issue into account. Therefore, for this special case, we formulate a modified scoring function as follows:

$$\hat{S} = \max\left(S(\mathbf{L}'), S(\mathbf{L})\right) \tag{6}$$

where $\mathbf{L}'$ is the labeling obtained by replacing all occurrences of the label "front" with either "left" or "right". This allows our optimization algorithm to commit to a label without being unfairly penalized. Note that this modified scoring function is only used when our optimizer is considering a "two-wall" layout scenario as determined during initialization. For a single-wall or three-wall case, the ambiguity issue does not apply.

4. Experimental Evaluation

4.1. Dataset

We train our network on the Large-scale Scene Understanding Challenge (LSUN) room layout dataset [21], a diverse collection of indoor scenes with layouts that can be approximated as boxes. It consists of 4000 training, 394

Figure 5: There is an inherent ambiguity in semantically labeling layouts. Two plausible labelings are shown above. A human may reasonably label the wall behind the bed as "front", whereas a labeling that enforces consistency based on left-to-right ordering may classify it as "right".

validation, and 1000 testing images. While these images have a wide range of resolutions, we rescale them anisotropically to 321×321 pixels using bicubic interpolation. The ground truth images are relabeled to be consistent with the ordering described in Sec 3.2.

We also test on the dataset published by Hedau *et al*. [6]. This consists of 209 training and 104 testing images. We do not use the training images.

4.2. Accuracy

We evaluate our performance by measuring two standard metrics:

1. The pixelwise accuracy between the layout and the ground truth, averaged across all images.

2. The corner error. This is the error in the position of the visible vertices p_i and e_i (as shown in Fig. 3), normalized by the image diagonal and averaged across all images.

We use the LSUN room layout challenge toolkit scripts to evaluate these. The toolkit addresses the labeling ambiguity problem by treating it as a bipartite matching problem, solved using the Hungarian algorithm, that maximizes the consistency of the estimated labels with the ground truth.

Our performance on both datasets are summarized in Tables 1 and 2. Our approach outperforms all prior methods and achieves state-of-the-art results.

4.3. Efficiency

The CNN used in our implementation can process 8 frames per second on an Nvidia Titan X. For optimization, we use a step size of 4 pixels for sampling lines and a grid of 200 vanishing points. With these parameters, the optimization procedure takes approximately 30 seconds per frame. The current single-threaded implementation is not tuned for performance, and significant improvements should be achievable (for instance, by parallelizing the sampling loops and utilizing SIMD operations).

Method	**Pixel Error**
Hedau *et al*. (2009) [6]	21.20
Del Pero *et al*. (2012) [14]	16.30
Gupta *et al*. (2010) [5]	16.20
Zhao *et al*. (2013) [22]	14.50
Ramalingam *et al*. (2013) [15]	13.34
Mallya *et al*. (2015) [12]	12.83
Schwing *et al*. (2012) [17]	12.8
Del Pero *et al*. (2013) [13]	12.7
DeLay	**9.73**

Table 1: Performance on the Hedau [6] dataset

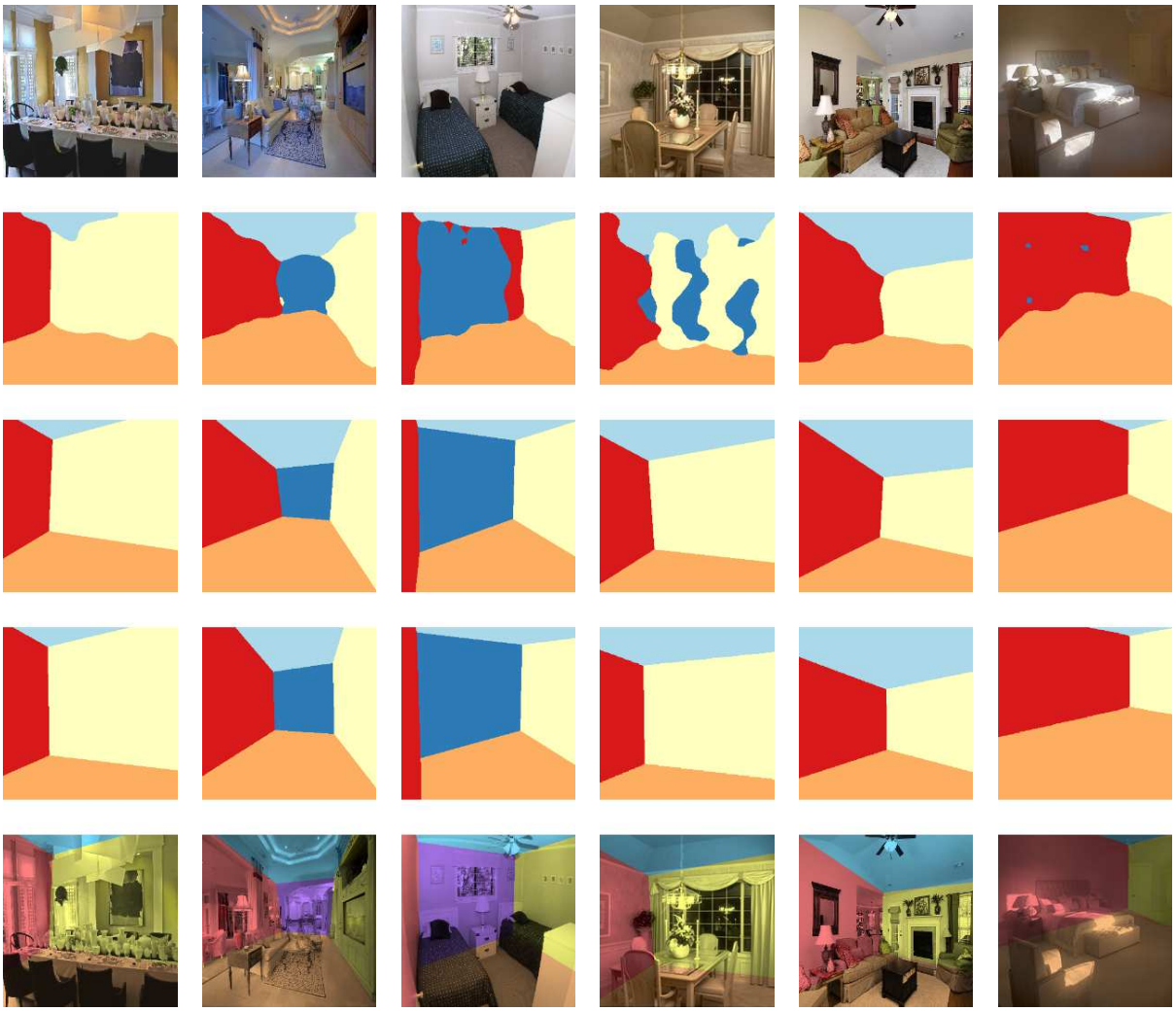
Method	**Corner Error**	**Pixel Error**
Hedau *et al*. (2009) [6]	15.48	24.23
Mallya *et al*. (2015) [12]	11.02	16.71
DeLay	**8.20**	**10.63**

Table 2: Performance on the LSUN [21] dataset

5. Qualitative Analysis

We analyze the qualitative performance of our layout estimator on a collection of scenes sampled from the LSUN validation set. We split our analysis into two broad themes: i) scenarios where our estimator performs well and demonstrates unique strengths of our approach, and ii) scenarios that demonstrate potential weaknesses of our framework and provide insight into future avenues for improvement.

Fig. 6 shows a collection of scenes where our estimator produces layouts that closely match the human-annotated ground truths. Fig. 6a shows the robustness of our estimator to a high degree of clutter. The ground plane's intersections with the walls are completely occluded by the table. The decorative fixture near the ceiling not only occludes the top corner but also includes multiple strong edges that can be easily confused for wall/ceiling intersections. Despite these challenges, our framework produces a highly accurate estimate. Fig. 6f shows a scene with illumination variation and nearly uniform wall, ground, and ceiling colors with almost no discernible edges at their intersections. Such scenes are



(a) (b) (c) (d) (e) (f)

Figure 6: Our results on the LSUN validation set. The first row shows the input images. The second row depicts $\hat{\mathbf{L}}$, the most probable label per pixel before optimization. The third row comprises our final layout estimate $\mathbf{L}^*$. The fourth row is the ground truth, while the fifth is our estimate superimposed on the input image. A detailed analysis of these images is provided in Sec. 5.

particularly challenging for methods that rely on low-level features and color consistency. However, our method is able to recover the layout almost perfectly. Fig. 6e shows an example where the Manhattan world assumption is violated. Our estimate, however, degrades gracefully. Arguably, it is no less valid than the provided ground truth image, which also attempts to fit a boxy layout to the non-conforming scene.

Fig. 6d and 6c show the effectiveness of our optimization procedure, and demonstrate that simply trusting the CNN output is insufficient. Directly using the estimate $\hat{\mathbf{L}}$ produces garbled results with inconsistencies like multiple disjoint components for a single label and oddly shaped boundaries. However, our optimizer is able to successfully recover the layout. It also shows the labeling ambiguity issue we described in Sec. 3.4.5. Observe that the CNN's confidence is split over the front and right wall classes in the ambiguous region. However, our modified scoring function is able to successfully handle this case.

In Fig. 7, we explore some of the scenarios where our estimator fails to produce results that agree with the human annotations. Fig. 7a is an interesting case where our estimate predicts a left wall that is absent from the ground truth. A closer observation of the image reveals that there is indeed a left wall present (this scene also violates the Manhattan assumption). Fig. 7c shows a scenario where our CNN

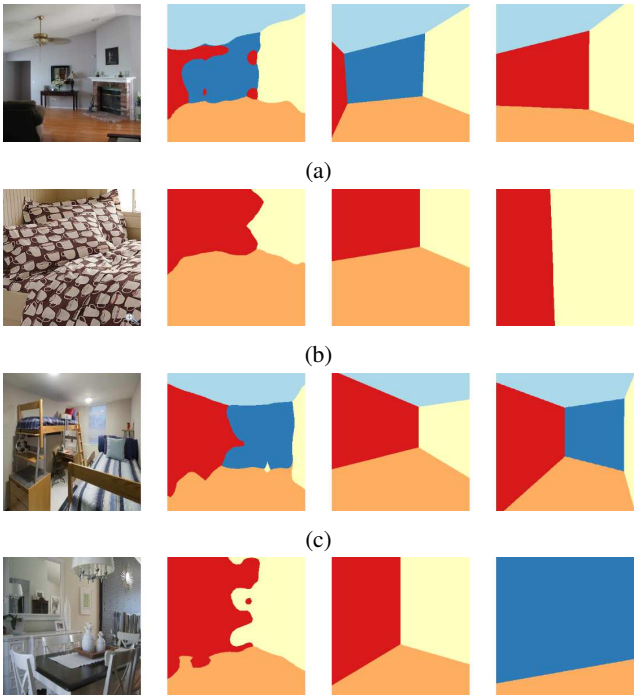


Figure 7: A set of challenging cases where our estimator's results do not match the human-annotated ground truths. The first column is the input image, the second is $\hat{\mathbf{L}}$, the third is our final layout estimate $\mathbf{L}^*$, while the fourth is the ground truth.

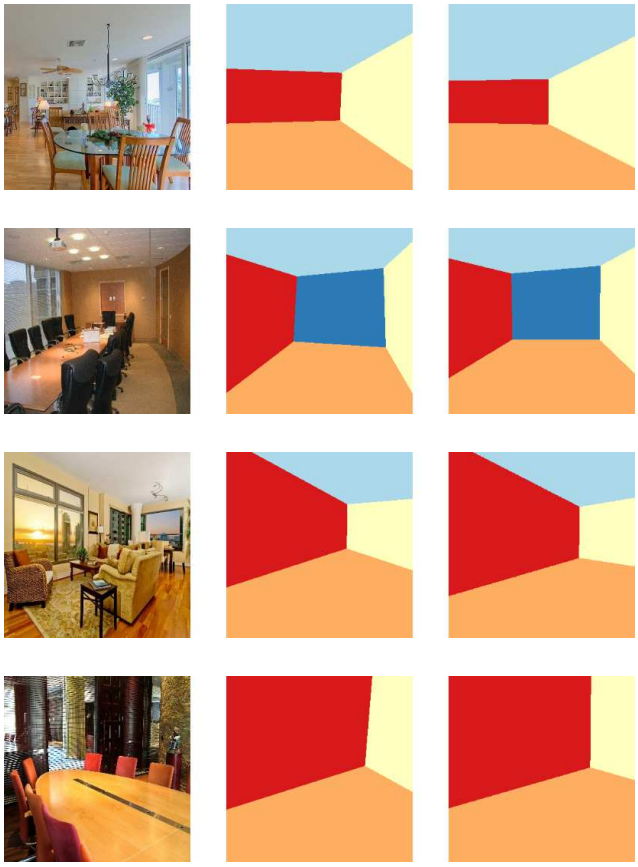


Figure 8: Further examples that demonstrate our estimator's performance on the LSUN validation set. Column 2 is our estimate, while column 3 is the ground truth.

produces a reasonably accurate output, but our optimizer incorrectly prunes the front wall as a spurious region. Occasionally, we encounter cases where the CNN predictions differ so drastically from the ground truth that the optimizer fails to provide any improvements. Such a case is shown in Fig. 7d where the presence of a strong color change in the wall causes our network to consider it as a separate wall. In Fig. 7b, we demonstrate a "semantic failure". The lower half of the scene is dominated by a bed. It is geometrically consistent with the notion of a ground plane, but not semantically.

Observing the results above, a few patterns emerge that lend themselves to future improvements. For instance, the CNN output suggests that the Manhattan world assumption is not strictly necessary. The issue with the bed in Fig. 7b illustrates the importance of incorporating broader semantics into the room layout estimation problem. A promising approach here would be to train a CNN for performing joint segmentation of both layout and object classes.

6. Conclusion

In this paper, we presented a framework for estimating layouts for indoor scenes from a single monocular image. We demonstrated that a fully convolutional neural network can be adapted to estimate layout labels directly from RGB images. However, as our results show, this output alone is insufficient as the neural network does not enforce geometric consistency. To address this issue, we presented a novel optimization framework that refines the neural network output to produce valid layouts. Our method is robust to clutter and works on a wide range of challenging scenes, achieving state-of-the-art results on two leading room layout datasets and outperforming prior methods by a large margin.

Acknowledgment. This research was supported by MURI grant WF911NF-15-1-0479 and Toyota Center grant 122282.

References

[1] Y.-W. Chao, W. Choi, C. Pantofaru, and S. Savarese. Layout estimation of highly cluttered indoor scenes using geometric

and semantic cues. In *Image Analysis and Processing–ICIAP 2013*, pages 489–499. Springer, 2013. 2

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *ICLR*, 2015. 2, 3, 4

[3] J. M. Coughlan and A. L. Yuille. The manhattan world assumption: Regularities in scene statistics which enable bayesian inference. In *NIPS*, pages 845–851, 2000. 3

[4] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 3

[5] A. Gupta, M. Hebert, T. Kanade, and D. M. Blei. Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. In *Advances in Neural Information Processing Systems*, pages 1288–1296, 2010. 6

[6] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In *Computer vision, 2009 IEEE 12th international conference on*, pages 1849–1856. IEEE, 2009. 2, 3, 6

[7] M. Hödlmoser and B. Micusik. Surface layout estimation using multiple segmentation methods and 3d reasoning. In *Pattern Recognition and Image Analysis*, pages 41–49. Springer, 2013. 2

[8] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 4

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 3

[10] C. Liu, A. G. Schwing, K. Kundu, R. Urtasun, and S. Fidler. Rent3d: Floor-plan priors for monocular layout estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3413–3421, 2015. 2

[11] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. *arXiv preprint arXiv:1411.4038*, 2014. 2, 4

[12] A. Mallya and S. Lazebnik. Learning informative edge maps for indoor scene layout prediction. In *International Conference on Computer Vision (ICCV)*, volume 1, 2015. 2, 6

[13] L. Pero, J. Bowdish, B. Kermgard, E. Hartley, and K. Barnard. Understanding bayesian rooms using composite 3d object models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 153–160, 2013. 2, 6

[14] L. D. Pero, J. Bowdish, D. Fried, B. Kermgard, E. Hartley, and K. Barnard. Bayesian geometric modeling of indoor scenes. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2719–2726. IEEE, 2012. 2, 6

[15] S. Ramalingam, J. K. Pillai, A. Jain, and Y. Taguchi. Manhattan junction catalogue for spatial reasoning of indoor scenes. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3065–3072. IEEE, 2013. 6

[16] A. G. Schwing, S. Fidler, M. Pollefeys, and R. Urtasun. Box in the box: Joint 3d layout and object reasoning from single images. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 353–360. IEEE, 2013. 2

[17] A. G. Schwing, T. Hazan, M. Pollefeys, and R. Urtasun. Efficient structured prediction for 3d indoor scene understanding. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2815–2822. IEEE, 2012. 2, 3, 6

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

[19] X. Y. Stella, H. Zhang, and J. Malik. Inferring spatial layout from a single image via depth-ordered grouping. 2008. 2

[20] H. Wang, S. Gould, and D. Koller. Discriminative learning with latent variables for cluttered indoor scene understanding. *Communications of the ACM*, 56(4):92–99, 2013. 2, 3

[21] Y. Zhang, F. Yu, S. Song, P. Xu, A. Seff, and J. Xiao. Large-scale scene understanding challenge: Room layout estimation. 5, 6

[22] Y. Zhao and S.-C. Zhu. Scene parsing by integrating function, geometry and appearance models. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3119–3126. IEEE, 2013. 6

[23] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr. Conditional random fields as recurrent neural networks. *CoRR*, abs/1502.03240, 2015. 2