Support surfaces prediction for indoor scene understanding

Anonymous ICCV submission

Paper ID 1506

Abstract

In this paper, we present an approach to predict the extent and height of supporting surfaces such as tables, chairs, and cabinet tops from a single RGBD image. We define support surfaces to be horizontal, planar surfaces that can physically support objects and humans. Given a RGBD image, our goal is to localize the height and full extent of such surfaces in 3D space. To achieve this, we created a labeling tool and annotated 1449 images with rich, complete 3D scene models in NYU dataset. We extract ground truth from the annotated dataset and developed a pipeline for predicting floor space, walls, the height and full extent of support surfaces. Finally we match the predicted extent with annotated scenes in training scenes and transfer the the support surface configuration from training scenes. We evaluate the proposed approach in our dataset and demonstrate its effectiveness in understanding scenes in 3D space.

1. Introduction

Knowledge of support surfaces is crucial to understand or interact with a scene. People walk on the floor, sit in chairs, eat on tables, and move objects around on desks. Our goal is to infer the heights and extents of support surfaces in the scene from a single RGBD image. The main challenge is that support surfaces have complex multi-layer structures and that much of the scene is hidden from view (Fig. 1). Often, surfaces such as tables, are littered with objects which limits the effectiveness of simple plane fitting strategies. Some support surfaces are not visible at all because they are above eye level or obstructed by other objects.

Our approach is to label visible portions of the scene, project these labels into an overhead view, and refine estimates and infer occluded portions based on scene priors and context. See Figure 4 for an overview. One barrier to our study was lack of a dataset that has complete 3D annotations. We undertook an extensive effort to create full 3D models that correspond to scenes in the NYU (v2) dataset [14], which we expect will be of interest to many other researchers. Our annotations complement the existing detailed 2D object and support relation annotations and can be used for experiments on inferring free space, support surfaces, and 3D object layout.

Figure 1: **Challenge.** Our goal is to predict the height and extent of all support surfaces, including occluded portions, from one RGBD image. As shown on the left, these surfaces are often covered with objects and are nearly invisible when they occur near eye-level. As shown on the right, we must perform inference over large portions of the scene that are blocked from view.

Another challenge is in how to represent support surfaces, which are often scattered and multilayered. For example, a shelf may be stacked on a computer desk that rests on the floor, with a chair pushed under the desk. When initially viewing a scene, we do not know how many surfaces there are or at which heights. Our solution is to infer a set of overhead support maps that indicate the extent of supports at various heights on the floor plan.

We need to specify the heights and extents of support surfaces, which is difficult due to clutter and occlusion. Our approach is analogous to that of Guo and Hoiem [3] who first label an RGB image according to the visible surfaces at each pixel and then infer occluded background labels. We label visible pixels into "floor", "wall", and "object" using the RGBD region classifier from Silberman et al. [14] and then project these pixels into an overhead view using the depth signal. Before projection, the scene is rotated so that walls and floor are axis-aligned, using the code from [14]. We then predict which heights are likely to contain a support surface based on the normals of visible surfaces and

straight lines in the image. One height could contain multiple surfaces such as table and counter tops or the seats of several chairs. For each height, we then predict the extent of support on the floor map using a variety of 2D and 3D features. These estimates are refined using an autocontext [16] approach and shape priors incorporated by matching whole surfaces from the training set, similarly to [3].

Our experiments investigate the accuracy of our estimates of support height and extent, the effects of occlusion due to foreground objects or surfaces above eye-level, and the effectiveness of our contextual inference and shape matching. Our method substantially outperforms a baseline of plane-fitting, but there is still much room for improvement by incorporating object recognition or more structured models of relations among surfaces.

1.1. Related work

Interpreting indoor scenes has recently become an active topic of research. One line of work is to predict boxy layouts from RGB images [6, 7, 10, 8, 5, 12, 11], in which occlusion and lack of 3D sensors is combatted with simple scene models and strong priors. By operating on RGB-D images that provide both color and depth signals, we aim to explore more detailed and complex scene representations. However, the depth signal does not trivialize the problem, since many support surfaces are obscured by clutter or completely hidden.

Our approach directly builds on recent efforts by Silberman et al. [14] and Guo and Hoiem [3]. Silberman et al. created the NYU v2 dataset and proposed algorithms to orient the scene, find major surfaces, segment the image into objects, label regions into "prop", "furniture", "structure", and "floor", and estimate which objects support which others. Our approach incorporates their algorithms for estimating 3D scene orientation and labeling visible pixels into geometric classes. We differ in that we predict the full 3D extent of support surfaces, and we extend their dataset with full 3D annotations of objects using a tool that enables users to markup scenes based on both image data and point clouds from the depth sensor. Guo and Hoiem [3] propose an approach to label both visible and occluded portions of a 2D image. We adopt their basic pipeline of labeling visible portions of the scene and inferring occluded portions based on autocontext and shape priors. However, our approach is applied to full 3D scenes, which requires new representations, features, and matching strategies.

Other works reason about support relations between objects and other surfaces. Gupta et al. infer support in outdoor scenes from RGB images to aid in geometric labeling [4], and Silberman et al. [14] infer support relations and types (e.g., supported from the side or from below) for pairs of regions. Our work differs in its aim to infer full 3D extent of support surfaces. Jiang et al. [9] propose an algorithm that trains a robot to place objects based on 3D point clouds. Our work on estimating underlying support surfaces would facilitate object placement and allow more complex interactions. Taylor and Cowley [15] estimate the layout of walls from an RGBD image based on plane fitting and inference in an overhead view but do not address the problem of support surfaces. Finally, our work has some relation to efforts in 3D scene modeling (e.g., [2, 17, 13]). We differ in that we are provided with only one viewpoint, and our goal is to recover extent of underlying support surfaces rather than a full model of visible objects and structures.

1.2. Contributions

This paper offers two main contributions: (1) detailed 3D annotations for the 1449 images of the NYU v2 Kinect dataset that can be used to study 3D reconstruction, free space estimation, and support inference; and (2) a model and approach to recover the extent of support surfaces, which is a fundamental but largely unexplored problem in computer vision.

2. Creating 3D Annotations for Indoor Scenes

We extend the NYU Depth Dataset V2, which contains 1449 images of roughly 500 distinct scenes, with complete, detailed 3D models (Fig. 2). Our annotation tool (Fig. 3) enables users to model scenes based on both image data and point clouds from the depth sensor while viewing the scene from multiple angles. Our 3D annotations, combined with the existing detailed 2D object annotations, will be useful for exploring a wide variety of scene understanding tasks.

2.1. 3D Scene Model

We categorize scene elements into three classes:

- **Layout structures** include floors, walls and ceilings. Because walls are always perpendicular to the floor, they are modeled as line segments in the overhead view and polygons in the horizontal view. Similarly, ceiling and floors are line segments in the horizontal view and polygons in the overhead view. We also model openings such as doorways and windows on the walls as polygons on the wall planes.

- **Furniture** objects are common in indoor scenes and tend to have complicated 3D appearance. For example, chairs and sofas cannot be modeled using cubic blocks, and their support surfaces are often not the top part. To accurately model them, we use Google SketchUp models from Google 3D Warehouse repository. We manually select 30 models from these models to model 6 categories of furniture that are most common: chair, table, desk, bed, bookshelf and sofa. The support surface of each object is labeled by hand on the 3D model.

Figure 2: **3D annotation examples.** Our annotations can be used to study estimation of 3D layout of support surfaces (as in this paper), objects, or occupied space.

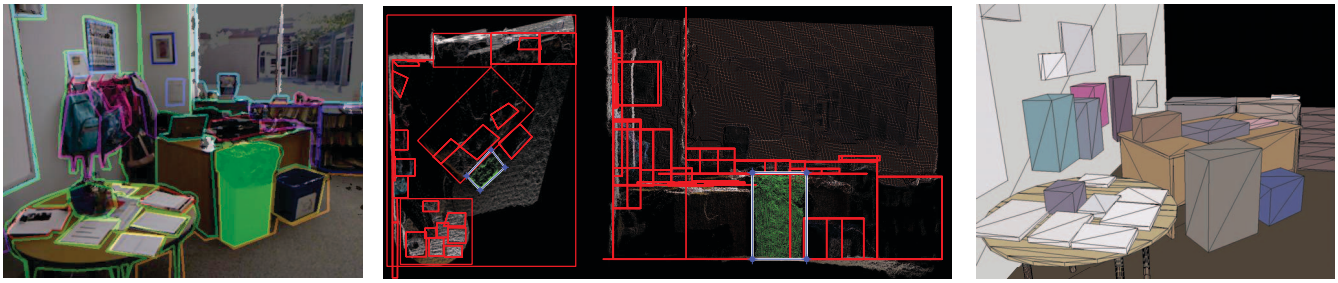


Figure 3: **Annotation tool.** Users annotate the 3D scene based on views of the RGB image (left), an overhead view (left-center), and a horizontal view (right-center), and with the help of 2D object segmentations and automatic initial guesses of the object footprint. The 3D model is shown on the right.

- **3D extruded models** are used to describe all other objects. Most clutter objects are small and do not have regular shape, and we simply model them as vertically extruded polygons.

2.2. Preprocessing

The RGBD scenes are first preprocessed to facilitate annotation as well as support surface inference. Because the preprocessing is automatic, we use the same procedure at the start of our inference pipeline. We start with the cropped RGBD scene, which corresponds to the area where information from both sensors are available. Then the surface normals are computed at each pixel, by fitting local planar surfaces in its neighborhood. These local planar surfaces take into account color information as well, in a procedure similar to bilateral filtering, which improves robustness of plane fits compared to using only depth information.

Next, we compute the dominant room orientation by iteratively aligning the surface normals to the x, y or z axes. Initially, surface normals of each point is assigned to the nearest axis. Then we fix the assignment and compute optimal rotation matrix R of the room is computed using SVD, based on all points that are aligned within a given threshold. The point cloud is then rotated using R and we repeat from the alignment step again until the rotation matrix does not change any more.

2.3. Annotation procedure

If the floor is visible, our system estimates the floor height from the depth information and object labels (from [14]). If necessary, the annotator can correct or specify the floor height by clicking on a scene point and indicating its height above the floor.

The annotator then alternates between labeling in an overhead view (from above the scene looking down at the floor) and horizontal view (from the camera looking at the most frontal plane) to model the scene:

1. The annotator is asked to click to select one region from the 2D annotated image plane by clicking.

2. In the overhead view, the annotator is shown highlighted 3D points and an estimated bounding box that correspond to the object. The annotator can fit a polygon to the footprint of the object.

3. The horizontal view is then shown, and the annotator specifies the vertical height of the object by drawing a line segment at the object's height.

4. Additionally, the annotator can supply more details of the object, such as adding openings to the wall or placing a detailed SketchUp model for furniture. The user can choose the SketchUp model and orientation with a single keystroke.

A major advantage of our annotation tool is that the annotation process can guided by depth values in RGBD image, improving ease and accuracy of layout recovery and object modeling. Furthermore, the tool uses existing 2D object label in the image plane (available in the NYU dataset) so that the 3D models are consistent with the 2D annotation. The tool also has a number of handy features that help users annotate quickly including (1) **a snapping feature** that snaps the polygon vertices to neighboring support surfaces or corners, and/or aligns annotations with the room axes; 2) an initial guess of the extent by fitting **default bounding boxes** to depth points included in the region. Therefore the users can often trust the default model configuration and have to edit only when the system's estimate is poor.

Our annotation tool is implemented in Matlab, and the annotation is also available in Protocol Buffer[1], an exchangeable data format. On average, it takes about 5 minutes to model a scene with more than 10 objects, depending on the complexity of the scene. We recruited four student annotators with little or no previous Matlab experience and obtained thousands of very high quality annotations shown in Figure 2.

3. Support surface prediction

In this section, we present an approach (Fig. 4) to predict the vertical height and horizontal extent of support surfaces in a scene. Scene parsing in an overhead view is very different than in the image plane. One challenge is that many locations are blocked from view and do not have observed evidence. On the other hand, since the room directions have been rectified, objects tend to have rectangular shapes in an overhead view. We design features that apply to the overhead view and use spatial context to improve parsing results.

3.1. Preprocessing

We first label pixels in the image plane using Silberman et al.'s pipeline [14] into four geometric categories: "floor", "ceiling", "wall" and "foreground". We then project the labeled pixels into the overhead view using the associated depth signal.

[1] http://code.google.com/p/protobuf/

We also construct a voxel map representation of the scene. We first voxelize the scene into grid and then project onto the camera plane. The projected depths are then compared to the depth values of the observed scene. All voxels having smaller depth values are observed free space, and the voxels larger depth values than the scene are not direct observed. The rest of the voxels are outside of the view scope and treated as "don't care".

3.2. Features

We use the following features, illustrated in Fig. 5.

1. **Observed 3D points with upward normals** are indicative of support surfaces at that height.

2. **Observed geometric labels** are recorded as the mean and max likelihood in each cell, after projecting to an overhead view.

3. **Edgemap** is the detected straight line segments projected onto the overhead view. Horizontal straight lines often occur at the edges of support surfaces.

4. **Voxel occupancy** is the amount of observed free space voxels near the near the surface height. If a voxel is observed to be free space, it cannot be a part of a support surface.

5. **Volumetric difference** is the difference of number of free space voxels above the height at this location subtracted by the number of the number of free space voxels below it. Support surfaces often have more air above them than below.

6. **Location prior** is the spatial prior of where support surfaces are in the training scenes, normalized by the scale of the scene.

7. **Viewpoint prior** is the spatial prior of support surfaces in training scenes with respect to the viewer.

8. **Support height prior** is the spatial distribution of the vertical height of support planes.

3.3. Predicting support height

We first predict the support heights. Our intuition is that at the height of a support plane, we are likely to see: (1) observed surfaces with upward normals; (2) a difference in the voxel occupancy above and below the plane; and (3) observed 3D points near the height. Also, we use an estimated prior, since some heights are more likely than others. We sum over the features (upward-facing points, volumetric difference, all 3d points) at the height of the proposed plane and estimate an empirical log-likelihood for each, which is used along with the prior probability in a linear SVM, which

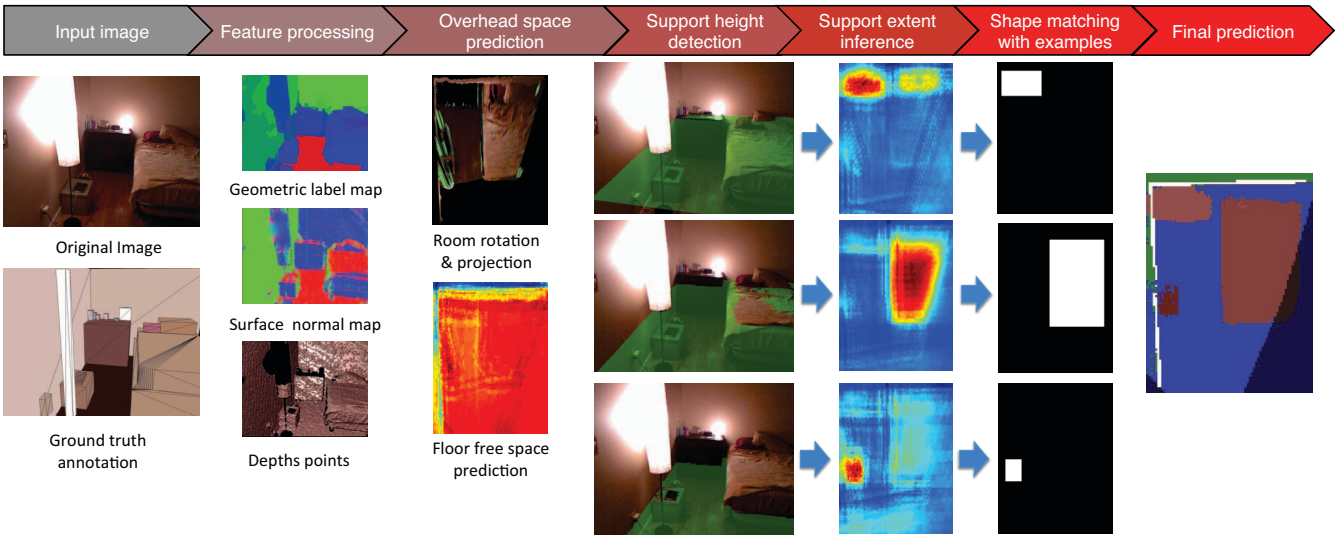


Figure 4: **Approach.** The input is an aligned RGBD image. We first compute features based on the depth image and estimated labels of image plane pixels into "floor", "ceiling", "wall" and "foreground". These features are projected into an overhead view and used to estimate the locations of walls (or floor free space). Next, a large number of horizontal planes are proposed and classified as "support" or "non-support". The horizontal extent of a supporting surface is then estimated for each support plane based on the features at each point and surrounding predictions. Template matching is performed with the footprints of training objects, which provides a more regularized estimate of support extent. In the rightmost image, green pixels are estimated walls, blue pixels are floor, and red pixels are support surfaces, with lighter colors corresponding to greater height. The darkly shaded portion corresponds to parts of the overhead map that are outside the camera's viewing angle. White lines correspond to wall estimates based on simple plane fitting.

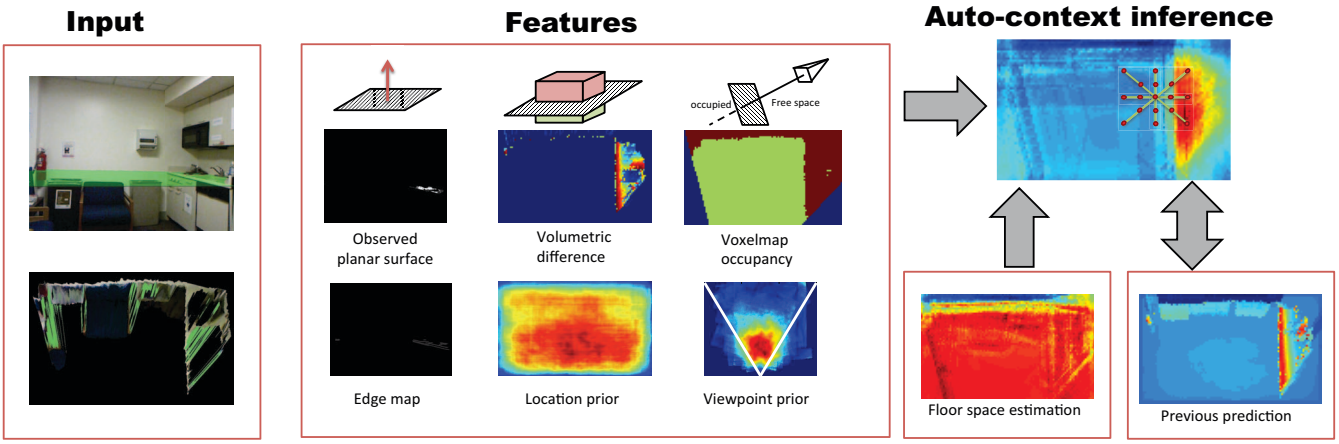


Figure 5: **Overhead scene parsing.** The feature set we used in our support surface prediction: observed up-pointing points, 3D geometric labels and edgemaps are computed in the image plane and then projected. Volumetric difference, occupancy and location/view prior are directly computed on the overhead grid. Auto context inference is applied to each predict support height.

classifies into "support" or "non-support". After classification, we perform non-maximum suppression to remove planes with very similar heights.

3.4. Estimating support extent

To estimate support extent, we divide the overhead view into grid cells. For each cell of the floor plane, all features are aggregated over a 3D vertical scene column and classified as "floor" or "wall" using a linear SVM. Likewise, for

each support plane, we classify the grid cells in the overhead view in that height into being part of support surface or not. We use the follow set of features: (1) observed point pointing up; (2) volumetric difference at each grid cell (3 levels); (3) projected surface map near the predicted support height; (4) view dependent and independent spatial priors in the overhead view; and (5) floor space prediction.

3.5. Integrating spatial context

The support surfaces have spatial contexts. When the the object is partly occluded or partly out of scope, we do not have direct appearance cues and need to rely on spatial contexts to fill in the missing part. Guo and Hoiem used auto-context procedure to overcome occlusion in 2D image space. We used the same strategy in the overhead parsing.

Autocontext process iteratively aggregates the features from the neighboring regions in a large template and applies a discriminative classifier to predict labels for each grid cell. Then the algorithm repeat by adding the output of previous prediction into the feature set to repredict until the prediction does not change anymore. We first compute feature at each overhead cell on the overhead view as in the previous subsection and apply autocontext. Figure. 5 visualizes this iterative procedure.

3.6. Shape prior transfer using template matching

Although spatial contexts helps to fill in the occluded part in the overhead view, it does not fully exploit the shape prior such as objects often being rectangular. We further harvest shape prior by matching the support extent probability map with the support surface layout template from training scenes.

The matched template should have similar shape, scale and spatial contexts with the probability map. We assign a positive score to locations that are support surfaces on both the prediction and the template; we assign a negative score to penalize where the template overlap with the free space in the probability map. We do not assign any score to location that are out of the view scope.

The template matching uses convolution to compute matching score and can be done efficiently using Fourier transform. The translation vector can be found by doing the inverse Fourier transform. We used the matched template as shape prior if it is the top k matches and that the matching score is above a threshold. Otherwise, we do not transfer any shape prior. For scenes with complicated support surface layout, it is often the case that there is not a single perfect match, but there exist multiple good partial matches. In such cases, we remove the probability from the current matched part and rematch the remaining of the probability map with the rest of the templates. And we repeat this to the point when there are no good matching anymore.

Compare to holistic scene matching strategies such as in case of Satkin et al. [11], our matching scheme is more flexible because we allow translation and partial matches. Our matching is also independent of the viewpoint since our scenes automatically aligned, further improving the chance of finding good matches.

4. Experiments

To verify the effectiveness of our propose method, we evaluate our support surface prediction by comparing to the ground truth support surfaces extracted from our annotated dataset.

4.1. Generating ground truth

We predefined categories of objects that are "capable of support", such as table, sofa, shelves, cabinet etc. Such definition are mostly obvious. Alternatively it is also possible to sort support relations and discover categories that are more often supporters. Here we just manually defined. Ground truth support surfaces are defined to be the top part of an object from "supporter" category unless they are within the 0.15m to the ceiling.

For prediction, we project the depth points to the overhead view and quantize the projected area into a grid with a spacing of 0.03 m, we perform prediction and evaluation based on this grid. The ground truth floor space is the union of the annotated area union observed area, subtracted by the area that has been occluded by walls from the viewpoint. The area that are out of the scope are marked as "don't care".

The ground truth of support plane is just the top surface of objects, or, in case of the SketchUp model objects, the annotated functional surface (e.g. the seat of a chair). To make evaluation less sensitive to noise in localization, we make the area around boundary of support surface within a thickness of 0.15m to be "don't care". As in floor space, we also do not evaluate the area that is out of the scope.

4.2. Experimental setup

We use the training/testing split from Silberman et al paper, with a training of 795 scenes and 654 scenes. For training support height classifier, we scan the vertical extent of the scene with spacing of 0.01m. The negative examples are the ones are far away from any ground truth heights by a threshold of 0.15m. The positive ones are just the annotated heights. We use a linear classifier to train and predict on support heights. At testing time, we also scan all vertical heights with 0.01m spacing and than apply non-maximum suppression, with a tolerance of 0.15m.

In autocontext, we used a template of sparse template points with a distance of 0, 0.03, 0.12, 0.27, 0.48, 0.75m away from the center pixel distributed in a star shape . This large template helps us to model the long range contextual

interactions. We repeat the autocontext for 3 iterations before it terminates.

To do template matching, we first aggregate the support surface configurations from training scenes, and obtain a total of 1372 templates. For the extent probability map we predict at each support height, we retrieve the top 10 templates with the highest matching scores. If the matching score is below a threshold of 200, we do not use the template to transfer shape prior. If the remaining score of the probability map is above this threshold of 200, we rematch until there is no good matching anymore.

4.3. Qualitative results

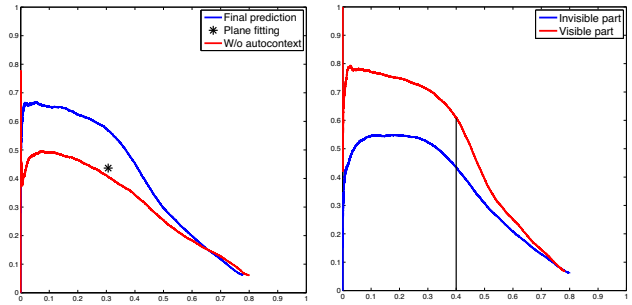
In Figure. 6, we display the visualization of support surface prediction in the overhead view. Although the scenes are cluttered and challenging, we are able to predict most of the support surface extents even if they are severely occluded (kitchen counter in first image on the top left) or is partly out of view (the chair in first top right image). Furthermore, the transferred support planes can give us a rough estimation of individual support objects.

However, there are cases where the support surfaces are hard to find because they are not obvious or not directly observed. For example, in the cabinet tops in the bottom left image are above the camera height. The seat of the chair in the bottom right image is out of the field of view and is hard to detect.

4.4. Quantitative results

We evaluate accuracy of support extent prediction with precision-recall curves. The ground truth consists of a set of known support pixels at some floor position and vertical height and a set of "don't care" pixels at the edges of annotated surfaces or out of the field of view. Our predictions consist of a set of confidence-valued support pixels. Computation of precision-recall is similar to that for object detection, as in the PASCAL VOC challenge [1]. The most confident predicted pixel at the same overhead position and within 0.15m height of ground truth support pixel is labeled positive (or "don't care" if that is the ground truth label). All other pixels are labeled as negative so that duplicate detections are penalized. Because there are many points per image in the dataset, we sample 1000 pixels from each image, so that each image has the same weight. Precision-recall curves are computed based on samples accumulated over the 654 test images.

Fig. 7 shows the results of our quantitative analysis. We compare to a baseline of plane-fitting, using the Silberman et al. [14] code for plane segmentation and selecting approximately horizontal surfaces as support surfaces. The baseline does not have confidences, so it is a single point on the curve. In Fig. 7(a), we see that our method outperforms the baseline by 12% precision at the same recall level



(a) PR curve of all support surfaces (b) PR curve on visible and invisible part

Figure 7: **Precision-Recall curve of support surface extent prediction.**

or 11% recall at the same precision. The gain due to use of autocontext and shape fitting is large. Most of the gain is due to autocontext; the improvement due to shape fitting affects qualitative results more than quantitative. In Fig. 7(b), we compare performance for occluded support surfaces to unoccluded (visible) ones. In qualitative results, we show predictions that have confidence greater than the value corresponding to the 0.4 recall threshold.

5. Conclusions

We propose 3D annotations for the NYU v2 dataset and an algorithm to find support planes and determine their extent in an overhead view. Our quantitative and qualitative results show that our prediction is accurate for nearby and visible support surfaces, but surfaces that are distant or near or above eye-level still present a major challenge. Because many surfaces are occluded, contextual reasoning is necessary to achieve good performance. Methods to improve detection of support planes above eye-level (which are not directly visible) and to recognize objects and use categorical information are two fruitful directions for future work. In addition, our dataset enables researchers to study problems of occupancy (or free space), estimation of navigable paths, 3D reconstruction, and other spatial understanding tasks.

References

[1] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2010 Results. http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html. 7

[2] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Reconstructing building interiors from images. In *ICCV*, pages 80–87, 2009. 2

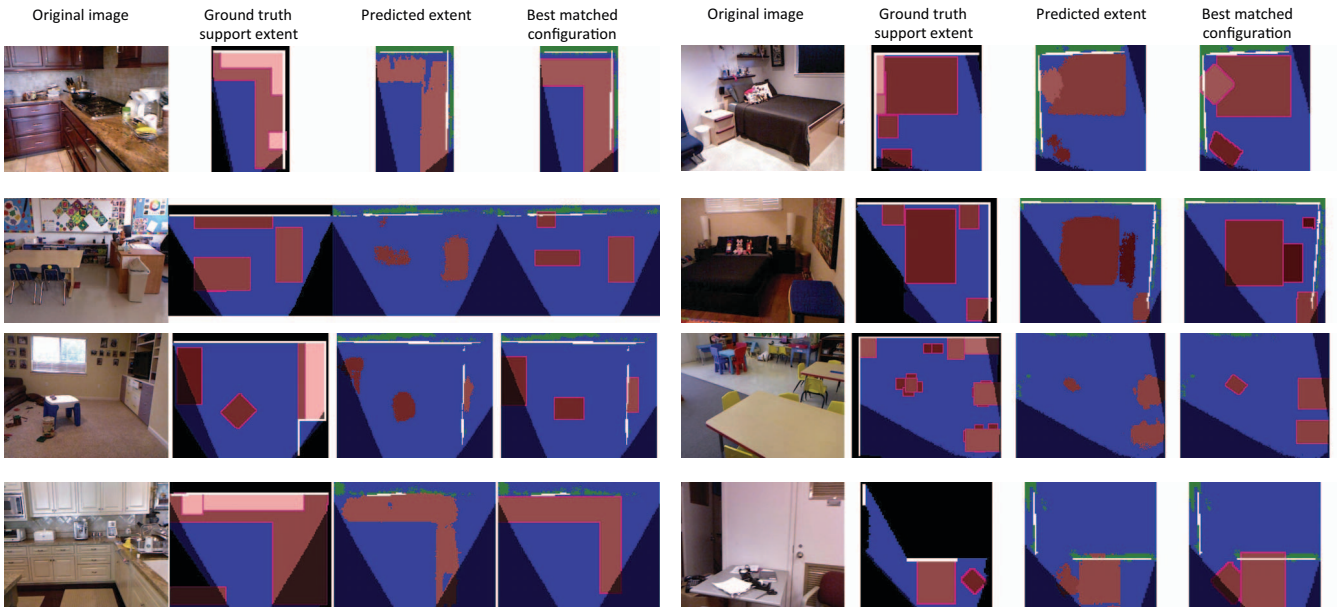


Figure 6: **Overhead visualization.** Green and blue and red areas are estimated walls, floor and support surfaces respectively. The brighter colors of support surfaces indicate higher vertical heights relative to the floor. Dark areas are out of the view scope.

[3] R. Guo and D. Hoiem. Beyond the line of sight: Labeling the underlying surfaces. In *ECCV*, pages 761–774, 2012. 1, 2

[4] A. Gupta, A. A. Efros, and M. Hebert. Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *ECCV*, 2010. 2

[5] A. Gupta, S. Satkin, A. A. Efros, and M. Hebert. From 3d scene geometry to human workspace. In *CVPR*, 2011. 2

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

[7] V. Hedau, D. Hoiem, and D. Forsyth. Thinking inside the box: Using appearance models and context based on room geometry. In *ECCV*, 2010. 2

[8] V. Hedau, D. Hoiem, and D. A. Forsyth. Recovering free space of indoor scenes from a single image. In *CVPR*, pages 2807–2814, 2012. 2

[9] Y. Jiang, M. Lim, C. Zheng, and A. Saxena. Learning to place new objects in a scene. *I. J. Robotic Res.*, 31(9):1021–1043, 2012. 2

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

[11] S. Satkin, J. Lin, and M. Hebert. Data-driven scene understanding from 3D models. In *BMVC*, 2012. 2, 6

[12] A. G. Schwing, T. Hazan, M. Pollefeys, and R. Urtasun. Efficient structured prediction for 3d indoor scene understanding. In *CVPR*, pages 2815–2822, 2012. 2

[13] T. Shao, W. Xu, K. Zhou, J. Wang, D. Li, and B. Guo. An interactive approach to semantic modeling of indoor scenes with an rgbd camera. *ACM Trans. Graph.*, 31(6):136, 2012. 2

[14] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, pages 746–760, 2012. 1, 2, 3, 4, 7

[15] C. J. Taylor and A. Cowley. Parsing indoor scenes using rgb-d imagery. In *Robotics: Science and Systems*, 2012. 2

[16] Z. Tu and X. Bai. Auto-context and its application to high-level vision tasks and 3d brain image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(10):1744–1757, 2010. 2

[17] J. Xiao and Y. Furukawa. Reconstructing the world's museums. In *ECCV (1)*, pages 668–681, 2012. 2