Data-Driven Scene Understanding from 3D Models

Scott Satkin
ssatkin@ri.cmu.edu

Jason Lin
jasonli1@andrew.cmu.edu

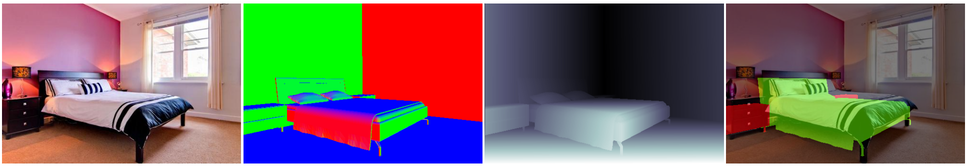
Martial Hebert
hebert@ri.cmu.edu

Carnegie Mellon University
The Robotics Institute
Pittsburgh, Pennsylvania

http://cmu.satkin.com/bmvc2012/

Abstract

In this paper, we propose a data-driven approach to leverage repositories of 3D models for scene understanding. Our ability to relate what we see in an image to a large collection of 3D models allows us to transfer information from these models, creating a rich understanding of the scene. We develop a framework for auto-calibrating a camera, rendering 3D models from the viewpoint an image was taken, and computing a similarity measure between each 3D model and an input image. We demonstrate this data-driven approach in the context of geometry estimation and show the ability to find the identities and poses of object in a scene. Additionally, we present a new dataset with annotated scene geometry. This data allows us to measure the performance of our algorithm in 3D, rather than in the image plane.



(a) Input image (b) Predicted surface normals (c) Predicted depth map (d) Predicted object labels

Figure 1: From a single image, we estimate detailed scene geometry and object labels.

1 Introduction

Over the past decade, researchers have demonstrated the effectiveness of data-driven approaches for complex computer vision tasks. Large datasets such as [32]'s 80 Million Tiny Images and [7]'s ImageNet have proven to be invaluable sources of information for tasks like scene recognition and object classification. Pioneering work such as [14, 15, 31] has shown the power of matching an image with similar ones from a large dataset for "transferring" information from one image to another.

Recently, large online repositories of 3D data such as Google 3D Warehouse [3] have emerged. These resources, as well as the advent of low-cost depth cameras [1], have sparked interest in geometric data-driven algorithms. At the same time, researchers have (re-)started investigating the feasibility of recovering geometric information, *e.g.*, the layout of a scene [5, 20, 30]. The success of data-driven techniques for tasks based on appearance features, *e.g.*,

http://dx.doi.org/10.5244/C.26.128

interpreting an input image by retrieving similar scenes [14, 25, 27, 32], suggests that similar techniques based on *geometric* data could be equally effective for 3D scene interpretation tasks. In fact, the motivation for data-driven techniques is the same for 3D models as for images: real-world environments are not random; the sizes, shapes, orientations, locations and co-location of objects are constrained in complicated ways that can be represented given enough data. In principle, estimating 3D scene structure from data would help constrain bottom-up vision processes. For example, in Figure 1, one nightstand is fully visible; however, the second nightstand is almost fully occluded. Although a bottom-up detector would likely fail to identify the second nightstand since only a few pixels are visible, our method of finding the best matching 3D model is able to detect these types of occluded objects. This is not a trivial extension of the image-based techniques. Generalizing data-driven ideas raises new fundamental technical questions never addressed before in this context: What features should be used to compare input images and 3D models? Given these features, what mechanism should be used to rank the most similar 3D models to the input scene? Even assuming that this ranking is correct, how can we transfer information from the 3D models to the input image? To address these questions, we develop a set of features that can be used to compare an input image with a 3D model and design a mechanism for finding the best matching 3D scene using support vector ranking. We show the feasibility of these techniques for transferring the geometry of objects in indoor scenes from 3D models to an input image.

The graphics community has begun harvesting data from online repositories such as Google 3D Warehouse in an effort to better understand and model how objects are typically arranged in homes [8, 9]. At the same time, the vision community has begun using 3D Warehouse data to learn about the geometric properties and affordances of objects [10]. There has also been work using this data for 3D to 3D matching with laser scans to aid in classification [22]. However, our work is one of the first to combine this geometric prior with image features in a framework capable of producing detailed 3D models from an image. Of course, this is not entirely new, the idea of relating 3D models to 2D projections was a foundation of earlier vision approaches [6, 11, 26]. The major difference here is our use of vast repositories of 3D data, which require novel vision and learning approaches.

This work is an important first step towards 3D data-driven techniques, which will contribute to addressing two major problems in image understanding. First, as most geometric scene understanding systems rely implicitly on sifting through a collection of hypotheses (iterative refinement [21], sampling [28], explicit search [12], structured prediction [16, 24]), matching and ranking mechanisms such as the ones we propose provide a data-driven way to *generate multiple hypotheses* which can be used as seeds for further processing. Second, 3D data offers potentially richer information for transfer. For example, one could predict not only object type and location, but also viewpoint and even occlusions from other objects.

While the problem of indoor geometry estimation has received considerable attention, it remains challenging with limited performance. Building upon the earlier work in single-view geometry estimation [20, 30], many researchers have tackled the problem of estimating the locations of the walls and floors from cluttered indoor images [16, 23, 35]. More recently, attention has been focused on estimating the locations of objects in these environments [17, 24, 28, 33], and on estimating a scene's free space [13]. New work [18, 29] aims to recover free space by localizing cuboids representing object categories and sizes using parametric models as their prior. In contrast, we recover more detailed object geometries and we use non-parametric priors that can capture complex interactions between objects.

This brief summary of previous work shows how vibrant this research area is and how much progress has been made in a short time. The data-driven techniques that we propose here should not be viewed as a substitute to any of the above approaches. Perhaps the most exciting aspect of our approach is that it can be used to augment *any* of these scene interpretation approaches: upstream, by providing a data-driven way to generate hypotheses; and downstream by providing richer mechanisms for information transfer. We show this by building upon the work of [16] and by demonstrating how prior 3D models can be integrated with this existing approach for room layout estimation to help discover the identity, locations and orientations of objects from a single image.

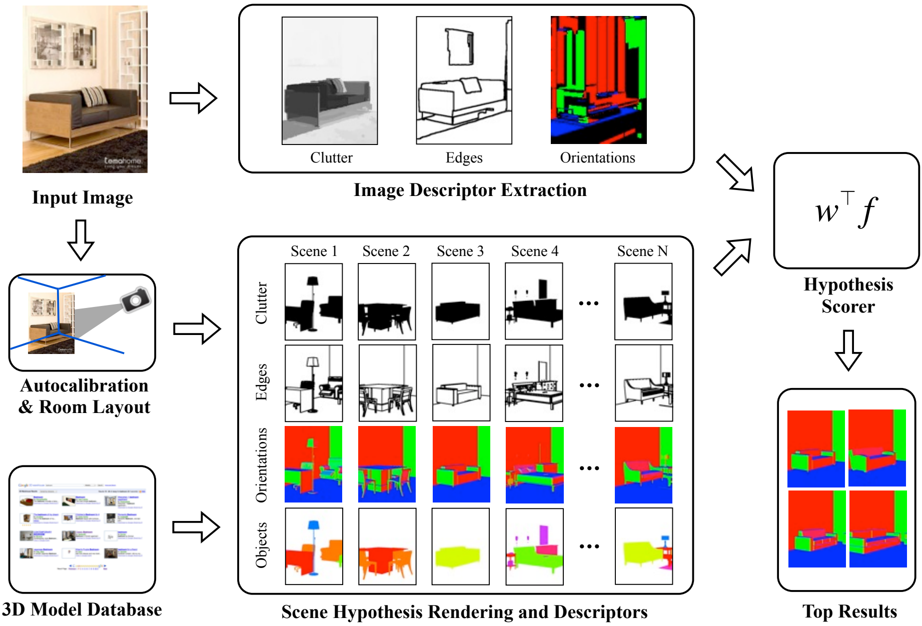


Figure 2: Overview of our approach for matching a 3D model with a monocular image.

2 Scene Understanding via 3D Model Matching

We now describe our framework for comparing 3D models to monocular images. Our ability to relate what we see in an image to a large collection of 3D models allows us to transfer the information from these models, creating a rich understanding of the scene. Naturally, we cannot compare 3D models directly to a 2D image. Thus, we first estimate the intrinsic and extrinsic parameters of the camera and use this information to render each of the 3D models from the same view as the image was taken from. We then compute similarity features between the models and the input image. Lastly, each of the 3D models is ranked based on how similar its rendering is to the input image using a learned feature weighting. See Figure 2 for an overview of this process.

2.1 Auto-calibration and Rendering

Given an image, we begin by auto-calibrating the camera by estimating vanishing points using the approach of [23]. The vanishing points are also used to estimate the orientation of the camera with respect to the three Manhattan-world axes in our scene. We run the pre-trained room layout estimation algorithm of [16] to determine the locations of the walls and the floor in the image. Lastly, relative to a fixed camera height, we recover the size of the room and focal length of the camera. For each model, we align the walls of the models with the estimated wall locations relative to the camera, and independently evaluate each of the four orthogonal rotations of the 3D model about the vertical axis.

We render hypothesized 3D models of scenes using OpenGL, incorporating our calibrated camera parameters (focal length and principal points) with a viewing frustum to create renderings which align with our input image. This setup allows us to project objects from our 3D model library into the image plane in a manner which is consistent with the estimated camera parameters. We use this renderer as a fundamental tool in computing similarity features from each 3D model. The following section details this process.

2.2 Feature Computation

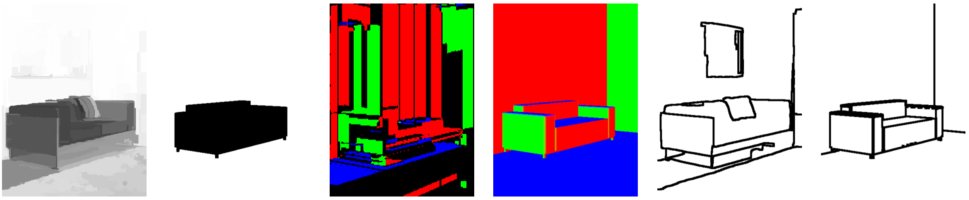

(a) p(object) (b) object mask (c) estimated normals (d) rendered normals (e) gPb (f) rendered edges

Figure 3: Descriptors extracted from an input image (a,c,e), and their corresponding rendered descriptors from the top-ranked 3D model (b,d,f).

An important question we address in this paper is, "What features are useful for matching 3D scenes with monocular images?" This issue is fundamentally complicated by the fact that we need to compare two objects of a completely different nature: an array of intensity/color pixels on the one hand, and a set of surfaces with little or no appearance information on the other hand.

We propose a series of image-based descriptors and their 3D counterparts. We use our renderer to produce synthetic image descriptors for each 3D model, and compare these to traditional image-based descriptors.

Object masks: We use the pre-trained geometric context model of [16] to estimate the likelihood that each of the pixels in an image contains an object (see Figure 3(a)). For each 3D model, we render a simple object mask (i.e., each polygon in the model is rendered black on a white background) as shown in Figure 3(b). After scaling each of these masks to be in the range $[-1,1]$, the normalized dot product between the predicted object locations and the rendered object masks indicate how well the model matches our image. This similarity measure is the first feature we use to compare 3D models to an input image.

Surface normals: We use the algorithm of [23] to predict the orientation of each pixel in an input image. For each 3D model, we render a surface normal image, by simply setting the red, green and blue color components of each polygon to the x, y and z components of the polygon's surface normal. See Figures 3(c) and 3(d) for examples of predicted surface normals, and rendered surface normals. The normalized dot product of these two descriptors quantifies their similarity. We use this value as a feature when scoring each 3D model.

Masked surface normals: We also combine our object masks and surface normal descriptors to create a highly-informative hybrid feature. For this feature, we multiply the object mask agreement score with the surface normal agreement score for each pixel (both scaled to be in the range $[0,1]$). This combined score aims to count how many pixels in the image satisfy two constraints: Firstly, objects in the renderings should appear only where they are predicted to be. Secondly, the surface normals of the 3D models at these locations should also agree with the predicted surface normals.

Edges: We extract edges from an input image using the globalPb algorithm [4] (thresholded at $p(\text{boundary}) > 0.5$). These edges are compared to Canny edges which are extracted from rendered surface-normal images of each scene hypothesis (Figures 3(e) and 3(f)). Pairs of edge images (extracted from the input image and each rendering) are compared using a modified symmetric Chamfer distance ($a \in A$ indicates a is an edge pixel in image A):

$$\Delta_{edge}(A,B) = \frac{1}{|A|}\sum_{a\in A}\min\left(\min_{b\in B}\|a-b\|,\tau\right) + \frac{1}{|B|}\sum_{b\in B}\min\left(\min_{a\in A}\|b-a\|,\tau\right). \qquad (1)$$

To reduce the influence of outlier edges which do not match well, we truncate individual edge distance penalties ($\tau \in \{10, 25, 50, \infty\}$ pixels). Intuitively, distances computed with smaller values of τ encourage fine-grain matching of edges, while distances computed with larger thresholds aim to penalize large errors. Each of the distances computed with a different value of τ is treated as a separate feature, for a total of four features.

2.3 Hypothesis Scoring with Support Vector Ranking

For a given input image, we render all of the 3D models in our scene library and compute similarity features from the renderings, as described above. We concatenate the object mask, surface normal, masked surface normal, and edge features into a 7-dimensional feature vector. A linear weighting of these features is computed to determine a matching score indicating how similar each 3D model is to the given image. We learn this weight vector using a max-margin learning framework. Using annotated training data, we can rank how well each 3D model in our library matches each training image. We use a modified version of the masked surface normal score presented in Section 2.2 to compute a similarity score for each pair of images and 3D models. For this scoring, we do not use the predicted surface normals and object masks, we use renderings of hand-annotated scene geometries which are treated as ground-truth.

Our goal is to find a weight vector w which can correctly rank pairs of 3D scenes (i.e.: $w^\top x^i_j > w^\top x^i_k$ if scene j matches image i better than scene k). We use the difference in masked surface normal scores as the hinge loss margin δ^i_{jk}. This optimization takes the form of support vector ranking [19]:

$$\min_{w,\xi} \frac{\lambda}{2}\|w\|^2 + \sum \xi^i_{jk} \qquad \text{s.t.:}\ w^\top x^i_j \geq w^\top x^i_k + \delta^i_{jk} - \xi^i_{jk},\ \ \xi^i_{jk} \geq 0. \qquad (2)$$

We optimize Equation 2 using stochastic gradient descent. In each iteration, we select a training image i and a pair of 3D models (j,k). If the current weight vector causes the pair of 3D models to be incorrectly ranked, or if their difference in scores is less the margin δ^i_{jk}, we compute a subgradient and update the weight vector. This process is repeated until convergence.

3 Experimental Dataset and Ground-truth Annotation

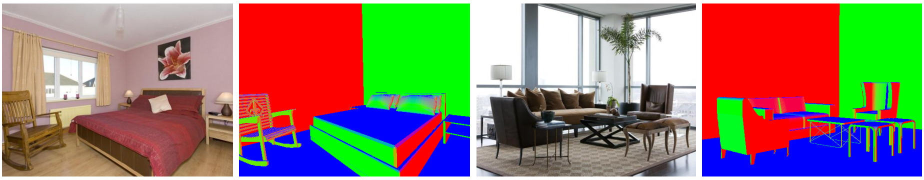
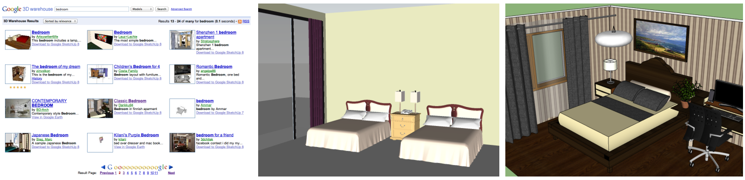


Figure 4: Example images and hand-crafted 3D models from our dataset.

Since the problem of monocular geometry estimation is relatively new, there does not yet exist an established dataset of images with detailed ground-truth object geometry and surface normals. Thus, we have created a new dataset with annotated scene geometry building upon the SUN database [34]. Our dataset consists of over 500 images from the categories "bedroom" and "living room." For each image, a detailed 3D model was constructed using Google SketchUp [2]. This software allows users to label vanishing points for camera autocalibration and insert existing 3D models of objects from the Internet to generate detailed models from an image.

We use these hand-crafted 3D models as ground-truth for training our scene ranker, as well as for evaluating the performance of our geometry estimation algorithm. Figure 4 includes example 3D models from our dataset. This dataset is in the process of being expanded to contain additional images and 3D models, and will be made publicly available.[1]

4 3D Model Library



Google 3D warehouse bedroom Models Search Advanced Search

3D Warehouse Results Sorted by relevance Results 13 - 24 of many for bedroom (0.1 seconds) - RSS

Bedroom
by Artsywriter4life
This bedroom includes a lamp,...
Download to Google SketchUp 8

Bedroom
by Lauz+Lachie
The most simple bedroom...
Download to Google SketchUp 8

Shenzhen 1 bedroom apartment
by Stratosphere
Shenzhen 1 bedroom apartment...
Download to Google SketchUp 8

The bedroom of my dream
by zinvolkon
This is the bedroom of my...
History
Download to Google SketchUp 8
★★★★★

Children's Bedroom for 4
by Costa Family
Bedroom layout with furniture...
Download to Google SketchUp 8

Romantic Bedroom
by angelpe86
Romantic Bedroom, one bed and...
Download to Google SketchUp 8

CONTEMPORARY BEDROOM
by BD-Arch
Contemporary style Bedroom...
View in Google Earth

Classic Bedroom
by Dankku94
Bedroom in finnish aparment
Download to Google SketchUp 8

bedroom
by Ammar
bedroom by Ammar
Download to Google SketchUp 7

Japanese Bedroom
by Sieg, Marc
A sample Japanese Bedroom
Download to Google SketchUp 8

Kilani's Purple Bedroom
by kilani
bed over dresser and mac book...
View in Google Earth

bedroom for a friend
by 3dchitek
facebook contest i did my my...
Download to Google SketchUp 8

◀ Goooooooooooogle ▶
Result Page: Previous 1 2 3 4 5 6 7 8 9 10 11 Next

Figure 5: An example query, and resulting "bedroom" models from Google 3D Warehouse.

We acquired our 3D indoor scenes through the public search engine of Google 3D Warehouse [3]. We perform queries for common indoor scenes such as "bedroom" and "living room," and download top ranking models. Due to the nature of data harvested from the web, a large number of scenes are irrelevant for our task. For example, a query for "bedroom" may return a 3D model for a "4 Bedroom House," which contains only an architectural model of the building exterior. Using simple heuristics, we discard models which are too large or small. Although over 8000 3D models matched our queries, the majority of them were rejected based on these criteria, leaving approximately 2000 models in our database. To increase the size of our model database, we include a reflected version of each scene. Each of these 3D models is then processed to segment individual components, save their polygonal faces, and identify object categories.

Each component has an associated label, such as "Couch," "Brown Leather Sofa," or "Love Seat." We automatically cluster the objects into groups by comparing their geometries with a simple 3D voxelized overlap score. This approach will discover that "Brown Leather Sofa" and "Love Seat" are synonyms of "Couch." To ensure accurate labeling of objects, we created a user-interface for quickly verifying or adjusting the label of each component.

[1] http://cmu.satkin.com/bmvc2012/

5 Experimental Results

In this section, we present a series of qualitative and quantitative results to demonstrate the capabilities of our scene matching technique. We partition our dataset and perform 10-fold cross-validation to report performance on the entire dataset. Figure 6 shows example results of our algorithm. Displayed from left to right are the input images, estimated surface normals from the top-ranking matched 3D model and color-coded object overlays.

Note that we are able to recover the labels, locations and orientations of objects, even from obscure viewpoints. For example, the image in the bottom left corner was captured from behind a couch; however, we still correctly identify this unique layout of furniture, recovering the position and orientation of both couches and the coffee table. Moreover, the general style of objects is also matched (*e.g.*, the headboard of each bed). Additionally, although appearance-based object detectors typically fail to detect mostly-occluded objects (such as nightstands), our holistic scene-matching approach is capable of finding these challenging objects.

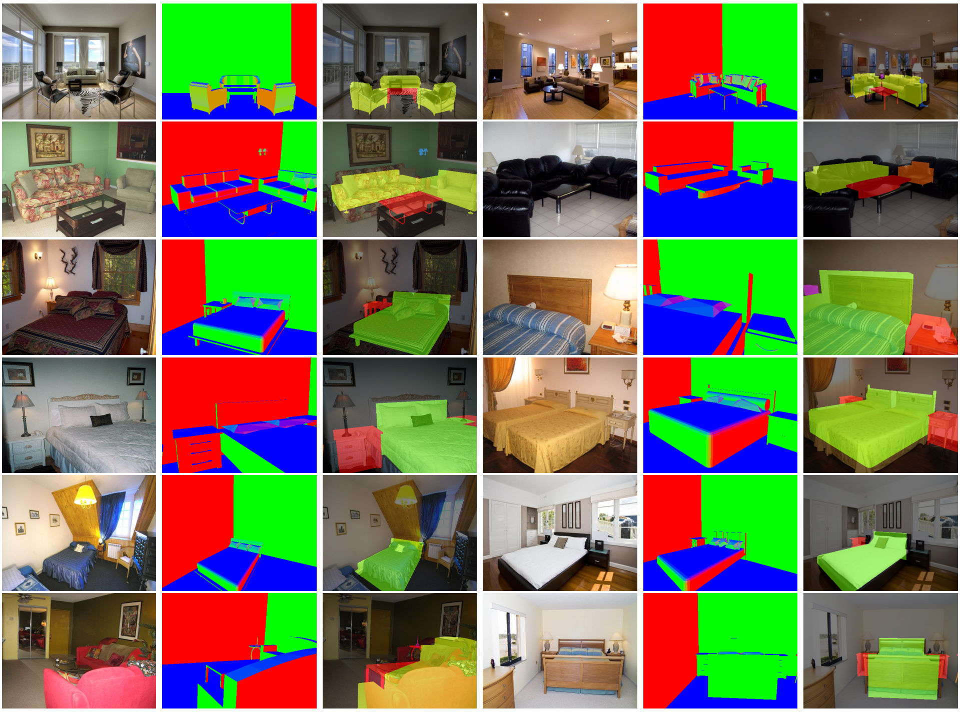


Figure 6: Input images, automatically-selected 3D models (surface normals displayed), and color-coded object labels (yellow=couch, red=table, green=bed).

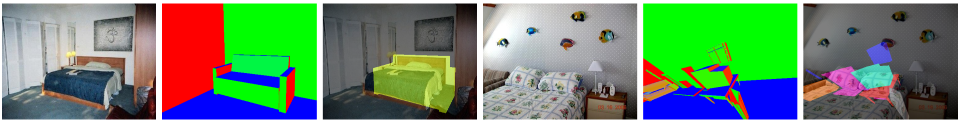


Figure 7: Failure cases of our scene matching system caused by incorrect room layout and vanishing point estimation.

5.1 Surface Normal Accuracy

One way to quantify the quality of our 3D models is to measure how accurately we can predict the orientation of surfaces in the scene. We score our 3D hypothesis by taking the dot product of the ground-truth surface normals (from our annotated models) and the orientations of pixels in our rendered hypothesis, normalized by the number of pixels. This score represents the percentage of the pixels for which we have correctly identified the orientation. Since the majority of pixels in most scenes correspond to walls or floor, which are not informative to the quality of object geometries, we also report the surface normal score for only those pixels which belong to objects.

A fundamental issue with our approach is our reliance on autocalibration and room layout estimation. If this stage of the pipeline fails, we will incorrectly estimate the arrangement of objects in a scene. For example, in Figure 7, the location of the wall behind the bed is predicted to be too close to the camera. This caused our scene matching process to fit a couch, which has less depth than a bed. The scene on the right shows a catastrophic failure due to incorrect vanishing point estimation. To decouple the effects of room layout estimation from our goal of determining the arrangement of objects, we report results using annotated room layouts and camera parameters as well as fully automatic results incorporating the room layout algorithm of [16].

Two annotators created 3D models for a subset of our images in our dataset. By comparing these different versions of the scene geometries, we can evaluate the subjectivity of the task and annotation process. Table 1 reports three sets of results. The first column measures how consistent humans are when annotating scene geometries. The next column reports the results of our algorithm for determining the arrangements of objects in a scene using annotated room layouts (camera parameters and wall locations). The last column reports the results of our fully-automated algorithm which uses the auto-calibration and room layout estimation techniques of [16].

	Human vs. Human	Annotated layout	Fully automatic
All pixels	0.88	0.82	0.70
Objects only	0.81	0.54	0.52

Table 1: Surface normal scores for all pixels, and only pixels belonging to objects.

5.2 Free Space Estimation

We also evaluate how accurately our algorithm can estimate the free space of a room from an image. Figure 8 shows our ability to recover an architectural floorplan of a room. We compare our estimated object locations to the ground-truth object locations from annotated images. For each square inch of the floor that we predict to be occupied, we compare to the ground-truth occupancy and report precision and recall. We run the pre-trained geometry estimation algorithm of Gupta et al. [13], and report their performance as a baseline. In addition, we use the evaluation metric proposed by [18]. Their metric provides a soft-measure of object overlap. Hypothesized objects which are close to ground-truth object locations, but do not overlap, are scored based on their distance to the closest ground-truth object.

Table 2 reports our precision and recall scores as well as [18]'s "δ precision" and "δ recall" scores. Additionally, we report the F-measure (harmonic mean of precision and recall) which aims to capture our performance with a single value.

We also compute a similar 3D free space evaluation by voxelizing our scenes and computing how precisely we estimate which voxels are occupied. Table 3 reports the results of this free space evaluation.

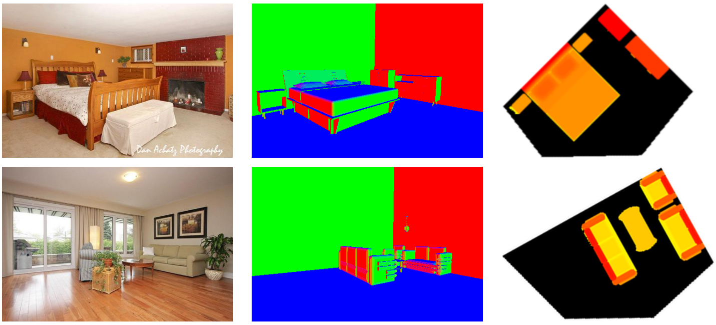


Figure 8: Input images, automatically-selected 3D models, and overhead views (color indicates height: yellow=low, red=high). Results shown use annotated camera parameters.

	Human vs. Human	Annotated layout	Fully automatic	Gupta et al. [13]
precision	0.88	0.59	0.50	0.37
recall	0.74	0.54	0.41	0.21
F-measure	0.80	0.53	0.40	0.25
δ precision	0.98	0.80	0.72	0.52
δ recall	0.92	0.77	0.63	0.38
δ F-measure	0.95	0.77	0.65	0.43

Table 2: 2D floorplan free space evaluation.

	Human vs. Human	Annotated layout	Fully automatic	Gupta et al. [13]
precision	0.73	0.47	0.39	0.12
recall	0.73	0.36	0.28	0.23
F-measure	0.72	0.36	0.27	0.14
δ precision	0.96	0.73	0.65	0.32
δ recall	0.94	0.66	0.56	0.42
δ F-measure	0.94	0.68	0.58	0.35

Table 3: 3D voxelized free space evaluation.

6 Conclusion

We have described a novel approach to scene understanding which leverages new large online repositories of 3D models. Our data-driven framework provides a mechanism for transferring rich information from these models to input images. We demonstrated the utility of our approach for the task of geometry estimation, and showed that we are able to produce accurate and detailed 3D models of images. Although this is only a first step in utilizing 3D data for scene understanding, we have shown that this is a promising paradigm for future experimentation.

Acknowledgements: This research is supported by MURI Grant N000141010934. The authors would like to thank Varsha Hedau and David Lee for making their code available. We also thank John Ulmer and Tricia Stahr from Google SketchUp for their support with exporting 3D models.

References

[1] Microsoft Corp. Redmond WA. Kinect for Xbox 360.

[2] Google SketchUp. http://sketchup.google.com/.

[3] Google 3D Warehouse. http://sketchup.google.com/3dwarehouse.

[4] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *PAMI*, May 2011.

[5] Sid Yingze Bao, Min Sun, and Silvio Savarese. Toward coherent object detection and scene layout understanding. In *CVPR*, 2010.

[6] Rodney Brooks. Symbolic reasoning among 3D models and 2D images. In *Artificial Intelligence*, volume 17, 1981.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009.

[8] Matthew Fisher and Pat Hanrahan. Context-based search for 3d models. *ACM Trans. Graph.*, 29:182:1–182:10, December 2010.

[9] Matthew Fisher, Manolis Savva, and Pat Hanrahan. Characterizing structural relationships in scenes using graph kernels. *ACM Trans. Graph.*, 30:34:1–34:12, August 2011.

[10] Helmut Grabner, Juergen Gall, and Luc J. Van Gool. What makes a chair a chair? In *CVPR*, 2011.

[11] W.E.L. Grimson, T. Lozano-Pérez, and D.P. Huttenlocher. *Object recognition by computer*. MIT Press, 1990.

[12] Abhinav Gupta, Alexei A. Efros, and Martial Hebert. Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *ECCV*, 2010.

[13] Abhinav Gupta, Scott Satkin, Alexei A. Efros, and Martial Hebert. From 3d scene geometry to human workspace. In *CVPR*, 2011.

[14] James Hays and Alexei A Efros. Scene completion using millions of photographs. *ACM Transactions on Graphics (SIGGRAPH 2007)*, 26(3), 2007.

[15] James Hays and Alexei A. Efros. im2gps: estimating geographic information from a single image. In *CVPR*, 2008.

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

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

[18] V. Hedau, D. Hoiem, and D. Forsyth. Recovering free space of indoor scenes from a single image. In *CVPR*, 2012.

[19] R. Herbrich, T. Graepel, and K. Obermayer. Support vector learning for ordinal regression. In *ANN*, volume 1, 1999.

[20] Derek Hoiem, Alexei A. Efros, and Martial Hebert. Recovering surface layout from an image. In *IJCV*, 2007.

[21] Derek Hoiem, Alexei A. Efros, and Martial Hebert. Closing the loop on scene interpretation. In *CVPR*, 2008.

[22] K. Lai and D. Fox. 3D laser scan classification using web data and domain adaptation. In *Proceedings of Robotics: Science and Systems*, June 2009.

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

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

[25] Ce Liu, Jenny Yuen, Antonio Torralba, Josef Sivic, and William T. Freeman. SIFT Flow: dense correspondence across different scenes. In *ECCV*, 2008.

[26] David Lowe. Three-dimensional object recognition from single two-dimensional images. In *Artificial Intelligence*, volume 31, 1987.

[27] Aude Oliva and Antonio Torralba. Building the gist of a scene: the role of global image features in recognition. In *Progress in Brain Research*, 2006.

[28] Luca Del Pero, Jinyan Guan, Ernesto Brau, Joseph Schlecht, and Kobus Barnard. Sampling bedrooms. In *CVPR*, 2011.

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

[30] Ashutosh Saxena, Min Sun, and Andrew Y. Ng. Make3D: Learning 3D scene structure from a single still image. *PAMI*, May 2009.

[31] Joseph Tighe and Svetlana Lazebnik. Superparsing: scalable nonparametric image parsing with superpixels. In *ECCV*, 2010.

[32] Antonio Torralba, Rob Fergus, and William T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *PAMI*, November 2008.

[33] H. Wang, S. Gould, and D. Koller. Discriminative learning with latent variables for cluttered indoor scene understanding. In *ECCV*, 2010.

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

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