Indoor Scene Segmentation using a Structured Light Sensor

Nathan Silberman and Rob Fergus
Dept. of Computer Science, Courant Institute, New York University
{silberman,fergus}@cs.nyu.edu

Abstract

In this paper we explore how a structured light depth sensor, in the form of the Microsoft Kinect, can assist with indoor scene segmentation. We use a CRF-based model to evaluate a range of different representations for depth information and propose a novel prior on 3D location. We introduce a new and challenging indoor scene dataset, complete with accurate depth maps and dense label coverage. Evaluating our model on this dataset reveals that the combination of depth and intensity images gives dramatic performance gains over intensity images alone. Our results clearly demonstrate the utility of structured light sensors for scene understanding.

1. Introduction

The use of depth or range sensors as well as depth-from-stereo has been the subject of a number of important, recent works for various vision-related tasks such as scene understanding and detection. Many approaches use a scene's depth as a channel for extracting features in a detection pipeline. Gould *et al.* [6] combine laser range finder data with images for detection of several small objects. Quigley *et al.* [16] use a laser-line scanner to recognize several classes and aid a robotic door-opening task.

Rather than use depth as a feature directly, a number of works use the depth of a scene to guide the detection process itself. Helmer and Lowe [8] explore how depth-from-stereo can limit the number of detection windows required while Hedau *et al.* [19] recovers the 3D structure of the room from a single image, which then provides context for recognition. Leibe *et al.* [12] use a car-mounted stereo rig to reason about the depth of a scene, detect pedestrians and cars, and track them over time.

In most works that utilize the depth signal of a scene, the dataset used is often collected from a very limited domain, specific to an application, and rarely made public. Consequently, while research has demonstrated that depth is a useful supplementary signal for vision tasks, competing approaches are rarely directly compared due to the lack of publicly available datasets. Additionally, researchers without access to the required, specialized hardware needed to produce these depth images cannot contribute to this area.

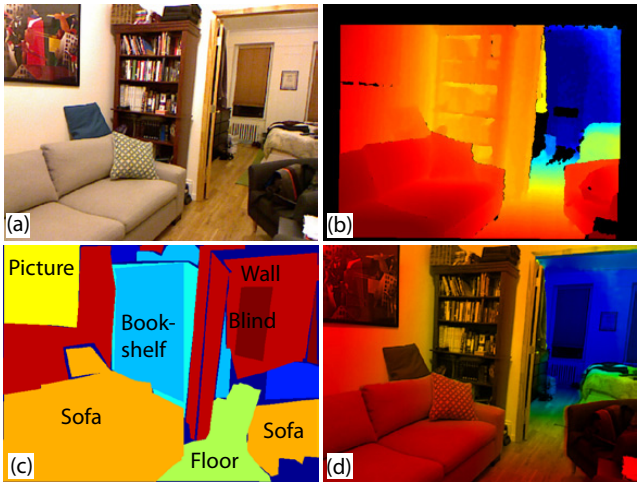

Figure 1. A typical indoor scene captured by the Microsoft Kinect. (a): Webcam image. (b) Raw depth map (red=close, blue=far). (c) Labels obtained via Amazon Mechanical Turk. (d) After a homography, followed by pre-processing to fill in missing regions, the depth map (hue channel) can be seen to closely aligned with the image (intensity channel).

To ammend this shortcoming, we introduce a new dataset complete with densely labeled pairs of RGB and depth images. These images have been collected using the Microsoft Kinect. This device uses structured light methods to give an accurate depth map of the scene, which can be aligned spatially and temporally with the device's webcam (see Fig. 1). The choice of this device over other depth-measurement tools (LIDARs and time-of-flight cameras) was motivated by its accuracy, compactness, portability (after a few modifications) and its price. These qualities make use of the device viable in numerous vision applications, such as assisting the visually impaired and robot navigation.

One clear limitation of the Kinect is that it can only operate reliably indoors, since the projected pattern is overwhelmed by exterior lighting conditions. We therefore focus our attention on indoor scenes.

The closest work to ours is that of Lai *et al.* [10] which contributed a depth-based dataset. Their images, however, are limited to isolated objects with uncluttered backgrounds rather than entire scenes making their dataset qualitatively similar to COIL [14].

This paper makes a number of contributions: (1) we introduce a new indoor scene dataset of which every frame has an accurate depth map as well as a dense manually-provided labeling - to our knowledge, the first of its kind, (2) we describe simple modifications that make the Kinect fully portable, hence usable for indoor recognition, (3) we provide baselines on the new dataset for the scene classification and multi-class segmentation tasks using several commonly used features and (4) we introduce a new 3D location prior improving recognition performance.

2. Approach

We now describe how the Kinect was made portable in order to facilitate data capture, as well as the image pre-processing necessary to make the Kinect output usable.

2.1. Capture Setup

The Kinect has two cameras: the first is a conventional VGA resolution webcam that records color video at 30Hz. The second is an infra-red (IR) camera that records a non-visible structured light pattern generated by the Kinect's IR projector. The IR camera's output is processed within the Kinect to provide a smoothed VGA resolution depth map, also at 30Hz, with an effective range of ~0.7–6 meters. See Fig. 1(a) & (b) for typical output.

The Kinect requires a 12V input for the Peltier cooler on the IR depth camera, necessitating a mains adapter to power the device (USB sockets only provide 5V at limited currents). Since the mains adapter severely limits portability of the device, we remove it and connect a rechargeable 4200mAh 12V battery pack in its place. This is capable of powering the device for 12 hours of operation. The output from the Kinect was logged on a laptop carried in a backpack, using open-source Kinect drivers [13] to acquire time synchronized image, depth and accelerometer feeds. The overall system is shown in Fig. 2(a). To avoid camera shake and blur when capturing data, the Kinect was strapped to a motion-damping rig built from metal piping, shown in Fig. 2(b). The weights damp the motion and have a significant smoothing effect on the captured video.

Both the depth and image cameras on the Kinect were calibrated using a set of checkerboard images in conjunction with the calibration tool of Burrus [4]. This also provided the homography between the two cameras, allowing us to obtain precise spatial alignment between the depth and RGB images, as demonstrated in Fig. 1(d).

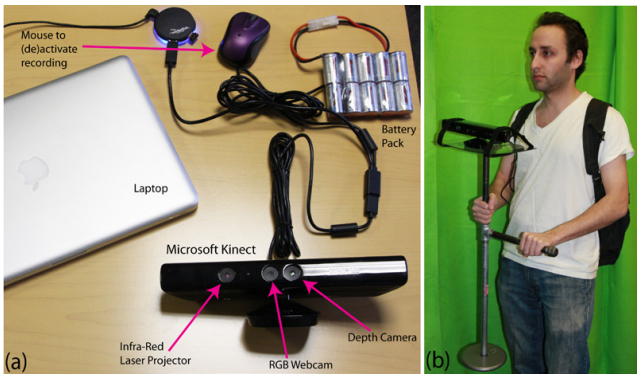

Figure 2. (a): Our capture system with a Kinect modified to run from a battery pack. (b) Our capture platform, with counter-weights to damp camera movements.

2.2. Dataset Collection

We visited a range of indoor locations within a large US city, gathering video footage with our capture rig. These mainly consisted of residential apartments, having living rooms, bedrooms, bathrooms and kitchens. We also captured workplace and university campus settings. From the acquired video, we extracted frames every 2–3 seconds to give a dataset of 2347 unique frames, spread over 64 different indoor environments. The dataset is summarized in Table 1. These frames were then uploaded to Amazon Mechanical Turk and manually annotated using the LabelMe interface [17]. The annotators were instructed to provide dense labels that covered every pixel in the image (see Fig. 1(c)). Further details of the resulting label set are given in Section 4.2.

2.3. Pre-processing

Following alignment with the RGB webcam images, the depth maps still contain numerous artifacts. Most notable of these is a depth "shadow" on the left edges of objects. These regions are visible from the depth camera, but not reached by the infra-red laser projector pattern. Consequently their depth cannot be estimated, leaving a hole in the depth map. A similar issue arises with specular and low albedo surfaces. The internal depth estimation algorithm also produces numerous fleeting noise artifacts, particularly near edges.

Before extracting features for recognition, these artifacts must be removed. To do this, we filtered each image using

Scene class	Scenes	Frames	Labeled Frames
Bathroom	6	5588	76
Bedroom	17	22764	480
Bookstore	3	27173	784
Cafe	1	1933	48
Kitchen	10	12643	285
Living Room	13	19262	355
Office	14	19254	319
Total	64	108617	2347

Table 1. Statistics of captured sequences.

the cross-bilateral filter of Paris [15]. Using the RGB image intensities, it guides the diffusion of the observed depth values into the missing shadow regions, respecting the edges in intensity. An example result is shown in Fig. 1(d).

The Kinect contains a 3-axis accelerometer that allows us to directly measure the gravity vector[1] and hence estimate the pitch and roll for each frame. Fig. 3 shows the estimate of the horizon for two examples. We rotate the RGB image, depth map and labels to eliminate any pitch and roll, leaving the horizon horizontal and centered in the image.

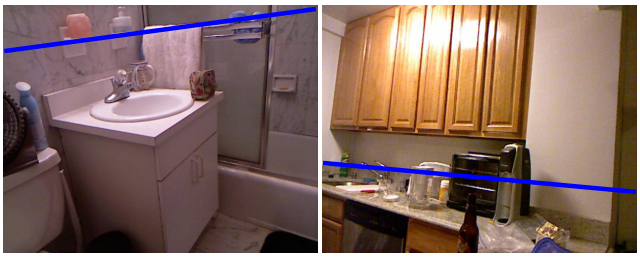

Figure 3. Examples of images with significant pitch and roll overlaid with horizon estimates, computed from the Kinect's accelerometer.

3. Model

We now describe the model used to measure baseline performance for the dataset. In common with several other multi-class segmentation approaches [7, 18], we use a conditional random field (CRF) model as its flexibility makes it easy to explore a variety of different potential functions. A further benefit is that inference can be performed efficiently with the graph-cuts optimization scheme of Boykov *et al.* [2][2].

The CRF energy function $E(\mathbf{y})$ measures the cost of a latent label y_i over each pixel i in the image N. y_i can take on a discrete set of values $\{1, \ldots, C\}$, C being the number of classes. The energy is composed of three potential terms: (1) a unary cost function ϕ, which depends on the pixel location i, local descriptor x_i and learned parameters θ; (2) a class transition cost $\psi(y_i, y_j)$ between pairs of adjacent pixels i and j and (3) a spatial smoothness term $\eta(i, j)$, also between adjacent pixels, that varies across the image.

$$E(\mathbf{y}) = \sum_{i \in N} \phi(x_i, i; \theta) + \sum_{i,j \in N} \psi(y_i, y_j)\eta(i, j) \quad (1)$$

Before applying the CRF, we generate super-pixels $\{s_1, \ldots, s_k\}$ using the low-level segmentation approach of Felzenszwalb and Huttenlocher [5]. We compute two different sets of super-pixels: S_{RGB} using the RGB image and S_{RGBD}, which is computed using both RGB and depth images[3]. We make use of super-pixels when aggregating the predictions from the unary potentials ϕ, as explained in Section 3.1.1. We also have the option of using them in the spatial smoothness potential η (Section 3.3).

3.1. Unary Potentials

The unary potential function ϕ is the product of two components, a local appearance model and a location prior.

$$\phi(x_i, i|\theta) = -\log(\underbrace{P(y_i|x_i, \theta)}_{\text{Appearance}} \underbrace{P(y_i, i)}_{\text{Location}}) \quad (2)$$

3.1.1 Appearance Model

Our appearance model $P(y_i|x_i, \theta)$ is discriminatively trained using a range of different local descriptors x_i of dimension D, as detailed below. For each descriptor type, we use the same training framework, which we now describe:

Descriptors are first extracted over the same dense grid[4] at S scales[5]. If $x_i^{(s)}$ is the descriptor extracted at grid point i and scale s, then $x_i = concat(x_i^{(1)}, x_i^{(2)}, ...x_i^{(S)})$

Given the set of descriptors $X = \{x_i : i = 1..N\}$ extracted from training images, we train a neural network with a single hidden layer of size $H(= 1000)$ and a softmax output layer of dimension C, which is interpreted as $P(y_i|x_i, \theta)$. It has parameters θ (two weight matricies of sizes $(D + 1) \times H$ and $(H + 1) \times C$) which are learned using back-propagation and a cross-entropy loss function.

The ground truth labels y_i^* for each descriptor x_i are taken from the dense image labels obtained from Amazon Mechanical Turk. The value of y_i^* is set to the label provided at grid location i.

Following training, the neural network model maps a local descriptor x_i directly to $P(y_i|x_i, \theta)$. Then, for each super-pixel s_k within an image, we average the probabilities $P(y_i|x_i, \theta)$ from each descriptor that falls into it and assign every pixel within s_k the resulting mean class probabilities.

We use a range of descriptor types as input x_i to the scheme above.

- **RGB-SIFT:** SIFT descriptors are extracted from the RGB image. This is our baseline approach.
- **Depth-SIFT:** SIFT descriptors are extracted from the depth image. These capture both large magnitude gradients caused by depth discontinuities, as well as small gradients that reveal surface orientation.
- **Depth-SPIN:** Spin image descriptors [9] are extracted from the depth map. To review, this is a descriptor designed for matching 3D point clouds and surfaces. Around each point in the depth image, a 2D histogram is built that counts nearby points as a function of radius and depth. The histogram is vectorized to form a descriptor.

[1] During capture the device was moved slowly to minimize direct accelerations.

[2] In practice we use Bagon's Matlab wrapper [1].

[3] Here, the input to [5] is the RGB image, with the blue channel replaced by an appropriately scaled depth map.

[4] Stride: 10 pixels; Patch size: 40×40 pixels.

[5] Scales: 1, .707, 0.5

We also propose several approach that combine information from the RGB and depth images:

- **RGBD-SIFT:** SIFT descriptors are extracted from both depth and RGB images. At each location, the 128D descriptors both images are concatenated to form a single 256D descriptor $x_i^{(s)}$ at each scale s.
- **RGB-SIFT/D-SPIN:** Spin image descriptors are extracted from the depth map, while SIFT is extracted from the RGB image.

3.1.2 Location Prior

Our location prior $P(y_i, i)$ can take on two different forms. The first captures the 2D location of objects, similar to other context and segmentation approaches (e.g [18]). The second is a novel 3D location prior that leverages the depth information.

2D location priors: The 2D priors for each class are built by averaging over every training image's ground truth label map $\mathbf{y}^*$. To provide a degree of 2D spatial invariance, we then smooth the averaged map with an 11×11 Gaussian filter. To compute the actual prior distribution $P(y_i, i)$, we normalize each map so it sums to $1/C$, i.e. $\sum_i P(y_i, i) = 1/C$. Note that this assumes the prior class distribution to be uniform[6]. Figure Fig. 4 shows the resulting distributions for 4 classes.

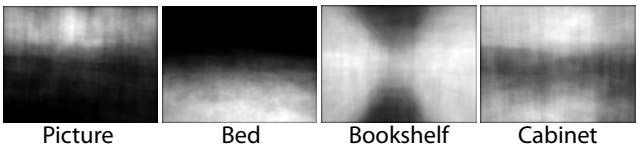

Figure 4. 2D location priors for select object classes.

3D location priors: The depth information provided by the Kinect allows us to estimate the 3D position of each object in the scene. However, the problem when building a 3D prior is how to combine this information from scenes of differing size and shape. The design of our 3D prior is motivated by three empirical constraints (C1-C3):

C1: While the absolute depth of an individual object in a scene is arbitrary with respect to the location of the viewer, objects of different classes exhibit a high degree of regularity with respect to their *relative* depths in a room. Fig. 6 highlights several examples. Walls are obviously at the farthest depths of rooms, televisions tend to placed just in front of them, and tables and beds are much more likely to occupy regions near the center of a scene.

C2: Many objects tend to be clustered near the edges of a room, such as walls, blinds, curtains, windows and pictures. Consequently, we want a non-linear scaling function that places increased emphasis on depths near the boundaries of a room.

C3: While objects show regularity in relative depth, any representation of an objects prior location must be somewhat invariant to the viewer moving around the room.

Our solution, therefore, is to normalize the depth of an object, using the depth of the room itself. We assume that in any given column[7] of the depth map, the point furthest from the camera is on the bounding hull of the room. Fig. 5 demonstrates the reliability of the procedure in separating the boundaries of the room from objects of similar depth. We scale the depths of all points in a given column so that the furthest point has relative depth $\tilde{z} = 1$. This effectively maps each room to a lie within a cylinder of radius 1. This allows us to build the highly regular depth profiles for each class.

Figure 5. A demonstration of our scheme for finding the boundaries of the room. In this scene, the blue channel has been replaced by a binary mask, set to 1 if the depth of point is within 4% of the maximum depth within each column (and 0 otherwise). The walls of the room are cleanly identified, while segmenting objects of similar depth such as the fire extinguisher and towel dispenser. On the right, the cabinets and sink are correctly resolved as being in the room interior, rather on the boundary.

Within this normalized reference frame, we then build histograms from the 3D positions of objects in the training set. These 3D histograms are over $(h, \omega, \tilde{z})$ where h is the absolute scene height relative to the horizon (in meters); ω is angle about the vertical axis and $\tilde{z}$ is relative depth.

In addition, we use a non-linear binning for $\tilde{z}$. This produces very fine bins near the boundaries of the room, allowing us to discriminate between the many objects at the extremal edges of the room (satisfying C2), and coarse bins at the center of the room, giving us a degree of invariance to the camera's position (satisfying C3).

[6] In practice, if the true class frequencies are used, common classes would be overly dominant in the CRF output.

[7] This is assisted by the pitch and roll correction made in pre-processing.

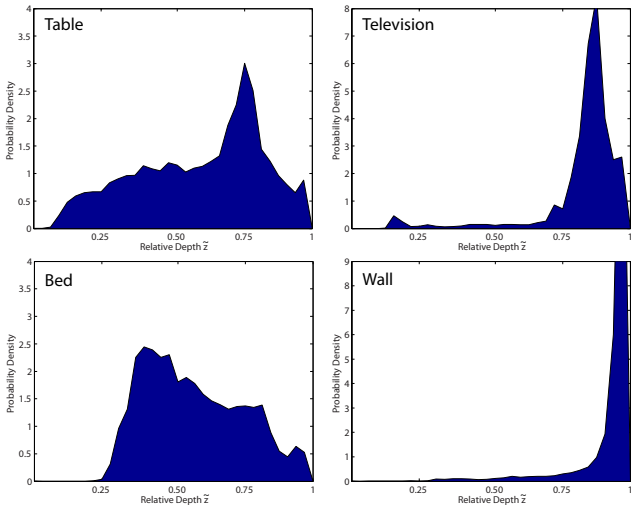

Figure 6. Relative depth histograms for table, television, bed and wall. As walls usually are on the boundary, they cluster near $\tilde{z} = 1$. Televisions lie just inside the room boundary, while tables and beds are found in the room interior.

Similar to the 2D versions, the 3D histograms are normalized so that they sum to $1/C$ for each class (see Fig. 7 for examples). During testing, the extremal depth for each column in the depth map is found and the relative 3D coordinate of each point can be computed. Looking up these coordinates in the 3D histograms gives the value of $P(y_i, i)$.

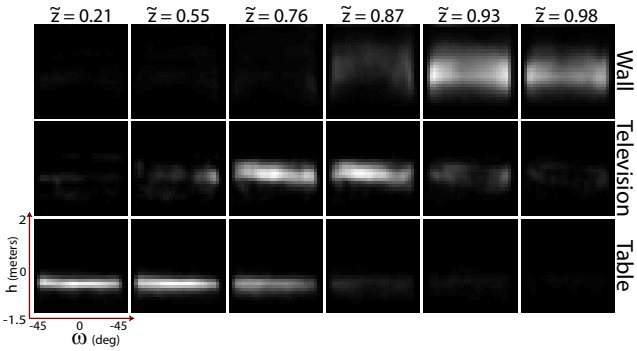

Figure 7. 3D location priors for wall, television and table. Each column shows a different relative depth $\tilde{z}$. For each subplot, the x-axis is orientation ω about the vertical and the y-axis is height h (relative to the horizon). The non-linear bin spacing in $\tilde{z}$ gives a more balanced distribution than linear spacing used in Fig. 6.

3.2. Class Transition Potentials

For this term we chose a simple Potts model [3]:

$$\psi(y_i, y_j) = \begin{cases} 0 & \text{if } y_i = y_j \\ d & \text{otherwise} \end{cases} \quad (3)$$

The deliberate use of a simple class transition model allows us to clearly see the benefits of the depth on the other two potentials in the CRF. In our experiments we use $d = 3$.

3.3. Spatial Transition Potentials

The spatial transition cost $\eta(i, j)$ provides a mechanism for inhibiting or encouraging a label transition at each location (independently of the proposed label class). We explore several options using a potential of the form:

$$\eta(i, j) = \eta_0 \, e^{-\alpha \max(|I(i)-I(j)|-t, 0)} \quad (4)$$

where $|I(i) - I(j)|$ is gradient between adjacent pixels i, j in image I, t is a threshold and α and η_0 are scaling factors. We use $\eta_0 = 100$ for all the following methods:

- **None:** The baseline method is to keep $\eta(i, j) = 1$ for all i, j in the CRF. The smoothness of the labels $\mathbf{y}$ is then solely induced by the class transition potential ψ.
- **RGB Edges:** We use I_{RGB} in Eqn. 4, thus encouraging transitions at intensity edges in the RGB image. $\alpha = 40$ and $t = 0.04$.
- **Depth Edges:** We use I_{Depth} in Eqn. 4, with $\alpha = 30$ and $t = 0.1$. This encourages transitions at depth discontinuities.
- **RGB + Depth Edges:** We combine edges from both RGB and depth images, with $\eta(i, j) = \beta \eta_{\text{RGB}}(i, j) + (1 - \beta)\eta_{\text{Depth}}(i, j)$ and $\beta = 0.8$.
- **Super-Pixel Edges:** We only allow transitions on the boundaries defined by the super-pixels, so set $\eta(i, j) = 1$ on super-pixel boundaries and η_0 elsewhere.
- **Super-Pixel + RGB Edges:** As for **RGB-Edges** above, but now we multiply $|I(i) - I(j)|$ in Eqn. 4 by the binary super-pixel boundary mask.
- **Super-Pixel + Depth Edges:** As for **Depth-Edges** above, but now we apply the binary super-pixel boundary mask to $|I(i) - I(j)|$.

4. Experiments

Before performing multi-class segmentation using our CRF-based model, we first try the simpler task of scene recognition to gauge the difficulty of our dataset.

4.1. Scene Classification

Table 1 shows the 7 scene-level classes in our dataset. After removing the 'Cafe' scene images (since using a single scene of this class would not make sense for scene classification) we split each of these into disjoint sets of equal size, careful to ensure frames from the same scene are not in both train and test sets. We apply the spatial pyramid matching scheme of Lazebnik *et al.* [11], using SIFT extracted from the RGB image (standard features), as well as SIFT on the depth image and both images (using the combination methods explained in Section 3.1.1). The mean confusion matrix diagonal is plotted in Fig. 8 as a function of k-means dictionary size for the different methods. Note that when using the RGB images, the accuracy is only 55%,

far less than the 81% achieved by the same method on the 15-class scene dataset used in [11]. This demonstrates the challenging nature of our data.

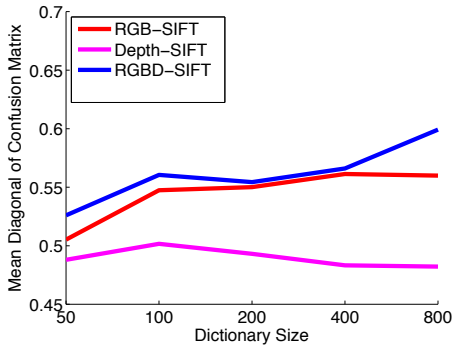

Figure 8. Scene classification performance for our dataset, using the approach of [11]. A significant performance gain is observed when depth and RGB information are combined with a large dictionary.

4.2. Multi-class Segmentation

We now evaluate our CRF-based model using the fully labeled set of 2347 frames. The annotations cover over 1000 classes, which we reduce using a Wordnet synonym/homonym structure to 12 common categories plus a generic background class (containing rare objects). We generated 10 different train/test splits, each of which divides the data into roughly 60% train and 40% test (see Table 2 for object counts). The error metric used throughout is the mean diagonal of the confusion matrix, computed for per-pixel classification over the 13 classes on the test set.

4.2.1 Unary Appearance

We first use our dataset to compare the local appearance models listed in Section 3.1.1, with the results shown in Table 3. We show the performance of the unary potential (with no location prior) in isolation, as well as the full CRF (sans spatial transition potential). The first row in the table, which makes no use of depth information achieves 43.4% accuracy. We note that: (i) combining RGB and depth information gives a significant performance gain of ∼5%; (ii) the CRF model gives a gain of ∼2.5% and (iii) the SIFT-based descriptors outperform the SPIN-based ones.

Object class	Train	Test	Overall	% Pixels
Bed	164.5	104.5	269	1.1
Blind	88.3	67.7	156	0.6
Bookshelf	685.5	283.5	969	6.4
Cabinet	520.0	311.0	831	3.1
Ceiling	947.0	525.0	1472	3.1
Floor	1213.8	578.2	1792	3.3
Picture	976.2	540.8	1517	1.4
Sofa	195.8	142.2	338	1.1
Table	1162.7	527.3	1690	3.1
Television	98.2	67.8	166	0.6
Wall	2564.1	1484.9	4049	22.4
Window	235.5	157.5	393	1.0
Background	-	-	-	34.1
Unlabeled	-	-	-	18.4
Object Total	8851.6	4790.4	13642	47.2

Table 2. Statistics of objects present in our 2347 frame dataset. Train and test counts are averaged over the 10 folds.

Descriptor	Unary Only	CRF
RGB-SIFT (S_{RGB})	40.9 ± 3.0	43.4 ± 3.3
RGB-SIFT (S_{RGBD})	40.4 ± 2.8	43.3 ± 3.1
Depth-SIFT	39.3 ± 2.2	41.1 ± 2.5
Depth-SPIN	34.0 ± 2.8	35.8 ± 3.1
RGBD-SIFT	45.8 ± 2.6	$\mathbf{48.1 \pm 2.9}$
RGB-SIFT/D-SPIN	42.5 ± 1.5	45.0 ± 1.6

Table 3. A comparison of unary appearance terms. Mean per-pixel classification accuracy (in %) using the test set of Table 2. All methods in this table compute appearance using S_{RGBD} super-pixels apart from the 1st row.

4.2.2 Unary Location

We now investigate the effect of location priors in our model. Table 4 compares the effect of the 2D and 3D location priors detailed in Section 3.1.2. All methods used S_{RGBD} super-pixels and no spatial transition potentials in the CRF. The 2D priors give a modest boost of 2.8% when used in the CRF model. However, by contrast, our novel 3D priors give a gain of 10.3%. We also tried bulding a prior using absolute 3D locations (3D priors (abs) in Table 4), which did not use our depth normalization scheme. This performed very poorly, demonstrating the value of our novel prior using relative depth. The overall performance gains for each of the 13 classes, relative to the RGB-SIFT (S_{RGB}) model (1st row of Table 3, which makes no use of depth information), is shown in Fig. 9. Using RGBD-SIFT and 3D priors, gains of over to 29% are achieved for some classes.

Descriptor	Unary Only	CRF
RGB-SIFT	40.9 ± 3.0	43.4 ± 3.3
RGB-SIFT+2D Priors	45.7 ± 2.8	46.2 ± 2.8
RGBD-SIFT	45.8 ± 2.6	48.1 ± 2.9
RGBD-SIFT+2D Priors	49.2 ± 2.2	49.9 ± 2.3
RGBD-SIFT+3D Priors	53.0 ± 2.2	$\mathbf{53.7 \pm 2.3}$
RGBD-SIFT+3D Priors (abs)	38.7 ± 3.2	39.9 ± 3.5

Table 4. A comparison of unary location priors.

In Fig. 10 we show six example images, each with label maps output by the RGB-SIFT+2D Priors and RGBD-SIFT+3D Priors models. The RGB model (2nd column) makes mistakes which are implausible based on the object's 3D location. The RGBD and 3D prior model gives a more powerful spatial context, with its label map (3rd column) being close to that of ground truth (4th column).

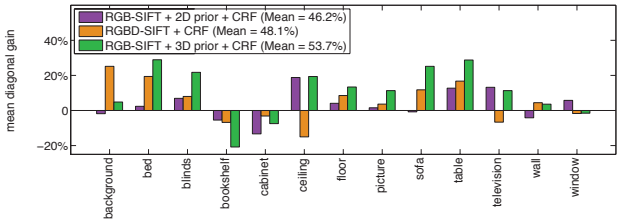

Figure 9. The per-class improvement over RGB-SIFT (S_{RGB}) + CRF (which uses no depth information), for models that add depth and 3D location priors. The large gains show benefits of adding depth information to both the appearance and location potentials.

4.2.3 Spatial Transition Potentials

Table 5 explores different forms for the spatial transition potential. All methods use unary potentials based on S_{RGBD} super-pixels and RGBD-SIFT + 3D prior. The results show that using an RGB-based spatial transition gives a performance gain of 2.8%. Using the depth or super-pixel constraints does not give a significant gain however.

Type	CRF
None	53.7 ± 2.3
RGB Edges	56.6 ± 2.9
Depth Edges	53.9 ± 3.1
RGB + Depth Edges	56.5 ± 2.9
Super-Pixel Edges	54.7 ± 2.4
Super-Pixel + RGB Edges	56.4 ± 3.0
Super-Pixel + Depth Edges	53.0 ± 3.0

Table 5. A comparison of spatial transition potentials. Mean per-pixel classification accuracy (in %).

5. Discussion

We have introduced a new indoor scene dataset that combines intensities, depth maps and dense labels. Using this data, our experiments clearly show that the depth information provided by the Kinect gives a significant performance gain over methods limited to intensity information. These gains have been achieved using a range of simple techniques, including novel 3D location priors. The magnitude of the gains achieved makes a compelling case for the use of devices such as the Kinect for indoor scene understanding.

Acknowledgements: This project is sponsored in part by NSF grant IIS-1116923.

References

[1] S. Bagon. Matlab wrapper for graph cut, December 2006. 3

[2] Y. Boykov, O. Veksler, and R. Zabih. Efficient approximate energy minimization via graph cuts. *IEEE PAMI*, 20(12):1222–1239, 2001. 3

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE PAMI*, 23:1222–1239, 2001. 5

[4] N. Burrus. Kinect rgb demo v0.4.0. Website, 2011. http://nicolas.burrus.name/index.php/Research/KinectRgbDemoV2. 2

[5] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2), 2004. 3

[6] S. Gould, P. Baurnstark, M. Quigley, A. Ng, and D. Koller. Integrating visual and range data for robotic object detection. In *ECCV Workshop (M2SFA2)*, 2008. 1

[7] X. He, R. Zemel, and M. Perpinan. Multiscale conditional random fields for image labeling. In *CVPR*, 2004. 3

[8] S. Helmer and D. G. Lowe. Using stereo for object recognition. In *ICRA*, 2010. 1

[9] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE PAMI*, 21(5):433–449, 1999. 3

[10] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *ICRA*, 2011. 2

[11] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006. 5, 6

[12] B. Leibe, N. Cornelis, K. Cornelis, and L. van Gool. Dynamic 3D scene analysis from a moving vehicle. In *CVPR*, 2007. 1

[13] H. Martin. Openkinect.org. Website, 2010. http://openkinect.org/. 2

[14] S. A. Nene, S. K. Nayar, and H. Murase. Columbia Object Image Library (COIL-20). Technical report, Columbia University, Feb 1996. 2

[15] S. Paris and F. Durand. A fast approximation of the bilateral filter using a signal processing approach. In *In Proceedings of the European Conference on Computer Vision*, pages 568–580, 2006. 3

[16] M. Quigley, S. Batra, S. Gould, E. Klingbeil, Q. Le, A. Wellman, and A. Y. Ng. High-accuracy 3d sensing for mobile manipulation: improving object detection and door opening. In *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, ICRA'09, pages 3604–3610, Piscataway, NJ, USA, 2009. IEEE Press. 1

[17] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: A database and web-based tool for image annotation. *MIT AI Lab Memo*, 2005. 2

[18] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006. 3, 4

[19] D. F. V. Hedau, D. Hoiem. Thinking inside the box: Using appearance models and context based on room geometry. In *ECCV*, 2010. 1

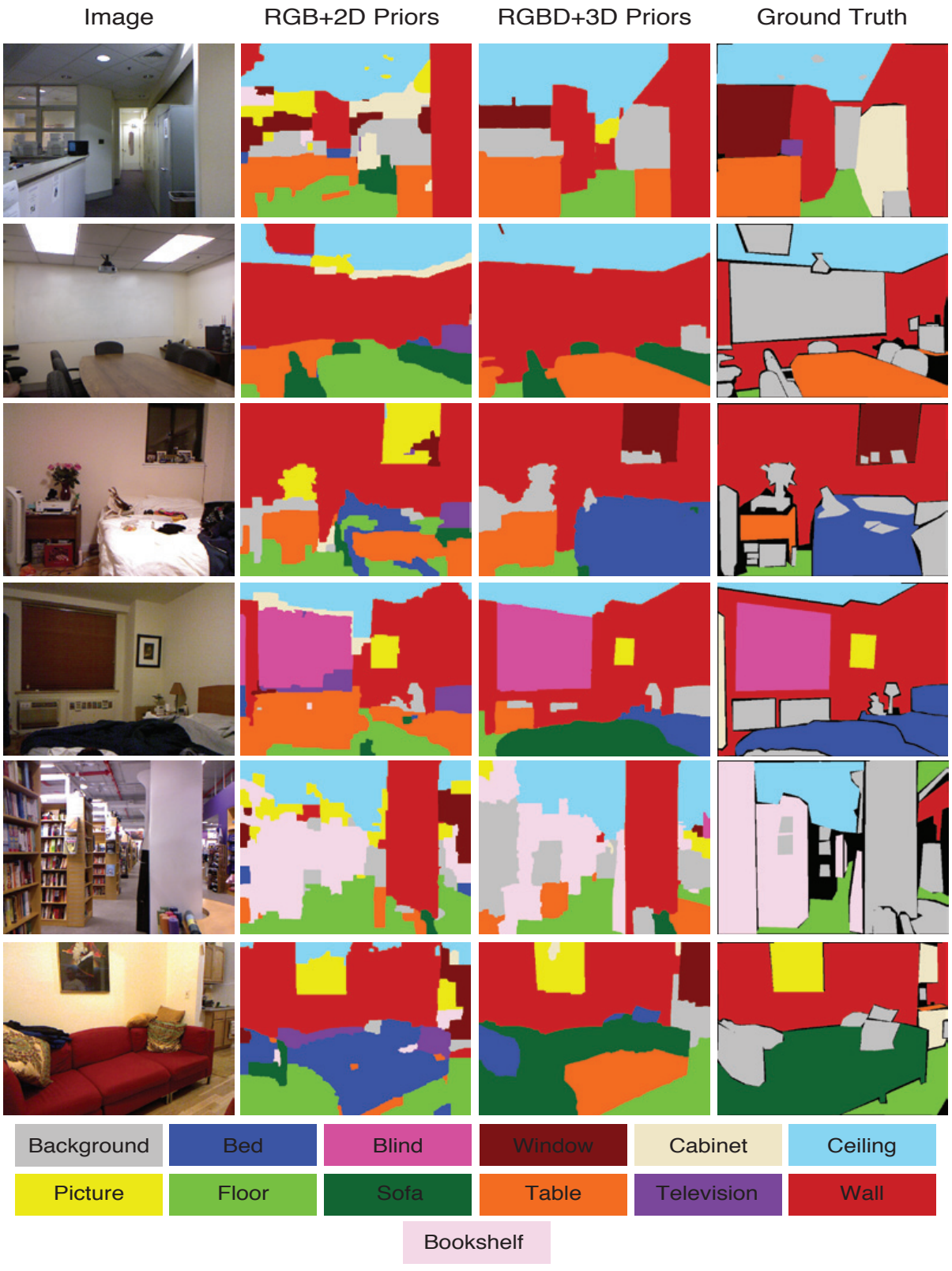

Figure 10. Six example scenes, along with outputs from 2 different models. See text for details. This figure is best viewed in color.