Real-time 3D Scene Layout from a Single Image Using Convolutional Neural Networks

Shichao Yang[1], Daniel Maturana[1] and Sebastian Scherer[1]

***Abstract*— We consider the problem of understanding the 3D layout of indoor corridor scenes from a single image in real time. Identifying obstacles such as walls is essential for robot navigation, but also challenging due to the diversity in structure, appearance and illumination of real-world corridor scenes. Many current single-image methods make Manhattan-world assumptions, and break down in environments that do not meet this mold. They also may require complicated hand-designed features for image segmentation or clear boundaries to form certain building models. In addition, most cannot run in real time.**

In this paper, we propose to combine machine learning with geometric modelling to build a simplified 3D model from a single image. We first employ a supervised Convolutional Neural Network (CNN) to provide a dense, but coarse, geometric class labelling of the scene. We then refine this labelling with a fully connected Conditional Random Field (CRF). Finally, we fit line segments along wall-ground boundaries and "pop up" a 3D model using geometric constraints.

We assemble a dataset of 967 labelled corridor images. Our experiments on this dataset and another publicly available dataset show our method outperforms other single image scene understanding methods in pixelwise accuracy while labelling images at over 15 Hz**.**

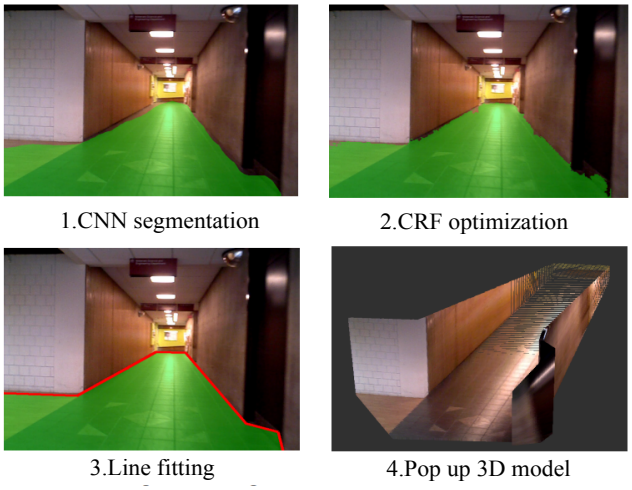

Fig. 1. Overview of our proposed method. We first segment the image into ground (in green) and walls using CNN, then refine it by CRF. After that, we detect the boundary, fit line segments and pop it up to 3D model.

I. Introduction

In order to safely navigate inside corridors, robots need to perceive the environment, detect wall obstacles and generate actions in real time. Cameras are a popular sensor on robots, as they provide rich information of the scene while having a small footprint and low cost. Thus, we are interested in using camera imagery to construct 3D representations for autonomous navigation.

A common approach to this problem is to use geometric 3D reconstruction techniques, but these require association across multiple images and often fail when there are few visual or geometric features in the scene, which is common in indoor scenes. Meanwhile, humans can effortlessly extract considerable geometric information from single images. For example, given the image in Figure 1 we can quickly interpret it as a corridor intersection surrounded by walls, and judge the approximate distance of these surfaces to the camera.

We wish to provide the same kind of abilities to robots, so they can easily find the free space and wall obstacles given a single image. The reasoning should be robust to various conditions such as poor lighting, homogeneous or even occluded situations. This ability can greatly help robots navigate in challenging featureless situations. It can also extend the sensing horizon beyond stereo cameras and other light-weight active sensors.

Our proposed approach is to combine machine learning with inference of geometric properties to achieve efficient scene understanding. It contains two parts: a learning algorithm to detect ground and wall planes and geometric modelling to build a simplified 3D plane model. For the learning part, we use a type of Convolutional Neural Network (CNN) and a Conditional Random Field (CRF) to predict a geometric layout class for each pixel in the image. We then use geometric constraints to compute the relative orientations of the wall and ground to pop up ground and wall planes into a simplified 3D model. In our evaluation, we show this system to be faster, more accurate and robust than other state-of-the-art systems for this task.

In summary, our main contribution is a method for scene layout prediction combining learning-based semantic segmentation with geometric modelling. It outperforms other state-of-the-art methods in our evaluation, while classifying frames at over 15Hz.

II. Related Work

A. Multiple images

1) Sparse 3D Reconstruction: Structure from Motion and Visual Simultaneous Localization and Mapping (vSLAM) [1] have been widely used to obtain 3D reconstructions from camera imagery. They track image features across multiple frames and build a globally consistent 3D map using

[1] The Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213, USA. shichaoy@andrew.cmu.edu,dimatura,basti@cmu.edu

optimization methods. These methods are mature but not suitable for some corridor environment because of its sparse visual and geometric features.

2) 3D Hypothesis Updating: Some works generate many candidate 3D model hypotheses and subsequently update their probability by feature tracking across multiple frames. Tsai et at. [2] build corridor models by connecting three edges with different orientations indicating left, center and right wall. Furlan et al. [3] fit planes from vSLAM point clouds and update the probability.

B. Single Image

1) Direct Depth Prediction: Many works predict pixel depth directly from a single image using machine learning method. Saxena et al.'s Make3D [4] system predicts image depth using a Markov Random Field framework on superpixels. Karsch et al. [5] propose to predict depth for single images by transferring depth from the closest matches in a large database of RGBD imagery. Recent methods using CNNs include Depth-Semantics-Normal (DSN) estimation networks [6]. These methods don't consider the scene layout constraints, and thus might yield unreasonable 3D maps.

2) Room Layout Parameterization: Lee et al. [7] detect line segments and extend them to generate fixed corridor models. Hedau et al. [8] parameterize room layouts by sampling rays from vanishing points and select the best candidate model based on the surface labels or orientation maps using structured prediction. These methods rely on the Manhattan assumption and specific image viewpoints. Moreover, most of these cannot achieve real-time performance, except for the sped-up implementation in [9].

3) Combining Geometry and Learning: The most similar work to ours is by Hoiem et al. [10]. They use region-based cues such as color, texture and edges, together with a superpixel segmentation, to classify the image into multiple geometry classes and then fold them into a 3D model. It does not assume a specific environment model and is applicable to various situations. But it is not obvious how to design effective image cues and it also cannot run in real-time. Compared to their method, our CNN segmentation with CRF refinement obtains significantly higher segmentation accuracy and the pop-up process is faster and more robust.

C. Robotics Applications

Our goal is to enable safe and robust navigation for robots in corridor environments. Some existing works use image cues to navigate such as following vanishing points direction [11], detecting wall-floor intersection landmarks [12] or building a simplified 3D model [2], as mentioned above. These methods usually work in specific environments with specific viewpoints and are not applicable for corners, object obstruction, curved corridors and poor lighting conditions.

III. Approach

A. CNN Model

CNNs are deep learning models that take advantage of of the spatial structure of 2D image data to learn rich representations with a relatively small number of parameters. They have recently revolutionized the state of the art in visual object recognition [13]. As mentioned in Section II, CNNs have also been recently adapted for the task of semantic segmentation. The Fully Convolution Network (FCN) architecture of Long et al. [14] and the DSN architecture of Eigen et al. [6] share the central ideas of using "skip connections" to integrate information across different scales and taking advantage of the convolutional nature of the CNNs to perform pixelwise labeling efficiently.

We design and implement a network based on the ideas of FCN and DSN models. Like one of the FCN variations, the network uses AlexNet [13] as its basis. We chose AlexNet rather than VGG [15] or other models as it is computationally more efficient. We also use *deconvolutional* layers at multiple scales to create a semantic segmentation. Our deconvolutional layers upsample the image in multiples of two, as in DSN. Like FCN, we perform multiscale fusion by channelwise summation, as opposed to concatenation. We found this strategy to result in more compact and efficient models, with little effect on accuracy.

Intuitively, this architecture first predicts a coarse output at a very small resolution and then progressively refines it by fusing it with finer-scale layers to provide both local and global reasoning. Since the computation is fully feed-forward and does not require any presegmentation or object proposal step, it is extremely efficient.

Our model structure is shown in Figure 2. Compared to the standard AlexNet, we decrease the strides of *convolution* and *pooling* to get larger output size. Compared to FCN, we use `conv1` and `conv4` as fusion layers instead of `conv3` and `conv4`, in order to get more diverse features and a larger output size. All the hidden layers use rectified linear units for activations. Dropout is applied to fully-connected layers `conv6` and `conv7`. The input to the network is $320\times240\times3$ RGB image and the first scale's output is $1/16$ of the input image size. We bilinearly upsample (*deconvolution*) it to $1/8$ scale and fuse it with `conv4` layer to get the second scale's prediction. Again, we upsample and fuse it with `conv1` layer to get the third $1/4$ scale output. It will be finally upsampled to the desired image size.

For training, we minimize the pixelwise cross-entropy loss:

$$L(C, C^*) = -\frac{1}{n}\sum_i C_i^* \log(C_i) \tag{1}$$

where $C_i = e^{z_i} / \sum_c e^{z_{i,c}}$ is the class softmax probability at pixel i given the CNN convolution output z.

B. CRF model

Our CNN model effectively predicts the geometric layout of the scene. However, it has some shortcomings. Due to the relatively coarse output resolution, its predictions do not always capture fine details in the image, as shown in Figure 4. Moreover, since we do not explicitly force smoothness, the CNN sometimes creates misclassified patches and discontinuities. This has an adverse effect on the line-fitting and pop-up stages of our methods (Section III-C). To fix these problems, we employ a fully connected dense CRF [16] to

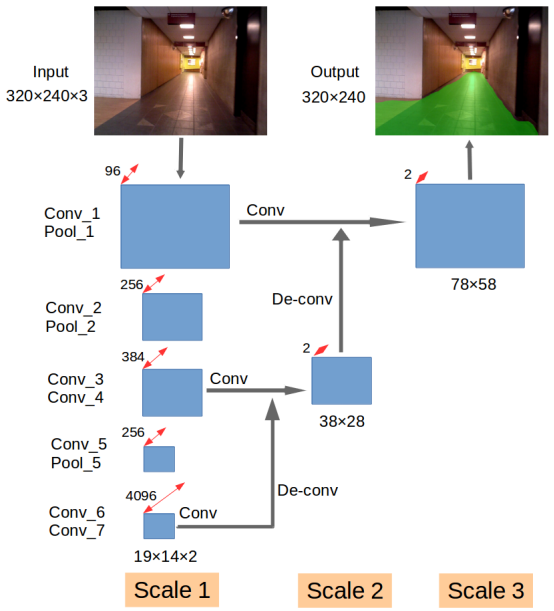

Fig. 2. Proposed CNN model containing three scales.

refine CNN segmentation. Unlike an adjacency (grid) CRF, the dense CRF models the pairwise connections between *all* the image pixels, allowing long-range reasoning. To make inference tractable, [16] proposes an efficient mean-field inference method using Gaussian edge potentials. This technique was also used in conjunction with CNN by [17], showing impressive improvement over pure CNN.

Here we briefly describe this method. Let the prediction for the n pixels be a vector $\mathbf{x} = (x_1, \cdots, x_n)$. The dense CRF assigns an energy function $E(\mathbf{x})$ to the prediction as a sum of unary potentials and binary potentials. The unary potentials are the negative log likelihood of the softmax probabilities computed by the CNN:

$$\psi_u(x_i) = -\log P(x_i) \tag{2}$$

The pairwise potentials enforce consistency between different pixels defined as a weighted sum of Gaussian kernels $\psi_b(x_i, x_j) = \mu(x_i, x_j)\lambda_{i,j}$, where $\mu(x_i, x_j) = 1$ if $x_i \neq x_j$, and $\lambda_{i,j}$ is a function of position p and color intensity I:

$$\lambda_{i,j} = w_1 \exp\left(-\frac{\|p_i - p_j\|^2}{2\sigma_\alpha^2} - \frac{\|I_i - I_j\|^2}{2\sigma_\beta^2}\right) + w_2 \exp\left(-\frac{\|p_i - p_j\|^2}{2\sigma_\gamma^2}\right) \tag{3}$$

C. Pop-Up 3D model

To create a 3D model from the segmented image, we implemented a simplified version of Hoiem's image Pop-Up [10]. We begin with a semantically labelled image, where each pixel is given a hard classification corresponding to the minimum energy in the Dense CRF (Section III-B). We then find line boundaries in the image; instead of Hough Transforms, we use Douglas-Peucker line simplification [18], as we found it to be more robust.

Fig. 3. Corridor dataset created from three sources, containing various scenes with various view points. Left to right: SUN RGBD, SUN Database, self-collected.

Finally, we project the ground plane and each wall plane to 3D space by assuming that walls are vertical relative to the ground, i.e. the soft Manhattan assumption [2]. The appendix details this step. The camera pose and calibration parameters are needed in order to perform the projection. We assume the camera is parallel to the ground with a height of h=1 m. If pose information is available from other sensors, this could be used instead.

IV. Implementation and Training

A. Training dataset

This paper focuses on corridor environments, which mobile robots operating indoors often have to traverse. Existing indoor datasets such as the NYU Depth V2 dataset [19] and the SUN RGBD dataset [20] are largely composed by images of cluttered rooms, which are of less interest for our purposes.

To our knowledge, there is no existing large image dataset specifically for corridors. Therefore, we assembled our own dataset[1] for this work. Examples images are shown in Figure 3. It contains 967 images from three sources: 349 images from the SUN RGBD [20] (category "corridor"); 327 images from SUN database [21] (category "corridor") and 291 images from self-collected video taken around the Carnegie Mellon University campus. For the SUN database images, we used annotations where available, and manually annotated an extra 250 images using LabelMe [22]. For benchmarking purposes, the dataset is split into 725 training images ($\sim 75\%$) and 242 testing images ($\sim 25\%$).

All images are resized to 320×240. Images are annotated with polygons corresponding to two classes: ground or non-ground (wall). Ceilings were not labelled as they are not important for most robot navigation purposes, but could be easily included if necessary.

B. CNN Training

We decouple CNN and CRF parameter training, assuming that the unary term in Equation 2 computed from the CNN

[1]Dataset is available at http://theairlab.org/cmu-corridor-dataset/

are fixed during CRF parameter searching. This is also the only connection between the CNN and the CRF.

For the CNN training, the parameters are learned through stochastic gradient descent to optimize the cross-entropy loss defined in Equation 1. There is no data augmentation such as random flip or rotations. We train the network in stages corresponding to the different scales. The first scale is initialized with the weights of the AlexNet model for the MIT Places205 dataset [23]. Then the first two scales are trained together. Finally, the full three scales are trained together. The batch size is set as 16, learning rate as 0.0001 and momentum as 0.9. Each training process is optimized for 300 epochs until converges. We use the Theano library [24] to compute the gradients and accelerate computation with the GeForce GTX 980 Ti GPU. It takes around five hours to train the network.

C. CRF parameter searching

CRF hyper parameters $w_1, w_2, \sigma_\alpha, \sigma_\beta, \sigma_\gamma$ in Equation 3 are searched by cross-validation on a small subset (100) images to achieve the highest mean IU metric (Intersection over Union). Default values of $w_2 = 3$ and $\sigma_\gamma = 3$ are used. The searching ranges of other parameters are: $w_1 \in \{5, 6, \dots 10\}, \sigma_\alpha \in \{2, 6, \dots 12\}, \sigma_\beta \in \{2, 4, \dots 10\}$. The maximum optimization iteration is set as 10 for all the experiments. Since CRF computation complexity grows linearly with pixel numbers, we both downsample the raw image and upsample the CNN output to 160×120 in order to speed up the prediction. We use the publicly available implementation of fully connected dense CRFs [16].

D. Pop-up

After getting binary labelled images (160×120) from the CRF, we resize them to raw camera image size in order to project them to 3D using camera calibration parameters. We extract the boundaries using the Suzuki *et al.* algorithm [25] and fit line segments using the Douglas-Peucker algorithm. For these two algorithms, we use the implementation in OpenCV.

V. Experiments and Results

We evaluate the proposed method on two datasets. The first is our mixture dataset test images (242 images) and the second is the public Michigan-Milan Indoor Dataset [3] (84 images).

We use three common semantic segmentation metrics: pixelwise accuracy, mean Intersection over Union (IU) and Frequency Weighted IU (F.W. IU), which are also adopted in [14]. For purposes of computation of these metrics, all prediction label images are resized with nearest neighbor resampling to the input image size.

A. Evaluation on our mixture data

1) Qualitative Results: We first qualitatively show the performance of each of the three main steps in our method. Examples of CNN prediction and CRF optimization are shown in Figure 4. We can see that CRF can refine the boundary, remove the extra misclassified ground regions and discontinuous hollow patches. Examples of 3D Pop-Up models are shown in Figure 5. Our algorithm works quite well in various corridor types with various lighting conditions and obstructions. To demonstrate the potential for robots' navigation, we apply our algorithm on a video where we pop up a 3D model for each frame independently. More results are provided in the supplementary materials.

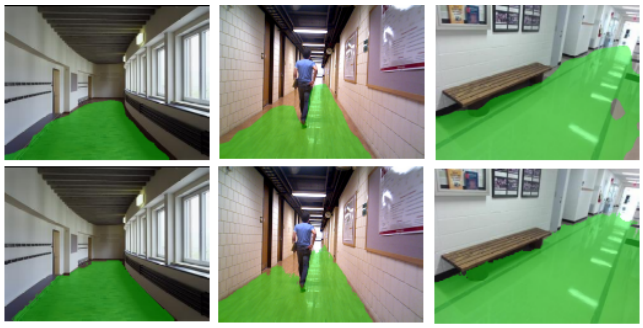

Fig. 4. CNN prediction and CRF optimization examples on our test dataset. Row 1: CNN prediction; Row 2: CRF optimization. CNN prediction captures the general location of ground. CRF further improves the spatial consistence and captures fine details.

Fig. 5. 3D pop up examples from our test dataset. The first row is the raw image and the second row is the pop-up 3D model. Since our algorithm doesn't predict the ceiling plane, we manually remove points above a constant height threshold just for visualization. Since we use assumed a camera pose, the 3D model is up to scale.

Finally, we compare with some other state-of-art methods: Lee *et al.* [7] , Hoiem *et al.* [10] and Hedau *et al.* [8]. in Figure 6. Our method works better, especially in curved, homogeneous, and poorly lit corridors.

2) Quantitative Results: We also report the quantitative results of each step in Table I. The number next to the name is image size for operations in each step. Pop-up accuracy in the last rows is the evaluation of the ground polygon, namely the re-projected label from 3D cloud to image pixels. All timings are measured on a desktop CPU (Intel i7, 4.0 GHz) and GPU (for CNN). The algorithm takes about 0.07 s which could run at 15 Hz, or at 30 Hz if CRF refinement is omitted. In our mixture dataset, the CNN prediction achieves over 95% pixel accuracy of the ground-wall segmentation and the CRF further improves mean IU by 1.5%. The CRF also has a beneficial effect when using the Pop-up model, increasing

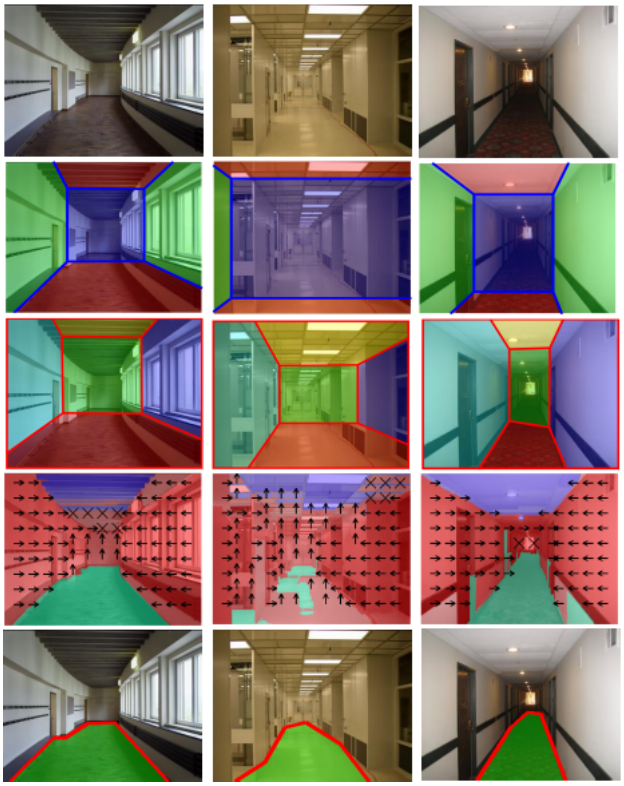

Fig. 6. Qualitative comparison of corridor scene understanding. The first row shows the input image and the outputs of four methods. Row 2: building model by Lee *et al.* [7]. Ground region is shown in red. Row 3: box layout estimates using Hedau *et al.*'s method [8]. Ground is in red. Row 4: surface label prediction by Hoiem et al.'s [10] method. Ground is in cyan. Row 5: our algorithm. Red lines are fitted line segments of ground boundaries.

mean IU by 2%.

On the other hand, the accuracy after pop up slightly decreased. This is often due to excessive simplification of segmentation boundaries when using line extraction in the pop-up process. However, we need this line simplification in order to pop up a 3D plane world for robot navigation. There is a trade-off between getting higher segmentation accuracy and building a simplified world model.

TABLE I

EVALUATION OF EACH STEP OF OUR METHOD ON OUR DATASET

Name	Pixel accuracy(%)	Mean IU (%)	F.W. IU (%)	Test time(s)
CNN 320×240	96.42	87.10	93.43	0.031
CRF 160×120	96.83	88.69	94.20	0.037
Pop-up (No CRF)	95.19	84.86	91.80	0.003
Pop-up (With CRF)	96.16	86.97	93.07	0.003

A quantitative comparison against other models is shown in Table II. Since other methods may predict the wall or ceiling part, we only evaluate "ground" and "non-ground" labels to make the metrics comparable. We use the publicly available implementation of these methods, in Matlab and C++. Since Hoiem *et al.* [10] also combine geometry modelling with learning, we retrain the surface classifier in [10] using our training dataset and show its results in the last row. The success rate is defined as the percentage of images which could generate valid 3D models by certain methods. Since our method, as well as [10], doesn't make assumptions on the specific environment model or view points, it is more robust than room layout [7], [8], which may not detect valid vanishing points or enough lines segments to form feasible room models. In all, our method performs much better than others in terms of accuracy, speed and robustness.

TABLE II

EVALUATION COMPARISON ON OUR TRAINING DATASET

Name	Pixel accu(%)	Mean IU (%)	F.W. IU (%)	Test time(s)	Success rate(%)
Our method	**96.16**	**86.97**	**93.07**	**0.071**	**100**
Lee [7]	88.31	68.38	80.49	8.403	88.3
Hedau [8]	88.04	69.60	80.65	17.45	85.8
Hoiem [10]	87.96	70.12	81.12	1.355	**100**
Hoiem*[10]	92.09	75.44	86.32	1.355	**100**

* Retrained on our mixture dataset.

B. Evaluation on Michigan-Milan Indoor dataset

In order to show the generalization abilities of our method, we directly test on this dataset without training or tuning any parameters. We evaluate three scenes in this dataset: Corridor, Entrance 1 and 2 since they are similar to corridor environments. Qualitative examples and pop-up models using the provided camera parameters are shown in Figure 7. Our method generates good 3D models even in poor lighting and occluded environments. Due to space constraints, we only report the F.W. IU evaluation result in Table III. The trend of other metrics is similar.

TABLE III

EVALUATION COMPARISON ON MICHIGAN-MILAN (F.W. IU %)

Name	Corridor	Entrance 1	Entrance 2
Our method	**96.66**	**91.17**	97.25
Lee [7]	79.99	80.54	**97.80**
Hedau [8]	87.39	90.70	92.94
Hoiem [10]	78.90	87.71	88.35

From the table, we can clearly see that our method outperforms others in the first two scenes. In the third scene, a very structured Manhattan environment with clear boundaries, the Building model collections of Lee *et al.* [7] outperforms our method by a small margin.

VI. ANALYSIS

In this section, we analyse how the CNN learns and some of our architectural choices, which contribute most to our segmentation accuracy. We also discuss some limitations of the algorithm.

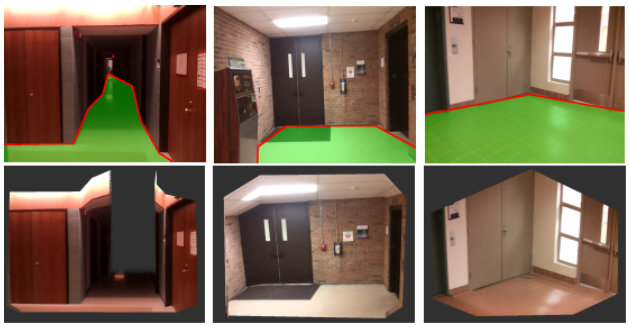

Fig. 7. Michigan-Milan dataset pop-up examples using our method. The first row is segmentation and line fitting. The second row is pop up 3D model. From left to right: Corridor, Entrance 1 and Entrance 2.

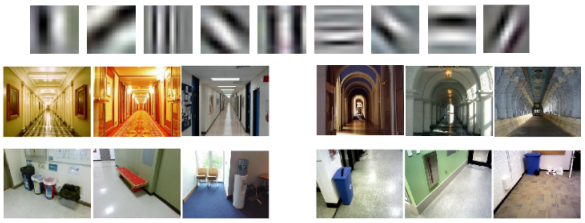

Fig. 8. CNN visualization. Top row: selected filters from first layer. Bottom two rows: top three activation images of selected four neurons in layer `pool5`. Each set represents certain corridor configurations such as straight forward corridors and left turning corridors.

A. What is the CNN learning?

We first visualize some first-layer filters of the CNN in the top row of Figure 8. The edges and corners filters in various orientations are important cues to extract and reason about the geometric structure. To visualize what the higher layers of the CNN learn, we retrieve images that maximally activate neurons in these layers. This gives us an understanding of what the neuron is "looking for" in its receptive field [26].

We only select four neurons from `pool5` layer due to space constraints, and for each neuron, we display the top three activation images as shown in the bottom two rows of Figure 8. We can see that each set of them represents certain corridor configurations such as long straight corridors, arched ceilings, and dominant left- and right- facing walls.

B. Why a multiscale CNN?

We use a three-scale CNN to capture both global and local information. Deconvolution layers increase not only the output image resolution, but also segmentation accuracy. We evaluate the contribution of different scales shown in the first two rows in Table IV. By adding the third scale, the F.W. IU increases by nearly 3%.

C. How does our model compare to other models?

We compare our model with another state-of-art multi-scale CNN model DSN by Eigen *et al.* [6]. Differences from this model are stated in Section III-A. We used our own implementation of the model, as at the time of submission there was no publicly available version. Since the DSN model has many more parameters, we train it until convergence, for 700 epochs. The result is shown in the last row of Table IV. Our model outperforms DSN model in terms of segmentation accuracy.

TABLE IV

COMPARISON OF CNN DIFFERENT SCALES AND MODELS

Name	Pixel accu(%)	Mean IU (%)	F.W. IU (%)	Output size
Scale 1+2	94.71	81.52	90.15	40×30
Scale 1+2+3	**96.16**	**86.97**	**93.07**	80×60
CNN [6]	95.58	85.15	92.25	147×109

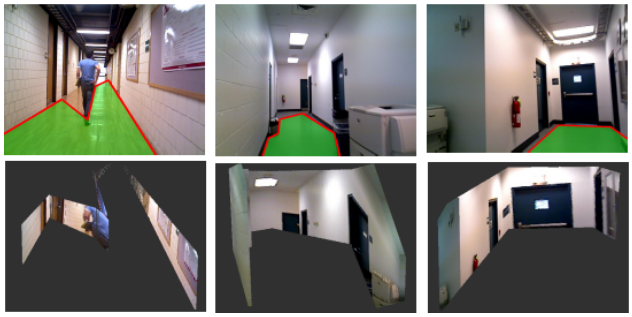

Fig. 9. Pop-up model in a cluttered environment. Left two columns: person and printer are wrongly popped as wall. Right column: the front-facing wall in left region is wrongly popped as right-facing.

D. Limitations and future work

Figure 9 shows some pop-up models in cluttered environments. Our algorithm can roughly detect the correct ground region, but the 3D model doesn't exactly match the scene geometry due to the following reasons. First, we only model ground and wall so the cluttered objects such as persons, chairs, printers, may be wrongly popped as wall planes in the left two images of Figure 9. An extra object detection step could be used to correctly pop these objects. Second, the wall's normal direction is computed based on the corresponding wall-ground boundary. If the boundary cannot be seen or detected correctly, the 3D model may not match the true geometry. For example, in the third image of Figure 9, the front facing wall in the left region is mistakenly popped as a right-facing wall. This is also a challenging problem for many existing methods [7], [10]. One possible solution is to separately model the wall and ground planes so that walls can still be popped without a visible ground plane.

VII. CONCLUSIONS

In this paper, we have presented a system for reliable real-time corridor layout understanding from a single image, which is applicable for robot navigation. The key components of our method are an efficient and accurate CNN+CRF classifier to segment indoor images into two geometric classes, and a pop-up algorithm that uses geometric constraints to create a simplified 3D model. We collected a large dataset of various corridors with nearly 1000 images, and use it to evaluate our method and other state-of-the-art algorithms

for this task. We show that our method outperforms other systems in accuracy while labeling frames at real-time rates.

In the future, we are interested in using multiple images in videos to refine the 3D model and obtain accurate state estimation. This could allow us to build a consistent 3D map. We also would like to improve the modelling of cluttered objects and wall planes to generate a more accurate and complete scene interpretation. We will also test our algorithm on real robots.

APPENDIX

A. Project segmented images to 3D model

Here, we show how to project ground and wall planes to 3D space efficiently as an extension to Section III-C. The representation of a plane is $\pi = (n^T, d)^T \in R^4$, where n, d is normal vector and distance to origin respectively. Then a 3D point $P = (X, Y, Z)^T$ lies on plane iff $n^T P + d = 0$. P is also related to its image pixel $p = (x, y, 1)^T$ by $P = \lambda K^{-1} p$, where λ is a parameter and K is calibration matrix. With these two constraints, we can solve for P from p and π:

$$P = \frac{-d}{n^T (K^{-1} p)} K^{-1} p \tag{4}$$

Using the assumed pose, ground plane equation is $\pi = (0, 1, 0, 1)$ in camera space. Then we can project all ground pixels including the boundary using Equation 4. The boundary points also lie on the walls. Using the assumption that wall is vertical to ground, we can thus compute all the plane's equation and project plane pixels using Equation 4.

ACKNOWLEDGMENTS

This research was sponsored by ONR (contract N0014-13-C-0259) and Chinese Scholarship Council. The authors would like to thank Yu Song for providing some suggestions.

REFERENCES

[1] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234. IEEE, 2007.

[2] Grace Tsai, Changhai Xu, Jingen Liu, and Benjamin Kuipers. Real-time indoor scene understanding using bayesian filtering with motion cues. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 121–128. IEEE, 2011.

[3] Axel Furlan, Stephen Miller, Domenico G Sorrenti, Li Fei-Fei, and Silvio Savarese. Free your camera: 3d indoor scene understanding from arbitrary camera motion. In *British Machine Vision Conference (BMVC)*, page 9, 2013.

[4] Ashutosh Saxena, Min Sun, and Andrew Y Ng. Make3d: Learning 3d scene structure from a single still image. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(5):824–840, 2009.

[5] Kevin Karsch, Ce Liu, and Sing Bing Kang. Depth extraction from video using non-parametric sampling. In *Computer Vision–ECCV 2012*, pages 775–788. Springer, 2012.

[6] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2650–2658, 2015.

[7] Daniel C Lee, Martial Hebert, and Takeo Kanade. Geometric reasoning for single image structure recovery. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2136–2143. IEEE, 2009.

[8] Varsha Hedau, Derek Hoiem, and David Forsyth. Recovering the spatial layout of cluttered rooms. In *Computer vision, 2009 IEEE 12th international conference on*, pages 1849–1856. IEEE, 2009.

[9] Alexander G Schwing, Tamir Hazan, Marc Pollefeys, and Raquel Urtasun. Efficient structured prediction for 3d indoor scene understanding. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2815–2822. IEEE, 2012.

[10] Derek Hoiem, Alexei A Efros, and Martial Hebert. Recovering surface layout from an image. *International Journal of Computer Vision*, 75(1):151–172, 2007.

[11] Cooper Bills, Joyce Chen, and Ashutosh Saxena. Autonomous mav flight in indoor environments using single image perspective cues. In *Robotics and automation (ICRA), 2011 IEEE international conference on*, pages 5776–5783. IEEE, 2011.

[12] Kyel Ok, Duy-Nguyen Ta, and Frank Dellaert. Vistas and wall-floor intersection features: Enabling autonomous flight in man-made environments. In *2nd Workshop on Visual Control of Mobile Robots (ViCoMoR): IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2012)*, pages 7–12, 2012.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[14] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *arXiv preprint arXiv:1411.4038*, 2014.

[15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[16] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *arXiv preprint arXiv:1210.5644*, 2012.

[17] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[18] John Edward Hershberger and Jack Snoeyink. *Speeding up the Douglas-Peucker line-simplification algorithm*. University of British Columbia, Department of Computer Science, 1992.

[19] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Computer Vision–ECCV 2012*, pages 746–760. Springer, 2012.

[20] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 567–576, 2015.

[21] Jianxiong Xiao, James Hays, Krista Ehinger, Aude Oliva, Antonio Torralba, et al. Sun database: Large-scale scene recognition from abbey to zoo. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 3485–3492. IEEE, 2010.

[22] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173, 2008.

[23] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in Neural Information Processing Systems*, pages 487–495, 2014.

[24] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a cpu and gpu math expression compiler. In *Proceedings of the Python for scientific computing conference (SciPy)*, volume 4, page 3. Austin, TX, 2010.

[25] Satoshi Suzuki et al. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985.

[26] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.