Designing Deep Networks for Surface Normal Estimation

Xiaolong Wang, David F. Fouhey, Abhinav Gupta
Robotics Institute, Carnegie Mellon University

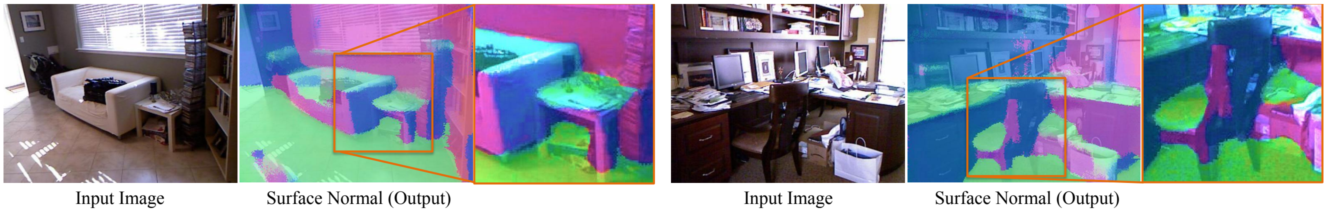


Figure 1: Given a single image, our algorithm estimates the surface normal at each pixel. Notice how our algorithm not only estimates the coarse structure but also captures fine local details. For example, on the left, the normals of the couch arm and side table legs are estimated accurately (see zoomed version). On the right, the chair surface and legs and even the top of the shopping bags are captured correctly. Normal legend: blue → X; green → Y; red → Z.

Abstract

In the past few years, convolutional neural nets (CNN) have shown incredible promise for learning visual representations. In this paper, we use CNNs for the task of predicting surface normals from a single image. But what is the right architecture? We propose to build upon the decades of hard work in 3D scene understanding to design a new CNN architecture for the task of surface normal estimation. We show that incorporating several constraints (man-made, Manhattan world) and meaningful intermediate representations (room layout, edge labels) in the architecture leads to state of the art performance on surface normal estimation. We also show that our network is quite robust and show state of the art results on other datasets as well without any fine-tuning.

1. Introduction

The last two years in computer vision have generated a lot of excitement: deep convolutional neural networks (CNNs) have broken the barriers of performance on tasks ranging from scene classification to object detection and fine-grained categorization. For instance, on object detection, performance on the standard dataset has gone up from a mAP of 33.7 to 58.5 in just two years. While CNNs have shown tremendous success on semantic tasks such as detection and categorization, their performance on other vision tasks such as 3D scene understanding and establishing correspondence has been not as extensively studied.

We want to explore the effectiveness of CNNs on the task of predicting surface orientation, or normals, from a single image. One could treat this as a per-pixel regression task and directly apply a CNN, for instance as was done for depth prediction in [8]. However, decades of research have shown that the output space of this task is governed by powerful physical constraints and researchers have exploited these constraints from the very beginning of computer vision [27] through the line-labeling era [17, 2, 19] all the way to recent investigations [14, 24, 12, 30, 40, 10].

In this paper, we demonstrate how to incorporate insights about 3D representation and reasoning into a deep learning framework for surface normal prediction. While deep networks have been particularly successful for learning image representations, we believe their design can benefit from past research in 3D scene understanding. We achieve this by developing CNNs that operate locally in a window as well as globally on the whole image; these predict not just surface normals, but also edges and cuboid room layout. A final CNN considers these predictions as well as evidence from vanishing points to yield a final prediction. Our method obtains state-of-the-art performance in surface normal estimation and shows a substantial improvement over a standard feed-forward architecture. Additionally, we show that our physical constraints enable 4.6 percentage points of our performance in the most strict evaluation metric. More importantly, our networks provide a deeper understanding of the scene in terms of not just surface normals, but also room layout and edge labels (Figure 2).

2. Related Work

The topic of 3D understanding goes back to the beginning of computer vision, starting from the first thesis, Roberts' Blocks World [27]. At the heart of this problem

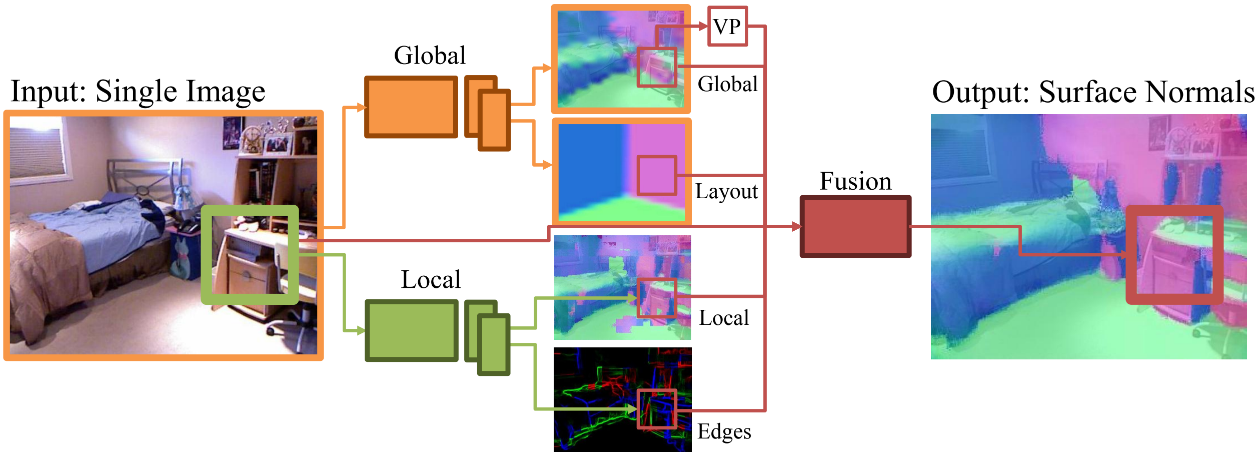


Figure 2: **An overview of our approach** to predicting surface normals of a scene from a single image. We separately learn global and local processes and use a fusion network to fuse the contradictory beliefs into a final interpretation. **Global processes:** our network predicts a coarse 20 × 20 structure and a vanishing-point-aligned box layout from a set of discrete classes. **Local processes:** our network predicts a structured local patch from a part of the image and line-labeling classes: convex-blue, concave-green, and occlusion-red. **Fusion process:** our network fuses the outputs of the two input networks, the rectified coarse normals with vanishing points (VP) and images to produce substantially better results.

are two related questions: (1) What are the right primitives for understanding? and (2) Given the local evidence, how can one obtain a global 3D scene understanding?

The problem of discovering primitives goes back to the early days of computer vision. The first primitives proposed took the form of lines [27, 36] and volumetric primitives such as geons [1], but these turned out to be too difficult to detect in natural images. Recent work has focused on using edges [25], super pixels [28] or segments [15] as primitives for reasoning. Most recently, [9] instead argued that data should determine the primitive instead of human intuition, and introduced a structured patch-based primitive; similarly [22] formulated the problem as per-pixel, using segments only as the data dictated. Unfortunately, while all of these local primitives work well on things like blinds, cabinets, and tile floors, they tend to be stymied by local ambiguities at less-textured regions.

In order to resolve ambiguities, most work turns to some form of reasoning to do top-down prediction. Most recent work is based on higher-order volumetric representations [14, 24, 30, 30] (e.g., the room should be an inside out box) or reasoning over volumes [24, 29] (e.g., two volumes should not intersect with each other) or edges [10] (e.g., via detected convex edges or occlusion boundaries) Typically, this representation is obtained via optimization over a domain-specific model and helps smooth predictions and resolve ambiguous areas, such as blank walls.

In this work, we address both threads. Instead of using primitives on manually designed features such as HoG [4], we use the data to derive a representation right from the pixels: inspired by the recent success of CNNs [23, 21] on the tasks of object detection [11, 31], segmentation [38], depth estimation [8], pose estimation [34], etc., we propose to adapt CNNs to learn representations and primitives for 3D scene understanding. Similarly, instead of hand-designing an optimizable model to reason about ambiguities, we learn a CNN to arbitrate between conflicting evidence. However, rather than abandon the insights learned in past work, we incorporate them into our design. In particular, we take into account the importance of:

Fusing global and local. We build local and global networks that handle these two forms of evidence. We combine their predictions with a fusion network that greatly outperforms either alone. Our fusion network can be viewed as a form of learned reasoning that replaces previous optimization-based attempts to reconcile evidence [12, 24, 29, 10] from conflicting sources.

Human-centric constraints. Past work has shown that the man-made nature of indoor scenes provides powerful constraints. For instance, it is common for there to be assumed three orthogonal directions in the scene (the Manhattan-world assumption [3]), as used in [14, 25, 24, 30, 29, 10, 39]. Similarly, it is common to assume that the scene is an inside-out box [14, 24, 30, 29]. Inspired by these approaches, our global network predicts a box layout in addition to coarse geometry, and we provide the vanishing-points to our fusion network. This enables the fusion network to softly apply the Manhattan or box constraint as the data dictates (e.g., on walls and floors but not chairs), and our results show that this leads to improved predictions.

Local structure. Another theme that has emerged both in the past [33, 17, 2, 19] and recently [16, 20, 10] is the reasoning between surface normals and the edges in the images. Inspired by these local constraints, we incorporate

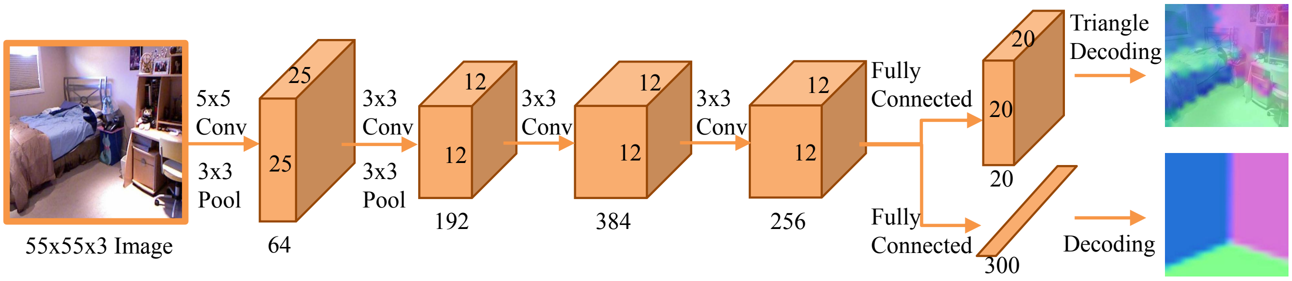


Figure 3: The architecture of our global network. Given an 55×55 image as input, it is passed though 4 convolutional layers. On the top of the last convolutional layer, the neurons are fully connected to two separate outputs: (i) global scene surface normals and (ii) room layouts.

them in learning of the local network and as an input in the fusion network. We demonstrate that the inclusion of predicted convex, concave and occlusion edges improve the performance over the simple feed-forward network.

A preliminary version of this work appeared on Arxiv [37]. At the same time, [7] introduced a stacked CNN model for surface normal estimation. Our contributions are complementary to [7] and combining both should provide further improvement.

3. Overview

This paper aims to combine the knowledge gleaned over the past decade in single-image 3D prediction with the representation-learning power of convolutional neural networks. Our overall objective is to frame the single-image 3D problem so that the structure we know is captured and convolutional networks can do what they do best – learn strong mappings from visual data to labels.

Following the lessons we described, we build a network with the following architecture (illustrated in Figure 2). We start with two networks: a global network that takes the whole image as input and predicts a coarse global interpretation (Section 4.2); and a local network that acts on local patches in a sliding-window fashion and maps them to local orientation (Section 4.3). Because the global and local processes have complementary errors, we combine their output with a fusion network that consolidates their predictions (Section 4.5). Each input network obtains strong performance by themselves, but by combining them, we obtain substantially better results, both quantitatively and qualitatively.

In addition to performing global/local fusion, we inject global human-centric constraints (including room layout, vanishing point) and local surface/edge constraints into the framework by introducing additional tasks. Our global network predicts room layout as well, and our local network predicts an edge label. Integrating these extra tasks leads to a more robust final network. We evaluate our approach in Section 5 and analyze what aspect of our designs gives what types of performance increases.

4. Method

We now describe each of the components of our method. For each, we describe their inputs, outputs, the intermediate layers, and the loss function they minimize.

4.1. Output: Regression as Classification

The outputs for the global and local networks are: surface normal for each pixel, room layout and edge labels. The edge label (convex, concave, occluding, no-edge) is a discrete output space and can be formulated as a classification problem. However, both surface normal and room layout are structured continuous output spaces (note a surface normal is on the unit sphere). Following past work such as [22], we reduce these problems to a classification problem.

Surface Normal: We use the surface normal triangular coding technique from Ladicky et al. [22] to turn normal regression into a classification problem. Specifically, we first learn a codebook with k-means and a Delaunay triangulation cover is constructed over the words. Given this codebook and triangulation, a normal can be re-written as a weighted combination of the codewords in whose triangle it lies. At training-time, we learn a softmax classifier on the codewords. At test-time, we predict a distribution over codewords; this is turned into a normal by finding the triangle in the triangulation with maximum total probability, and using the relative probabilities within that triangle as weights for reconstructing the normal.

Room Layout: Room layout is continuous structured output space. We reformulate the problem as classification by learning a codebook over box layouts. The codewords are learned with k-medoids clustering over 6000 room layouts; each codeword is a category for classification.

4.2. Global Network

The goal of this network is to capture the coarse structure, enabling the interpretation of ambiguous portions of the image which cannot be decoded by local evidence alone.

Input: Whole image rescaled to $55 \times 55 \times 3$.

Output: Given the whole image as an input, we produce two complementary global interpretations as outputs: (i) a structural estimation of surface normals for the image and

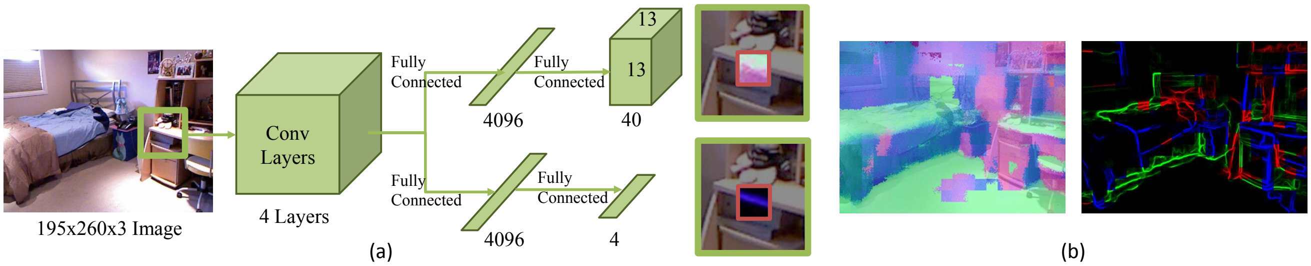


Figure 4: Our local network. (a) Given a full image, we perform sliding window on it and extract 55×55 image content as input. By forward propagation, the network produces the local surface normals and convex/concave/occlusion edge labels for the middle 13×13 image patch; (b) After sliding window, we obtain the surface normals and edge labels for the whole image. We plot the edge labels on the output of Structured Edges [6]. The colors blue, green and red represent the convex, concave and occlusion edge labels, respectively.

(ii) a cuboidal approximation of the image as introduced by [14] and used in [24, 30], among others. For surface normal estimation, the output layer is $M_t \times M_t \times K_t$ where $M_t \times M_t$ is the size of output image for surface normals and K_t is the number of classes uses in codebook. For room layout, we use simple classification over 300 categories. We use $M_t = 20, K_t = 20$.

Architecture: The global global network includes four convolutional layers; these layers are shared by the two tasks (surface normal and room layout estimation). The output of the neurons in the fourth convolutional layers are then fully connected to the two labels. To simplify the description, we denote the convolutional layer as $C(k, s)$, which indicates the there are k kernels, each having the size of $s \times s$. During convolution, we set all the strides to 1. We also denote the local response normalization layer as LRN, and the max-pooling layer as MP. The stride for pooling is 2 and we set the pooling operator size as 3×3. Then the network architecture for the convolutional layers can be described as: $C(64, 5) \rightarrow MP \rightarrow LRN \rightarrow C(192, 3) \rightarrow MP \rightarrow LRN \rightarrow C(384, 3) \rightarrow C(256, 3)$. For surface normal estimation, neurons in the fourth convolutional layers are fully connected to the output space of $M_t \times M_t \times K_t$, which is $20 \times 20 \times 20 = 8000$. For the room layout estimation, we connect the same set of neurons to the $K_l = 300$ labels. The architecture of the network is shown in Figure 3.

Loss function: We treat both tasks as classification problems. For the room layout classification, we simply employ the softmax regression to define the loss.

For surface normal estimation, we denote $F_i(I)$ as a K_t-class classification output for ith pixel on surface normal output map. We also apply softmax regression to optimize the function $F_i(I)$. Then the loss for the structural outputs of surface normals can be represented as,

$$L(I, Y) = - \sum_{i=1}^{M \times M} \sum_{k=1}^{K} (\mathbb{1}(y_i = k) \log F_{i,k}(I)), \quad (1)$$

where $F_{i,k}(I)$ represents the probability that the ith pixel should have the normal defined by the kth codeword, $\mathbb{1}(y_i = k)$ is the indicator function, $Y = \{y_i\}$ are the groundtruth labels for the surface normals, $M = M_t$ and $K = K_t$.

During training, we learn the networks with these two losses simultaneously. As we have structural outputs for surface normals and only one prediction for the room layout, we need to balance the learning rate for both losses. If σ denotes the learning rate for surface normal estimation, we set the learning rate for layout estimation to 50σ.

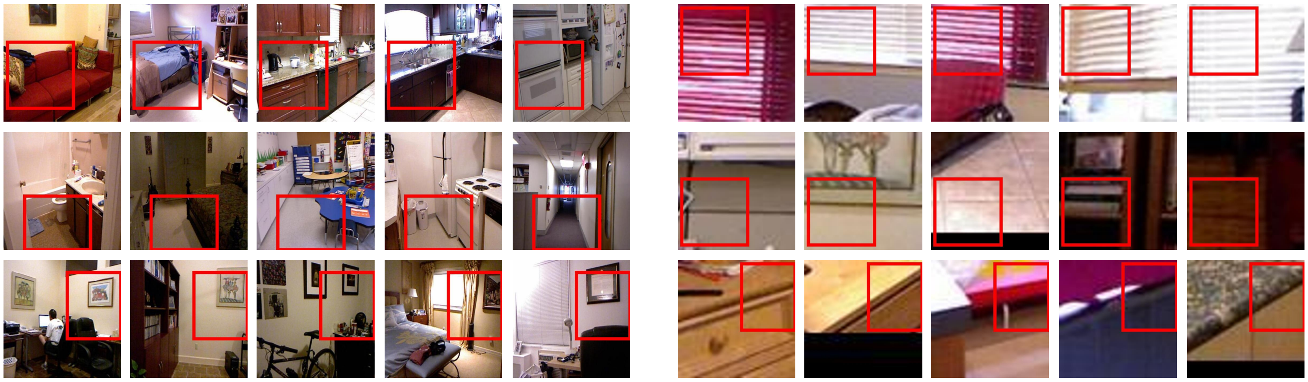
4.3. Local Network

The goal of this network is to capture local evidence at a higher resolution that might be missed by the global network. We take a sliding window approach where we extract features in a window and predict the image properties in the center of the window. This type of model has been applied successfully for generating local image interpretations in the form of normals [9, 22] and semantic edges [6].

Input: Given an image with size 195×260, we perform sliding window on it with a window of size 55×55 and stride of 13 (defined to match our output size).

Output: The local network produces two types of outputs: (i) surface normals and (ii) an edge label. Each local sliding window predicts the surface normal for $M_b \times M_b$ pixels at the center of the window. We use $M_b = 13$. As Figure 4 illustrates, our network takes a smaller part of the image as input and predicts the surface normals in the middle of the patch, thus predicting the local normals from local texture and its context. We use $K_b = 40$ codewords to define the output space. We use a larger number of codewords since we expect local network to capture finer details. For the edges, we use the classic categories of convex, concave, occlusion or not an edge. Note we just predict one edge label for 13×13 pixels. For visualization purposes, we project these edge labels onto the output of Structured Edges [6].

Architecture: The architecture of the local network includes 4 convolutional layers and 2 sets of fully connected layers. The convolutional layers are shared by the two tasks, and we use the same parameter settings mentioned in the global network. At the end of the convolutional layers, we



(a)

(b)

Figure 5: Top regions for the 4th convolutional layer units in global and local networks. The receptive field for the neurons in the 4th layer is 31×31. We use red bounding boxes to represent the regions with top responses for different units. (a) The neurons from the global network tend to capture the structure information in the global scene; (b) The neurons from the local network respond to local texture and edges.

stack two separate fully connected layers with 4096 neurons, each of which corresponds to a task. For local surface normal estimation, the output size is $M_b \times M_b \times K_b = 13 \times 13 \times 40 = 6760$; for edge labels, there are 4 outputs.

Loss Function: Both tasks are defined as classification. For edge label estimation, we apply softmax regression to define the loss. For the local surface normal estimation we apply the loss defined in Eq.1 by setting $M = M_b$ and $K = K_b$. Similar to the coarse network, we optimize these two tasks jointly during training, and the learning rate for local surface normals and edge label estimation are σ and 50σ, respectively.

4.4. Visualization

We now attempt to analyze what the global and local networks learn. Figure 5 shows the top 5 activations for the units in the fourth convolutional layer of (a) global network and (b) local network. Note that these two networks share the same structure in the convolutional layers, and the size of receptive fields of the unit is 31×31. We select representative samples for illustration. For the global network, the units capture high-level structures such as the side of beds, hallways and paintings on walls. For the local network, the units respond to local texture and edges.

4.5. Fusion Network

The goal of this network is to fuse the results of the two earlier networks and refine their results. Each approach has complementary failure modes and by fusing the two networks, we show that better results can be obtained than either by themselves. Additionally, both coarse and local networks treat every pixel independently; our fusion network can also be thought of applying a form of learned reasoning akin to [26, 35] on our outputs.

Input: As input to this fusion network, we concatenate outputs of the global and local networks with the input image. The concatenation process is as follows:

- Global Coarse Output: The output of global network is 20×20 with 20 classes. We decode the output to a 3-dimensional continuous surface normal map and upscale it to $195 \times 260 \times 3$.
- Layout: We select the room layout corresponding to the label with highest probability. The layout is a 3-channel feature map representing the surface normals in the layout. We resize it to $195 \times 260 \times 3$.
- Local Surface Normals: The output of local network in the sliding window format is $195 \times 260 \times 3$.
- Edge Labels: We obtain the 4 probabilities of edge labels per window. As the probabilities sum to 1, we do not pass the no-edge output to the fusion network. We upsample this three dimension vector to size $13 \times 13 \times 3$ for each window and obtain $195 \times 260 \times 3$ inputs.
- Vanishing Point-Aligned Coarse Output: We adjust our global output's interpretation to match vanishing points estimated by [14], yielding another feature representation with the same size.

In addition to 15 channels described above, we also concatenate the original image and therefore our final input to the deep network is $195 \times 260 \times 18$.

Output: The fusion network is also applied in a sliding window scheme on the 195×260 image. By taking the inputs with size of 55×55, we estimate the surface normals of the $M_b \times M_b$ center patch via the fusion network. Note that we use the same output window size $M_b = 13$ as in the local network, and the output space is defined by $K_b = 40$ codewords.

Architecture: The architecture of this network is 4 convolutional layers and 2 fully connected layers. The convolutional layers share the same parameter settings as the global and local networks. The last convolutional layer are

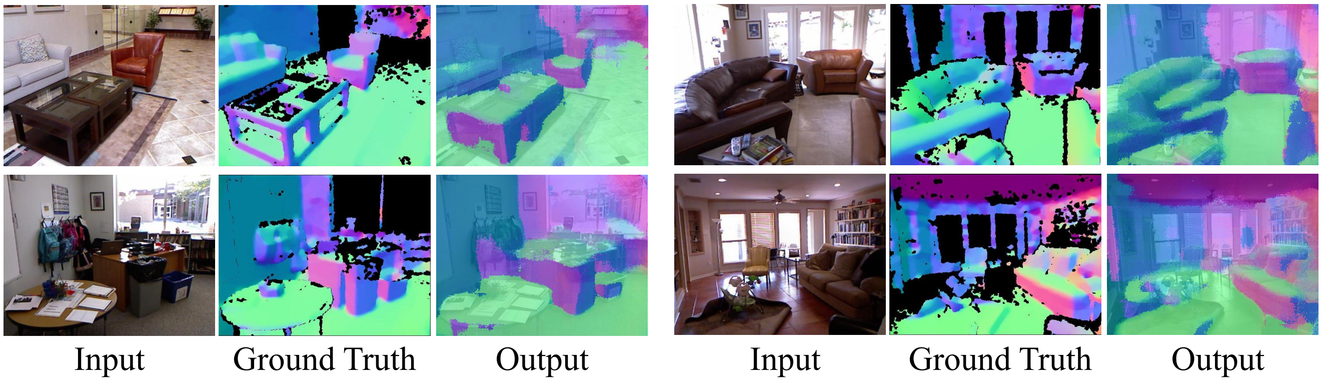


Figure 6: Qualitative results of surface normal estimation using our complete architecture. Input images are shown on the left, ground truth surface normals from Kinect are shown in middle and the predicted surface normals are shown on right. Our network not only captures the coarse layout of the room but also preserves the fine details. Notice that fine details like the top of couches and the legs of table are captured by our algorithm.

fully connected to 4096 neurons, which in turn lead to the $13 \times 13 \times 40$ outputs representing the surface normals. At testing time, we apply the fusion network on the feature maps with the stride of M_b.

Loss Function: At training time, we fix the parameters of the global and local networks and obtain the feature maps of the training data though them. The loss function is defined as Eq.1 by setting $M = M_b$ and $K = K_b$. To train the network, we apply the stochastic gradient descent with learning rate σ.

5. Experiments

We now describe our experiments. We adopt the protocols introduced in [9] and used by state-of-the-art methods on this task [9, 10, 22].

Dataset and Settings: We evaluate our method on the NYU Depth v2 dataset [32]. However, to train our models we use the corresponding raw video data for the training images. We process the video data using the provided development kit, but improve the normals with TV-denoising similar to [22]. We use the official split with 249 scenes for training and 215 scenes for testing. We extract 200K frames from the 249 scenes for training and test on the 654 images from the standard test set. We extract the room layout by fitting an inside-out box from [14] to the estimated surface normals. The edge labels are estimated using the ground-truth depth data in a procedure like [13].

During training, we fine-tune the network with stochastic gradient descent with learning rate $\sigma = 1.0 \times 10^{-6}$. Note that during joint tuning with the layouts and edges we set the learning rate as 50σ for these losses. For training our coarse networks, we augment our data by flipping, color changes and random crops. For training the local and fusion network, we rescale the training images to 195×260 and randomly sample 400K patches with size 55×55 from them.

Evaluation Criteria: Following [9], we evaluate a per-pixel error over the whole dataset, ignoring values that are unknown due to missing depth data. We summarize this population of per-pixel errors with statistics: the mean and median, as well as percent-good-pixel (PGP) metrics, or what fraction of the pixels are correct within a threshold t (for $t = 11.25, 22.5, 30$). In the interest of easy comparison, we report results on the ground-truth provided by [22]; relative performance is similar on the ground-truth of [9] and our denoised normals.

Baselines: Our primary baselines are the published state-of-the-art in surface normal prediction [9, 10, 22]. Each is state-of-the-art in at least one metric that we evaluate on. Additionally, since there have been no published CNN results, we adapt the coarse network of the Eigen et al. [8] to surface normals by using the negative dot-product as loss. This coarse network nearly matches the full system's performance on depth, and as a single feed-forward CNN with no intermediate representations or designed structures, it is a good baseline.

5.1. Experimental Results

Qualitative: First, we demonstrate our qualitative results. Figures 6 and 7 show the results of our complete architecture. Notice how our results capture the fine details of the input image. Unlike many past approaches, our algorithm is able to estimate even the legs of the tables etc. Our algorithm is able to even estimate how the surface normal changes across the couch (last column, figure 6).

Quantitative: Table 1 compares the performance of our algorithm against several baselines. As the results indicate, our approach is significantly better than all the baselines in all metrics. For many cases, our results show as much as 15% improvement over previously state of the art results.

For the sake of completeness, we also report results from a contemporary Arxiv paper [7] which uses stacked CNNs.

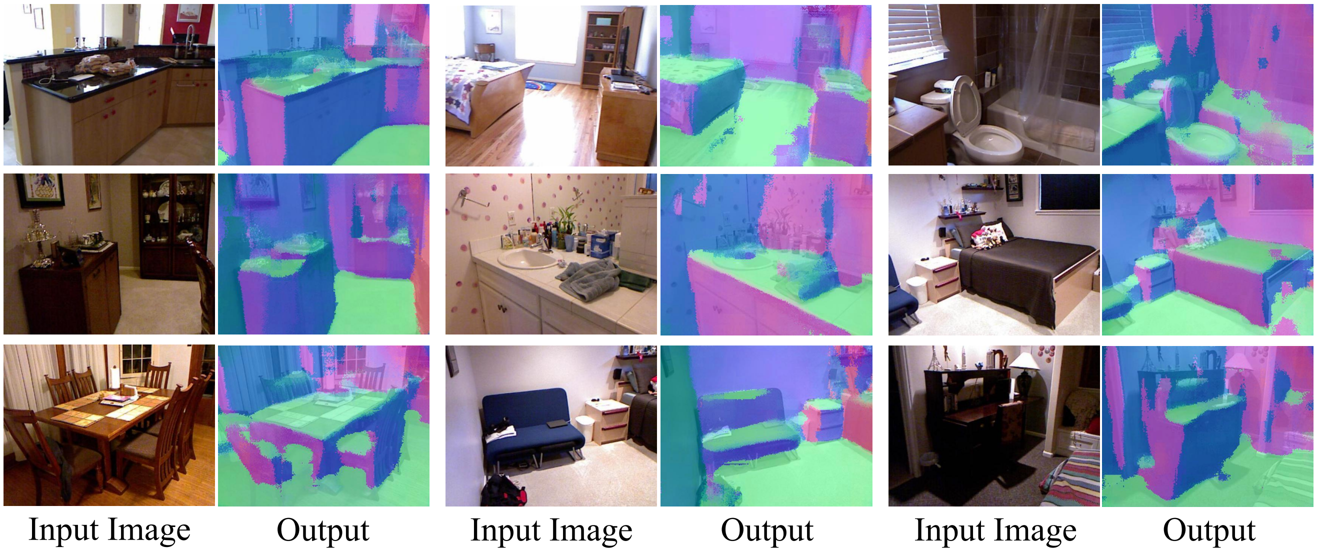


Figure 7: More qualitative results to show the performance of our algorithm. Again notice the details captured such as the top of night-stands, the counters and even the legs of the chair are captured by our algorithm.

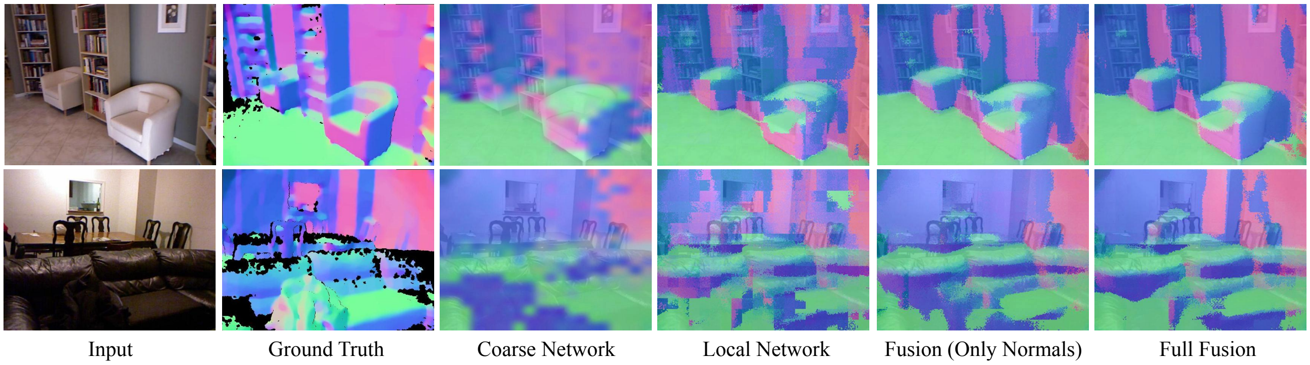


Figure 8: Qualitative Ablative Analysis: The the global and local network estimation results have complementary failure modes. By combining both normal outputs we obtain better results via fusion network. With more information feeding in, the full fusion network reasons among them and improve the performance.

The performance on all metrics are comparable, although we do not use any Imagenet [5] data for pretraining. At ambiguous pixels, their approach tends to produce an averaged and smooth output whereas our approach picks one of the interpretations. Therefore, their mean error is lower and our median error is lower.

Ablative Analysis: Next, we perform a comprehensive ablative analysis to explore which component of the network helps in improving performance. This ablative analysis helps evaluate the hypothesis that designing networks based on meaningful intermediate representations and constraints can help improve the performance.

First, we discuss some qualitative results shown in Figure 8. As seen in the figure, the global network just captures the coarse structure of the room. For example, in the top figure, it misses the vertical surface on the inner side of the couch or it misses how the vertical orientations change due to bookshelves between couches. On the other hand, a local network indeed captures those details. However, since it only observes local patches, it completely misclassifies the wall patches below the picture frame. Fusing the two networks preserves the finer details (inner side of the couch and changing vertical orientations of the wall), but still mis-

Table 1: Results on NYU v2 for per-pixel surface normal estimation, evaluated over valid pixels.

	(Lower Better)		(Higher Better)		
	Mean	Median	11.25°	22.5°	30°
Our Network	26.9	**14.8**	**42.0**	61.2	68.2
Stacked CNN [7]	**23.7**	15.5	39.2	**62.0**	**71.1**
UNFOLD [10]	35.2	17.9	40.5	54.1	58.9
Discr. [22]	33.5	23.1	27.7	49.0	58.7
3DP (MW) [9]	36.3	19.2	39.2	52.9	57.8
3DP [9]	35.3	31.2	16.4	36.6	48.2

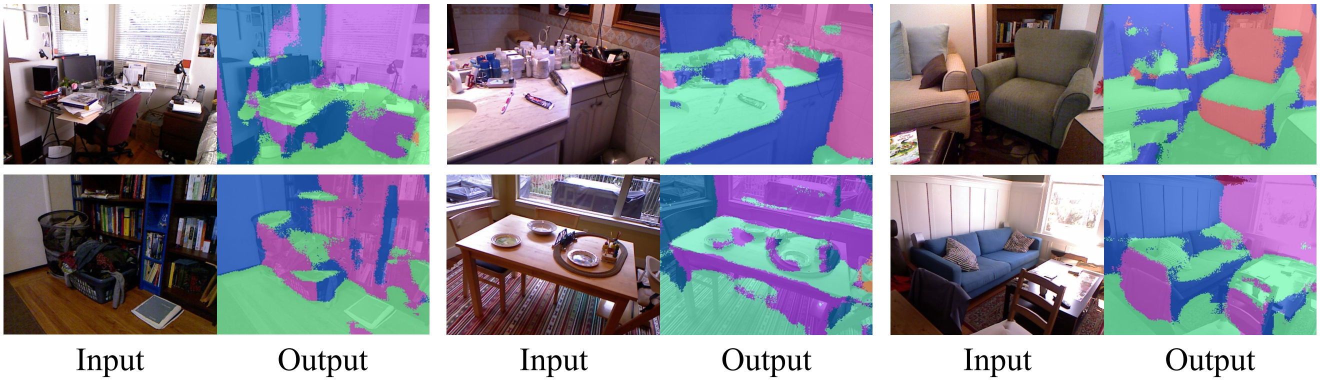


Figure 9: Results on B3DO dataset[18]. We obtain state of the art performance by applying our model trained on the NYU dataset without further fine-tuning on the B3DO dataset.

Table 2: Ablative Analysis

	Mean	Median	11.25°	22.5°	30°
Full	**26.9**	**14.8**	**42.0**	**61.2**	**68.2**
Full w/o Global	28.8	17.7	34.6	57.8	66.0
Fusion (+VP)	27.3	15.6	40.2	60.1	67.5
Fusion (+Edge)	27.8	16.4	37.5	59.4	67.4
Fusion (+Layout)	27.7	16.0	38.8	59.9	67.4
Fusion	27.9	16.6	37.4	59.2	67.1
Local	34.0	25.1	25.6	46.4	56.2
Global	30.9	20.8	31.4	52.3	60.5
Coarse CNN [8]	30.1	24.7	24.1	46.4	57.9

classifies a big patch on the wall near the picture frame. However, once the network uses the edge labels (e.g., the convex edge of the shelf and the missing edge on the wall) to improve the boundaries.

Quantitatively, we compare all the components one by one in Table 2. The fusion network, which combines the raw images, surface normal predictions from the local and global networks provides a significant boost in performance. Furthermore, adding layout (+Layout), edges (+Edge) and vanishing points (+VP) independently improve the performance of the network. By combining all of them together in the full fusion network, we obtain better results in all metrics, and a 4.6% gain in the most strict metric.

We note that adding components accounts only for the marginal gain of each component over the base system. While it is easy to improve bad systems, it is difficult to improve on a strong system like the fusion network: by itself, it would be state-of-the-art in most metrics. We therefore report fusing our constraints with just the local network (Full w/o Global), which leads to a 9% gain across all PGP metrics. This underscores the effectiveness and value of our constraints. Finally, we note that our performance is significantly better than our implementation of the coarse network of Eigen et al. [8], a single feed-forward CNN.

Table 3: B3DO

	Mean	Median	11.25°	22.5°	30°
Full	**34.5**	**20.1**	**36.7**	**52.4**	**59.2**
3DP(MW) [9]	38.0	24.5	33.6	48.5	54.5
Hedau et al. [14]	43.5	30.0	32.8	45.0	50.0
Lee et al. [25]	41.9	28.4	32.7	45.7	50.8

5.2. Berkeley B3DO Dataset

To show our model can generalize well, we apply it directly on the B3DO [18] dataset. There is significant mismatch in dataset bias between the two: NYU contains almost exclusively full scenes while the B3DO contains many close-ups. Also, since B3DO contains many scenes with down-facing views, unlike NYU, we rectify our results to detected vanishing points to compensate. We report our results of our full fusion network in Table 3 and some some qualitative results in Figure 9. Our method outperforms the baselines from [9] by a substantial margin in all metrics.

6. Conclusion

We have presented a novel CNN-based approach for surface normal estimation. By injecting insights into 3D representation, our model achieves state of the art performance. Qualitatively, our model works well and not only captures the coarse scene structure but even captures fine details such as table legs and curved surfaces of couches.

Acknowledgments: This work was partially supported by NSF IIS-1320083, ONR MURI N000141010934, Bosch Young Faculty Fellowship to AG and NDSEG fellowship to DF. This material is also based on research partially sponsored by DARPA under agreement number FA8750-14-2-0244. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government. The authors thank NVIDIA for GPU donations.

References

[1] I. Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94:115–147, 1987. 2

[2] M. Clowes. On seeing things. *Artificial Intelligence*, 2:79–116, 1971. 1, 2

[3] J. Coughlan and A. Yuille. The Manhattan world assumption: Regularities in scene statistics which enable bayesian inference. In *NIPS*, 2000. 2

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 2

[5] 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. 7

[6] P. Dollár and C. L. Zitnick. Structured forests for fast edge detection. In *ICCV*, 2013. 4

[7] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. *CoRR*, abs/1411.4734, 2014. 3, 6, 7

[8] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. *CoRR*, abs/1406.2283, 2014. 1, 2, 6, 8

[9] D. F. Fouhey, A. Gupta, and M. Hebert. Data-driven 3D primitives for single image understanding. In *ICCV*, 2013. 2, 4, 6, 7, 8

[10] D. F. Fouhey, A. Gupta, and M. Hebert. Unfolding an indoor origami world. In *ECCV*, 2014. 1, 2, 6, 7

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 2

[12] A. Gupta, A. Efros, and M. Hebert. Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *ECCV*, 2010. 1, 2

[13] S. Gupta, P. Arbelaez, and J. Malik. Perceptual organization and recognition of indoor scenes from RGB-D images. In *CVPR*, 2013. 6

[14] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In *ICCV*, 2009. 1, 2, 4, 5, 6, 8

[15] D. Hoiem, A. Efros, and M. Hebert. Geometric context from a single image. In *ICCV*, 2005. 2

[16] D. Hoiem, A. Stein, A. Efros, and M. Hebert. Recovering occlusion boundaries from a single image. In *ICCV*, 2007. 2

[17] D. Huffman. Impossible objects as nonsense sentences. *Machine Intelligence*, 8:475–492, 1971. 1, 2

[18] A. Janoch, S. Karayev, Y. Jia, J. Barron, M. Fritz, K. Saenko, and T. Darrell. A category-level 3-d object dataset: Putting the kinect to work. In *Workshop on Consumer Depth Cameras in Computer Vision (with ICCV)*, 2011. 8

[19] T. Kanade. A theory of origami world. *Artificial Intelligence*, 13(3), 1980. 1, 2

[20] K. Karsch, Z. Liao, J. Rock, J. T. Barron, and D. Hoiem. Boundary cues for 3D object shape recovery. In *CVPR*, 2013. 2

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 2

[22] L. Ladický, B. Zeisl, and M. Pollefeys. Discriminatively trained dense surface normal estimation. In *ECCV*, 2014. 2, 3, 4, 6, 7

[23] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In *NIPS*, 1990. 2

[24] D. C. Lee, A. Gupta, M. Hebert, and T. Kanade. Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. In *NIPS*, 2010. 1, 2, 4

[25] D. C. Lee, M. Hebert, and T. Kanade. Geometric reasoning for single image structure recovery. In *CVPR*, 2009. 2, 8

[26] D. Munoz, J. A. Bagnell, and M. Hebert. Stacked hierarchical labeling. In *ECCV*, 2010. 5

[27] L. Roberts. Machine perception of 3D solids. In *PhD Thesis*, 1965. 1, 2

[28] A. Saxena, S. H. Chung, and A. Y. Ng. Learning depth from single monocular images. In *NIPS*, 2005. 2

[29] A. G. Schwing, S. Fidler, M. Pollefeys, and R. Urtasun. Box In the Box: Joint 3D Layout and Object Reasoning from Single Images. In *ICCV*, 2013. 2

[30] A. G. Schwing and R. Urtasun. Efficient Exact Inference for 3D Indoor Scene Understanding. In *ECCV*, 2012. 1, 2, 4

[31] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *ICLR*, 2014. 2

[32] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from RGBD images. In *ECCV*, 2012. 6

[33] K. Sugihara. *Machine Interpretation of Line Drawings*. MIT Press, 1986. 2

[34] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *CVPR*, 2014. 2

[35] Z. Tu and X. Bai. Auto-context and its application to high-level vision tasks and 3d brain image segmentation. *TPAMI*, 32(10):1744–1757, 2010. 5

[36] D. Waltz. Understanding line drawings of scenes with shadows. In *The Psychology of Computer Vision*. McGraw-Hill, 1975. 2

[37] X. Wang, D. F. Fouhey, and A. Gupta. Designing deep networks for surface normal estimation. *CoRR*, abs/1411.4958, 2014. 3

[38] X. Wang, L. Zhang, L. Lin, Z. Liang, and W. Zuo. Joint task learning via deep neural networks with application to generic object extraction. In *NIPS*, 2014. 2

[39] Y. Zhang, S. Song, P. Tan, and J. Xiao. Panocontext: A whole-room 3d context model for panoramic scene understanding. In *ECCV*, 2014. 2

[40] Y. Zhao and S. Zhu. Scene parsing by integrating function, geometry and appearance models. In *CVPR*, 2013. 1