PlaneNet: Piece-wise Planar Reconstruction from a Single RGB Image

Chen Liu\textsuperscript{1} Jimei Yang\textsuperscript{2} Duygu Ceylan\textsuperscript{2} Ersin Yumer\textsuperscript{3} Yasutaka Furukawa\textsuperscript{4}

\textsuperscript{1}Washington University in St. Louis \quad \textsuperscript{2}Adobe Research \quad \textsuperscript{3}Argo AI \quad \textsuperscript{4}Simon Fraser University

chenliu@wustl.edu \quad \{jimyang,ceylan\}@adobe.com \quad meyumer@gmail.com \quad furukawa@sfu.ca

Figure 1: This paper proposes a deep neural architecture for piece-wise planar depthmap reconstruction from a single RGB image. From left to right, an input image, a piece-wise planar segmentation, a reconstructed depthmap, and a texture-mapped 3D model.

Abstract

This paper proposes a deep neural network (DNN) for piece-wise planar depthmap reconstruction from a single RGB image. While DNNs have brought remarkable progress to single-image depth prediction, piece-wise planar depthmap reconstruction requires a structured geometry representation, and has been a difficult task to master even for DNNs. The proposed end-to-end DNN learns to directly infer a set of plane parameters and corresponding plane segmentation masks from a single RGB image. We have generated more than 50,000 piece-wise planar depthmaps for training and testing from ScanNet, a large-scale RGBD video database. Our qualitative and quantitative evaluations demonstrate that the proposed approach outperforms baseline methods in terms of both plane segmentation and depth estimation accuracy. To the best of our knowledge, this paper presents the first end-to-end neural architecture for piece-wise planar reconstruction from a single RGB image. Code and data are available at https://github.com/art-programmer/PlaneNet.

1. Introduction

Human vision has a remarkable perceptual capability in understanding high-level scene structures. Observing a typical indoor scene (e.g., a floor, walls, and a ceiling), we can instantly parse a room into a few number of dominant planes (e.g., a floor, walls, and a ceiling), perceive major surfaces for a furniture, or recognize a horizontal surface at the table-top. Piece-wise planar geometry understanding would be a key for many applications in emerging domains such as robotics or augmented reality (AR). For instance, a robot needs to identify the extent of a floor to plan a movement, or a table-top segmentation to place objects. In AR applications, planar surface detection is becoming a fundamental building block for placing virtual objects on a desk \cite{17}, replacing floor textures, or hanging artworks on walls for interior remodeling. A fundamental problem in Computer Vision is to develop a computational algorithm that masters similar perceptual capability to enable such applications.

With the surge of deep neural networks, single image depthmap inference \cite{8, 7, 20, 35, 36} and room layout estimation \cite{21} have been active areas of research. However, to our surprise, little attention has been given to the study of piece-wise planar depthmap reconstruction, mimicking this remarkable human perception in a general form. The main challenge is that the piece-wise planar depthmap requires structured geometry representation (i.e., a set of plane parameters and their segmentation masks). In particular, we do not know the number of planes to be inferred, and the order of planes to be regressed in the output feature vector, making the task challenging even for deep neural networks.

This paper proposes a novel deep neural architecture "PlaneNet" that learns to directly produce a set of plane...
parameters and probabilistic plane segmentation masks from a single RGB image. Following a recent work on point-set generation [9], we define a loss function that is agnostic to the order of planes. We further control the number of planes by allowing probabilistic plane segmentation masks to be all 0 [33]. The network also predicts a depthmap at non-planar surfaces, whose loss is defined through the probabilistic segmentation masks to allow back-propagation. We have generated more than 50,000 piece-wise planar depthmaps from ScanNet [6] as ground-truth by fitting planes to 3D points and projecting them to images. Qualitative and quantitative evaluations show that our algorithm produces significantly better plane segmentation results than the current state-of-the-art. Furthermore, our depth prediction accuracy is on-par or even superior to the existing single image depth inference techniques that are specifically trained for this task.

2. Related work

Multi-view piece-wise planar reconstruction. Piece-wise planar depthmap reconstruction was once an active research topic in multi-view 3D reconstruction [12, 31, 13, 40]. The task is to infer a set of plane parameters and assign a plane-ID to each pixel. Most existing methods first reconstruct precise 3D points, perform plane-fitting to generate plane hypotheses, then solve a global inference problem to reconstruct a piece-wise planar depthmap. Our approach learns to directly infer plane parameters and plane segmentations from a single RGB image.

Learning based depth reconstruction. Saxena et al. [28] pioneered a learning based approach for depthmap inference from a single image. With the surge of deep neural networks, numerous CNN based approaches have been proposed [8, 23, 27]. However, most techniques simply produce an array of depth values (i.e., depthmap) without plane detection or segmentation. More recently, Wang et al. [35] enforce planarity in depth (and surface normal) predictions by inferring pixels on planar surfaces. This is the closest work to ours. However, they only produce a binary segmentation mask (i.e., if a pixel is on a planar surface or not) without plane parameters or instance-level plane segmentation.

Layout estimation. Room layout estimation also aims at predicting dominant planes in a scene (e.g., walls, floor, and ceiling). Most traditional approaches [16, 22, 14, 29, 10, 34] rely on image processing heuristics to estimate vanishing points of a scene, and aggregate low-level features by a global optimization procedure. Besides low-level features, high-level information has been utilized, such as human poses [4, 10] or semantics [4, 2]. Attempts have been made to go beyond room structure, and predict object geometry [14, 34, 2, 42]. However, the reliance on hand-crafted features makes those methods less robust, and the Manhattan World assumption limits their operating ranges. Recently, Lee et al. [21] proposed an end-to-end deep neural network, RoomNet, which simultaneously classifies a room layout type and predicts corner locations. However, their framework is not applicable to general piece-wise planar scenes.

Line analysis. Single image 3D reconstruction of line drawings date back to the 60s. The earliest attempt is probably the Robert’s system [26], which inspired many follow-up works [32, 37]. In real images, extraction of line drawings is challenging. Statistical analysis of line directions, junctions, or image segments have been used to enable 3D reconstruction for architectural scenes [25] or indoor panoramas [38]. Attributed grammar was used to parse an image into a hierarchical graph for 3D reconstruction [24]. However, these approaches require hand-crafted features, grammar specification, or algorithmic rules. Our approach is purely data-driven harnessing the power of deep neural networks.

3. PlaneNet

We build our network upon Dilated Residual Networks (DRNs) [39, 5] (See Fig. 2), which is a flexible framework for both global tasks (e.g., image classification) and pixel-wise prediction tasks (e.g., semantic segmentation). Given the high-resolution final feature maps from DRN, we compose three output branches for the three prediction tasks.

Plane parameters: For each scene, we predict a fixed number ($K$) of planar surfaces $S = \{S_1, \cdots, S_K\}$. Each surface $S_i$ is specified by the three plane parameters $P_i$ (i.e., encoding a normal and an offset). We use $D_i$ to denote a depth image, which can be inferred from the parameters $P_i$.

Non-planar depthmap: We model non-planar structures and infer its geometry as a standard depthmap. With abuse of notation, we treat it as the $(K+1)^{th}$ surface and denote the depthmap as $D_{K+1}$. This does not explain planar surfaces.

Segmentation masks: The last output is the probabilistic segmentation masks for the $K$ planes ($M_1, \cdots, M_K$) and the non-planar depthmap ($M_{K+1}$).

To summarize, the network predicts 1) plane parameters ($P_1, \cdots, P_K$), 2) a non-planar depthmap ($D_{K+1}$), and 3) probabilistic segmentation masks ($M_1, \cdots, M_{K+1}$). We now explain more details and the loss function for each task.

3.1. Plane parameter branch

The plane parameter branch starts with a global average-pooling to reduce the feature map size to $1 \times 1$ [39], followed by a fully connected layer to produce $K \times 3$ plane parameters. We do not know the number of planes as well as their order in this prediction task. By following prior works [9, 33], we predict a constant number ($K$) of planes, then allow some

*The depth value calculation requires camera intrinsic parameters, which can be estimated via vanishing point analysis, for example. In our experiments, intrinsics are given for each image through the database information.
predictions to be invalid by letting the corresponding probabilistic segmentation masks be 0. Our ground-truth generation process (See Sect. 4) produces at most 10 planes for most examples, thus we set $K = 10$ in our experiments. We define an order-agnostic loss function based on the Chamfer distance metric for the regressed plane parameters:

$$L^P = \sum_{i=1}^{K^*} \min_{j \in [1, K]} \| P^*_i - P^*_j \|_2^2.$$  

(1)

The parameterization $P_i$ is given by the 3D coordinate of the point that is closest to the camera center on the plane. $P^*_i$ is the ground-truth. $K^*$ is the number of ground-truth planes.

### 3.2. Plane segmentation branch

The branch starts with a pyramid pooling module [43], followed by a convolutional layer to produce $K + 1$ channel likelihood maps for planar and non-planar surfaces. We append a dense conditional random field (DCRF) module based on the fast inference algorithm proposed by Krahenbuhl and Koltun [19], and jointly train the DCRF module with the precedent layers as in Zheng et al. [44]. We set the number of meanfield iterations to 5 during training and to 10 during testing. Bandwidths of bilateral filters are fixed for simplicity. We use a standard softmax cross entropy loss to supervise the segmentation training:

$$L^M = \sum_{i=1}^{K+1} \sum_{p \in I} (1(M^*(p) = i) \log(1 - M_i^{(p)}))$$  

(2)

The internal summation is over the image pixels ($I$), where $M_i^{(p)}$ denotes the probability of pixel $p$ belonging to the $i^{th}$ plane. $M^*(p)$ is the ground-truth plane-id for the pixel.

### 3.3. Non-planar depth branch

The branch shares the same pyramid pooling module, followed by a convolution layer to produce a 1-channel depthmap. Instead of defining a loss specifically for non-planar regions, we found that exploiting the entire ground-truth depthmap makes the overall training more effective. Specifically, we define the loss as the sum of squared depth differences between the ground-truth and either a predicted plane or a non-planar depthmap, weighted by probabilities:

$$L^D = \sum_{i=1}^{K+1} \sum_{p \in I} (M_i^{(p)}(D_i^{(p)}(p) - D^*(p))^2)$$  

(3)

$D_i^{(p)}$ denotes the depth value at pixel $p$, while $D^*(p)$ is the ground truth depth value.

### 4. Datasets and implementation details

We have generated 51,000 ground-truth piece-wise planar depthmaps (50,000 training and 1,000 testing) from ScanNet [6], a large-scale indoor RGB-D video database. A depthmap in a single RGB-D frame contains holes and the quality deteriorates at far distances. Our approach for ground-truth generation is to directly fit planes to a consolidated mesh and project them back to individual frames, while also exploiting the associated semantic annotations.

Specifically, for each sub mesh-models of the same semantic label, we treat mesh-vertices as points and repeat extracting planes by RANSAC with replacement. The inlier distance threshold is 5 cm, and the process continues until 90% of the points are covered. We merge two (not necessarily adjacent) planes that span different semantic labels if the plane normal difference is below 20°, and if the larger plane fits the smaller one with the mean distance error below 5 cm. We project each triangle to individual frames if the three vertices are fitted by the same plane. After projecting all the triangles, we keep only the planes whose projected area is larger than 1% of an image. We discard entire frames if the ratio of pixels covered by the planes is below 50%. For training samples, we randomly choose 90% of the scenes from ScanNet, subsample every 10 frames, compute piece-wise planar depthmaps with the above procedure, then use
the final random sampling to produce 50,000 examples. The
same procedure generates 1,000 testing examples from the
remaining 10% of the scenes.

We have implemented PlaneNet using TensorFlow [1]
based on DeepLab [5]. Our system is a 101-layer
ResNet [15] with Dilated Convolution, while we have fol-
lowed a prior work and modified the first few layers to deal
with the degridding issue [39]. The final feature map of
the DRN contains 2096 channels. We use the Adam opti-
mizer [18] with the initial learning rate set to 0.0003. The
input image, the output plane segmentation masks, and the
non-planar depthmap have a resolution of 256x192. We train
our network for 50 epochs on the 50,000 training samples.

5. Experimental results

Figure 3 shows our reconstruction results for a variety of
scenes. Our end-to-end learning framework has successfully
recovered piece-wise planar and semantically meaningful
structures, such as floors, walls, table-tops, or a computer
screen, from a single RGB image. We include many more
examples in the supplementary material. We now provide
quantitative evaluations on the plane segmentation accuracy
and the depth reconstruction accuracy against the competing
baselines, followed by more analyses of our results.

5.1. Plane segmentation accuracy

Piece-wise planar reconstruction from a single RGB im-
age is a challenging problem. While existing approaches
have produced encouraging results [11, 24, 25], they are
based on hand-crafted features and algorithmic designs, and
may not match against big-data and deep neural network
(DNN) based systems. Much better baselines would then
be piece-wise planar depthmap reconstruction techniques
from 3D points [12, 31, 13, 40], where input 3D points are
either given by the ground-truth depthmaps or inferred by a
state-of-the-art DNN-based system [20].

In particular, to infer depthmaps, we have used a variant
of PlaneNet which only has the pixel-wise depthmap branch,
while following Eigen et al. [7] to change the loss. Table 1
shows that this network, PlaneNet (Depth rep.), outperforms
the current top-performers on the NYU benchmark [30].

For piece-wise planar depthmap reconstruction, we have
used the following three baselines from the literature.

* “NYU-Toolbox” is a plane extraction algorithm from the
  official NYU toolbox [30] that extracts plane hypotheses
  using RANSAC, and optimizes the plane segmentation via a
  Markov Random Field (MRF) optimization.

* Manhattan World Stereo (MWS) [12] is very similar to
  NYU-Toolbox except that MWS employs the Manhattan
  World assumption in extracting planes and exploits vanishing
  lines in the pairwise terms to improve results.

* Piecewise Planar Stereo (PPS) [31] relaxes the Manhattan
  World assumption of MWS, and uses vanishing lines to gen-
  erate better plane proposals. Please see the supplementary
doctor for more algorithmic details on the baselines.

Figure 4 shows the evaluation results on two recall met-
rics. The first metric is the percentage of correctly predicted
ground-truth planes. We consider a ground-truth plane be-
ing correctly predicted, if one of the inferred planes has 1)
more than 0.5 Intersection over Union (IOU) score and 2)
the mean depth difference over the overlapping region is less
than a threshold. We vary this threshold from 0 to 0.6m with
an increment of 0.05m to plot graphs. The second recall
metric is simply the percentage of pixels that are in such
overlapping regions where planes are correctly predicted.
The figure shows that PlaneNet is significantly better than all
the competing methods when inferred depthmaps are used.
PlaneNet is even better than some competing methods that
use ground-truth depthmaps. This demonstrates the effec-
tiveness of our approach, learning to infer piece-wise planar
structures from many examples.

Figure 5 shows qualitative comparisons against existing
methods with inferred depthmaps. PlaneNet produces signifi-
cantly better plane segmentation results, while existing meth-
ods often generate many redundant planes where depthmaps
are noisy, and fail to capture precise boundaries where the
intensity edges are weak.

5.2. Depth reconstruction accuracy

While the capability to infer a plane segmentation mask
and precise plane parameters is the key contribution of the
work, it is also interesting to compare against depth predic-
tion methods. This is to ensure that our structured depth
depth prediction does not compromise per-pixel depth prediction
accuracy. PlaneNet makes (K+1) depth value predictions
at each pixel. We pick the depth value with the maximum
probability in the segmentation mask to define our depthmap.

Depth accuracies are evaluated on the NYUv2 dataset
at 1) planar regions, 2) boundary regions, and 3) the entire
image, against three competing baselines. ‡ Eigen-VGG [7]
is a convolutional architecture to predict both depths and
surface normals. SURGE [35] is a more recent depth infer-
ence network that optimizes planarity. FCRN is the current
state-of-the-art single-image depth inference network [20] †.

Depthmaps in NYUv2 are very noisy and ground-truth
plane extraction does not work well. Thus, we fine-tune our
network using only the depth loss (3). Note that the key
factor in this training is that the network is trained to generate
a depthmap through our piece-wise planar depthmap represen-

‡ Following existing works, we choose NYUv2 to evaluate depth accu-

racy and consider only the valid 561x427 area as the entire image evaluation.

† The numbers are different from the numbers reported in [20] since
[20] evaluate on the original resolution, 640x480, and their numbers
are influenced by the issue reported at https://github.com/iro-cp/FCRN-
DepthPrediction/issues/42.
Figure 3: Piece-wise planar depthmap reconstruction results by PlaneNet. From left to right: input image, plane segmentation, depthmap reconstruction, and 3D rendering of our depthmap. In the plane segmentation results, the black color shows non-planar surface regions.
Figure 4: Plane segmentation accuracy against competing baselines that use 3D points as input [30, 12, 31]. Either ground-truth depthmaps or inferred depthmaps (by a DNN-based system) are used as their inputs. PlaneNet outperforms all the other methods that use inferred depthmaps. Surprisingly, PlaneNet is even better than many other methods that use ground-truth depthmaps.

Figure 5: Qualitative comparisons between PlaneNet and existing methods that use inferred depthmaps as the inputs. From left to right: an input image, plane segmentation results for [30], [12], [31], and PlaneNet, respectively, and the ground-truth.

tation. To further verify the effects of this representation, we have also fine-tuned our network in the standard per-pixel depthmap representation by disabling the plane parameter and the plane segmentation branches. In this version, denoted as “PlaneNet (Depth rep.)”, the entire depthmap is predicted in the \((K + 1)^{\text{th}}\) depthmap \(D_{K+1}\).

Table 1 shows the depth prediction accuracy on various metrics introduced in the prior work [8]. The left five metrics provide different error statistics such as relative difference (Rel) or rooted-mean-square-error (RMSE) on the average per-pixel depth errors. The right three metrics provide the ratio of pixels, for which the relative difference between the predicted and the ground-truth depths is below a threshold. The table demonstrates that PlaneNet outperforms the state-of-the-art of single-image depth inference techniques. As observed in prior works [35, 3], the planarity constraint makes differences in the depth prediction task, and the improvements are more significant when our piece-wise planar representation is enforced by our network.
Table 1: Depth accuracy comparisons over the NYUv2 dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Rel</th>
<th>Rel(sqr)</th>
<th>$\log_{10}$ RMSE</th>
<th>$\log_{10}$ $\text{RMSE}_{\text{sin}}$</th>
<th>$\log_{10}$ $\text{RMSE}_{\text{log}}$</th>
<th>Lower the better (LTB)</th>
<th>Higher the better (HTB)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Evaluation over planar regions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Eigen-VGG [7]</td>
<td>0.143</td>
<td>0.088</td>
<td>0.061</td>
<td>0.490</td>
<td>0.193</td>
<td>80.1</td>
<td>96.4</td>
</tr>
<tr>
<td>SURGE [35]</td>
<td>0.142</td>
<td>0.087</td>
<td>0.061</td>
<td>0.487</td>
<td>0.192</td>
<td>80.2</td>
<td>96.6</td>
</tr>
<tr>
<td>FCRN [20]</td>
<td>0.140</td>
<td>0.087</td>
<td>0.065</td>
<td>0.460</td>
<td>0.183</td>
<td>79.2</td>
<td>95.6</td>
</tr>
<tr>
<td>PlaneNet (Depth rep.)</td>
<td>0.130</td>
<td>0.080</td>
<td><strong>0.054</strong></td>
<td>0.399</td>
<td>0.156</td>
<td><strong>84.4</strong></td>
<td>96.7</td>
</tr>
<tr>
<td>PlaneNet</td>
<td>0.129</td>
<td>0.079</td>
<td><strong>0.054</strong></td>
<td>0.397</td>
<td>0.155</td>
<td><strong>84.2</strong></td>
<td>96.8</td>
</tr>
<tr>
<td><strong>Evaluation over edge areas</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Eigen-VGG [7]</td>
<td>0.165</td>
<td>0.137</td>
<td>0.073</td>
<td>0.727</td>
<td>0.228</td>
<td>72.9</td>
<td>74.3</td>
</tr>
<tr>
<td>SURGE [35]</td>
<td>0.162</td>
<td>0.133</td>
<td>0.071</td>
<td>0.697</td>
<td>0.221</td>
<td>74.7</td>
<td>94.7</td>
</tr>
<tr>
<td>FCRN [20]</td>
<td>0.154</td>
<td>0.111</td>
<td>0.073</td>
<td>0.548</td>
<td>0.208</td>
<td>74.7</td>
<td>94.1</td>
</tr>
<tr>
<td>PlaneNet (Depth rep.)</td>
<td>0.145</td>
<td>0.099</td>
<td>0.061</td>
<td>0.480</td>
<td>0.179</td>
<td><strong>80.9</strong></td>
<td>95.9</td>
</tr>
<tr>
<td>PlaneNet</td>
<td>0.145</td>
<td>0.099</td>
<td>0.061</td>
<td>0.479</td>
<td>0.178</td>
<td><strong>80.7</strong></td>
<td><strong>96.1</strong></td>
</tr>
<tr>
<td><strong>Evaluation over the entire image</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Eigen-VGG [7]</td>
<td>0.158</td>
<td>0.121</td>
<td>0.067</td>
<td>0.639</td>
<td>0.215</td>
<td>77.1</td>
<td>95.0</td>
</tr>
<tr>
<td>SURGE [35]</td>
<td>0.156</td>
<td>0.118</td>
<td>0.067</td>
<td>0.643</td>
<td>0.214</td>
<td>76.8</td>
<td>95.1</td>
</tr>
<tr>
<td>FCRN [20]</td>
<td>0.152</td>
<td>0.119</td>
<td>0.072</td>
<td>0.581</td>
<td>0.207</td>
<td>75.6</td>
<td>93.9</td>
</tr>
<tr>
<td>PlaneNet (Depth rep.)</td>
<td>0.143</td>
<td>0.107</td>
<td>0.060</td>
<td>0.518</td>
<td>0.180</td>
<td><strong>81.3</strong></td>
<td>95.5</td>
</tr>
<tr>
<td>PlaneNet</td>
<td>0.142</td>
<td>0.107</td>
<td>0.060</td>
<td>0.514</td>
<td>0.179</td>
<td><strong>81.2</strong></td>
<td><strong>95.7</strong></td>
</tr>
</tbody>
</table>

5.3. Plane ordering consistency

The ordering ambiguity is a challenge for piece-wise depthmap inference. We found that PlaneNet automatically learns a consistent ordering without supervision, for example, the floor is always regressed as the second plane. In Fig. 3, colors in the plane segmentation results are defined by the order of the planes in the network output. Although the ordering loses consistency for small objects or extreme camera angles, major common surfaces such as the floor and walls have a consistent ordering in most cases.

We have exploited this property and implemented a simple room layout estimation algorithm. More specifically, we look at reconstruction examples and manually select the entries of planes that correspond to the ceiling, the floor, and the left/middle/right walls. For each possible room layout configuration [21], (e.g., a configuration with a floor, a left wall, and a middle wall visible), we construct a 3D concave hull based on the plane parameters and project it back to the image to generate a room-layout. We measure the score of the configuration by the number of pixels, where the constructed room layout and the inferred plane segmentation (determined by the winner-takes-all) agree. We pick the constructed room layout with the best score as our prediction. Figure 6 shows that our algorithm is able to generate reasonable room layout estimations even when the scene is cluttered and contain many occluding objects. Table 2 shows the quantitative evaluations on the NYUv2 303 dataset [41], where our method is comparable to existing techniques which are designed specifically for this task. §

5.4. Failure modes

While achieving promising results on most images, PlaneNet has some failure modes as shown in Fig. 7. In the first example, PlaneNet generates two nearly co-planar vertical surfaces in the low-light region below the sink. In the second example, it cannot distinguish a white object on the floor from a white wall. In the third example, it misses a column structure on a wall due to the presence of object§

Table 2: Room layout estimations. Quantitative evaluations against the top-performers over the NYUv2 303 dataset.

<table>
<thead>
<tr>
<th>Input</th>
<th>Layout error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schwing et al. [29]</td>
<td>RGB</td>
</tr>
<tr>
<td>Zhang et al. [41]</td>
<td>RGB</td>
</tr>
<tr>
<td>Zhang et al. [41]</td>
<td>RGB+D</td>
</tr>
<tr>
<td>RoomNet [21]</td>
<td>RGB</td>
</tr>
<tr>
<td>PlaneNet</td>
<td>RGB</td>
</tr>
</tbody>
</table>

§RoomNet paper [21] does not provide code or evaluation numbers for the NYUv2 benchmark. We have implemented their system using Torch7 and trained on LSUN dataset as described in their paper.
Figure 6: Room layout estimations. We have exploited the ordering consistency in the predicted planes to infer room layouts.

Figure 7: Typical failure modes occur in the absence of enough image texture cues or at the presence of small objects and clutter.

Figure 8: Texture editing applications. From top to bottom, an input image, a plane segmentation result, and an edited image.

We first extract Manhattan directions by using the predicted plane normals through a standard voting scheme [12]. Given a piece-wise planar region, we define an axis of its UV coordinate by the Manhattan direction that is the most parallel to the plane, while the other axis is simply the cross product of the first axis and the plane normal. Given a UV coordinate, we insert a new texture by alpha-blending or completely replace a texture with a new one. Please see the supplementary material and the video for more AR application examples.

6. Applications

Structured geometry reconstruction is important for many application in Augmented Reality. We demonstrate two image editing applications enabled by our piece-wise planar representation: texture insertion and replacement (see Fig. 8).

6. Applications

Structured geometry reconstruction is important for many application in Augmented Reality. We demonstrate two image editing applications enabled by our piece-wise planar representation: texture insertion and replacement (see Fig. 8).

7. Conclusion and future work

This paper proposes PlaneNet, the first deep neural architecture for piece-wise planar depthmap reconstruction from a single RGB image. PlaneNet learns to directly infer a set of plane parameters and their probabilistic segmentation masks. The proposed approach significantly outperforms competing baselines in the plane segmentation task. It also advances the state-of-the-art in the single image depth prediction task. An interesting future direction is to go beyond the depthmap framework and tackle structured geometry prediction problems in a full 3D space.

8. Acknowledgement

This research is partially supported by National Science Foundation under grant IIS 1540012 and IIS 1618685, Google Faculty Research Award, and Adobe gift fund. We thank Nvidia for a generous GPU donation.
References


