VPLNet: Deep Single View Normal Estimation with Vanishing Points and Lines

Rui Wang[1], David Geraghty[2], Kevin Matzen[2], Richard Szeliski[2], and Jan-Michael Frahm[1,2]

[1]University of North Carolina at Chapel Hill
[2]Facebook, Inc.

Abstract

We present a novel single-view surface normal estimation method that combines traditional line and vanishing point analysis with a deep learning approach. Starting from a color image and a Manhattan line map, we use a deep neural network to regress on a dense normal map, and a dense Manhattan label map that identifies planar regions aligned with the Manhattan directions. We fuse the normal map and label map in a fully differentiable manner to produce a refined normal map as final output. To do so, we softly decompose the output into a Manhattan part and a non-Manhattan part. The Manhattan part is treated by discrete classification and vanishing points, while the non-Manhattan part is learned by direct supervision.

Our method achieves state-of-the-art results on standard single-view normal estimation benchmarks. More importantly, we show that by using vanishing points and lines, our method has better generalization ability than existing works. In addition, we demonstrate how our surface normal network can improve the performance of depth estimation networks, both quantitatively and qualitatively, in particular, in 3D reconstructions of walls and other flat surfaces.

1. Introduction

Single-view surface normal estimation has been extensively studied in the past few decades. A traditional approach for solving this problem is based on vanishing point and line estimation [14, 22, 16]. However, this approach has certain limitations, for example: 1) large textureless surfaces, common in indoor scenes, are challenging, 2) due to degeneracies, it is common to find lines in the image that are compatible with two distinct vanishing points . Recently, researchers have focused on deep learning methods for tasks in single-view geometry estimation such as surface normals, depth, room layout and canonical frames [2, 5, 38, 12]. These methods typically produce dense outputs and work well for featureless regions, where most traditional methods fail. However, deep learning-based methods tend to be data-hungry and to not generalize well to unseen datasets. In contrast, traditional geometric methods do not suffer from the generalization problem. In this paper, we demonstrate that the benefits of deep learning-based methods and traditional vision methods are complementary, and that combining them produces significant improvements.

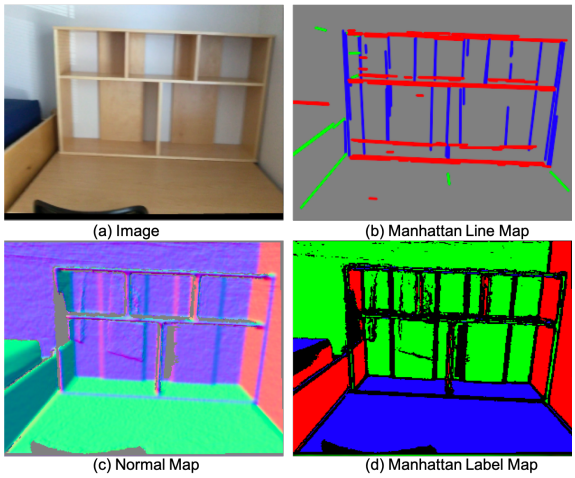


Figure 1: Example inputs and outputs of our method. The first row shows the input image and the Manhattan line map. The second row shows the output normal and Manhattan label maps. The colors in the Manhattan line and label maps represent the unsigned Manhattan directions: blue corresponds to vertical, while red and green correspond to the two orthogonal horizontal directions.

In this paper, we develop a single-view normal estimation framework that combines line and vanishing point analysis with a deep neural network in both the training and prediction phases. We softly decompose the normal map into a Manhattan part (treated by discrete classification and vanishing points) and a non-Manhattan part (learned directly). Our method outperforms the state-of-the-art on typical benchmarks and generalizes well to unseen datasets.

In more details, we use a line detection and vanishing point estimation method to compute the dominant vanish-

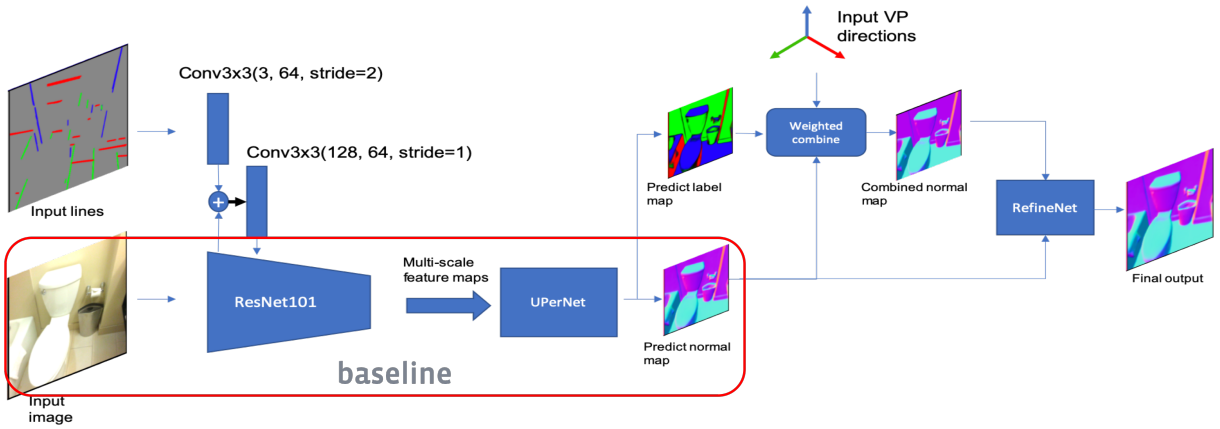


Figure 2: The full pipeline of our proposed model. The network takes an RGB image and a Manhattan line map as input, and produces a Manhattan label map and a raw normal prediction as intermediate output. These intermediate outputs are then combined with the analytically computed dominant vanishing points to generate a "combined normal map". This operation is differentiable. Finally, the combined and raw normal maps are fused through a refinement network to produce the final normal prediction.

ing points and "Manhattan lines", i.e. lines in the Manhattan directions, for each image. As shown in Figure 2, we provide a Manhattan line map and color image together with vanishing points as input to the network and output a raw normal estimate, a Manhattan label map, and finally a refined normal map. The ground truth Manhattan label map is generated using both the ground truth normal map and the computed vanishing points, which we detail in Section 3.2. The estimation of the Manhattan label map is equivalent to normal estimation in the coordinate system aligned to the Manhattan frame. Since the Manhattan label map only contains normals in the dominant vanishing point directions, we also perform raw normal estimation, which fills in the normals for surface points that are not in the dominant directions. The refined normal map is generated by fusing the Manhattan label map with the raw normal estimation. Figure 1 shows an example of the inputs and outputs of our proposed method. We use the ScanNet [4] dataset for training and validation, and we use the NYUD-v2 [23] and Replica [29] datasets to show the generalization ability of our proposed method.

In summary, our main contributions are:

- We introduce the idea of a Manhattan label map, which only represents the dominant directions but is easier to learn than a dense normal map.
- We combine a traditional vision method with a deep neural network by providing an analytically computed Manhattan line map and label map as input and as ground truth training label for the network, respectively, which achieves higher accuracy and better generalization.
- We use multi-task learning and a refinement network to jointly predict the Manhattan label map and a raw normal map and fuse them in a fully differentiable manner.

2. Related Works

Inferring 3D geometry from a single image is a long-standing task in computer vision. By utilizing the property of parallel lines in perspective geometry, vanishing point detection algorithms [3, 1, 28, 26] can be used to estimate plane normals. However, this typically succeeds only in regions containing lines that point in two distinct 3D directions. Shape-from-shading algorithms [37, 27] solve the single view 3D reconstruction as an inverse problem, with a set of assumptions that often limits their applicability.

Convolutional neural networks (CNNs) have begun to produce better results than traditional methods on single view geometry estimation tasks. Not relying on handcrafted features or heuristic assumptions, they can work on more complex environments. For tasks such as depth and surface normal estimation, they produce dense output. Works by Eigen *et al.* [6] and Liu *et al.* [21] proposed supervised training pipelines for single view depth estimation. Godard *et al.* [9] and Garg *et al.* [8] proposed unsupervised single view depth estimation pipelines using stereo pairs. Wang *et al.* [31] proposed using RNNs to leverage multiple consecutive video frames for depth estimation and demonstrated both supervised and unsupervised training.

It has been shown by Zamir *et al.* [36] that the surface normal is the visual representation that has the most direct connection to the majority of vision tasks in deep learning. Marr revisited [2] proposed using synthetic data to augment the normal estimation training. Wang *et al.* [32] proposed estimating surface normals, room layout and edges at both local and global scales, and fusing them together to produce the final normal estimation. GeoNet [24] and Wang *et al.*, [30] proposed estimating depth and normal maps simultaneously. Wei *et al.* [35] recently proposed using "virtual normals" to regularize and improve the depth estimation.

They randomly sample triplets of 3D points from their predicted geometry; each triplet forms a virtual plane whose normal is a virtual normal. They showed that the virtual normal is more robust to noise than the surface normal.

Normal estimation in different coordinate systems has also been studied. FrameNet [12] proposed jointly estimating local canonical frames and their projections, together with the normal map. This joint estimation has been shown to improve results. Xian *et al.* [33] estimate the 2DoF camera orientation by computing the alignment between the local camera coordinate system, and a global reference coordinate system which is aligned with gravity. The two coordinate systems are represented by the two surface geometries estimated from a deep neural network. We also leverage the normal estimation in different frames to impose more geometric constraints. We provide analytically computed Manhattan line map as input to help our normal estimation in Manhattan coordinates. In addition, since the Manhattan directions (dominant vanishing points) are analytically computed from lines, we turn the normal estimation in Manhattan coordinates into a classification problem to further reduce its difficulty.

Besides per-pixel normal estimation, piece-wise planar surface estimation is another popular way for surface normal estimation under the assumption that our world is piecewise planar. PlaneNet [20] represents each plane as a normal plus an offset. It uses a CNN to predict a segmentation into planes as well as the parameters for each plane segment. Whereas it assumes a fixed number of planes, PlaneRCNN [19] eliminates that constraint. It uses Mask-RCNN [11] to segment out an arbitrary number of planes and then estimates parameters for each plane. Compared to per-pixel dense estimation, piece-wise planar methods produce more regularized predictions but have the drawback that an error in an estimated parameter will result in an accumulated error in the whole plane.

LayoutNet [38] uses Manhattan lines as input to help the network predicting room layouts. Li *et al.* [18] used multi-view stereo on a large collection of internet photos to create a large dataset for training CNNs for depth estimation. Both papers show the usefulness of traditional geometric reasoning based methods in deep neural network training. In our work, in both the training and prediction phases, we combine the vanishing point and line method with the deep learning method. Our experiments demonstrate that this combination leverages the advantage of both methods and produces better results than the state of the art.

3. Method

In this section, we introduce our single-view normal estimation framework. First, we briefly introduce the vanishing point and line detection method, and our method for generating the Manhattan label map. Then, we discuss the overall network architecture. Finally, we elaborate on the energy terms and training details.

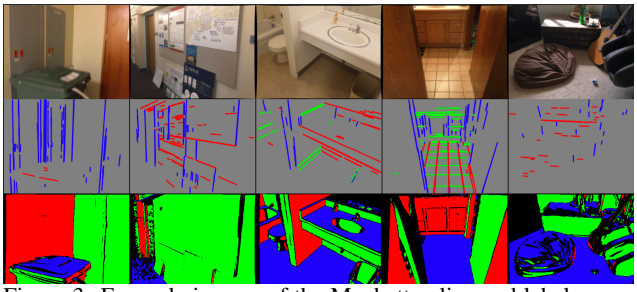


Figure 3: Example images of the Manhattan line and label maps. For better visualization, the parallel planes with opposite directions are assigned to the same color in the Manhattan label map. The actual Manhattan label map is a seven-class label map ($\pm$ 3 Manhattan directions plus one non-Manhattan direction).

3.1. Line and vanishing point detection

Under a pinhole camera model, parallel lines in 3D space project to converging lines in the image plane. The common point of intersection in the image plane is called the vanishing point (VP). The steps for detecting VPs are 1) detect line segments, 2) cluster the lines into groups with the assumption that a cluster will share a common vanishing point, 3) find the three dominant pair-wise orthogonal vanishing points.

Line segments are detected using the LSD line detector [10]. We use the Expectation-Maximization approach of [13] to fit vanishing points. Before running EM, the vanishing points are initialized as follows: in our training data, the vertical vanishing point is initialized from a gravity-aligned camera pose. We then form the corresponding horizon line in the projective image plane, and divide it into equal-angle bins. To initialize horizontal vanishing points, we intersect image line segments with the horizon line, and consider peaks in the resulting histogram. Between EM iterations, we purge candidate vanishing points with low evidence, and merge vanishing points that become sufficiently close.

3.2. Manhattan line and label map

After the dominant vanishing points have been detected, we classify the corresponding line segments to form a color-coded Manhattan line map. We use blue for line segments in the vertical vanishing point direction, and green and red for line segments in the two orthogonal horizontal vanishing point directions. We always render the lines in a fixed order based on the Manhattan directions. Row 2 of Figure 3 shows an example of the Manhattan line map, which is used as an input to our normal estimation network.

Given the classified line segments, one can determine the plane normals near the intersection of two groups of lines. However, it is non-trivial to determine whether the

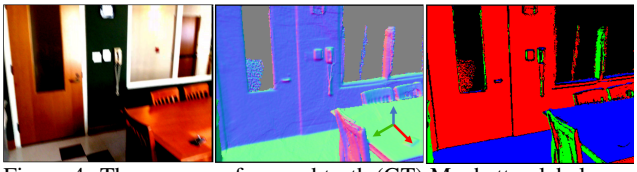


Figure 4: The process of ground truth (GT) Manhattan label map generation. Middle: a GT normal map in which the coordinates in the lower right corner correspond to the three dominant vanishing points. Right: the GT Manhattan label map generated by classifying each surface normal in the GT normal map into one of the dominant vanishing point directions based on a threshold. For better visualization, the parallel planes with opposite directions are assigned to the same color in the Manhattan label map.

two groups are coplanar and to select the appropriate region near the line intersections where normals can be estimated. Therefore, we introduce a discrete classification problem where each pixel is classified into seven classes: six classes for surfaces aligned to the signed Manhattan directions, and one extra class for non-Manhattan surfaces. For each training image we create a seven-class one-hot encoded Manhattan label map M^{gt}, using the given dominant vanishing points $\mathbf{v}$ and ground truth normal map $\mathbf{n}_{gt}$.

Figure 4 shows an example of generating M^{gt} given $\mathbf{n}_{gt}$ and $\mathbf{v}$. The third row in Figure 3 shows more examples of the Manhattan label maps. In the visualizations, we ignore sign differences in the label map. As above, the Manhattan label map is a seven-class label map with each label specifying a dominant vanishing point direction, or a non-Manhattan direction. Given the ground truth normal map and dominant vanishing points, we classify each normal direction into one of the dominant vanishing point directions if it is within a certain angle; otherwise it is classified as an unknown (non-Manhattan) direction. The one-hot encoding of the labels if given by:

$$M_c^{gt}(i) = \begin{cases} 1 & \text{if } \angle(\mathbf{n}_{gt}(i), \mathbf{v}_c) < T \\ 0 & \text{otherwise} \end{cases}$$
$$M_u^{gt}(i) = 1 - \sum_{c=1}^{C} M_c^{gt}(i), i = 1 \dots N, \quad (1)$$

where $C = 6$ is the number of signed Manhattan directions, c runs over these directions, $\mathbf{v}_c$ is the signed Manhattan direction associated to label c, $u = 7$ corresponds to an unknown (non-Manhattan) direction, $\angle(\cdot, \cdot)$ is the angle between two vectors, T is a selected angle threshold, N is the number of pixels in the image, and i runs over all pixels. We are using consistent labeling of the directions, e.g. $c = 0$ is always corresponding to the up direction.

The advantages of converting the normal map representation into the Manhattan label map representation are that 1) the problem of regressing arbitrary normal direction is converted to a classification problem; 2) the color coding

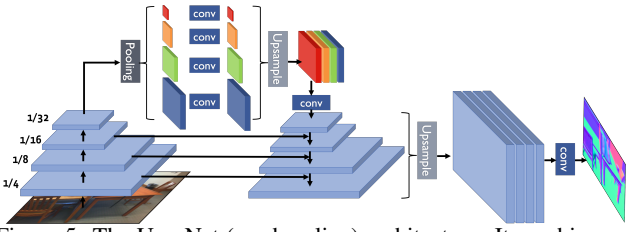


Figure 5: The UperNet (our baseline) architecture. It combines a Feature Pyramid Network (FPN) with a Pyramid Pooling Module (PPM) attached at the last layer of the encoder of the FPN.

of the Manhattan label map is complementary to the Manhattan line map and thus can directly take advantage of the input Manhattan line map, for example, Manhattan normal directions should be orthogonal to nearby Manhattan line directions. In the next subsection, we introduce our network architecture for the Manhattan label map and normal map estimation, given the analytically computed dominant vanishing points and Manhattan line map.

3.3. Manhattan Label Map and Normal Estimation

As mentioned in Section 3.2, the Manhattan label map is a seven-class label map. It is natural to adopt a semantic segmentation network architecture for the task of learning this labeling. We adopt the UPerNet [34] architecture, which combines a Feature Pyramid Network (FPN) and a Pyramid Pooling Module (PPM). It has large receptive fields while remaining efficient compared to the UNet [25] or DORN [7] architectures. A more detailed architecture is shown in Figure 5. The encoder is a ResNet101 backbone, and the decoder is a FPN. Between the encoder and decoder is a PPM that further increases the receptive field of the network. Please refer to [34] for more details.

Baseline model. We first trained a baseline model that takes a single image as input and outputs a dense normal map. We initialize all the weights in our baseline model, except the last layer, using the pretrained UPerNet weights from a semantic segmentation task. Then all layers are fine-tuned (except for the last layer, which is freshly trained) on the normal prediction task. From our experiments, our baseline model already outperforms FrameNet.

vp-line model. Our full model is shown in Figure 2; we name it the vp-line model. The input to the vp-line model is an RGB image together with analytically computed vanishing points and Manhattan line map. The 3-channel Manhattan line map contains color-coded lines, where blue corresponds to the vertical direction and the other two colors correspond to two orthogonal horizontal dominant vanishing point directions. The Manhattan line map provides useful information for the Manhattan label map estimation. The RGB image and the Manhattan line map are processed separately using two convolutional blocks and then concatenated for use in the third convolutional block. The decoder of the

vp-line model is the same as the baseline model except for the last convolutional layer, where the vp-line model has a ten-channel output layer instead of three. The seven additional channels correspond to the predicted Manhattan label map: after the decoder, for each pixel i, we have a raw predicted surface normal $\mathbf{n}^{\text{raw}}(i)$, and a set of softmax probabilities $\mathbf{p}(i) = (p_1(i), \dots, p_7(i))$ for the labels described above. The Manhattan label map is similar to a dense version of the Manhattan line map. The Manhattan label map estimation can be interpreted as semantic segmentation; it is also equivalent to normal estimation in an aligned coordinate system since it classifies surface normals into Manhattan directions.

Next, we convert the Manhattan label map into a surface normal map in camera coordinates. As above, the predicted Manhattan label map contains probabilities for each of the six Manhattan directions, and a non-Manhattan direction. An easy way to convert the Manhattan label map to a normal map is to directly assign a Manhattan direction to each pixel in the label map based on the highest probability. However, such an assignment is non-differentiable and thus cannot be integrated into the network training process. Therefore, instead of direct assignment, we make full use of the softmax probabilities and the raw predicted normals, to produce a "combined normal" $\mathbf{n}^{\text{comb}}(i)$ at each pixel i by taking:

$$\mathbf{n}^{\text{comb}}(i) = \langle \sum_{c=1}^{6} p_c(i)\mathbf{v}_c + p_7(i)\mathbf{n}^{\text{raw}}(i) \rangle, i = 1 \dots N, \tag{2}$$

where each $\mathbf{v}_c$ is the (constant) signed Manhattan direction corresponding to label c, and $\langle \cdot \rangle$ is a normalization operation. For pixels strongly associated to a Manhattan direction c, the network can leverage the provided direction $\mathbf{v}_c$ by increasing the weight $p_c(i)$. Many object surfaces are irregularly shaped, so we cannot expect only the dominant Manhattan directions to represent the whole scene. For pixels not lying on a Manhattan aligned surface, the weight $p_7(i)$ should be large, and thus $\mathbf{n}^{\text{raw}}(i)$ should dominate in the sum.

At this point, our vp-line model has two normal map predictions: $\mathbf{n}^{\text{raw}}$ and $\mathbf{n}^{\text{comb}}$. To produce a final output normal map $\mathbf{n}^{\text{out}}$, we follow the idea in GeoNet [24] of using a refinement network, as shown in Figure 2. The refinement network we use is a three-layer residual network with a 6-channel input obtained by concatenating $(\mathbf{n}^{\text{raw}}, \mathbf{n}^{\text{comb}})$, and whose output is our final prediction $\mathbf{n}^{\text{out}}$.

3.4. Loss Functions

As mentioned in the previous subsection, there are three intermediate outputs, and one final output in the vp-line model: 1) the softmax output $\mathbf{p}(i)$ for Manhattan label classification, 2) the raw normal map $\mathbf{n}^{\text{raw}}(i)$, which is the other output of the decoder, 3) the combined normal map $\mathbf{n}^{\text{comb}}(i)$, defined by (2), and 4) the final output normal map $\mathbf{n}^{\text{out}}(i)$ from the refinement network. We use a negative log likelihood for the Manhattan label map estimation:

$$L_{\text{label}} = -\sum_{i=1}^{N}\sum_{c=1}^{7} M_c^{gt}(i)\log(p_c(i)), \tag{3}$$

where $M_c^{gt}(i)$ are the ground truth one-hot labels from (1).

We use angular distance between predicted normals and ground truth normals as the loss for all normal predictions:

$$L_{\text{norm}} = \sum_{*}\sum_{i=1}^{N} \omega_* \arccos(\mathbf{n}^*(i) \cdot \mathbf{n}_{gt}(i)), \tag{4}$$

where $*$ runs over $\{\text{raw}, \text{comb}, \text{out}\}$, and ω_* controls the importance of each normal prediction; in this work we weigh them equally. The overall loss is defined as:

$$L = L_{\text{norm}} + \lambda L_{\text{label}}, \tag{5}$$

4. Experiments

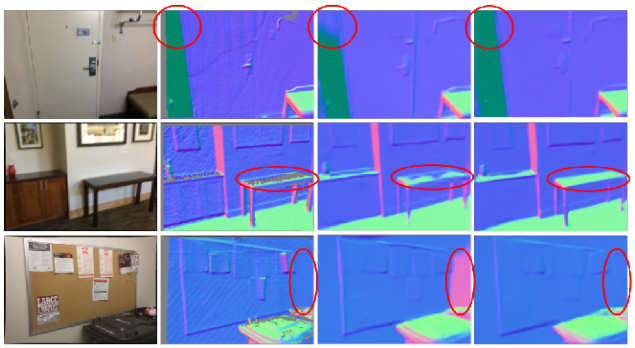
In this section, we show quantitative and qualitative evaluation results of our proposed vp-line model. We perform comparisons with state-of-the-art normal estimation methods using the ScanNet dataset, showing the superiority of our method. We also demonstrate the generalization ability of our method using the Replica and NYUD-v2 datasets. Through our experiments, we conclude that the combination of vanishing points and axis-aligned lines with a deep neural network, and our decomposition of the normal into the Manhattan part (learned by discrete classification) and non-Manhattan part (learned by regression) not only improves the normal estimation accuracy but also improves generalization ability.

4.1. Implementation Details

To generate the ground truth Manhattan label map, we used a hard threshold $T = 15$ degrees for classifying a normal direction into one of the Manhattan directions. We set the weights $\omega_* = 1.0$ (in Eq. 4) and the weight $\lambda = 0.2$ (in Eq. 5). We used the Adam optimizer with $\beta_1 = 0.5$ and $\beta_2 = 0.999$. We set the initial learning rate to be 0.0001, with 0.95 exponential decay rate. The input image size was 640×480, and the prediction size was 320×240. The batch size we used was 6 on an Nvidia V100 GPU. For data augmentation, we used random center cropping.

4.2. Datasets

We used the ScanNet dataset [4], which contains more than 2.5 million images, to train our models. Similar to FrameNet, we used 2/3 of the dataset for training and the rest for validation and testing. We also generated all the



(a) Image (b) GT (c) Baseline (d) vp-line model

Figure 6: Visual comparison of the results on ScanNet [4] dataset. By incorporating the vanishing points and lines, our vp-line method improves the baseline method.

Manhattan line maps during data preparation using the vanishing point and line method described in Section 3.1.

NYUD-v2 [23] is another large benchmark dataset; we directly applied our ScanNet pretrained model on its official test set to demonstrate the generalization ability of our model, and also compared with the state-of-the-art methods.

In addition to NYUD-v2, we also demonstrate the generalization ability of our method using the Replica dataset [29]. Replica has the highest ground truth normals quality compared to the other datasets, but it has a lot fewer scenes.

4.3. Normal estimation

To evaluate the performance of our vp-line model, we compare to the state-of-the-art deep learning-based normal estimation methods. We use mean and median angle error as well as the percentage of normals within certain angles as the evaluation metrics. As shown in Table 1, our baseline model, based on UPerNet architecture, already outperforms the state-of-the-art. The proposed vp-line model achieves the best result, demonstrating the effectiveness of incorporating Manhattan lines and vanishing points into normal estimation as well as the joint estimation of normals in both Manhattan frame aligned coordinates and camera coordinates. Figure 6 shows the qualitative results; as can be seen, the vp-line model produces more correct results.

Methods	Error metric		Accuracy metric		
	mean	median	11.25°	22.5°	30°
GeoNet [24]	19.77	11.34	49.7	70.4	77.7
FrameNet [12]	15.28	8.14	60.6	78.6	84.7
Our baseline	14.23	7.03	64.8	81.2	86.5
vp-line model	**13.76**	**6.68**	**66.3**	**81.8**	**87.0**

Table 1: Normal prediction on the ScanNet [4] dataset. Our baseline model outperforms the state-of-the-art, and our vp-line model achieves the best results.

Generalization to unseen datasets. One advantage of combining analytically computed Manhattan lines and vanishing points into a deep learning-based normal estimation framework is the improved generalization ability. To demonstrate this, we directly applied our vp-line model, which is trained on the ScanNet dataset, to other datasets. Table 2 shows the quantitative results on the NYUD-v2 dataset. It can be seen that our vp-line model performs even better than methods trained on the NYUD-v2 dataset, demonstrating its good generalization ability. We also directly ran FrameNet, trained on ScanNet, on the NYUD-v2 dataset; it can be seen that the degradation of its performance when applied to an unseen dataset is much larger than our vp-line model. Figure **??** shows a qualitative comparison with the-state-of-the-art methods. It can be seen that our predicted normal maps capture many more details than other methods.

Methods	Training	Error metric		Accuracy metric		
		mean	median	11.25°	22.5°	30°
Eigen *et al.* [5]	NYUD-v2	23.7	15.5	39.2	62.0	71.1
GeoNet [24]	NYUD-v2	19.0	11.80	48.4	71.5	79.5
FrameNet [12]	ScanNet	21.60	13.52	43.7	65.7	74.2
vp-line model	ScanNet	**17.98**	**9.83**	**54.3**	**73.8**	**80.7**

Table 2: Normal prediction on the NYUD-v2 [23] dataset. Both FrameNet and our vp-line model are trained on ScanNet. Our vp-line model achieves the best results, demonstrating its good generalization ability.

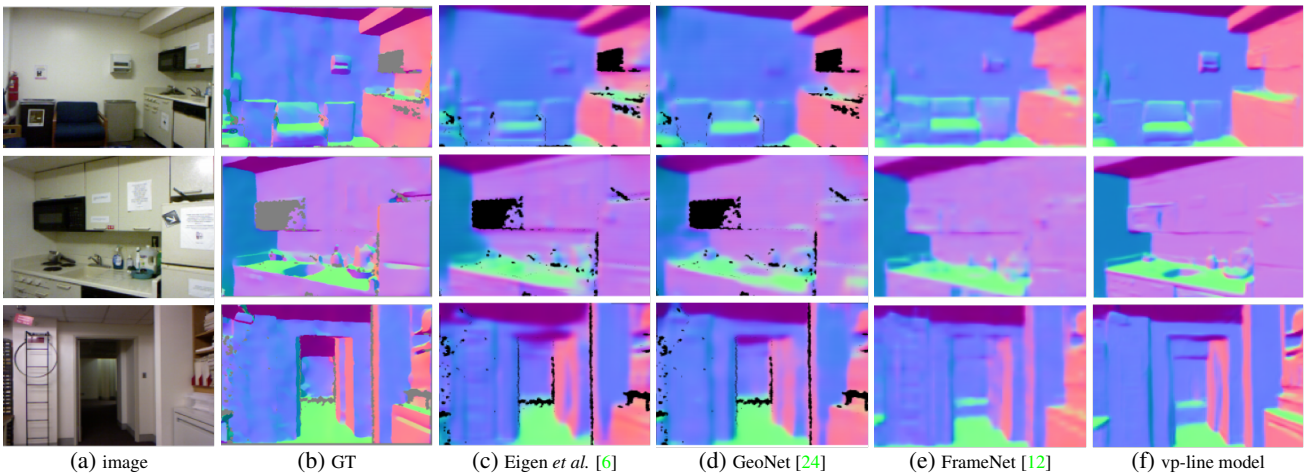
To further show the generalization ability of our proposed vp-line model, we used the Replica dataset that has much larger scene variation compared to the ScanNet dataset. Table 3 and Figure 8 show the comparison results of FrameNet, our baseline model, and our vp-line model. All three models were trained on the ScanNet dataset and directly applied to the Replica dataset. Compared to the evaluation result on the ScanNet dataset, the performance degradation of the FrameNet and our baseline model is larger than that of the vp-line model, demonstrating that the incorporation of the vanishing points and the Manhattan line map improves the generalization ability of the model.

Methods	Error metric		Accuracy metric		
	mean	median	11.25°	22.5°	30°
FrameNet [12]	19.51	13.60	39.1	72.7	82.0
Our baseline	18.23	12.77	44.5	76.2	84.6
vp-line model	**17.08**	**10.50**	**51.9**	**77.5**	**85.2**

Table 3: Normal prediction on the Replica [29] dataset. All methods are trained on ScanNet to compare the generalization ability. Our vp-line model achieves best results and lowest performance degradation, demonstrating its good generalization ability.

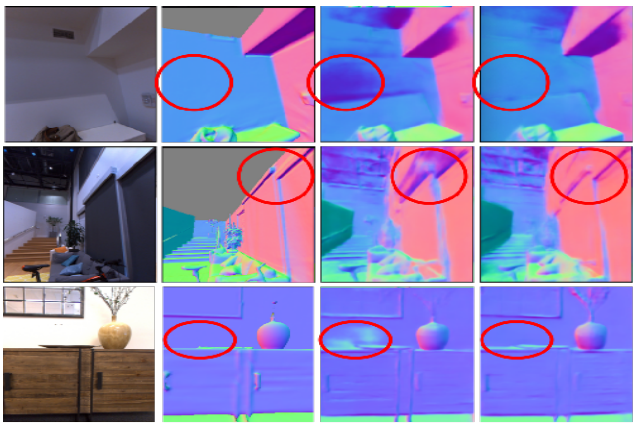
4.4. Ablation Study

To investigate our architecture choices, we conduct a series of ablation studies. To explore the impact of the input Manhattan line map, we trained a model with only an RGB



(a) image (b) GT (c) Eigen *et al.* [6] (d) GeoNet [24] (e) FrameNet [12] (f) vp-line model

Figure 7: Visual comparison with the state-of-the-art on NYUv2 [23]. Our vp-line model produces more detailed results, such as the box on the wall in the first image and the sink in the second image.



(a) Image (b) GT (c) Baseline (d) vp-line model

Figure 8: Visual comparison on the Replica [29] dataset. Our vp-line model can handle difficult cases such as shadow and low light conditions.

Methods	Error metric		Accuracy metric		
	mean	median	11.25°	22.5°	30°
vp-line model (no line)	13.93	6.79	65.5	81.6	86.8
vp-line model (no label)	14.14	7.08	64.7	81.2	86.5
Our baseline	14.23	7.03	64.8	81.2	86.5
vp-line model	**13.76**	**6.68**	**66.3**	**81.8**	**87.0**

Table 4: Evaluation of the impact of different components in vp-line model. Only the combination of the Manhattan line and label map can improve the performance over our baseline.

Outputs	Error metric		Accuracy metric		
	mean	median	11.25°	22.5°	30°
$\mathbf{n}^{\text{comb}}$	14.08	7.01	65.1	81.6	86.8
$\mathbf{n}^{\text{raw}}$	13.92	6.97	65.8	81.8	86.9
$\mathbf{n}^{\text{out}}$	**13.76**	**6.68**	**66.3**	**81.8**	**87.0**

Table 5: Evaluation of different normal predictions in the vp-line model. The final fused result $\mathbf{n}^{\text{out}}$ has the best result.

image and vanishing points as input; the output remains the same as from the vp-line model (vp-line model (no line)). Similarly, to show the impact of the Manhattan label map, we trained a model with an RGB image and a Manhattan line map as input and only a normal map as output(vp-line model (no label)): in other words our decoder produces a 3-channel output $\mathbf{n}^{\text{raw}}(i)$, which we take as the final output. Table 4 shows the quantitative results of the ablation studies. It can be seen that only the combination of the Manhattan line and label map can produce significantly better normal estimation results than the baseline model. One explanation is that the input Manhattan lines are helpful for the Manhattan label map estimation but not directly for the normal estimation.

As mentioned in Section 3.3, our vp-line model outputs multiple normal estimates. The accuracy of the normals from the Manhattan label map $\mathbf{n}^{\text{comb}}$, the direct normal prediction $\mathbf{n}^{\text{raw}}$ and the fused final normal prediction $\mathbf{n}^{\text{out}}$ is shown in Table 5. The fused normal map $\mathbf{n}^{\text{out}}$ takes advantage of both $\mathbf{n}^{\text{comb}}$ and $\mathbf{n}^{\text{raw}}$ and thus has the best result. The network converges after twenty epochs of training. During the first ten epochs, the accuracy of $\mathbf{n}^{\text{comb}}$ is higher than $\mathbf{n}^{\text{raw}}$, which suggests the Manhattan label map is easier to learn than to the direct normal map. However, after ten epochs, $\mathbf{n}^{\text{raw}}$ starts to achieve better results, possibly because the accuracy of $\mathbf{n}^{\text{comb}}$ is bounded by the analytically computed vanishing points.

We observed that inheriting weights from a network pre-trained on semantic segmentation makes a significant difference. We suspect this is because normal estimation is closely related to semantic segmentation; for example, if a region is identified as a ground plane, the normal is very likely to be pointing upward. That was also our motivation

Methods	Error metric		Accuracy metric		
	mean	median	11.25°	22.5°	30°
Baseline (no init)	16.90	8.69	58.7	76.4	82.5
Baseline	14.23	7.03	64.8	81.2	86.5
vp-line model (no init)	15.33	7.59	62.4	78.9	84.5
vp-line model	**13.76**	**6.68**	**66.3**	**81.8**	**87.0**

Table 6: Evaluation of normal estimation with and without weights pretrained from semantic segmentation tasks.

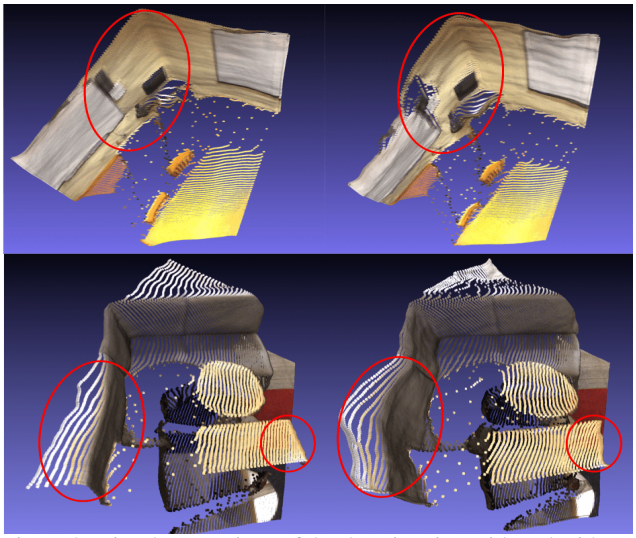


Figure 9: Visual comparison of depth estimation with and without our predicted normal map as input. The left image is with predicted normal map as input. It can be seen that the wall and the sofa are flatter and the corner is sharper.

for choosing a semantic segmentation network as the backbone for our normal estimation model. Table 6 shows the quantitative results of our proposed model with and without pretrained semantic segmentation weights as initialization.

4.5. Depth estimation

In recent years, single-view depth estimation has received even more interest than normal estimation. However, most current deep learning-based depth estimation methods suffer from large scale surface irregularities, as shown in Figure 9. It can be seen that the estimated geometry from an ordinary single-view depth prediction network tends not to be smooth, even in flat regions. Since surface normals are closely related to depth, they can provide large scale regularization for depth estimation. In order to prove that the predicted normal map is helpful for depth estimation, we conducted a series of depth estimation experiments. We evaluate the performance of depth prediction based on the following metrics: mean absolute relative error (ARD), root mean squared error (RMSE) , root mean squared log error (RMSE (log)) and the accuracy under threshold ($\delta_i < 1.25^i, i = 1, 2, 3$).

Methods	RMSE	RMSE (log)	ARD
model-d	0.342	0.087	0.120
model-nd	0.280	0.057	0.104
model-pd	**0.267**	**0.053**	**0.098**

Table 7: Depth prediction on ScanNet [4] dataset. The numbers are lower the better. Both *model-nd* and *model-pd* perform significantly better than *model-d*, demonstrating the usefulness of the normal map in depth prediction.

As shown in Table 7, we trained three networks using the ScanNet dataset, with the same training and test split as in the normal estimation evaluation: 1) a baseline depth estimation network (*model-d*), which takes a single image as input and outputs a depth map, 2) a joint depth and normal estimation network (*model-nd*), which predicts both depth and normal maps simultaneously, and 3) a depth estimation network (*model-pd*) that takes a color image together with a normal map from our pretrained normal estimation network as inputs. It can be seen that both *model-nd* and *model-pd* perform significantly better than *model-d*, which supports our idea that normals are helpful for depth estimation. Figure 9 shows that with an additional normal map as input, the predicted depth map is more regularized.

We further ran *model-pd* on the NYUD-v2 dataset without retraining or fine-tuning. As shown in Table 8, it has comparable performance with the state-of-the-art models trained on NYUD-v2.

Methods	Error metric			Accuracy metric		
	RMSE	RMSE (log)	ARD	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Laina *et al.* [15]	0.584	0.164	0.136	82.2	95.6	98.9
DORN [7]	0.547	0.158	0.116	85.6	96.1	98.6
Lee *et al.* [17]	0.538	0.148	0.131	83.7	97.1	99.4
Yin *et al.* [35]	**0.416**	-	0.108	87.5	**97.6**	**99.4**
model-d	0.508	0.162	0.132	83.0	96.1	99.0
model-pd	0.438	**0.136**	**0.103**	**88.4**	97.0	99.2

Table 8: Depth prediction on NYUD-v2 [23] dataset. With a pretrained normal map as input, our proposed *model-pd*, trained on ScanNet, has comparable performance with the state-of-the-art trained on NYUD-v2.

5. Conclusion

In this paper, we have presented a novel single-view normal estimation method that combines a robust vanishing point and line detection method with a deep convolutional neural network. We introduced the Manhattan label map that can serve as a bridge to connect the Manhattan lines and vanishing points with the direct normal prediction in a fully differentiable manner. We demonstrated that such a combination not only allows our model to produce superior results over the state of the art but also leads to better generalization ability . Furthermore, by providing the normal map from our pretrained model as input to a depth estimation network, the performance of the depth estimation network can be significantly improved.

References

[1] M. Antunes and J. P. Barreto. A global approach for the detection of vanishing points and mutually orthogonal vanishing directions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1336–1343, 2013. 2

[2] A. Bansal, B. Russell, and A. Gupta. Marr revisited: 2d-3d alignment via surface normal prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5965–5974, 2016. 1, 2

[3] B. Caprile and V. Torre. Using vanishing points for camera calibration. *International journal of computer vision*, 4(2):127–139, 1990. 2

[4] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 2, 5, 6, 8

[5] D. Eigen and R. 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. 1, 6

[6] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014. 2, 7

[7] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2002–2011, 2018. 4, 8

[8] R. Garg, V. K. BG, G. Carneiro, and I. Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European Conference on Computer Vision*, pages 740–756. Springer, 2016. 2

[9] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 270–279, 2017. 2

[10] R. Grompone von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. LSD: a Line Segment Detector. *Image Processing On Line*, 2:35–55, 2012. 3

[11] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 3

[12] J. Huang, Y. Zhou, T. Funkhouser, and L. Guibas. FrameNet: Learning local canonical frames of 3d surfaces from a single RGB image. *arXiv preprint arXiv:1903.12305*, 2019. 1, 3, 6, 7

[13] J. Kosecka and W. Zhang. Video compass. pages 476–490, 2002. 3

[14] J. Košecká and W. Zhang. Extraction, matching, and pose recovery based on dominant rectangular structures. *Computer Vision and Image Understanding*, 100(3):274–293, 2005. 1

[15] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, pages 239–248. IEEE, 2016. 8

[16] D. C. Lee, M. Hebert, and T. Kanade. Geometric reasoning for single image structure recovery. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2136–2143. IEEE, 2009. 1

[17] J.-H. Lee and C.-S. Kim. Monocular depth estimation using relative depth maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2019. 8

[18] Z. Li and N. Snavely. MegaDepth: Learning single-view depth prediction from internet photos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2041–2050, 2018. 3

[19] C. Liu, K. Kim, J. Gu, Y. Furukawa, and J. Kautz. Planercnn: 3d plane detection and reconstruction from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4450–4459, 2019. 3

[20] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa. Planenet: Piece-wise planar reconstruction from a single RGB image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2579–2588, 2018. 3

[21] F. Liu, C. Shen, G. Lin, and I. Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE transactions on pattern analysis and machine intelligence*, 38(10):2024–2039, 2015. 2

[22] B. Micusk, H. Wildenauer, and M. Vincze. Towards detection of orthogonal planes in monocular images of indoor environments. In *2008 IEEE International Conference on Robotics and Automation*, pages 999–1004. IEEE, 2008. 1

[23] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from RGBD images. In *ECCV*, 2012. 2, 6, 7, 8

[24] X. Qi, R. Liao, Z. Liu, R. Urtasun, and J. Jia. Geonet: Geometric neural network for joint depth and surface normal estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 283–291, 2018. 2, 5, 6, 7

[25] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 4

[26] C. Rother. A new approach to vanishing point detection in architectural environments. *Image and Vision Computing*, 20(9-10):647–655, 2002. 2

[27] E. Rouy and A. Tourin. A viscosity solutions approach to shape-from-shading. *SIAM Journal on Numerical Analysis*, 29(3):867–884, 1992. 2

[28] S. N. Sinha, D. Steedly, and R. Szeliski. Piecewise planar stereo for image-based rendering. In *Twelfth IEEE International Conference on Computer Vision (ICCV 2009)*, pages 1881–1888, September 2009. 2

[29] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, A. Clarkson, M. Yan, B. Budge, Y. Yan, X. Pan, J. Yon, Y. Zou,

K. Leon, N. Carter, J. Briales, T. Gillingham, E. Mueggler, L. Pesqueira, M. Savva, D. Batra, H. M. Strasdat, R. D. Nardi, M. Goesele, S. Lovegrove, and R. Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 2, 6, 7

[30] P. Wang, X. Shen, B. Russell, S. Cohen, B. Price, and A. L. Yuille. Surge: Surface regularized geometry estimation from a single image. In *Advances in Neural Information Processing Systems*, pages 172–180, 2016. 2

[31] R. Wang, S. M. Pizer, and J.-M. Frahm. Recurrent neural network for (un-)supervised learning of monocular video visual odometry and depth. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[32] X. Wang, D. Fouhey, and A. Gupta. Designing deep networks for surface normal estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 539–547, 2015. 2

[33] W. Xian, Z. Li, M. Fisher, J. Eisenmann, E. Shechtman, and N. Snavely. Uprightnet: Geometry-aware camera orientation estimation from single images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9974–9983, 2019. 3

[34] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 418–434, 2018. 4

[35] W. Yin, Y. Liu, C. Shen, and Y. Yan. Enforcing geometric constraints of virtual normal for depth prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5684–5693, 2019. 2, 8

[36] A. R. Zamir, A. Sax, W. Shen, L. J. Guibas, J. Malik, and S. Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3712–3722, 2018. 2

[37] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah. Shape-from-shading: a survey. *IEEE transactions on pattern analysis and machine intelligence*, 21(8):690–706, 1999. 2

[38] C. Zou, A. Colburn, Q. Shan, and D. Hoiem. Layoutnet: Reconstructing the 3d room layout from a single rgb image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2051–2059, 2018. 1, 3