Region-Support Depth Inference from a Single Image

Anonymous Author(s)
Affiliation
Address
email

Abstract

Depth inference from a single image is a long-standing problem in the computer vision community. It is technically ill-posed since monocular cues are ambiguous for the depth inference. Using semantic segmentation result is beneficial to resolve some ambiguities of monocular cues, but it also introduces new ambiguities between semantic labels and depth values. To address this issue, we propose a new depth estimation method using region support as the inference guidance and design a region support network to realize the depth inference. The region support network consists of two modules: the generation module for region support and the refinement module for coarse depth. The generation module employs a pyramid unit to determine the region support from the RGB image. The region support concatenates the RGB image to form the inference guidance and provides the initial coarse depth for the refinement. With the inference guidance, the refinement module implements the coarse-to-fine strategy in a novel iterative manner by a simplified pyramid unit. The experiments on the NYU dataset demonstrate that the region support can significantly resolve the ambiguities and improve the inference accuracy.

1 Introduction

The depth estimation methods are widely used in robotics, autonomous vehicles, recognition tasks, visual localization and scene analysis [1–4]. As monocular images are the most readily available data in computer vision, the depth inference from a single image has attracted considerable attention in the past decades [5–7]. Most methods infer the depth according to the monocular cues such as scale ratio and feature variance of objects [6, 8], but these cues are not clear enough to guide the depth inference. Using the semantic segmentation results as the inference guidance is proven to be beneficial to resolve some ambiguities because the semantic guidance can fuse monocular cues to regularize the depth inference space [9–11]. Currently, neural networks improve the depth inference with their powerful representations [3, 7, 12, 13]. Especially, many methods find that it is profitable to combine the depth estimation with the semantic segmentation using multi-task neural networks [14–16], where neural networks effectively leverage the semantic information to guide the depth inference. However, there are still many ambiguous situations where the semantic guidance is not helpful. For example, when objects lying on different depths have the same semantic label, the same label is ambiguous to infer the different depth values. Besides, since the semantic labels are the same for pixels among one object if this object strides over a large variance of depths, the same labels is also ambiguous for the depth inference. To this end, we propose a new depth estimation method using region support as the inference guidance and carry out the depth inference by a novel end-to-end neural network.

The guidance of region support mainly comes from the division of regions which are determined by pixels at the same depth. It can be obtained by refining the semantic segmentation results and replacing semantic labels. Compared with directly using semantic segmentation results as discussed

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

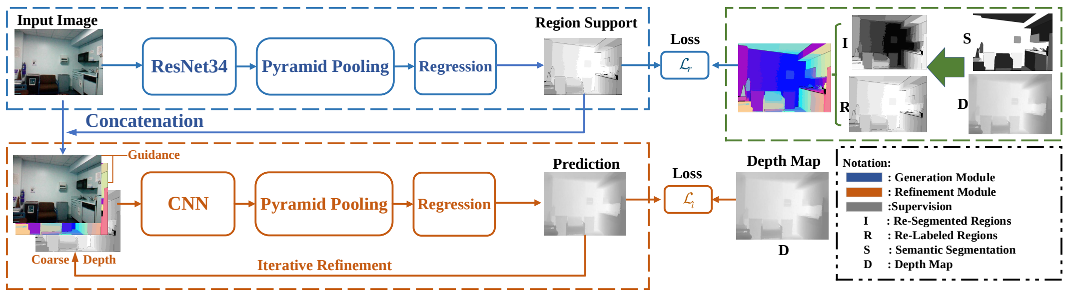

Figure 1: The overview of our depth estimation method. The generation module uses a pyramid-based architecture to generate the region support, as shown in the blue legend. The supervision of region support is obtained from the segmentation results and depth map as shown in the green legend and a region-based loss function $\mathcal{L}_r$ is designed for the training. The region support concatenates with the RGB image as the inference guidance for the refinement to infer the accurate depth. As shown in orange legend, The refinement carries out in an iterative manner with a simplified pyramid unit. In addition, an iterative loss $\mathcal{L}_i$ is designed for the training. This figure is better shown in the colorized view.

in [9–11, 14–16], we re-segment the huge object into small regions according to the local depth variance. The re-segmentation ensures the depth variance of each region is stable, so the ambiguity between labels and depths among each region are remarkably reduced. Furthermore, the increased quantity of regions is conducive for the inference of the mutual relationships between different regions. The additional relationships effectively contribute to the regularization of the depth inference space. Then, the refined regions are re-labeled by their average depths. The re-labeling operation resolves the ambiguity of regions at different depths but with the same label. It is worth noting that we also employ the region support as the initial coarse depth for the refinement module. The region support helps to carry out the depth inference in a new coarse-to-fine manner, which has been proven successful to resolve the ill-posed problem [6, 7, 17].

In this work, we present a novel neural network called region support network (RSN) to carry out the depth inference with the region support. As shown in Figure 1, the RSN consists of two modules: the generation of the region support and the refinement of the coarse depth. We design a new pyramid network to use multi-scale features for determination of the region support. We gain the supervision for the training from semantic segmentation results and the depth map. A new region-based loss $\mathcal{L}_r$ is designed to supervise the learning process. The obtained region support concatenates with original RGB images as the inference guidance for the refinement module. With this guidance, the refinement iteratively uses a simplified pyramid unit to infer the accurate depth from the coarse depth. The region support also works as the initial coarse depth, and then the later refined depth map replaces the previous coarse depth to form the new input. The region support network finally runs in an end-to-end manner by seamlessly integrating two modules. With the region support, our method reaches appealing performance on the NYU [18, 19] dataset. The comparison of the different guidance shows that the region support can significantly resolve the ambiguities and regularize the inference space.

2 Related Work

The depth estimation methods follow the human's monocular cues such as texture variations, texture gradients, occlusion, objects scales, etc. [20–23] The depth inference based on scaling laws proved the multi-scale feature is useful for the depth inference [6, 8]. Many works found that the depth estimation could be improved by semantic segmentation results [9, 15, 16]. Based on these observations, we design a pyramid unit to capture the multi-scale feature for the depth inference and propose the region support as the new inference guidance based on the semantic segmentation results.

The works in [6, 9, 10, 24] are mostly related to our method. Eigen et al. [6] designed an auto-encoder architecture to capture the multi-scale feature for depth inference and took a fully connected layer to achieve the final inference. Liwicki et al. [24] and Eigen et al. [6] respectively implemented the coarse-to-fine strategy using different neural networks. Liu et al. [10] and Wang et al. [9] used semantic segmentation results as the inference guidance for depth estimation and carried out the inference with the Markov Random Field (MRF) which slowed down the speed of depth inference. Compared with these methods, we design a pyramid pooling unit to capture the multi-scale features for the depth inference and achieve the whole depth inference using an end-to-end convolutional network. We give a new implementation of coarse-to-fine strategy by the iterative refinement module. The refinement module takes both region support and RGB information into consideration to infer the accurate depth from coarse depth.

3 Region Support Network

The region support network is illustrated in Figure 1. We first introduce the generation module with the pyramid pooling unit to determine the region support. The ground truth is obtained from semantic segmentation and depth map, as shown in Algorithm 1. A region-based $\mathcal{L}_r$ is proposed for the training. Then we present the interactive refinement module to achieve the coarse-to-fine strategy with the region-support guidance according to $\mathcal{L}_i$. Finally, we elaborate how to seamlessly integrate the two modules into the end-to-end region support network.

3.1 Generation for the Region Support

The guidance of region support mainly comes from the division of regions and the labels of the coarse depth. The division of regions helps the depth inference to determine the mutual relationships between regions, while the coarse depth makes the complex inference running in a much simpler manner. To obtain the region support using the network, we provide the supervision of the region support from the depth map and semantic segmentation results. We first re-segment the semantic segmentation results into unrelated regions where different regions have different labels according to the indexing order. Then we refine the regions with a stable variance of depths, in the same time, each region is re-labeled by its average depth. The Algorithm 1 shows the detailed operation to obtain the ground truth.

Algorithm 1: Generating the ground truth of the region support

```
Input: Semantic Segmentation Results S,
Depth Map D
Output: Region Support R, Re-Segmented Regions I
Notations of Functions:
max : Max Value
min : Min Value
abs : Absolute Value
where : Find the Correct Positions
sort : Sort from Left-Top to Right-Bottom
mean : Mean Value
len : Length of Input
dis : The Euclidean Distance of Two Positions
zeros: The Zero Matrix as Shape of Input
1.Semantic Regions to Instance Regions
I = S;
label = -1;
while (max(S)>0) do
    category = max(S);
    while (max(S)==category) do
        positions = where(S == category);
        positions = sort(positions);
          I(positions[0]) = label;
        for (i=1;i<len(positions);i++) do
            if (dis(position[i]-position[i-1])<=2) then
                I(positions[i]) = label;
            end
            else
                break;
            end
        end
    end
    label = label - 1;
end
I = abs(I);
R = I;
2.Refining Regions and Labeling with Depth
var = (max(D) - min(D))/10;
S = zeros(S);
for (i=1;i<=max(R);i++) do
    positions = where(R == i);
    d = D(positions);
    if (max(d)-min(d)<2*var) then
        S[positions] = mean(d);
    else
        start = min(d), end = min(d) + var;
        while (end<max(d)) do
            p = where(R == i, d > start, d <
              end);
            if (len(D(p))>len(d)/(max(d)-min(d))/var) then
                S(p) = mean(D(p));
                start = end; end = end + var;
                continue;
            end
            else
                start = end; end = end + var;
                continue;
            end
            if (end+var>max(d)) then
                S(p) = mean(D(p));
                break;
            end
        end
    end
end
R = S;
end
```

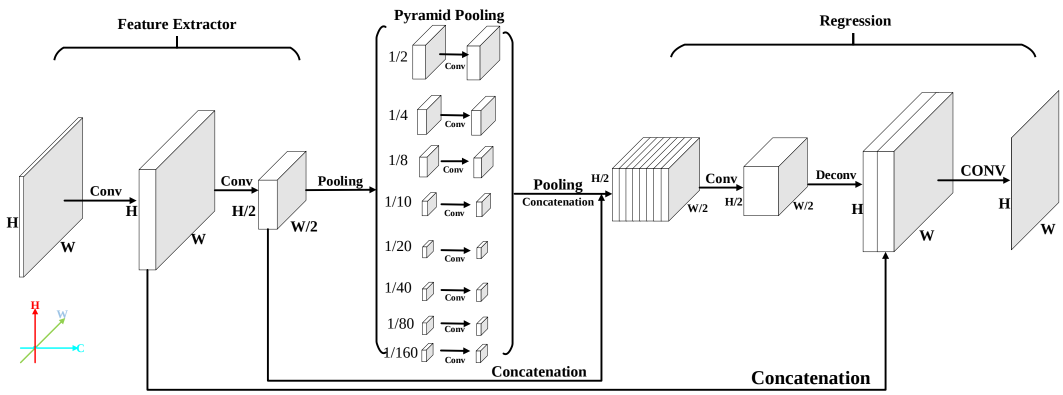

Figure 2: The proposed pyramid network. The pyramid network is used for both generation module and refinement module. It consists of feature extractor, pyramid pooling unit and depth regression. The detailed layer setting is shown in Table 1. For generation module, a ResNet34 is employed as the feature extractor while the refinement only uses six convolutional layers. The pyramid pooling unit pools the feature map into eight scales, i.e., $1/2, 1/4, 1/8, 1/10, 1/20, 1/40, 1/80, 1/160$ to gain the multi-scale feature. The final eight convolutional layers are employed to regress the depth value.

We design a novel pyramid architecture to utilize multi-scale features for the determination of the region support. The proposed pyramid network is shown in Figure 2. It consists of feature extractor, pyramid pooling and depth regression. For the generation module, we adopt a modified ResNet34 [25] as the feature extraction. We remove all sub-sampling operations in the residual layers and conduct the sub-sampling only after the first residual layer. Through the residual layers, we get a sub-sampled feature map with the half-resolution size. Then we employ the pyramid pooling unit to offer the multi-scale features for the depth inference. The half-resolution feature map pools into eight different scales, i.e., $1/2, 1/4, 1/8, 1/10, 1/20, 1/40, 1/80$ and $1/160$. The sub-sampled feature maps respectively pass through one convolutional layer. After that, the sub-sampled feature maps are resized into the half-resolution size and concatenated together with the original half-resolution feature map. With the multi-scale features, we adopt three convolutional layers to carry out the regression on half-resolution and then use a de-convolutional layer to up-sample the feature map back to the full-resolution. The obtained full-resolution feature map concatenates with the full-resolution feature map before the second module of residual modules. The final regression is carried out on full-resolution using additional five convolutional layers. It is worth noting that during two concatenations, we ensure the channel of the high-resolution feature map holds half of channels of the concatenated feature map. Except for the last layer, all the convolutional layers are followed with a batch-normalization and a ReLu unit. The detailed setting of generation module is shown in Section 3.3.

Instead of directly training the two modules together, we pre-train the two modules separately. The $\mathcal{L}_2$ loss function for depth inference is defined as

$$\mathcal{L}_2 = \frac{1}{N}\sum_{n=1}^{N}(y_n - {y_n}^*)^2. \tag{1}$$

The predicted depth map and ground truth are represented by y and y^*, respectively, and N pixels are indexed by n. Equation 1 computes the pixel-wise loss among the whole image, but for the determination of region support, this kind of statistics is not suitable. To this end, we calculate the loss among each region, which can be expressed as

$$\mathcal{L}_r = \frac{1}{M}\frac{1}{N}\sum_{m=1}^{M}\sum_{n=1}^{N}(y_n^m - {y^*}_n^m), \tag{2}$$

where y^m represents the predicted depth at the m-th region and y^{*m} represents the ground truth depth in the m-th region. N indexes the pixels in m-th region and M indexes the regions for computation. More explanations of the loss functions are dicussed in Section 3.3.

Table 1: The parameter setting for the region support network

	Generation of Region Support					Refinement of Depth				
	index	kernel	stride	in	out	index	kernel	stride	in	out
Feature Extractor	1	7	1	3	64	1	7	1	4	64
	2-4	3	1	64	64	2-4	3	1	64	64
		residual unit					residual unit			
	5	3	1	64	128	5	3	1	64	128
	6	3	1	128	128	6	3	1	128	128
	7-10	3	2	128	256	Each layer is with BN and ReLu. We adopt ResNet34 as the pre-trained model to initialize the layers. For refinement, we adopt a simple feature extractor consisting of six convolutional layers.				
	11-16	3	1	256	512					
	17-20	3	3	512	512					
	7-20	residual unit								
pyramid pooling	21	pyramid pooling scale				7	pyramid pooling scale			
		1/2,1/4,1/8,1/10,1/20,1/40,1/80,1/160					1/2,1/4,1/8,1/10,1/20,1/40,1/80,1/160			
depth inference	22	3	1	512	512	8	3	1	128	128
	concatenate with 20					concatenate with 6				
	23	3	1	1024	512	9	3	1	256	128
	24	3	1/2	512	256	10-15	3	1	128	1
	25	3	1	256	128	The pyramid pooling unit pools the feature map into eight scales. The concatenation fuses the multi-scale feature.				
	concatenate with 6									
	26-30	3	1	128	1					

3.2 Refinement Module for the Coarse Depth

The region support combines with the RGB image as the inference guidance for refinement. Besides, it is also used as the initial coarse depth for the iterative refinement. The division of regions helps the inference to find out the mutual relationship between regions. Compared with computing the mutual relations between pixels, the region-level relationship is much more reliable. Instead of refining the coarse depth like Eigen and Fergus [17], our refinement module infers the depth value regarding the average depth of each region, which effectively constrains the inference space.

We refine the coarse depth in an iterative manner because the refinement is seen as a general inference from coarse depth to refined depth. The refinement of coarse depth should be adequate not only for the region support but also for the usual coarse depth in [6, 24]. The iterative refinement should be at least two times, in this paper, the refinement module iterates for three times to reach a sufficient performance. In the first iteration, the refinement module infers the accurate depth from the average depth in region support. Then in the second iteration, the refinement module handles the usual coarse depth. The final iteration makes the refinement module to have a better generalization.

As shown in Figure 1, we concatenate the RGB image with the region support as the inference guidance. In the first iteration, the region support also works as the coarse depth to concatenate with the inference guidance. After each iteration, the refined depth map replaces the previous to form the new input. The refinement module uses a simplified architecture to infer the accurate depth. We replace the pre-trained residual network (ResNet34) with six convolutional layers and remove the sub-sampling units, so the obtained feature map is full-resolution. Then the proposed pyramid pooling unit is applied as shown in Figure 2. After that, the depth inference is carried out as a regression task using eight convolutional layers. All convolutional layers are followed by a batch-normalization and a ReLu unit except the last layer. The detailed layer setting is shown in Section 3.3. The iterative regression loss function for refinement module is defined as

$$\mathcal{L}_i = \lambda_1 \mathcal{L}_r^1 + \lambda_2{}_r^2 + \lambda_3 \mathcal{L}_r^3, \tag{3}$$

where $\mathcal{L}_r^i$ indicates the $\mathcal{L}_r$ loss 2 in the iteration i. The λ_i are the weights for the losses of different iterations. In the experiments, we set $\lambda_1 = 0.2$, $\lambda_2 = 0.3$, $\lambda_3 = 0.5$.

3.3 Implementation Details

The parameter setting for region support network is illustrated in Table 1. We first respectively train the generation of region support with $\mathcal{L}_r$ 2 and refinement of depth with loss $\mathcal{L}_i$ 3. Then we freeze the generation module and train the refinement module with $\mathcal{L}_i$ 3. After that, we freeze refinement part and train generation part only with $\mathcal{L}_i$ 3. Finally, we train the two modules together according to a combination loss fuction

$$\mathcal{L}_c = \lambda_0 \mathcal{L}_r + \lambda_4 \mathcal{L}_i, \tag{4}$$

Table 2: The comparison with state-of-the-art methods on the NYU dataset

Method	Error (lower is better)			Accuracy (higher is better)		
	rel	RMSE-log	RMSE-lin	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Karsch et al. [26]	0.349	-	1.214	0.447	0.745	0.897
Zhuo et al. [11]	0.305	0.122	1.04	0.525	0.838	0.962
Liu et al. [27]	0.230	0.095	0.824	0.614	0.883	0.975
Xu et al. [7]	**0.143**	0.065	0.613	**0.811**	**0.984**	**0.987**
Wang et al. [9]	0.220	0.094	0.745	0.605	0.890	0.970
Eigen et al. [6]	0.215	0.283	0.907	0.611	0.887	0.971
Laina et al. [28]	0.129	**0.056**	**0.583**	0.801	0.950	0.986
Chakrabarti et al. [13]	0.149	0.43	0.620	0.806	0.958	**0.987**
Our Generation Module	0.2350	0.2639	0.7367	0.6517	0.8932	0.9715
Our Refinement Module	**0.0851**	**0.1057**	**0.2917**	**0.9528**	**0.9938**	**0.9988**
Our Region Support Network	0.196	**0.172**	0.681	0.792	0.961	**0.987**

where the $\lambda_0 = 0.3$ and $\lambda_4 = 0.7$. After computing on the linear loss between y and y^*, we transform the y and y^* by a logarithm function to reach more accurate results. We find the log loss behaves more stable than linear loss when linear loss comes to a small value. The results are shown in Table 3. During training process, we adopt a standard SGD optimizer with a fixed learning rate of 0.001.

4 Experiments

In Section 4.1, we compare our depth estimation method with the state-of-the-art methods on NYU [18, 19] dataset. The impressive results show the region support network effectively resolves the ambiguities in indoor scenes. The analysis of region support network is shown in Section 4.2. To demonstrate the great effectiveness of the region support, we carry out the depth inference with different guidance. The experiments find out the effectiveness of each module. A qualitative visualization of depth estimation results is depicted in Figure 3.

4.1 Results on NYU Dataset

The NYU-Depth dataset [18, 19] is comprised of video sequences from a variety of indoor scenes recorded by both the RGB and Depth cameras from the Microsoft Kinect. It consists of RAW data and labeled data. The raw dataset contains the raw images and accelerometer dumps from the Kinect. The labeled data is a subset of the video data accompanied by dense multi-class labels which have also been preprocessed to fill in missing depth labels. In this paper, we only use the labeled data to form the training and testing sets. We combine the NYU-Depth V1 [18] and NYU-Depth V2 [19] to form the final NYU dataset used for our experiments. The NYU-Depth V1 has 64 scenes while the NYU-Depth V2 has 464 scenes. The fused NYU dataset has 478 scenes. For scenes with many images, we randomly select several images as testing. And for scenes which only have few images, we directly use them as testing images even there is no similar data for training. The division ensures each scene is well evaluated and makes the evaluation harder but more reliable. Finally, we select 3264 images for training and 483 for testing. Compared with the previous methods [6, 27, 13] which use raw data or data augmentation to form a large quantity of training data, we only use a small quantity of images as training data. For evaluation, we use several general metrics to access the performance of our method. That is, linear RMSE-lin: $\sqrt{\frac{1}{N}\sum_{n=1}^{N}(y_n - {y_n}^*)^2}$, log RMSE-log: $\sqrt{\frac{1}{N}\sum_{n=1}^{N}(log(y_n) - log({y_n}^*)^2)}$, related error (rel): $\frac{1}{N}\sum_{n=1}^{N}|y - y^*| \div y^*$ and threshold accuracy: $\max(\frac{y}{y^*}, \frac{y^*}{y}) = \delta < thres.$

The results on the NYU dataset are shown in Table 2. The generation module and refinement module are modified to directly infer the depth. The generation module reaches comparable results on RSME and threshold accuracy. This proves the pyramid network can capture the multi-scale feature for depth inference. The refinement module gets very impressive performance with the ground truth of region support. The region support from Algorithm 1 help the refinement module to significantly outperform

Table 3: The analysis of region support network on NYU dataset

Experiments		Error (lower is better)			Accuracy (higher is better)		
		rel	RMSE-log	RMSE-lin	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Baseline1	For Region Support	0.2334	0.2852	0.7737	0.6111	0.8769	0.9648
Generation Module	For Depth Inference	0.2350	0.2639	0.7367	0.6517	0.8932	0.9715
	With Linear Loss	0.2572	0.2855	0.7805	0.6114	0.8456	0.9650
	With $\mathcal{L}_2$ Loss	0.2503	0.2784	0.7681	0.6217	0.8682	0.9663
Baseline2	Analysis of Ground Truth	0.5063	0.4763	1.29	0.615	0.9016	0.9742
Refinement Module	With Ground Truth Region Support Guidance	**0.0851**	**0.1057**	**0.2917**	**0.9528**	**0.9938**	**0.9988**
	With Semantic Guidance	0.3471	0.3764	1.056	0.4497	0.7643	0.9191
	With Generated Coarse Depth Guidance	0.3345	0.3235	0.9806	0.4847	0.8005	0.9345
Baseline3	Initialized from Baseline1	0.2922	0.3472	0.8673	0.4648	0.7935	0.9372
Region Support Network	Freeze Refinement	0.2563	0.2852	0.7737	0.6111	0.8769	0.9648
	Freeze Generation	0.2692	0.2993	0.7922	0.5725	0.8596	0.9601
	End-to-End	**0.196**	**0.172**	**0.681**	**0.792**	**0.961**	**0.987**

the state-of-the-art method in all matrics. Especially in RSME and related error, it outperforms more than 50% than the state-of-the-art. This result demonstrates the region support is extremely instructive to resolve the ambiguities in the depth inference and the iterative refinement effectively uses the guidance to achieve the coarse-to-fine strategy. But directly using the region support might be a little unfair for other methods. So we also combine the two modules to infer depth end-to-end. Although using the fewest training data, the final RSN still reach a comparable performance with the state-of-the-art methods and the threshold accuracy reaches state-of-the-art performance.[1]

4.2 Analysis of Region Support Network

To demonstrate the effectiveness of the region support, we use the ablation analysis of the RSN. The results are shown in Table 3. First, we focus on the generation module. The *Baseline1* is the generation module trained for the region support with the log $\mathcal{L}_r$. Despite the validation in Table 3, we compute the mean value of RMSE and variance among each region respectively according to depth map and region support, which is 0.3613, 0.1196 and 0.2852, 0.117. The results show that even though the mean value is close to the ground truth, the generated region support still has an unstable variance more than 41%. The generation module for *depth inference* is directly trained by log $\mathcal{L}_r$ loss function on the depth map to figure out the ultimate performance of our pyramid unit. After 40 epochs, the generation module reaches 0.7367 linear RMSE. The results show that the pyramid architecture effectively utilizes the multi-scale feature for depth inference. We test the *linear loss* of $\mathcal{L}_r$ for the generation module. We can see that after both trained of 40 epochs, the log loss is 6.67% lower than the linear loss. But we also find the linear loss converges faster than the log loss, only after 17 epochs, the linear loss can reach 0.837 RMSE. To this end, the final end-to-end RSN is first trained by linear $\mathcal{L}_r$ for 17 epochs and then trained for by the log $\mathcal{L}_r$. The comparison of the proposed $\mathcal{L}_r$ loss and $\mathcal{L}_2$ *loss* is in log space. The 5.67% improvement in log RSME shows that the $\mathcal{L}_r$ loss is better for the depth estimation task.

For the refinement module, we first analysis the ground truth of region support then compare the effectiveness different guidance and coarse depth. To measure the original attributes of region support as the guidance and coarse depth, we compare the ground truth region support and the depth map. The *Baseline2* shows the region support is obviously different from the accurate depth in RMSE and related loss, but they are very similar to the depth in threshold measure which is because of the average depth label. We first use the *ground truth region support* as guidance and initial coarse depth. Then we use the *semantic segmentation results* as the guidance with the generated coarse depth. Finally, we use the *coarse depth* from generation module for depth inference as guidance. The refinement is trained for 63 epochs when the loss of semantic guidance does not come down anymore. The *region support guidance* reaches the performance of RMSE: 0.4055, log: 0.1665, rel: 0.1251, thre1: 0.8659, thre2: 0.9661,thre3: 0.9843. It remarkably outperforms the other two guidance in all metrics, which strongly proves that the region support can resolve the ambiguities where semantic guidance is not useful. The bold result is shown in Table 3 is trained for 120 epochs, which shows the

[1]The bold log error of RSN is because the log function of log error has two class: log_{10} and log_e. The 0.172 is the best log_e result while its log_{10} result is 0.112.

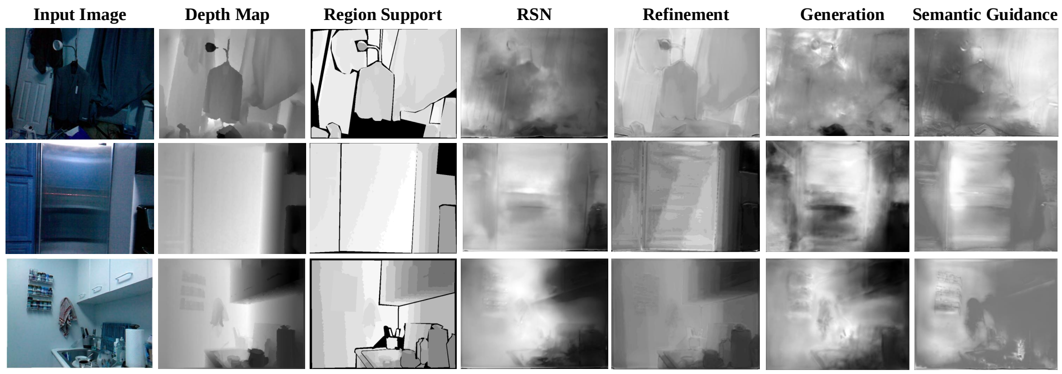

Figure 3: The visualization results on the NYU dataset. We visualize the predicted depth map from the region support network, refinement module with ground truth region support guidance, generation module and refinement module with semantic guidance.

significant performance of the region support. Compared to the *Baseline2*, the refinement module improves more than 78.7% on RMSE and 84% in related error, respectively, which demonstrates the great effectiveness of iterative refinement.

In the end, we validate the end-to-end region support network. The *Baseline3* is initialized from the pre-trained model of *Baseline1* and the best refinement module. But it is weaker than either module. The reason is that the depth inference of refinement is based on the average depth of each region, but the obtained region support may not perfectly satisfy this condition. To shorten the difference, we try to *freeze the refinement module* and fine-tune the generation module. We can see that only after 3 epochs, the network is remarkably improved by 11.2% on RMSE. We also try to *freeze the generation module* and use the generated region support as the inference guidance. After 5 epochs, it also reaches a better performance than *Baseline3* of 8.7% on RMSE and 19.3% better than the *semantic guidance*. Even though the generated region support is weaker than the ground truth, it still effectively guide the depth inference. After end-to-end training the region support network, the final result reaches an 11.6% promotion than the *Baseline1* and 25.4% than the *Baseline3* on RMSE. The visualization is illustrated in Figure 3. We can see the region support can always lead to a better performance while the ambiguities existing in the semantic guidance are obviously resolved.

4.3 Discussion and Future Work

The region support from generation module is unavoidable to have a unstable variance among each region, since the regression loss is employed for the determination. Compared with the segmentation methods which use classification loss to determine the label of the region, the regression results have an unstable variance. But the depth value is continuous and infinity, using the classification loss will greatly limit the generalization of the depth estimation method. We can see using the generated region support obviously limits the significant effectiveness of refinement module. The decay is from the variance of the obtained region support. So the end-to-end region support network can be improved by a better generation module which provides a more stable and low-varicance region support. In the future work, we will study a better generation module to genuinely obtain the region support. Beyond the benefits to depth estimation from a single image, we will extend more applications of the region support. The guidance of division of regions and coarse depth from the region support could serve as a mid-level representation for tasks requiring 3D guidance such as video analysis, object detection, scene understanding, etc.

5 Conclusion

In this paper, we have presented a novel depth estimation method using the end-to-end region support network. The network can carry out the depth inference using the region support as the guidance. We designed a new pyramid unit which can provide the multi-scale feature for depth inference. The refinement module can implement the coarse-to-fine strategy with region support. The experiments on the NYU dataset demonstrate the great effectiveness of the proposed method.

References

[1] David Marr and Tomaso Poggio. A computational theory of human stereo vision. *Proc. R. Soc. Lond. B*, 204(1156):301–328, 1979.

[2] David A Forsyth and Jean Ponce. *Computer vision: a modern approach*. Prentice Hall Professional Technical Reference, 2002.

[3] Weifeng Chen, Zhao Fu, Dawei Yang, and Jia Deng. Single-image depth perception in the wild. In *Advances in Neural Information Processing Systems*, pages 730–738, 2016.

[4] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3354–3361. IEEE, 2012.

[5] Ashutosh Saxena, Sung H Chung, and Andrew Y Ng. Learning depth from single monocular images. In *Advances in neural information processing systems*, pages 1161–1168, 2006.

[6] David Eigen, Christian Puhrsch, and Rob 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.

[7] Dan Xu, Elisa Ricci, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Multi-scale continuous crfs as sequential deep networks for monocular depth estimation. In *Proceedings of CVPR*, 2017.

[8] Tai-sing Lee and Brian R Potetz. Scaling laws in natural scenes and the inference of 3d shape. In *Advances in Neural Information Processing Systems*, pages 1089–1096, 2006.

[9] Peng Wang, Xiaohui Shen, Zhe Lin, Scott Cohen, Brian Price, and Alan L Yuille. Towards unified depth and semantic prediction from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2800–2809, 2015.

[10] Beyang Liu, Stephen Gould, and Daphne Koller. Single image depth estimation from predicted semantic labels. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1253–1260. IEEE, 2010.

[11] Wei Zhuo, Mathieu Salzmann, Xuming He, and Miaomiao Liu. Indoor scene structure analysis for single image depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 614–622, 2015.

[12] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, volume 2, page 7, 2017.

[13] Ayan Chakrabarti, Jingyu Shao, and Greg Shakhnarovich. Depth from a single image by harmonizing overcomplete local network predictions. In *Advances in Neural Information Processing Systems*, pages 2658–2666, 2016.

[14] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4): 834–848, 2018.

[15] Yuanzhouhan Cao, Chunhua Shen, and Heng Tao Shen. Exploiting depth from single monocular images for object detection and semantic segmentation. *IEEE Transactions on Image Processing*, 26(2):836–846, 2017.

[16] Omid Hosseini Jafari, Oliver Groth, Alexander Kirillov, Michael Ying Yang, and Carsten Rother. Analyzing modular cnn architectures for joint depth prediction and semantic segmentation. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 4620–4627. IEEE, 2017.

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

[18] N. Silberman and R. Fergus. Indoor scene segmentation using a structured light sensor. In *Proceedings of the International Conference on Computer Vision - Workshop on 3D Representation and Recognition*, 2011.

[19] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[20] David Marr and Tomaso Poggio. A theory of human stereo vision. Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE ARTIFICIAL INTELLIGENCE LAB, 1977.

[21] Jeff Michels, Ashutosh Saxena, and Andrew Y Ng. High speed obstacle avoidance using monocular vision and reinforcement learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 593–600. ACM, 2005.

[22] Isabelle Bülthoff, Heinrich Bülthoff, and Pawan Sinha. Top-down influences on stereoscopic depth-perception. *Nature neuroscience*, 1(3):254, 1998.

[23] Bing Wu, Teng Leng Ooi, and Zijiang J He. Perceiving distance accurately by a directional process of integrating ground information. *Nature*, 428(6978):73, 2004.

[24] Stephan Liwicki, Christopher Zach, Ondrej Miksik, and Philip HS Torr. Coarse-to-fine planar regularization for dense monocular depth estimation. In *European Conference on Computer Vision*, pages 458–474. Springer, 2016.

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[26] Kevin Karsch, Ce Liu, and Sing Bing Kang. Depth transfer: Depth extraction from videos using nonparametric sampling. In *Dense Image Correspondences for Computer Vision*, pages 173–205. Springer, 2016.

[27] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5162–5170, 2015.

[28] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 239–248. IEEE, 2016.