Semantic Segmentation and Single Image Depth Estimation

Final Report

Presented in Partial Fulfillment of the Requirements for the Degree Master of Science in the

Graduate School of The Ohio State University

By

Jiangtian Pan, B.S.

Graduate Program in Electrical and Computer Engineering

The Ohio State University

2019

Final Exam Committee

Irem Eryilmaz, Advisor

Lisa Fiorentini

Table of Contents

Section 1. Executive summary

Estimating the depth of a single image is a challenging task which has attracted researchers increasingly in recent years. When the three-dimensional data is projected into a two-dimensional space, information of the third dimension is lost. Hence, without the geometric constraint information of the 3D-to-2D projection, single depth estimation is a challenge.

One approach to solve this problem is called Binocular depth estimation method. If we have two images from different perspectives of the same scene, we can get the matching points by feature matching, and then the depth of the matching points can be estimated through triangulation.

However, estimating the depth of a single image, without the contra-geometric constraint information, also without the overall information of the scene, is a challenging task. Moreover, the main difficulty in this task is to estimate the depth of the image accurately. In this report, we propose a method to use semantic segmentation to enhance the performance of depth estimation. Also, we apply our method to our proposed model and evaluate it on two other different models.

The rest of this report will be organized as follows. Section 2 introduces the knowledge about Convolutional Neural Network (CNN) based Depth Estimation task and literature review of it. Section 3 introduces the project goal and objectives. Section 4 introduces some commonly used criteria for the evaluation of Depth Estimation task. Section 5 introduces our proposed method and the challenge of it. Section 6 is the result

of our project, based on three different models and their result analysis. Section 7 is the conclusion part of this report.

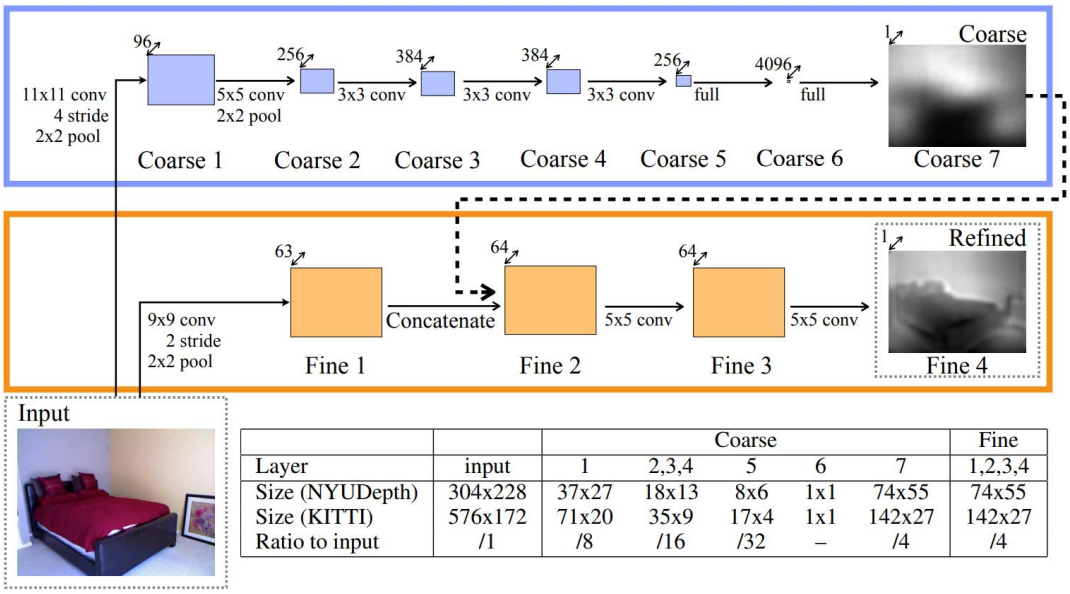
Section 2. Introduction of Depth Estimation Task

Depth estimation aims to predict the third-dimensional information from a two-dimensional image. It is a challenging task which has attracted researchers increasingly in recent years. When the three-dimensional data is projected into a two-dimensional space, information of the third dimension is lost. Without the geometric constraint information of the 3D-to-2D projection, single depth estimation is a challenge.

This section detailly introduces the best CNN based models for single image depth estimation in recent few years. In this section, we mainly introduce CNN based models.

2.1. Depth Map Prediction based on a Multi-Scale Deep Network [1]

In NIPS 2014, a straightforward method is proposed. In this model, the image can be sent directly into CNN for estimation, and the network can directly return the depth estimate. It divides the estimation process into two parts. It first estimates the global structure of the scene and then using local information for fine-tuning. The structure of this model is shown in Figure 1.



		Coarse					Fine
Layer	input	1	2,3,4	5	6	7	1,2,3,4
Size (NYUDepth)	304x228	37x27	18x13	8x6	1x1	74x55	74x55
Size (KITTI)	576x172	71x20	35x9	17x4	1x1	142x27	142x27
Ratio to input	/1	/8	/16	/32	–	/4	/4

Figure 1. The model structure of [1]

This model is a coarse to the fine based structure. The coarse-scale network is in the purple box part, which is the standard AlexNet [2]. Its task is to predict the overall structure of the depth map by using the global feature of the scene (such as vanishing point, object position, room alignment information). This kind of information is only available from a global perspective.

On the contrary, the fine-scale sub-network is in the orange box. Its task is to modify the coarse prediction received so that it is aligned with the edges of the object and the wall.

This model only contains the convolutional layer, and there is a pooling operation in the first convolutional part. After being input into the first layer of the convolutional part, the input image is merged with the depth map generated by the coarse-scale network and then passed through two convolutional layers to obtain a fine-tuned depth map.

2.2. Predicting Depth with a Common Multi-Scale Convolutional Architecture [3].

In ICCV 2015, Eigen et al. proposed a three-scale model based on the model they proposed in NIPS 2014. The structure of this model is remarkably similar to the previous one. However, it uses a deeper network (VGG-16) as the backbone and switches the course-to-fine structure to a three-scale structure, which can obtain a larger output image.

In scale-1 part, the input of this scale is the full RGB image. It uses a VGG or AlexNet network to extract the feature map. Then it uses the 1x1 convolution kernel to reduce the dimension and reduce the calculations.

The scale-2 is set for course prediction of the depth map. The input is the ground truth's depth and the normal vector. They tried to use the prediction to get depth and normal, but the joint training scale1 and scale2 get almost no improvement, but depth and norms can be used to train the semantic scale2 separately.

As for the scale-3, it is set for up-sampling to obtain a higher accurate output.

2.3. Height estimation from fully residual convolutional-deconvolutional network [4].

In this report, a full convolution-deconvolution network architecture is proposed, which is trained end-to-end. This model contains residual layer to simulate a fuzzy mapping between a monocular remote image and a height map. Precisely, it consists of two parts, a convolution subnet, and a deconvolution subnet.

The convolution subnet corresponds to a feature extractor that transforms the input remote sensing image into an advanced multidimensional feature representation, while

the latter acts as a height generator that generates a height map based on features extracted from the convolution subnetwork.

Furthermore, to preserve the fine edge detail of the estimated height map, a skip connection structure is also added that can pass through the low-level visual information. For example, the object boundaries and edges can be obtained directly over the network through the skip connections.

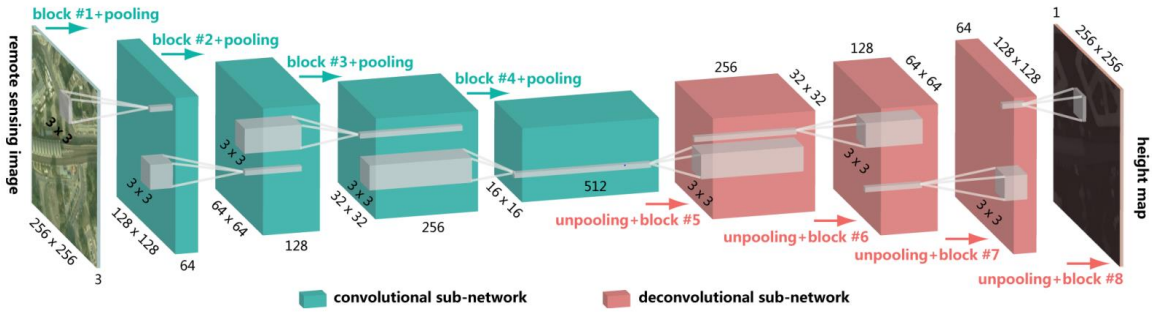


Figure 2. The structure of IM2HEIGHT model [4].

2.4. MegaDepth: Learning Single-View Depth Prediction from Internet Photos [5].

Compared with the previous research, this model first time combined the single depth estimation with the semantic segmentation task. They propose three new ideas for semantic segmentation in dataset generation. First, they use the segmentation results to remove the false MVS depth in the foreground region. Second, they use segmentation as a standard label to classify each photo to provide Euclidean depth or ordinal depth. Moreover, they combine semantic information with MVS depth and mark the ordinal depth relationship to help the area where MVS cannot be reconstructed.

In this research, they first use the semantic segmentation result for filtering. They use PSPNet [6] to obtain the semantic result, and then divide them into the foreground

object, background object, and sky. Besides that, the architecture of the model is constructed with an Hourglass [7] based backbone.

Section 3. Project Goal and Objectives

The depth estimation task is becoming increasingly important with the development of the self-driving car in recent years. Taken real-time camera-generated-images as the input, the depth estimation system can predict the distance of the objects behind the car. And this technology is crucial for autonomous driving, autonomous braking, and collision avoidance.

Since the CNN based method was firstly proposed in NIPS 2014 [1], deep learning methods become increasingly popular for depth estimation. However, at present, the depth estimation from a single image is still not developed well enough to be applied to use in real life. Since recovering the third-dimensional information from a two-dimensional image is still a challenge for us. People are interested in building a deeper network and modifying the network structure to improve the performance of depth estimation. However, designing a deeper model means more powerful hardware is required for calculation, and also, the deeper model means more difficult to generate real-time depth information. Hence, we try to search for another way to deal with this task.

We try to adopt the multi-task approach and use the information from segmentation to enhance the performance of depth estimation. The objective of this project is firstly designing an encoder model that can obtain features for both depth estimation and segmentation. Secondly, we will design a decoder model that can use the features generated in the segmentation task to boost the performance of the depth estimation. Also, in the project, multi experiments will be done to evaluate the generation of this method.

Section 4. Success evaluation criteria for deliverables

The most commonly used metrics for evaluating the performance of depth estimation tasks are as follows. They are RMSE (root mean square error), log RMSE (root mean square logarithmic error), abs rel (absolute relative error) and δ (accuracy under a threshold) are adopted to evaluate the depth estimation. Their mathematical formula can be described from (18) to (21).

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum\left(y_{pred} - y_{gt}\right)^2} \tag{1}$$

$$\log \text{RMSE} = \sqrt{\frac{1}{n}\sum\left(log\left(y_{pred}\right) - log\left(y_{gt}\right)\right)^2} \tag{2}$$

$$\text{abs rel} = \frac{1}{n}\sum\left|\frac{y_{pred} - y_{gt}}{y_{gt}}\right| \tag{3}$$

$$\max\left(\frac{y_{gt}}{y_p}, \frac{y_p}{y_{gt}}\right) = \delta < \text{threshold} \quad (4)$$

In particular, y_{pred} and y_{gt} are the predicted depth value and the true depth value of each pixel. *Abs rel, RMSE* and *log RMSE* are used to describe the difference between the predicted data and the ground truth. *Accuracy under a threshold* is used to estimate the percentage of accurate data under different standards. In general, the values adopted for the threshold are 1.25, 1.25^2, 1.25^3, where 1.25 is the most accurate and 1.25^3 is the least accurate.

Currently, researchers have designed lots of new models for depth estimation and they have achieved great success in recent years. For example, in [8], the state of the art method in CVPR 2014, Liu et al. have obtained 0.335, 0.137, 9.49 accuracy on the criteria *abs real*, *log RMSE* and *RMSE*. And up to now, the state-of-the-art method has obtained an accuracy of 0.136, 0.210 and 6.127 based on criteria of *abs real*, *log RMSE* and *RMSE* [9].

In this project, we aim to use Semantic Segmentation approach to boost the performance of Depth Estimation task. In general, Semantic Segmentation is a pixel level classification task. The system can generate the class of each pixel based on the RGB input image. In order to evaluate our method, we decide to compare the accuracy with and without depth estimation task. If the Semantic Segmentation task can help to obtain a more than 10% accuracy boosting on several different models (our proposed model included), we think our approach can work well.

Section 5. Implementation approach

In this section, we mainly introduce our proposed model in detail, including the model structure, the detailed parameters of the model and the theoretical analysis of the feasibility of our proposed model.

5.1. Model Description

The structure of our proposed model is a CNN based encoder-decoder network. It contains two parts, and one is the encoder part, a sub-convolutional network that contains five blocks. The other one is the decoder part; a sub-deconvolutional network that contains five blocks either. It is shown in Figure 3 as below.

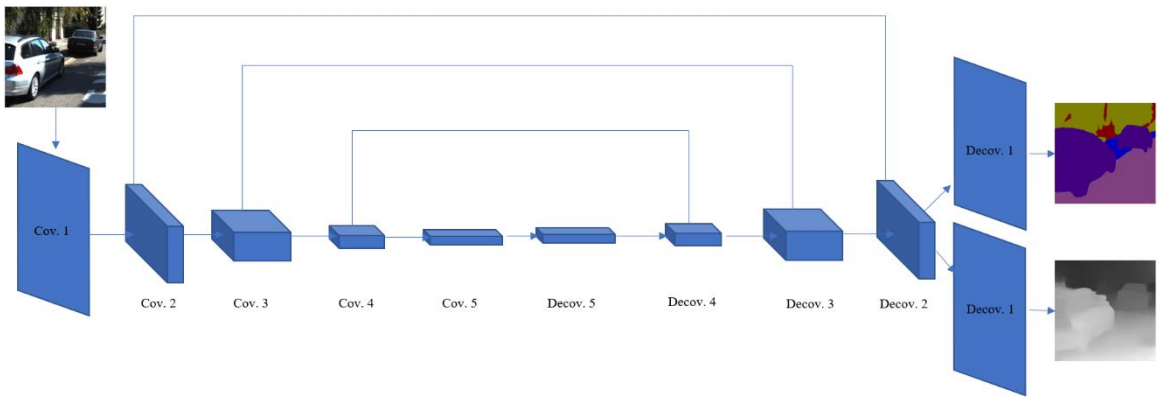


Figure 3. The structure of the proposed model.

Each convolutional block contains three layers including convolutional layer, ReLu activation layer, and max-pooling layer. The convolution layer is set to extract the features of images. The first block extracts low-level image information, like the edge, the angle. The following blocks extract high-level information, like the texture and material information. However, after passing through several convolutional blocks,

the low-level information may get lost, and only the high-level information remains. The problem is that in the depth estimation, the low-level information is as significant as the high-level information. Without edge information, the profile of the objects can be lost, which will cause a bad estimation.

Considering the potential problems described above, we decide to use a skip connection between the encoder and the decoder part. Due to the skip connections, features can be directly transferred from one block to another one. Hence, those features extracted from each layer in the sub-convolutional block will be transferred into the corresponding sub-deconvolutional block, which features map has the same dimension with the transferred one. In this way, the potential low-level information can remain in the deconvolutional process. Moreover, the details of the model structure and parameters are in Table 1 and Table 3.

In Table 1, our encoder sub-network contains five convolutional blocks. In each block, a convolutional layer with a ReLU activation layer and followed by a max-pooling layer. To maintain the shape of the input and output of each layer, we use padding with 1 pixel and use a 3×3 convolutional kernel with stride equals to 1.

	Name	input size	kernel size	stride	output size
Conv. Block 1	Conv 1	256, 256, 3	3, 3, 3, 64	1	256, 256, 64
	ReLU				
	max-poling 1	256, 256, 64	2, 2	2	128, 128, 64
Conv. Block 2	Conv 2	128, 128, 64	3, 3, 64, 128	1	128, 128, 128
	ReLU				
	max-poling 2	128, 128, 128	2, 2	2	64, 64, 128
Conv. Block 3	Conv 3	64, 64, 128	3, 3, 128, 256	1	64, 64, 256
	ReLU				
	max-poling 3	64, 64, 256	2, 2	2	32, 32, 256
Conv. Block 4	Conv 4	32, 32, 256	3, 3, 256, 512	1	32, 32, 512
	ReLU				
	max-poling 4	32, 32, 512	2, 2	2	16, 16, 512
Conv. Block 5	Conv 5	16, 16, 512	3, 3, 512, 1024	1	16, 16, 1024
	ReLU				
	max-poling 5	16, 16, 1024	2, 2	2	8, 8, 1024

Table 1. The structure of our convolutional sub-network.

Take the first convolutional layer, for example, the input of it is an RGB image with a shape of $256 \times 256 \times 3$. After adding the 1-pixel padding, the shape is $258 \times 258 \times 3$. We use 64 convolutional kernels with the shape of $3 \times 3 \times 3$ to slide and scan on the image and do the convolution between the kernel and the corresponding pixel. Hence, each kernel will generate a 256×256 output. With the output of 64 kernels, we obtain an output with the shape of $256 \times 256 \times 64$. Then, we choose to use a kernel with the size of 2×2 and stride equals 2. In this way, the kernel slides and scans the input data without any overlapping area. With the max-pooling

calculation, the input data is changed from $256 \times 256 \times 3$ to $128 \times 128 \times 64$. With five convolutional blocks implemented, we can obtain a feature map with a shape of $8 \times 8 \times 1024$.

	Name	input size	kernel size	stride	output size
Conv Block 5	up-conv 5	8, 8, 1024	2, 2	2	16, 16, 1024
	Conv 6	16, 16, 1024	3, 3, 1024, 512	1	16, 16, 512
	ReLU				
Conv Block 4	up-conv 4	16, 16, 512	2, 2	2	32, 32, 512
	Conv 7	32, 32, 512*2	3, 3, 1024, 256	1	32, 32, 256
	ReLU				
Conv Block 3	up-conv 3	32, 32, 256	2, 2	2	64, 64, 256
	Conv 8	64, 64, 256*2	3, 3, 512, 128	1	64, 64, 128
	ReLU				
Conv Block 2	up-conv 2	64, 64, 128	2, 2	2	128, 128, 128
	Conv 9	128, 128, 128*2	3, 3, 256, 64	1	128, 128, 64
	ReLU				
Conv Block 1	up-conv 1	128, 128, 64	2, 2	2	256, 256, 64
	Conv 10	256, 256, 64*2	3, 3, 128, 32	1	256, 256, 32
	ReLU				
Depth task	Conv 11	256, 256, 32	1, 1, 32, 3	1	256, 256, 3
Seg. task	soft-max	256, 256, 32	N/A	N/A	256, 256, 8

Table 2. The structure of our deconvolutional sub-network.

Also, the detail of the deconvolutional blocks is shown in Table 2. In the deconvolutional block, we use the up-sampling method for recovering the feature map. Similar to the convolutional block, we set one up-convolutional layer followed with a convolutional layer with a ReLU activation layer. To map the feature information into the correct location, we need the information of each max-pooling layer in all convolutional parts.

Notably, we need to set the last layer for different tasks. For the depth estimation task, it is a convolutional layer to mapping the feature into a depth image with the same size as the original image. Meanwhile, the last layer for the semantic segmentation task is set as a soft-max layer. It can output the possibility of each pixel corresponding to the classes.

5.2. Dataset

We choose to train and test our model on the KITTI dataset [10]. KITTI dataset is captured by driving around the mid-size city of Karlsruhe, in rural areas and on highways. Also, in this dataset, all data in raw format is provided in several tasks, including depth estimation and semantic segmentation, which are all required in this project.

In our model, the training set and testing set are all required for semantic segmentation and depth estimation. Hence, the KITTI dataset is the optimal one for us. We download 400 images for semantic segmentation, 1000 images for depth estimation.

The semantic segmentation of KITTI benchmark consists of 200 semantically annotated training as well as 200 testing images. The data format and validate metrics (to calculate the validate accuracy) are confirmed with The Cityscapes Dataset. Also, the segmentation dataset contains eight classes in each image, including resident, vehicle, road, sidewalk, building, tree, road signal, and sky.

The depth estimation benchmark contains over 93 thousand depth maps and the corresponding RGB images, aligned with the "raw data" of the KITTI dataset.

5.3. Training Details

With only 400 images provided in semantic segmentation task, we need to do the data augmentation. There are 1242*375 pixels in each original image, and we crop them into 256*256 pixels of each image with 10% overlap. Hence, we obtain a dataset of 4800 images. Then, we split the 4800 images into two parts for the semantic segmentation task: we use 4000 images for training and 800 images for testing.

Meanwhile, there are over 14 GB images in KITTI dataset for depth estimation task. We manually screen out 400 images which are matched with the dataset for semantic segmentation task. Furthermore, we crop and split the depth images in a way, which has been introduced in semantic segmentation task, to obtain 4000 images for training and 800 images for testing.

Furthermore, we set the hyperparameters as Shuffle = True, Rotation = True in training to make the input image randomly shuffle and rotate to augment the training data and avoid overfitting.

Training is done end-to-end, with the semantic segmentation task and depth estimation task processing at the same time. As described in Section 4, the final layer of the segmentation task is a soft-max layer for classification, and the final layer of the depth estimation task is a convolutional layer for regression.

Moreover, the loss function of each task is well designed in formula (5) and (6).

$$L_{seg} = -\frac{1}{m}\sum_{i=1}^{m}\left[-\frac{1}{N}\sum_{j=1}^{N} y_{gt}^{i,j}\log\left(y_{pred}^{i,j}\right)\right] \quad (5)$$

$$L_{depth} = -\frac{1}{N}\sum_{i=1}^{N}\left|y_{gt}^{i} - y_{pred}^{i}\right| \quad (6)$$

In particular, y_{pred} and y_{gt} are the predicted depth value and the true depth value of each pixel. The (5) is a pixel-wise cross entropy loss, which is a standard segmentation loss function. The *m* stands for the total number of classes, and the *N* stands for the total number of pixels. It computes the logistic loss between the actual value and the predicted value of *N* pixel in *m* classes. Meanwhile, the loss function (6) is a pixel-wise *L1-loss*, which computes the difference between predicted values and ground-truth value of each pixel. Moreover, the total loss function is a combination of the segmentation loss and the depth estimation loss in (7).

$$L_{total} = \lambda L_{seg} + (1 - \lambda) L_{depth} \quad (7)$$

The total loss function contains the segmentation loss and the depth estimation loss with a weight of λ. In our experiments, we set the λ as 0.5 for simplicity.

As for the optimization part, several optimizers have been proposed in recent years. At present, Adam [11] is the most popular one for gradient descent. The parameters in our training are listed in Table 3.

α	$\beta 1$	$\beta 2$	ε
0.001	0.9	0.999	10^{-8}

Table 3. The set of parameters for Adam optimizer.

In Adam optimizer, α stands for the learning rate, which is set to control the speed of the gradient descent. $\beta 1$ stands for the 1st order exponential decay rate, $\beta 2$ stands for the 2nd order exponential decay rate, and they are set to correct the deviation of the descent. ε is a small parameter which sets to avoid the exponential decay mean to

become 0. Also, to guarantee an efficient training, we set the α changing with the epoch number *i* in formula (8).

$$\alpha_i = \alpha_0/\sqrt{i} \tag{8}$$

With the epoch number *i* increasing, the learning rate will be reduced at a decreasing rate. This can make the loss function converge fast at the beginning to reduce the training time. Also, it can make the loss function converge slowly to find the minimum solution.

Furthermore, with a GTX 1080 8G GPU processes, we set the epoch number as 50 and set the iteration number as 100 in each epoch. The batch size of training is set as 2, which will consume 6.9G memory of the GPU.

5.4. Rationality Analysis

We decide to use the semantic segmentation task to boost the accuracy of the depth estimation task. The reasons make us believe this idea can work well based on the theoretical supports as below.

Firstly, the models for depth estimation are also generals for semantic segmentation task. In our previous research, we found that some models designed for depth estimation are modified from the model proposed for the segmentation task. For example, in [13], they use FCN as the backbone combined with residual connections for depth estimation. Also, in [14], J Jiao, Y Cao and et al. add their proposed mechanisms on the RefineNet [12] to obtain the depth map. Furthermore, in our

previous research, we find the U-net, introduced in Section 4, also performs well on depth estimation for remote sense images.

Secondly, the training mechanism of the network guarantees that a network can be trained for different tasks. In the BP process, the weights of the network will update based on the loss function. Hence if the loss function is corresponding to both tasks, the network can be trained to fit both segmentation and depth estimation.

Moreover, the feature maps of the segmentation and depth estimation are partially similar including the outline, edge, color, context and so on. The difference is that the segmentation task has better performance to extract outline and context features whereas depth estimation task performs better in extracting some high-level information like locations. However, the low-level information like outline and edge are equally crucial for the two tasks. Hence, we think that the training process of segmentation can help to maintain the outline information in the feature map, which may get lost with the depth task alone.

Finally, we think that the priority information from segmentation can boost the accuracy of the depth estimation. For example, all depth value of one object should not vary too much in a small area. Moreover, by segmentation, networks can learn the outline features, which is also necessary for the depth estimation task. Furthermore, the segmentation task can distinguish the foreground and the background. Meanwhile, in the depth estimation task, knowing the foreground, midground and background can be useful to constrain the range of depth values directly. For example, the depth values of foreground objects should be much smaller than that of the background objects.

Hence, we use the same structure for both segmentation and depth estimation task except the last layer of each task. We use the soft-max layer and cross-entropy loss for segmentation. Also, we use the convolutional layer and $L1$-loss for depth estimation.

5.5. Challenges and Solutions

The biggest challenge in our project is to design the experiments based on our goal and objectives. To show that the Semantic Segmentation task can do boost the performance of the Depth Estimation task, we need to carefully design the testing group to compare the results with and without Depth Estimation task.

Moreover, we need to explain the rationality of our method from a theoretical perspective. However, until now, the development of deep learning is mostly based on experiments and observation. With limited mathematical support, people can only try to explain the principles of it based on the observation results. Hence, it is a challenge to explain our rationality from the mathematical perspective.

To solve the two principal challenges mentioned above, at first, we apply our method to three different models. One is our proposed model and two others are all popular models in the field of Depth Estimation and Semantic Segmentation. And we will calculate the accuracy boosting from the Semantic Segmentation task.

Secondly, we try to describe the rationality by referring to other researchers' theory and experiments. Also, we try to explain our approach from the principal of convolutional neural networks. This can be seen from the Rational Analysis part and the Results Analysis part.

Section 6. Results and Evaluation.

In this part, we introduce the results of our experiments. Mainly, we compare the results with and without segmentation task to prove to evaluate our approach that segmentation can be used to enhance the performance of segmentation.

6.1. Experimental Results.

The testing results of our proposed model are shown in Figure 4. From left to right, they are the original image, segmentation label, depth estimation with segmentation task, depth estimation without segmentation task and the ground truth.

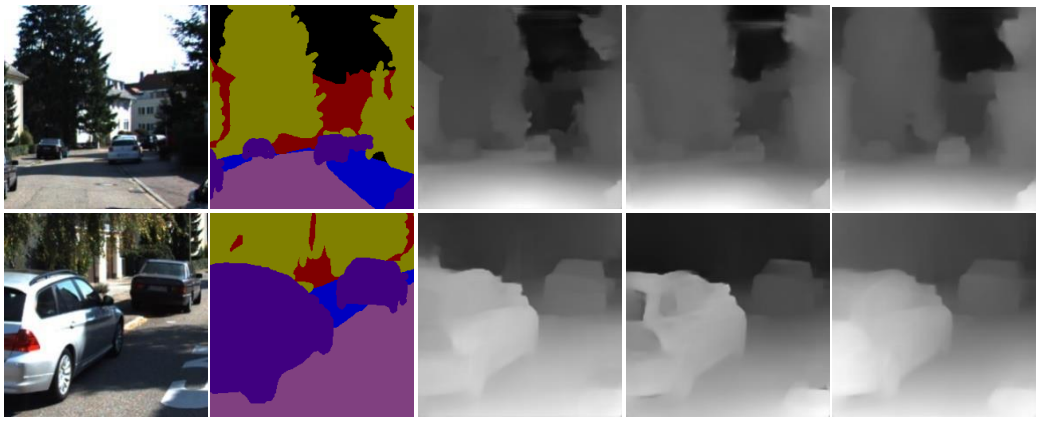


Figure 4. The results of the proposed model.

We observe that the segmentation task is effective to separate the foreground and background. Hence, the depth estimation task can enhance accuracy when predicting the depth value of the foreground and background.

Specifically, in the first row, it can be observed that the depth of the white car should be different from the background house. However, without segmentation, its

depth value is predicted close to the house. Also, with the segmentation task, the depth values of the car and the house are distinguishable by the model. It is observed that with segmentation, the depth estimation can perform better on the objects with a common class number.

Specifically, in the second row, it can be observed that with the segmentation task, all the pixels of the car will be fixed into the same class by segmentation. Hence, the values of its depth should not vary too much. It makes the depth estimation robust to the color, light, and shadow in some cases. However, the results from the original model are less accurate with the interference of shadow and light. Precisely, with the color varies between the window and the car, the depth of the car is predicted into two different sets. On the contrary, with the segmentation task, they are considered as the same class, rather than based on the color and shadow. Hence, the depth value of them is predicted more accurately.

To make our evaluation more reliable, we use several popular indices to evaluate the accuracy of depth estimation which has been introduced in section 4. Their mathematical formula can be described from (1) to (4).

The estimated results corresponding with the indices are in Table 4.

For results in the blue blocks, less is better and for results in the gray blocks, larger is better. These results are obtained from the evaluation matrix provided on the KITTI website. From the evaluation indices listed above, it is observed that the depth evaluation can become more accurate with the segmentation task.

	Depth with Segmentation	Depth without Segmentation
RMSE (linear)	**6.07**	6.49
RMSE (log)	0.229	**0.216**
abs rel	**0.183**	0.197
$\delta < 1.25$	**0.741**	0.725
$\delta < 1.25^2$	**0.896**	0.867
$\delta < 1.25^3$	**0.972**	0.968

Table 4 The statistic of depth estimation performance.

Moreover, it is observed that with the segmentation task, the model can perform better when implementing the background objects in this model, like buildings located far away and the sky.

6.2. General Evaluation

Our proposed method, using a segmentation task to enhance the accuracy of depth estimation task, shows a noticeable effect in our proposed model. Considering that our model is just a basic encoder-decoder one, the results may not be convincing enough. Hence, we further use our method in some popular models to evaluate the general of it, including the U-net [15] and SegNet [16], which are the most popular networks for segmentation task.

For the training details, the loss functions for segmentation and depth estimation are consistent with our proposed model. Also, the optimizer is Adam with parameters as $\alpha = 0.001$, $\beta 1 = 0.9$, $\beta 2 = 0.999$, $\varepsilon = 10^{-8}$ and $\alpha_i = \alpha_0/\sqrt{i}$.

To make the U-net suitable for both segmentation and depth estimation. A pair of layers is designed as the last layer. One is a 2×2 up-convolutional layer, and the other is a soft-max layer. As the class number of KITTI dataset is 8, the dimension of the soft-max layer is set as 8.

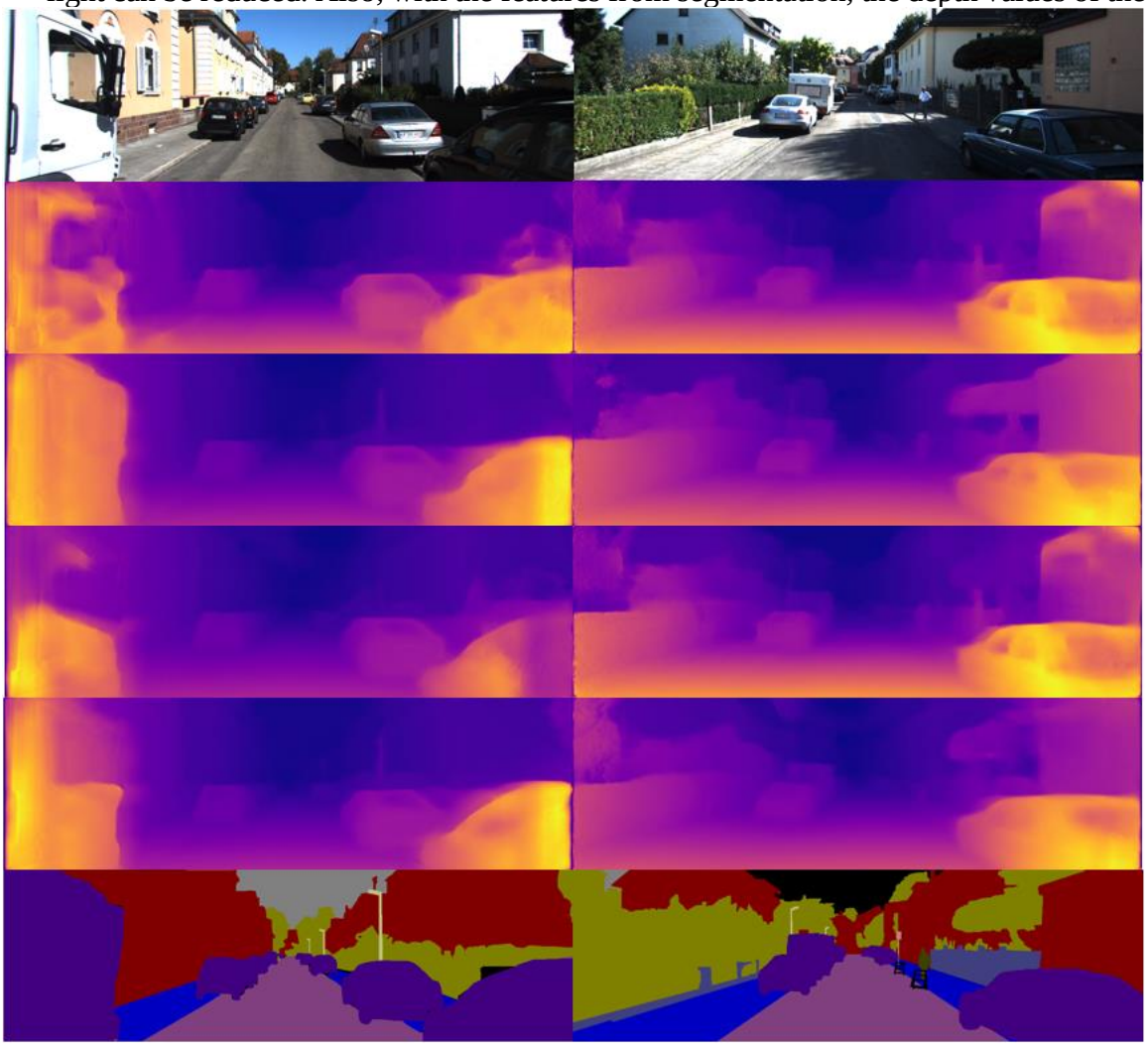
On the contrary, the last layer of the SegNet is a soft-max layer. Hence, we add a convolutional layer with a 1×1 kernel and keep it parallel with the soft-max layer for multi-tasks training.

The testing results are in Figure 5. From above to bottom, they are RGB image, SegNet without segmentation, SegNet with segmentation, U-net without segmentation, U-net with segmentation and the segmentation label provided in KITTI dataset.

For the left set of images, it can be observed that the window of the truck is transparent, and the depth of it may highly possible to be predicted the same depth with the objects behind the window. However, with the features from the segmentation, the whole truck will be recognized as one part. This can make the depth prediction more reliable when dealing with objects that are transparent or easily affected by color, light, and shadow.

For the right column of images, the car located at the right side of the image also equipped with windows. However, the windows of this car are opaque, and the object behind it cannot be seen. From the test results, it can be observed that with the feature

of segmentation, the car is also recognized as one object. Thus, the effects of color and light can be reduced. Also, with the features from segmentation, the depth values of the



tree in front of the wall on the right side of the image are more accurate.

Figure 5. The test results of U-net and SegNet with and without segmentation.

To evaluate the accuracy of the depth with and without the segmentation task. We adopt indices of RMSE (root mean square error), log RMSE (root mean square logarithmic error), abs rel (absolute relative error) and δ (accuracy under a threshold) again to evaluate the SegNet and U-net with and without segmentation. The evaluation results are in Table 5.

	SegNet		U-net	
	Depth with Segmentation	Depth without Segmentation	Depth with Segmentation	Depth without Segmentation
RMSE (linear)	**3.13**	3.49	**4.92**	5.17
RMSE (log)	**0.184**	0.205	**0.214**	0.225
abs rel	**0.153**	0.167	**0.174**	0.187
$\delta < 1.25$	0.825	**0.829**	**0.817**	0.795
$\delta < 1.25^2$	**0.937**	0.904	**0.902**	0.891
$\delta < 1.25^3$	**0.961**	0.959	**0.956**	0.943

Table 5. The evaluation results of SegNet and U-net.

In Table 5, it is observed that the overall performance of SegNet is better than U-net. Moreover, the performance of depth estimation with segmentation is better than that without segmentation. However, we also find that for the evaluation index $\delta < 1.25$, SegNet without segmentation performs the best, and we suppose that the segmentation can implement with the interference from light, shadow and color, but may not so effective on some other situations.

6.3. Results Analysis

In this experiment, we apply our method to three different models, including our proposed model, U-net and SegNet. Our approach works well on these three different models. Specifically, the accuracy of depth estimations is ranked as SegNet > U-net > our model. Since our model only contains ten convolutional layers in total while the other two models are much deeper. However, the enhancements of segmentation are all effective on three different models.

Note that segmentation can be effective to deal with the outline information for depth estimation. Since it is observed that with the features of segmentation, objects can be distinguished from its background. Hence, the depth values of the objects are more accurate. On the contrary, without segmentation, some objects may be predicted as the same depth value as its background.

Moreover, it is observed that with segmentation, depth estimation can be more robust to the noise information like color, light, and shadow. In some cases, the reflection of light on the dark window can disturb the judgment of the network. For example, it is difficult to estimate the depth of an opaque window. However, with the segmentation information, regardless of these noises, a reasonable depth map can be predicted.

Furthermore, our method may weaken the depth estimation in some certain occasions. Segmentation concerns the overall information of one object but aborts some internal details of parts of the object. Moreover, most objects in our world are in three-dimensional. In some rare cases, the 3D information of parts of an object is essential. However, the segmentation will weaken the third-dimensional information and make the object become a whole part. Hence, with the segmentation, it may be not reliable to predict the depth value of internals of an object.

However, this shortcoming is not so influential as we supposed before. Firstly, as a multi-task training, features are from both tasks, and none of it can become the decisive one. Secondly, we can adjust the weight parameter λ, to balance the weight of each task to make a trade-off between the advantages and the disadvantages from

segmentation. Moreover, this kind of circumstances is rare in the KITTI dataset, which is about road scenes. Hence, our method is still meaningful for the development of the autonomous vehicle.

Section 7. Conclusions

In this report, we propose a CNN based model for depth estimation. Particularly, we use the encoder-decoder model with skip connections to extract both low-level features and high-level features.

Moreover, we propose a multi-task approach to enhance the depth estimation. Both the segmentation and the depth estimation task can be trained simultaneously on the same network backbone. The last layer for segmentation is a soft-max layer for classification, while the last layer for depth estimation is the convolutional layer for regression.

Furthermore, we apply our method to three different models to evaluate its general. We test our method on SegNet, U-net, and our proposed model. In most situations, our method is effective.

Finally, we analyze the experimental results. It is observed that with segmentation, our method can be more effective when outline information is essential but less effective when internal details of objects are required. Also, our method is useful to keep the model more robust to the interference from some noises, like light, shadow, and color.

References

[1] Eigen, D., Puhrsch, C., & Fergus, R. (2014). Depth map prediction from a single image using a multi-scale deep network. In Advances in neural information processing systems (pp. 2366-2374).

[2] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

[3] Eigen, D., & Fergus, R. (2015). Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In Processings of the IEEE International Conference on Computer Vision (pp. 2650-2658).

[4] Mou, L., & Zhu, X. X. (2018). IM2HEIGHT: Height estimation from single monocular imagery via fully residual convolutional-deconvolutional network. arXiv preprint arXiv:1802.10249

[5] Li, Z., & Snavely, N. (2018, April). MegaDepth: Learning Single-View Depth Prediction from Internet Photos. In Processings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 2041-2050).

[6] Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017, July). Pyramid scene parsing network. In IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (pp. 2881-2890).

[7] Newell, A., Yang, K., & Deng, J. (2016, October). Stacked hourglass networks for human pose estimation. In European Conference on Computer Vision (pp. 483-499). Springer, Cham.

[8] Liu, M., Salzmann, M., & He, X. (2014). Discrete-continuous depth estimation from a single image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 716-723).

[9] Ramirez, P. Z., Poggi, M., Tosi, F., Mattoccia, S., & Di Stefano, L. (2018). Geometry meets semantics for semi-supervised monocular depth estimation. arXiv preprint arXiv:1810.04093.

[10] Uhrig, J., Schneider, N., Schneider, L., Franke, U., Brox, T., & Geiger, A. (2017). Sparsity invariant cnns. arXiv preprint arXiv:1708.06500.

[11] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[12] Lin, G., Milan, A., Shen, C., & Reid, I. D. (2017, July). RefineNet: Multi-path Refinement Networks for High-Resolution Semantic Segmentation. In CVPR (Vol. 1, No. 2, p. 5).

[13] Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., & Navab, N. (2016, October). Deeper depth prediction with fully convolutional residual networks. In 3D Vision (3DV), 2016 Fourth International Conference on (pp. 239-248). IEEE.

[14] Mousavian, A., Pirsiavash, H., & Košecká, J. (2016, October). Joint semantic segmentation and depth estimation with deep convolutional networks. In 3D Vision (3DV), 2016 Fourth International Conference on (pp. 611-619). IEEE.

[15] Ronneberger, O., Fischer, P., & Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241). Springer, Cham.

[16] Badrinarayanan, V., Kendall, A., & Cipolla, R. (2015). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. arXiv preprint arXiv:1511.00561.