Towards Good Practice for CNN-Based Monocular Depth Estimation

Zhicheng Fang[1] Xiaoran Chen[1] Yuhua Chen[1] Luc Van Gool[1,2]

[1]Computer Vision Laboratory, ETH Zurich [2]VISICS, ESAT/PSI, KU Leuven

zfang@student.ethz.ch {chenx, yuhua.chen, vangool}@vision.ee.ethz.ch

Abstract

Monocular depth estimation has gained increasing attention in recent years, and various techniques have been proposed to tackle this problem. In this work, we aim to provide a comprehensive study on the techniques widely used in monocular depth estimation, and examine their individual influence on the performance. More specifically, we provide a study on: 1) network architectures, including different combinations of encoders/decoders. 2) supervision losses, including fully supervised losses and self-supervised losses and 3) other practices such as input resolution. The experiments are conducted on two commonly used public datasets, KITTI and NYU Depth v2. We also provide an analysis on the errors produced by different models, to reveal the limitations of current methods. Furthermore, by a careful redesign, we present a model for depth estimation, which achieves competitive performance on KITTI and state-of-the-art performance on NYU Depth v2. Our code is publicly available at https://github.com/zenithfang/supervised_dispnet.

1. Introduction

Monocular depth estimation is a fundamental task in computer vision, which aims to estimate the depth of each pixel for an input image. It is closely related to many downstream applications, such as 3D modeling, robotics, autonomous driving, *etc*.

Over the years, various techniques have been proposed for monocular depth estimation. Early efforts mainly rely on hand-crafted features and probabilistic models [31, 32], while recent approaches [6, 7, 19, 20, 38] are mostly based on convolutional neural networks (CNNs) due to the strong performance of the learned representations. CNN-based models typically formulate depth estimation as a per-pixel regression problem which is then solved using fully convolutional networks (FCNs). In order to learn such networks, depth sensors are often used to collect the ground-truth depth values. However, collecting such labels can be costly in many cases. To alleviate the demand for ground-truth labels, self-supervised methods have been proposed to essentially leverage photometric error as the supervision source, either between temporal frames [4, 41], or between left and right images in the stereo setup [8, 10, 17].

Though these techniques have shown promising results for monocular depth estimation, they are usually evaluated with different configurations. This makes it arduous to directly compare the effectiveness of each component and identify the limitation of the current methods. For example, many works are trained with images of different resolutions while the resolution itself may be an influential factor for the benchmark performance.

Therefore, in this work, our aim is to provide an empirical analysis for each influential factor/component by decoupling the commonly used components in depth estimation methods and then examining their individual influence with controlled experiments.

Our contributions can be summarized as follows:

- We experimentally evaluate the influence of different components/factors for monocular depth prediction. For network architectures, we test different combinations of encoders/decoders. We also examine the effect of different supervision losses and other factors such as input image resolution, number of training images *etc*.

- To reveal the limitations of the current models and provide insights on model design, we analyze the errors made by different models. In more details, we perform studies on distributions of image-wise and pixel-wise error and also show the correlation on errors between different models.

- Driven by our study, we present a model for monocular depth estimation which achieves the state-of-the-art results on NYU Depth v2 and competitive results on KITTI.

2. Related Works

Classical Monocular Depth Estimation Early efforts for monocular depth estimation are mainly based on hand-crafted features and probabilistic methods. Saxena *et al*. [31, 32] proposed to use global features and multi-scale features with Markov Random Field (MRF). Achanta *et al*. [1] utilize constraint between neighboring pixels and Liu *et al*. [21] fuse semantic information into depth estimation.

Supervised Monocular Depth Estimation Driven by the efficacy of deep learning, the performance of depth estimation has been significantly improved by CNN-based methods [6, 7, 19, 20, 38]. Such models are typically trained with supervision signals from depth sensors [9, 27] or synthetic datasets. Eigen *et al*. [6] introduce convolutional networks (CNN) to depth estimation with dense depth map regression predicting target depth values. Specifically, the architecture consists of two parts, a coarse prediction network based on AlexNet [16] and a refinement network. The approach is later improved by techniques such as Conditional Random Fields (CRF) [23], hierarchical CRF refinement [20] and multi-scale CRF method [38]. Better losses are also proposed to improve the depth estimation, such as reverse Huber (Berhu) loss [28, 43], and ordinal regression loss [7]. Depth estimation can also be trained jointly with other tasks such as semantic segmentation [3, 40].

Self-supervised Monocular Depth Estimation To address the expenses and effort of label acquisition for depth supervision, self-supervised methods are developed. Garg *et al*. [8] propose to train models using the supervision from image alignment with an encoder-decoder architecture. Particularly, the stereo image is warped according to the predicted depths such that an image alignment loss can be obtained from the difference between the warped image and the ground-truth. This method is further extended by Godard *et al*. [10] with a loss calculated from both left and right images. Unlike these methods, Zhou *et al*. [41] propose to simultaneously predict both pose and depthand train the network with an alignment loss computed from the warped images for pose and depth. Kuznietsov *et al*. [17] combine the loss from supervised learning and unsupervised learning, and report further improvement on the prediction accuracy. Yang *et al*. [39] combine stereo training with video training. Godard *et al*. [12] propose a full resolution multi-scale loss.

3. Methodology

3.1. Problem Formulation

Firstly, we briefly introduce the formulation of monocular depth estimation. Monocular depth estimation aims to map an input image $I \in \mathbb{R}^{H\times W\times C}$ to an output pixel-wise depth map as $\hat{Y} \in \mathbb{R}^{H\times W}$, with H, W being the height and width of the image and C being the number of RGB channels. CNN-based methods typically establish such a mapping by learning a neural network F such that $\hat{Y} = F(I)$. To learn the parameters of such networks, direct supervision losses $L(\hat{Y})$ are applied on the output $\hat{Y}$ to approximate the ground-truth depth map Y. Self-supervision losses $L(I, \tilde{I})$ are applied on the reconstructed image $\tilde{I}$ which is synthesized from $\hat{Y}$ and other source images to approximate the target image I.

In this work, we study several aspects involved in monocular depth estimation. As for architecture, we research the influence of the encoder and decoder choice in terms of model performance. As for network parameters, we investigate the supervision loss with those parameters learned from training progress. Generally, supervision loss L is typically categorized as direct supervision loss $L(Y, \hat{Y})$ and self-supervision loss $L(I, \tilde{I})$. Thus, we first study different types of supervisions and their combinations and then further study different direct supervision loss functions $L(Y, \hat{Y})$ which are typically computed from $\hat{Y}$ and Y. Specifically, the supervision with multi-scale output of DispNet and $L(\hat{Y}, Y)$ of different forms are studied. In addition, we look into the influence of the input image size (*e.g*. H and W) and dataset size. Lastly, we evaluate the advantages, correlation and limitations of the methods at both image- and pixel-wise level with mapping functions F learned by different architectures and losses.

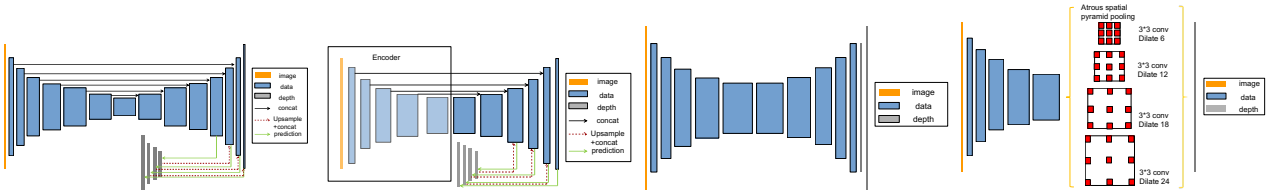
3.2. Network Architectures

Networks used for depth estimation typically follow an encoder-decoder design. We compare the performance with combinations of different encoder and decoder structures. We base the empirical study on the architectures of DispNet [26] and FCRN [19].

DispNet with ResNet-based and VGG-based encoders DispNet [26] is one of the most widely used architectures for depth estimation with the encoder-decoder structure, skip connections and multi-scale intermediate predictions as in Figure 1a. We adopt the original DispNet decoder architecture and replace its original encoder with different structures like VGG [34] and ResNet [14].

Unlike earlier works [10, 12, 17] that build DispNet with ResNet, we apply different settings for the skip connection and set the strides in the encoder as in Figure 1b. The encoder is based on VGG-16 [34] similarly to [13] and performs a 5-time down-sampling in the spatial direction. The decoder is symmetrical to the encoder with the same number of blocks and skip connections. Overall, VGG, VGG with batch normalization and ResNet are implemented as the encoder.

FCRN with ResNet-ASPP decoder Fully Convolutional Residual Network (FCRN) is another prevalent architecture



(a) DispNet (b) DispNet with other encoder (c) FCRN (d) ResNet-50 with ASPP

Figure 1: **Architecture overview**

proposed for depth estimation. The network comprises a decoder concatenated with a pre-trained ResNet-50. Specifically, the decoder has 4 up-convolution blocks with each block mapping each entry into the top left corner of a 2×2 kernel. After this mapping, it performs a convolution with initially a filter size of 5×5 and then 3×3, as well as a projection connection from the lower resolution feature map ss shown in Figure 1c. Note that FCRN is not equipped with skip connections.

Atrous Spatial Pyramid Pooling (ASPP) is firstly proposed in DeepLab v2 [2] for semantic segmentation, and shows promising performance improvement. Due to the success in modelling contextual information, we also test ASPP module with the ResNet-based encoder.

With the addition of ASPP to the ResNet-50-based encoder, we use a dilation rate of 6, 12, 18 and 24 in the decoder. Meanwhile, the dilation is introduced to the encoder to give a dense feature map for the ASPP module so that no skip connection is required between the encoder and the decoder.

3.3. Loss Functions

We categorize the proposed loss functions for depth estimation into direct supervision losses which require ground-truth depth and self-supervision losses which require no ground-truth depth. We briefly describe the losses of each type with the same notations as in subsection 3.1 and denote a pixel-wise depth prediction $\hat{Y}$ and ground-truth Y for the pixel i as $\hat{y}_i$ and y_i.

Losses for direct supervision To study the effect of direct supervision losses, we adopt the following losses: 1) scale invariant loss [6], 2) Berhu loss [28, 43], 3) ordinal regression loss [7].

Scale invariant loss Motivated by the scale ambiguity in depth prediction, scale invariant loss is defined as the mean of scale invariant error and ℓ_2 error (Eq.(1)).

$$\mathcal{L}_{SI}(Y, \hat{Y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 - \frac{1}{2n^2}\left(\sum_{i=1}^{n}(y_i - \hat{y}_i)\right)^2 \quad (1)$$

Berhu loss Berhu loss is defined as Eq.(2). As the definition shows, Berhu loss assigns large weights for samples with large residuals. More explicitly, it behaves similarly as ℓ_2 loss when the residual is above a threshold and equals ℓ_1 loss when the error is below the threshold which accounts for more impact than ℓ_2 loss.

$$\mathcal{L}_{BerHu}(Y, \hat{Y}) = \begin{cases} |Y - \hat{Y}|, & |Y - \hat{Y}| \leq c \\ \frac{(Y-\hat{Y})^2 + c^2}{2c}, & |Y - \hat{Y}| > c \end{cases}, \quad (2a)$$

$$c = \frac{1}{5} max_i(\hat{y}_i - y_i)\,. \quad (2b)$$

Ordinal regression loss Ordinal regression loss employs a Space-Increasing Discretization (SID) to discretize the depth. Given the depth interval $[\alpha, \beta]$ is discretized into K intervals, SID is formulated as:

$$t_i = e^{\log\alpha + \frac{\log(\beta/\alpha) * i}{K}} \quad (3)$$

where t_i is discretization threshold with $t_i \in t_0, t_1, ..., t_K$.

To train with the ordinal regression loss, we obtain the ordinal output $\hat{D}$ of size $2K$ with d_i denoted as the i-th component of $\hat{D}$. In addition, $l \in 0, 1, ..., K-1$ is the discretized target label produced by SID. Ordinal regression loss is defined as:

$$\mathcal{L}_{OR}(l, \hat{D}) = -\left(\sum_{k=0}^{l-1}\log(\mathcal{P}^k) + \sum_{k=l}^{K-1}\log(1 - \mathcal{P}^k)\right) \quad (4a)$$

$$\mathcal{P}^k = \frac{e^{d_{2k+1}}}{e^{d_{2k}} + e^{d_{2k+1}}} \quad (4b)$$

Losses for self-supervision Unlike direct supervision, self-supervision loss is computed with image reconstruction where the pixels in two images with similar RGB information are assumed to be corresponding pixels. Within this setting, it is desirable to have smooth changes in the output. A smoothness loss can be incorporated prevent pixels from matching to irregular positions. The smoothness can also be imposed by an appearance matching loss by using a similarity metric between the input image and warped image.

Smoothness loss In order to encourage smooth changes in disparity, the smoothness loss is proposed in the work [35]. The loss is configured as the sum of ℓ_1 norm of second-order gradient of disparity. Specifically, it is defined in Eq.(5) where $y_{i,j}$ is the prediction, i is the row ordinal number of $y_{i,j}$, j is the column ordinal number of $y_{i,j}$,

n is the number of pixels and k is the scale ordinal which represents final output when $k = 0$.

$$\mathcal{L}_{Smoothness}(y) = \frac{1}{n}\sum_{k=0}^{3}\sum_{i,j}^{n}\frac{1}{2^k}\left(\frac{\partial y_{ij}^2}{\partial i^2} + \frac{\partial y_{ij}^2}{\partial j^2} + 2\frac{\partial y_{ij}^2}{\partial ij}\right) \tag{5}$$

Appearance matching loss In our unsupervised training, we adopt the combination of SSIM [36] and ℓ_1 as in previous works [11, 12] as the similarity metric in the appearance matching loss. The loss is formulated as Eq.(6), where I_i is the input image pixel, $\tilde{I}_i$ is the reconstructed one and n is the number of pixels. Additionally, the SSIM here is a 3×3 block filter rather than a Gaussian one, and α is set as 0.85.

$$\begin{aligned}&\mathcal{L}_{AML}(I_i, \tilde{I}_i)\\ &= \frac{1}{n}\sum_{i=0}^{n}\alpha\frac{1-\text{SSIM}(I_i,\tilde{I}_i)}{2} + (1-\alpha)\left\|I_i - \tilde{I}_i\right\|\end{aligned} \tag{6}$$

4. Experiments

4.1. Experiment Setup

We evaluate the methods on KITTI [9] and NYU Depth v2 [27] to check the performance of the methods on both indoor and outdoor scenes.

KITTI The KITTI dataset contains outdoor scenes captured by cameras and depth sensors in a driving car, with a typical image with a size of 375 × 1242. During training, we resize the images to 128 × 416. Due to the training data requirement for different supervision methods, we use different number of images for training while the difference in data volume has no significant influence on the results according to our study in subsection 4.4. We train the network with 38564 samples and validate with 5164 samples for the supervised training. For the monocular video training, we have 39810 monocular triplets for training and 4424 for validation. For the stereo training, we have 45200 training pairs and 1776 validation pairs. We clamp the depth output to obtain a value range from 0 to 80 meters and evaluate the single-view depth performance on the 697 samples on the Eigen test split [6].

NYU Depth v2 The NYU Depth v2 dataset contains indoor scenes captured by Microsoft Kinect camera, with a typical image of 480 × 640. The images are then resized to 284 × 392 and randomly cropped to 256 × 352 during training. We train the network with 69837 images and evaluate the single-view depth performance on the 654 samples from the Silberman test split [27].

Implementation details We implement the methods using the publicly available PyTorch framework [29]. During training, we use Adam optimizer [15] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, learning rate of 0.0001 and a mini-batch size of 4. The iteration number should be set to 250K for KITTI and 750k for NYU Depth v2.

4.2. Architecture Study and Comparison with State-of-the-art

Despite the proposal of various architectures, the performances of the proposed methods are often evaluated in different settings. In order to understand the influence of architecture design, we experimentally test different architectures in the same setting on KITTI dataset.

Architecture study The effectiveness of different encoders and decoders is studies in the setting of direct supervision. According to Table 1, VGG-16 with batch normalization achieves the best result among all the encoders while DispNet decoder outperforms the other two decoders. Observing such results, we recommend pre-trained VGG-16 with batch normalization as the encoder and DispNet decoder as the decoder.

Comparison	Method	rel	rms	rms log	δ_1	δ_2	δ_3
		lower is better			higher is better		
encoder	Dispnet	0.139	5.416	0.232	0.810	0.930	0.971
	VGG-16 pt	0.109	4.583	0.190	0.867	0.958	0.982
	VGG-16 BN pt	**0.105**	**4.537**	**0.186**	**0.873**	**0.959**	**0.983**
	ResNet-18 pt	0.119	4.921	0.204	0.847	0.947	0.978
	ResNet-50 pt	0.111	4.762	0.197	0.861	0.951	0.979
	ResNet-101 pt	0.118	4.880	0.201	0.849	0.949	0.980
decoder	Dispnet	**0.111**	**4.762**	0.197	**0.861**	0.951	0.979
	FCRN	0.138	5.318	0.226	0.805	0.934	0.974
	ASPP	0.119	4.801	**0.196**	0.846	**0.953**	**0.982**

Table 1: **Comparison of different encoders and decoders.** Comparison of encoder is based on same Dispnet decoder and comparison of decoder is based on same pretrained ResNet-50 encoder. pt: using model weight pretrained on ImageNet [30], BN: batch normalization.

Performance on KITTI In Table 2, our network trained with ℓ_1 loss in the direct supervision setting and its variant are compared with the previous works on KITTI dataset. Our network achieves the second best performance.

Performance on NYU Depth v2 In Table 3, our network trained with ℓ_1 loss under direct supervision setting is compared with previous works on NYU Depth v2 dataset. Our network obtains the best performance among all works.

4.3. Supervision Study

Besides the architectures, the type of supervision is another key factor in depth estimation methods. Different supervision methods make use of different image information,

Method	supervision	rel	rms	rms log	δ_1	δ_2	δ_3
		lower is better			higher is better		
Saxena *et al.* [33]	direct	0.280	8.734	0.361	0.601	0.820	0.926
Eigen *et al.* [6]	direct	0.203	6.307	0.282	0.702	0.890	0.958
Liu *et al.* [23]	direct	0.201	6.471	0.273	0.680	0.898	0.967
Zhou *et al.* [41]	video	0.208	6.856	0.283	0.678	0.885	0.957
Fu *et al.* [7]	direct	**0.072**	**2.727**	**0.120**	**0.932**	**0.984**	**0.994**
Luo *et al.* [25]	stereo	0.127	5.008	0.209	0.841	0.946	0.979
Godard *et al.* [12]	stereo	0.109	4.960	0.209	0.864	0.948	0.975
Watson *et al.* (high reso) [37]	stereo	0.096	4.393	0.185	0.890	0.962	0.981
Kuznietsov *et al.* [17]	DS	0.113	4.621	0.189	0.862	0.960	0.986
Guo *et al.* [13]	DS	0.105	4.422	0.183	0.874	0.959	0.983
Our VGG-16 BN	direct	0.105	4.486	0.183	0.873	0.960	0.984
Our VGG-16 BN (high reso)	direct	0.098	4.075	0.174	0.889	0.963	0.985
Our VGG-16 BN (high reso)	DMS	0.096	3.966	0.167	0.893	0.969	0.987

Table 2: **Performance on KITTI.** $\delta_i : \delta < 1.25^i$. DS: direct and stereo supervision. DMS: monocular video and stereo supervision with direct supervision finetuning. High reso: the model is trained with image being 1024×320 pixels in size. Best results in each category are in **bold**; second best are underlined.

Method	rel	rms	rms log	δ_1	δ_2	δ_3
	lower is better			higher is better		
Ladicky *et al.* [18]	-	-	-	0.542	0.829	0.941
Liu *et al.* [24]	0.335	1.06	-	-	-	-
Zhuo *et al.* [42]	0.305	0.104	-	0.525	0.838	0.962
Li *et al.* [20]	0.232	0.824	-	0.621	0.886	0.968
Liu *et al.* [22]	0.230	0.824	-	0.614	0.883	0.975
Eigen *et al.* [6]	0.215	0.907	0.285	0.611	0.887	0.971
Eigen & Fergus [5]	0.158	0.641	0.214	0.769	0.950	0.988
Laina *et al.* [19]	0.127	0.573	0.195	0.811	0.953	0.988
Xu *et al.* [38]	0.121	0.586	-	0.811	0.954	0.987
Li *et al.* [20]	0.113	0.821	-	0.621	0.886	0.968
Fu *et al.* [7]	0.115	0.509	-	0.828	**0.965**	**0.992**
Our VGG-16 BN	**0.101**	**0.412**	**0.160**	**0.868**	0.958	0.986

Table 3: **Performance on NYU DEPTH v2.**

that is, the type of supervision may not only influence the performance but also reveal the useful information in the images for depth estimation.

Supervision method study In order to find out whether different supervision methods use image information differently and which supervision method is the best, we compare direct supervision, supervision by stereo, supervision by video, and their combinations. The supervision methods are compared in the same setting with the image resolution of 128×416. We use ℓ_1 loss as the loss for direct supervision and the loss as in Godard [12] as the loss for self-supervised training. Comparison with a common encoder architecture of pre-trained VGG-16 with batch normalization is given in Table 4. Among all four different supervision methods, combination of video and stereo training with direct supervision fine-tuning gives the best result. Direct supervision reports the second best result, followed by stereo and monocular video. These results suggest that direct supervision performs better than self-supervision. On the other hand, self-supervision does learn different but useful information for depth estimation as the combination of self-supervision and direct supervision outperforms only direct supervision.

Method	rel	rms	rms log	δ_1	δ_2	δ_3
	lower is better			higher is better		
Direct Sup.	0.105	4.537	0.186	0.873	0.959	0.983
Video(VGG)	0.116	4.850	0.192	0.871	0.959	0.982
Stereo(VGG)	0.109	4.942	0.207	0.861	0.949	0.976
Video & Stereo(VGG)	0.108	4.773	0.197	0.869	0.955	0.980
Video & Stereo(VGG) $\rightarrow$ Direct	**0.104**	**4.475**	**0.181**	**0.877**	**0.962**	**0.985**

Table 4: **Comparison of different supervised setting.** Sup.: supervised, Video: monocular video supervision, $\rightarrow$: finetune

Direct supervision loss study We experimentally examine the performance of different supervision losses and identify the optimal one. With direct supervision, the losses are evaluated with DispNet based on pre-trained VGG-16 with batch normalization. For the deep ordinal regression network (DORN), we have a classification layer taking the input from the output of DispNet and we set the interval number as 80. According to Table 5, Berhu loss performs the best and ℓ_1 the second. It is observsed that both ℓ_1 and Berhu losses often result in smaller errors for low residual pixels while the other loss with square calculation does not address such errors well. Overall, Berhu loss is recommended as the best loss.

Method	rel	rms	rms log	δ_1	δ_2	δ_3
	lower is better			higher is better		
L1	**0.105**	4.537	0.186	**0.873**	0.959	0.983
L2	0.114	4.508	0.183	0.857	0.960	**0.986**
Berhu	**0.105**	4.486	0.183	**0.873**	0.960	0.984
Scale Inv.	0.112	**4.468**	**0.182**	0.858	0.960	**0.986**
DORN	0.107	4.514	0.184	0.867	**0.961**	0.985

Table 5: **Comparison of different unitary loss under original DispNet.** Scale Inv.: scale invariant loss.

Multi-scale direct supervision study As the DispNet decoder outputs depth estimation in four different scales, we study the influence of additional supervision on the lower resolution output. We evaluate different multi-scale supervision methods under direct supervision using pre-trained VGG-16 batch normalization and the weight of the loss for lower resolution output is halved by each scale change. The target data of the low resolution output are generated in order to enforce supervision on the outputs of multiple scales. In details, different low resolution target are generated using maxpooling, average pooling and bilinear interpolation and are summarized in Table 6. According to Table 6, bilinear interpolation performs the best among all three generation methods and it also slightly improves the results of the single supervision. Given comparison about multi-scale

supervision, we find that there is little difference between multi-scale and single supervision since the loss on full resolution output has already penalized the side output prediction. As suggested by Table 6, we recommend single direct supervision over multi-scale direct supervision.

Method	rel	rms	rms log	δ_1	δ_2	δ_3
	lower is better			higher is better		
maxpool	0.108	4.541	0.188	0.868	0.958	**0.983**
avgpool	0.106	4.652	0.190	0.867	0.956	0.982
bilinear	0.106	**4.428**	**0.183**	**0.873**	**0.959**	**0.983**
L1	**0.105**	4.537	0.186	**0.873**	**0.959**	**0.983**

Table 6: **Comparison of different methods to produce ground truth data of different scales for direct supervision**

4.4. Data Property Study

Due to the architecture difference and supervision requirement, existing works use different training resolutions and different numbers of images. However, such data properties may affect the estimation performance. We now present an analysis of such data properties to indicate their influences,

Training resolution study For monocular depth training, we use the images of different resolutions as input. The images are down-sampled by Antialias filter and fed into the model based on pre-trained VGG-16 with batch normalization. The model is trained with ℓ_1 loss under different supervision in Table 7. The training of resolution 1024×320 is first performed on a resolution of 640×192 for 10 epochs and then fine-tuned on the original resolution 1024×320 for 5 epochs. The results in Table 7 show that, under all four different settings, the model performance improves with higher resolution. It could be that, the training with high resolution images preserves more useful information than training with low resolution images and also leads to more accurate high resolution prediction. Therefore, training with high resolution data is recommended if computing power allows.

Data amount study The training results can also be affected by the difference in data amount. Here, we study this factor on the KITTI dataset using pre-trained VGG-16 with batch normalization and ℓ_1 loss for direct supervision under different supervision in Table 8. As is observed in Table 8 and Figure 2, the performance gradually degrades as fewer data is used. Moreover, the data amount does not significantly influence the performance but does affect the training time. Given these results, it is more efficient to search for the optimal networks and losses on a small partition of the full dataset and then train on the full dataset with the optimal choice. The results also indicate that it is beneficial to use a dataset with more diverse scenes rather than a large set of similar scenes.

Method	rel	rms	rms log	δ_1	δ_2	δ_3
	lower is better			higher is better		
D 416×128	0.105	4.537	0.186	0.873	0.959	0.983
D 640×192	0.099	4.227	0.175	0.885	**0.963**	**0.985**
D 1024×320	**0.098**	**4.075**	**0.174**	**0.889**	**0.963**	**0.985**
S 416×128	0.109	4.942	0.207	0.861	0.949	0.976
S 640×192	0.104	4.719	0.201	0.875	0.954	**0.977**
S 1024×320	**0.102**	**4.598**	**0.199**	**0.877**	**0.955**	**0.977**
M 416×128	0.116	4.850	0.192	0.871	0.959	0.982
M 640×192	0.111	4.660	0.186	0.884	0.962	0.982
M 1024×320	**0.109**	**4.581**	**0.185**	**0.890**	**0.964**	**0.983**
MS 416×128	0.108	4.773	0.197	0.869	0.955	0.980
MS 640×192	0.101	4.512	**0.188**	0.881	0.961	**0.981**
MS 1024×320	**0.099**	**4.461**	**0.188**	**0.883**	**0.961**	**0.981**
DMS 416×128	0.104	4.475	0.181	0.877	0.962	0.985
DMS 640×192	0.097	4.134	0.170	0.891	0.967	0.986
DMS 1024×320	**0.096**	**3.966**	**0.167**	**0.893**	**0.969**	**0.987**

Table 7: **Comparison of different resolution input data.** D: direct supervision, M: monocular video supervision, MS: monocular video and stereo supervision, DMS: monocular video and stereo supervision with direct supervision finetuning

Method	rel	rms	rms log	δ_1	δ_2	δ_3
	lower is better			higher is better		
D 100%	**0.105**	**4.537**	**0.186**	**0.873**	**0.959**	**0.983**
D 50%	0.110	4.705	0.192	0.861	0.955	0.982
D 25%	0.111	4.765	0.196	0.858	0.953	0.980
D 12.5%	0.116	4.607	0.193	0.859	0.955	0.981
D 6.25%	0.118	4.608	0.194	0.850	0.953	0.982
D 3.125%	0.126	4.763	0.205	0.841	0.949	0.980
S 100%	**0.109**	4.943	**0.207**	**0.861**	**0.949**	0.976
S 50%	0.111	**4.929**	**0.207**	0.860	0.948	0.976
S 25%	0.112	4.950	**0.207**	0.860	0.948	0.976
S 12.5%	0.116	5.011	**0.207**	0.848	0.947	**0.977**
S 6.25%	0.122	5.091	0.210	0.834	0.944	**0.977**
S 3.125%	0.128	5.227	0.212	0.823	0.940	**0.977**
M 100%	**0.116**	4.850	**0.192**	**0.871**	**0.959**	**0.982**
M 50%	0.119	4.896	0.195	0.866	0.958	0.981
M 25%	0.117	**4.784**	**0.192**	0.865	0.958	**0.982**
M 12.5%	0.125	4.900	0.197	0.849	0.955	**0.982**
M 6.25%	0.132	4.957	0.201	0.836	0.952	**0.982**
M 3.125%	0.147	5.318	0.215	0.800	0.942	0.981

Table 8: **Performance on different percentage of KITTI training data**

4.5. Error Analysis

Only quantitative results might not be sufficient to explain all the differences among different methods. We additionally perform an analysis of model correlation to clarify

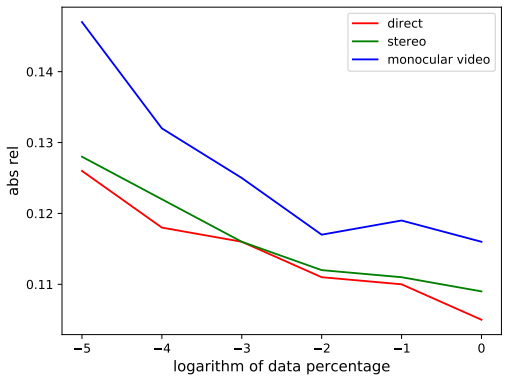


Figure 2: **Performance on different percentage of KITTI training data.** X-axis is set as the binary logarithm of data percentage and absolute relative error is chosen as the performance metric.

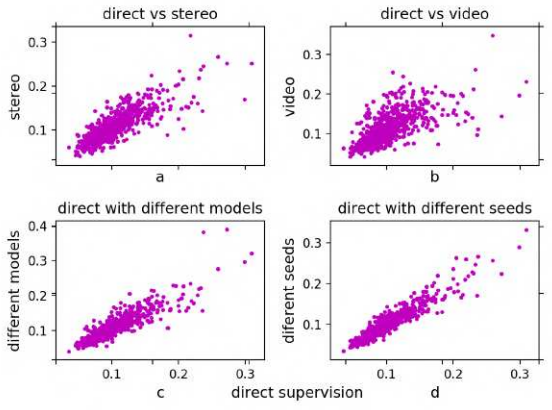


Figure 3: **Comparison of the image-wise performance of different supervision and architectures.** X-axis represents the absolute relative errors under direct supervision ℓ_1 loss and VGG model. Y-axis represents their performance under different supervision methods and neural networks

the advantage of each method on an image-wise and pixel-wise level. The errors on typical image are also visualized in plots for comparison.

Image-wise performance study In this section, we study whether different models and supervision methods would influence the performance of a certain scene. Specifically, we choose the absolute relative error as the comparison metric. The control group uses the same model with supervision initialized with different seeds. According to the Figure 3c and Figure 3d, different encoders under direct supervision produce highly correlated results. As for supervision method comparison in Figure 3a and Figure 3b, stereo and monocular video supervision give better results than direct supervision in some specific scenes but their performance is still correlated with direct supervision. Thus we suppose that learning under different supervision gives limited different insight for our model.

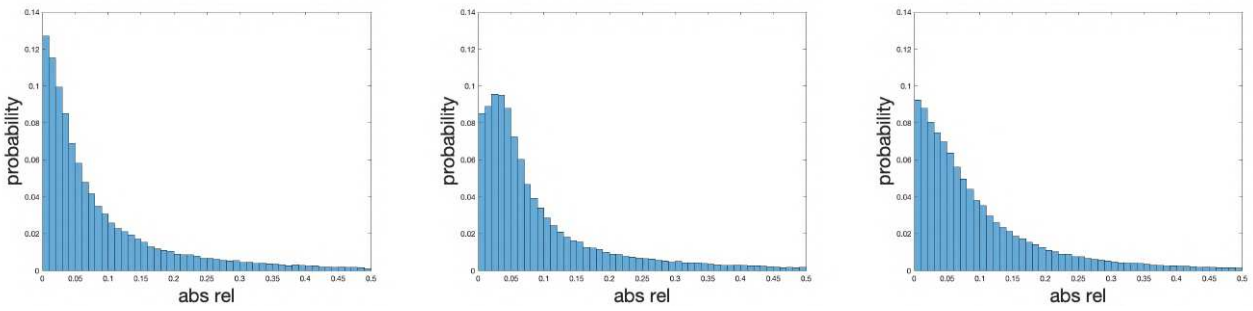
Distribution of pixel-wise performance Since image-wise performance is highly correlated, we further study the pixel-wise performance under the same setting as section 4.5. We also use the absolute relative error as the performance metric. Out of the approximately 17k valid pixels in a single test image, we randomly choose 100 pixels and plot the histogram of absolute relative errors within 0 to 0.5 to represent the distribution of error. As shown in Figure 5a and Figure 5c, direct and video supervision have histograms of similar shapes where most pixels have small errors and the probability drops faster as the error goes larger. For the absolute value, the probability of direct model is larger in the low error area. As shown in Figure 5b, stereo based supervision has more pixels lying in the area about 0.05 in terms of the metric. We suppose that direct supervision has more low error pixels and the unsupervised training results tend to be focused on relative low error area which may be caused by the error in the stereo method. The good results of video-based method in low error area implies the positive effect of depth median scaling. Such distribution of pixel performance implies the complementarity between the direct supervision and self-supervision.

Study of pixel-wise performance with supervision Having observed the distribution of all pixels, we further study pixel-wise performance with different supervision. As it is difficult to evaluate on all pixels in a single image, we select only 100 pixels in each image to visualize this supervision comparison. As Figure 6a shows that the scatter points are mostly placed above the diagonal line, which confirms that direct supervision results in better performance than stereo. Figure 6b shows that the video supervision performs worse than the direct one by a small margin. Since the training with video supervision requires median scaling during evaluation, the good performance may be attributed to the median scaling instead of the supervision itself.

Pixel-wise visual study Besides quantitative results, we reveal drawbacks of different supervision methods by plotting images and marking large errors in red. According to the images of the first three rows in Figure 4, neither of three supervision methods performs well for moving objects where video supervision performs worse than the other two supervision methods with its assumption for static scenes on moving camera. As for the last three rows in Figure 4, neither of three supervision methods is good at predicting porous objects such as the guarding rails and blocking nets. Such objects may exhibit rapid and large depth change, making them difficult to learn. In general, the dynamic ob-

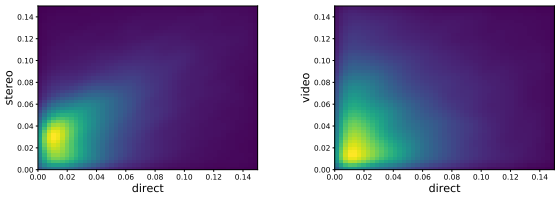
(a) Input (b) Direct error (c) Stereo error (d) Video error (e) groundtruth

Figure 4: **Image visual comparison.** Pixels with absolute relative error larger than 0.3 are denoted by red markers



(a) Histogram of direct supervision (b) Histogram of stereo supervision (c) Histogram of video supervision

Figure 5: **Distribution of pixel-wise performance**



(a) Direct vs stereo (b) Direct vs video

Figure 6: **Comparison of pixel-wise performance of different supervision methods.** The heat map are represented by the scatter points where the x-axis represents the pixel-wise absolute relative errors under direct supervision and y-axis represents the errors under stereo or video supervision.

jects are highly challenging for depth prediction even under direct supervision.

5. Conclusion

In this paper, we have presented a study on various factors for monocular depth estimation, including architectures, supervision losses and other aspects such as resolution, data property and data size. An error analysis is also conducted aiming to provide further insight for model design. Lastly, we have introduced an improved encoder architecture which achieves competitive performance on KITTI and state-of-the-art performance on NYU Depth v2. The empirical results in work can be used as a guideline for more efficient depth estimation model design and training.

Acknowledgments The authors thank the generous support by armasuisse.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Süsstrunk, et al. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012. 2

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. 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. 3

[3] Y. Chen, W. Li, X. Chen, and L. V. Gool. Learning semantic segmentation from synthetic data: A geometrically guided input-output adaptation approach. In *CVPR*, 2019. 2

[4] Y. Chen, C. Schmid, and C. Sminchisescu. Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera. In *ICCV*, 2019. 1

[5] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015. 5

[6] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2014. 1, 2, 3, 4, 5

[7] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, 2018. 1, 2, 3, 5

[8] R. Garg, V. K. BG, G. Carneiro, and I. Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *ECCV*, 2016. 1, 2

[9] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013. 2, 4

[10] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017. 1, 2

[11] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017. 4

[12] C. Godard, O. Mac Aodha, M. Firman, and G. Brostow. Digging into self-supervised monocular depth estimation. *ICCV*, 2019. 2, 4, 5

[13] X. Guo, H. Li, S. Yi, J. Ren, and X. Wang. Learning monocular depth by distilling cross-domain stereo networks. In *ECCV*, 2018. 2, 5

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2

[15] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. 2015. 4

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 2

[17] Y. Kuznietsov, J. Stuckler, and B. Leibe. Semi-supervised deep learning for monocular depth map prediction. In *CVPR*, 2017. 1, 2, 5

[18] L. Ladicky, J. Shi, and M. Pollefeys. Pulling things out of perspective. In *CVPR*, 2014. 5

[19] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. In *3DV*, 2016. 1, 2, 5

[20] B. Li, C. Shen, Y. Dai, A. Van Den Hengel, and M. He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *CVPR*, 2015. 1, 2, 5

[21] B. Liu, S. Gould, and D. Koller. Single image depth estimation from predicted semantic labels. In *CVPR*, 2010. 2

[22] F. Liu, C. Shen, and G. Lin. Deep convolutional neural fields for depth estimation from a single image. In *CVPR*, 2015. 5

[23] F. Liu, C. Shen, G. Lin, and I. Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(10):2024–2039, Oct. 2016. 2, 5

[24] M. Liu, M. Salzmann, and X. He. Discrete-continuous depth estimation from a single image. In *CVPR*, 2014. 5

[25] C. Luo, Z. Yang, P. Wang, Y. Wang, W. Xu, R. Nevatia, and A. Yuille. Every pixel counts ++: Joint learning of geometry and motion with 3d holistic understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2019. 5

[26] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 2

[27] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 2, 4

[28] A. B. Owen. A robust hybrid of lasso and ridge regression. *Contemporary Mathematics*, 443(7):59–72, 2007. 2, 3

[29] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017. 4

[30] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 4

[31] A. Saxena, S. H. Chung, and A. Y. Ng. Learning depth from single monocular images. In *Advances in neural information processing systems*, pages 1161–1168, 2006. 1, 2

[32] A. Saxena, S. H. Chung, and A. Y. Ng. 3-d depth reconstruction from a single still image. *International journal of computer vision*, 76(1):53–69, 2008. 1, 2

[33] A. Saxena, M. Sun, and A. Y. Ng. Learning 3-d scene structure from a single still image. In *ICCV*, 2007. 5

[34] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. 2015. 2

[35] S. Vijayanarasimhan, S. Ricco, C. Schmid, R. Sukthankar, and K. Fragkiadaki. Sfm-net: Learning of structure and motion from video. *arXiv preprint arXiv:1704.07804*, 2017. 3

[36] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 4

[37] J. Watson, M. Firman, G. J. Brostow, and D. Turmukhambetov. Self-supervised monocular depth hints. In *ICCV*, 2019. 5

[38] D. Xu, E. Ricci, W. Ouyang, X. Wang, and N. Sebe. Multi-scale continuous crfs as sequential deep networks for monocular depth estimation. In *CVPR*, 2017. 1, 2, 5

[39] Z. Yang, P. Wang, Y. Wang, W. Xu, and R. Nevatia. Every pixel counts: Unsupervised geometry learning with holistic 3d motion understanding. In *ECCV*, 2018. 2

[40] Z. Zhang, Z. Cui, C. Xu, Z. Jie, X. Li, and J. Yang. Joint task-recursive learning for semantic segmentation and depth estimation. In *ECCV*, 2018. 2

[41] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017. 1, 2, 5

[42] W. Zhuo, M. Salzmann, X. He, and M. Liu. Indoor scene structure analysis for single image depth estimation. In *CVPR*, 2015. 5

[43] L. Zwald and S. Lambert-Lacroix. The berhu penalty and the grouped effect. *arXiv preprint arXiv:1207.6868*, 2012. 2, 3