# A Deep Learning Approach to Single-Image Depth Prediction

Nathaniel Lacelle
University of Michigan
natelac@umich.edu

Ramsey Nofal
University of Michigan
rnofal@umich.edu

## Abstract

*Estimating the depth information of a scene is vital to a variety of computer vision applications. Traditionally, predicting the depth of a single scene has required either multiple perspectives of that scene or specialized sensors [1]. In contrast, monocular depth estimation attempts to deduce the depth information of a scene from a single image. Powered by recent advancements in deep-learning techniques, monocular depth estimation is now capable of yielding useful, applicable depth predictions. In this paper, we outline an approach to monocular depth estimation using a multi-scale deep network which predicts coarse depth information at a global-scale, then refines this prediction locally. We train our network on the NYU Depth V1 dataset [2], utilizing a scale-invariant error to evaluate its performance. The techniques used in this paper were originally proposed by Eigen et al [3].*

## 1. Introduction

Depth estimation is an essential component to determining the 3D configuration of a scene, with applications including 3D modeling, augmented reality, and autonomous systems. Direct approaches to depth estimation often rely on specialized sensors, such as LIDAR and RGBD cameras [1]. However, these methods can be expensive, physically large, and susceptible to adverse weather conditions [4]. Image-based depth prediction methods, in contrast, rely solely on stills or videos of scenes. Stereo depth estimation, which necessitates multiple perspectives of a given scene, is one such method, providing robust measures of correspondence between objects in a scene [1]. Monocular depth estimation, on the other hand, extracts pixel-wise depth predictions from a single image [5]. This method has recently been considered as a serious alternative to traditional approaches due to the availability of still images and the rapid development of deep learning techniques [3].

Single-image depth prediction, however, has inherent ambiguity. Given an image, an infinite number of scenes could have produced it. Therefore, single-image depth prediction techniques must rely on determining the depths between objects and surfaces within a scene relative to each other, as opposed to absolutely. Our method addresses this issue through the use of a scale-invariant error which controls for variations in the physical sizes of scenes.

Figure 1. Example input image and depth map prediction from a neural network shown in Li *et al*. [1]

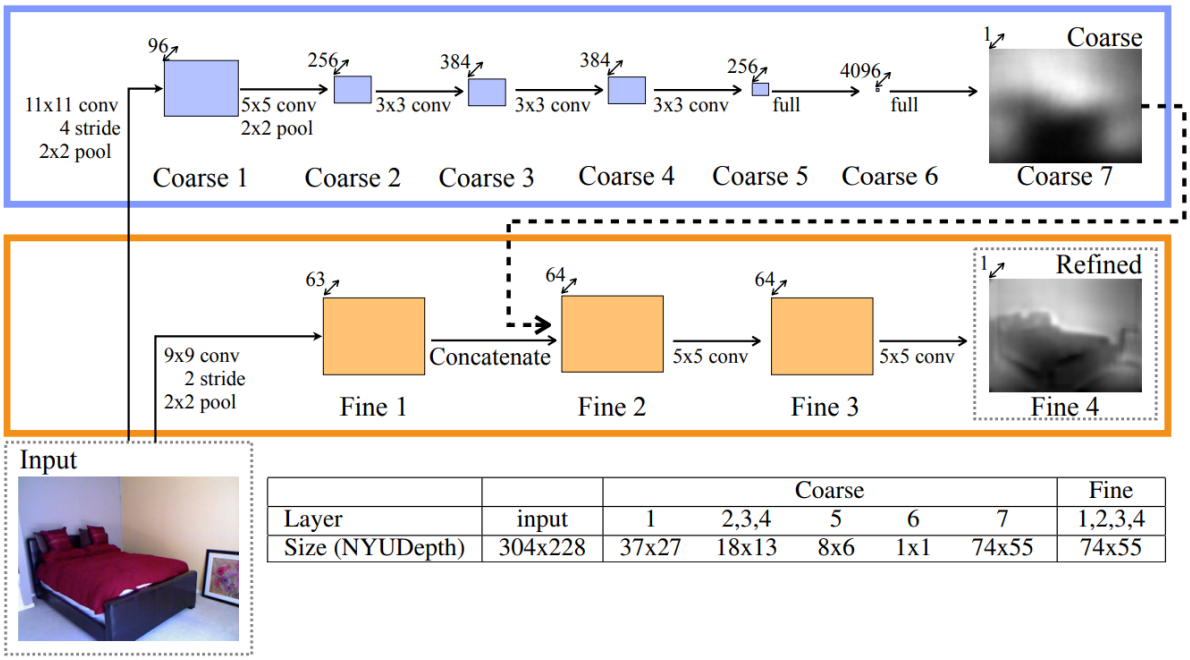
The method outlined in this paper is built on the network proposed, at a high-level, by Eigen *et al*. [3] They describe a multi-scale deep network, which extracts global depth predictions through a coarse network, which are further refined through a locally-focused fine network.

The structure of our network follows layer-by-layer that of Eigen *et al*. [3], but the exact implementation is wholly our own and makes extensive use of Pytorch methods and paradigms. We compare the accuracy of our results to those of Eigen *et al*. [3] quantitatively, through a variety of error functions, and qualitatively, based on visual cues.

## 2. Related Work

### 2.1. Markov Random Fields

A method tackling the same problem as us, but implemented using a different approach, was developed by Saxena *et al*. [6]. Their method involves creating quantitatively accurate 3D models of scenes from monocular images via supervised learning. The images are first processed using a superpixel segmentation algorithm. This algorithm groups pixels together by features detected using a convolutional network. A Markov Random Field (MRF) is formulated using these superpixels in order to predict plane parameters




| | | Coarse | | | | | Fine |
|---|---|---|---|---|---|---|---|
| Layer | input | 1 | 2,3,4 | 5 | 6 | 7 | 1,2,3,4 |
| Size (NYUDepth) | 304x228 | 37x27 | 18x13 | 8x6 | 1x1 | 74x55 | 74x55 |

Figure 2. Network Architecture from Eigen *et al*. [3]

for patches. These patches include 3D locations and 3D orientations, which are used to render them in 3D space in order to reconstruct the scene. This method is not as powerful as ours, since it requires hand-crafted convolutional filters and horizontal alignment of the images.

### 2.2. Unsupervised Learning

Unsupervised prediction methods often rely on the geometry of a scene to eliminate the need for annotated ground-truth depths. Garg *et al*. [3], for example, utilize stereo inputs to train a network on pairs of images with some small, known camera motion between them. This small amount of motion enables them to apply geometric calculations to estimate depth. Their network architecture is considerably simpler than equivalent systems and gives comparable performance to that of supervised methods.

### 2.3. Focus and Defocus Cues

Certain methods make use of the geometry of a camera lens to make depth predictions. These may either be used on their own or incorporated into a neural network to augment and improve estimates.

Gur and Wolf [7] utilize an encoder-decoder network architecture to extract depth from focus. Their network makes use of a novel convolutional layer, built on a Point Spread Function (PSF), which makes use of optical geometry. They evaluate their model on a host of data sets, including NYU Depth V2 [8] and KITTI [9]. Their approach quantitatively outperforms a host of well-known, supervised, single-image deep-learning models.

Levin *et al*. [10] used a coded aperture to estimate depth for selective blurring of pictures. Unlike similar methods, this method generates both image and depth from a single picture being taken. The aperture of the camera is modified with a aperture pattern that selectively blocks light from entering the camera. This selective blocking of light allows for simple depth to be extracted from the image. This method does reduce the quality of the images.

## 3. Method

### 3.1. Architecture

Our network is composed of two convolutional neural networks, a coarse network and a fine network, which are used in conjunction to predict the depth of a image. The image is first fed through the coarse network, which outputs a depth prediction at the global scale. The image and the output of the coarse network are then fed into the fine network, which outputs a final depth map prediction for the entire image. See Figure 2 for specific details, and Eigen *et al*. [3] for more details.

#### 3.1.1 Coarse Network

The coarse network is composed of 7 layers. The first 5 layers are convolutional layers, and the first 2 of those use max pooling to reduce the size of the layers. The final 2 layers are fully connected layers. Altogether, this architecture allows global information to be compressed to a smaller size. In turn, this forces the coarse network to learn global depth

information rather than fine scale information.

#### 3.1.2 Fine Network

The fine network is composed of 4 layers. All layers but the second layer are convolutional layers. The first layer uses pooling to reduce the input image to the same size as the coarse network. The second layer concatenates the output of the coarse network onto the output of the first layer of the fine network. Altogether, this architecture allows the fine network to focus on fine details across the image, while also taking into account the global depth/trends of the image.

### 3.2. Training Loss

Our predictions for depth are all relative depth since it is impossible to recover distances in a monocular image. We therefore must avoid punishing the network when the predicted depth is scaled incorrectly compared to the actual depth. We use a scale-invariant error (in log space) proposed by Eigen *et al*. [3], *scale-invariant mean squared error*, to calculate the loss of our network. The equation punishes pixel-by-pixel mistakes that are in the same direction less, and punishes mistakes that are opposite each other more. This reduces error attributed to scale differences between the ground truth and the prediction. This punishment/reward is the second term in Eqn 1.

$$L(y, y^*) = \frac{1}{n}\sum_i (d_i)^2 - \frac{\lambda}{n^2}(\sum_i d_i)^2 \qquad (1)$$

$$d_i = \log y_i - \log y_i^* \qquad (2)$$

Equation 1 is also known as the log scale-invariant RMSE, it calculates the loss of the network for an image pixel by pixel. During training we use $\lambda = 0.5$ as an average between scale-invariant loss ($\lambda = 1$) and elementwise $l_2$ loss ($\lambda = 0$). During validation we use $\lambda = 1$.

Since the Kinect has depth holes from the optical disparity of its IR and CMoS sensor, we must mask out erronous pixels. This is done during training by excluding pixels from the loss function with no depth value.

### 3.3. Weighting

The weights of each layer are drawn from a normal distribution. We employ Kaiming initialization on convolutional layers and Xavier initialization on linear layers. We settled on these schemes through trial-and-error; other initialization regimes result in exploding or vanishing gradients.

## 4. Experiments

### 4.1. Setup

We train our model on the NYU Depth V1 labeled dataset [2]. This dataset contains, among other variables, 2284 still images and their associated depths, which were recorded by the RGB and Depth cameras from a Microsoft Kinect. Values within depth variables are given in meters. We reduce this dataset to 800 randomly selected image-depth pairs, which are then partitioned into a training set of 640 pairs, with the remainder serving as a validation set. We find that this split yields well-fit results without exhausting our limited hardware runtime resources. Our train-time data loader runs on batches of 32 shuffled images.

Similar to Eigen *et al*. [3], we augment our data with random transformations. Inputs and targets are rotated by between -5 and 5 degrees and flipped horizontally with 0.5 probability. These transformations reduce overfitting; they effectively increase the amount of training data without using valuable hardware resources, all while preserving the geometry of a scene. We intentionally avoided transformations, such as scaling and cropping, which might distort image geometry.

We use an SGD optimizer with learning rates and momentum set according to Eigen *et al*. [3]; further tuning these base values has no positive effect on the quality of our output. Learning rates are 0.001 for coarse layers 1-5, 0.1 for coarse layers 6 and 7, 0.001 for fine layers 1 and 3, and 0.01 for fine layer 2. Additionally, we add a scheduled learning rate decay not mentioned in Eigen *et al*. [3] We find that our model plateaus fairly early without this decay, which reduces the learning rates by a factor of 0.1 with a loss threshold of 0.0001 and a patience of 10 epochs. Momentum is set to 0.9.

We train the coarse network for 41 epochs on the 640 pre-selected images, then feed it into the fine network, which runs for another 41 epochs without backpropagation into the coarse network. This is close to the minimum number of epochs and inputs which results in useful output.

### 4.2. Results

#### 4.2.1 Quantitative

Results of the trained network are provided in Table 1. We compare our results to those of Eigen *et al*. [3] along various error measurements. Our implementation achieves slightly lower performance than Eigen *et al*. [3] across all metrics. Its RMSE (log, scale inv.) error on the validation set lags behind the same metric in Eigen *et al*. [3] by 26%. General under performance compared to the baseline is no surprise as we train on orders of magnitude fewer samples than Eigen *et al*. [3]. The small degree of this under performance is explained by the diminishing returns of the errors, which are

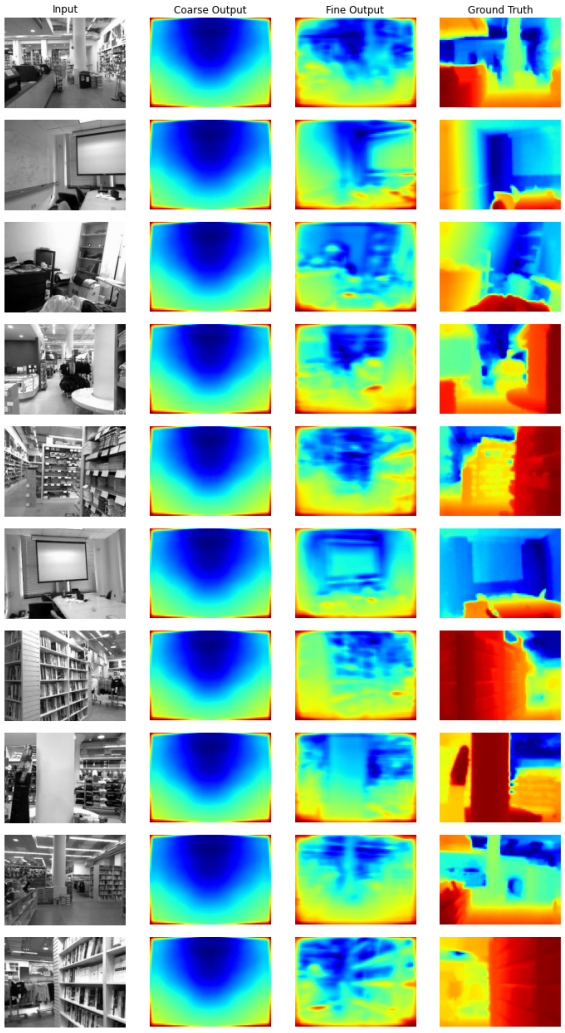



Figure 3. Selected validation outputs from the trained network

computed on a pixel-by-pixel basis; fine-grain details have far less impact on error than large, uniform areas. Training the model for many more samples produces visually much cleaner output but has a comparatively small impact on error.

#### 4.2.2 Qualitative

Selected outputs from our trained network are provided in Figure 3. Visually, our network produces rough estimates of scene depths; areas which are close are predicted close, those which are far are predicted far. However, our trained network struggles in three aspects:

**Coarse Output:** The coarse output always considers the middle to be farther away than the edges. It is likely overfitting to the dataset: in most scenes the camera is pointed down a hallway or toward the far corner of a room.

**Fine Details:** Large areas of similar depth are generally accurate. Highly focused, localized changes in depth are ignored.

**Magnitude:** Our network inadequately assesses the degree by which one area is deeper than another, especially areas far from the scene border. Sections which should be comparatively much deeper than other sections are only somewhat deeper.

| Average Loss | Coarse | Coarse + Fine |
|---|---|---|
| RMSE (log, scale inv.) Training | 0.316 | 0.278 |
| RMSE (log, scale inv.) Validation | 0.327 | **0.285** |
| RMSE (log) Validation | 0.452 | 0.413 |

Table 1 Our losses

| Average Loss | Coarse | Coarse + Fine |
|---|---|---|
| RMSE (log, scale inv.) Validation | 0.221 | **0.219** |
| RMSE (log) Validation | 0.283 | 0.285 |

Table 2 Eigen *et al*. [3] losses

## 5. Conclusions

Based on our results, we conclude that single-image depth estimation through this multi-scale network is a viable approach to producing reasonable depth estimates, potentially outperforming other network designs. However, all current single-image depth estimators, including ours, fall substantially short of matching the accuracy of physical depth sensors and are incapable of providing absolute depth information.

Our approach is largely the same as that of Eigen *et al*. [3], with some key optimizations. Through extensive trial-and-error, we found parameters which make implementing their method viable on small sample sizes and limited hardware resources. It is important to note that implementation details in Eigen *et al*. [3] are sparse; every line of our model's code is either our own or borrowed from the official Pytorch documentation.

In the future we would like to extend this approach by incorporating it into a generative adversarial network (GAN). Jung *et al*. [11]. have demonstrated that adversarial networks are an effective tool to improve supervised depth prediction methods, achieving lower errors and better qualitative outputs than non-adversarial networks. We also plan to further test this method on other depth data sets, such as KITTI [9] and Make3D [6].

The future of single-image depth prediction is promising. Current methods demonstrate incredible aptitude in their ability to compete with traditional methods in certain applications. As the field of deep-learning rapidly develops, we are sure to see dramatic improvements in single-image depth prediction.

## References


[1] Q. Li, J. Zhu, J. Liu, R. Cao, Q. Li, S. Jia, and G. Qiu, "Deep learning based monocular depth prediction: Datasets, methods and applications," *CoRR*, vol. abs/2011.04123, 2020. [Online]. Available: https://arxiv.org/abs/2011.04123

[2] 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.

[3] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *CoRR*, vol. abs/1406.2283, 2014. [Online]. Available: http://arxiv.org/abs/1406.2283

[4] Admin, "Advantages and disadvantages of lidar." [Online]. Available: https://lidarradar.com/info/advantages-and-disadvantages-of-lidar

[5] A. Mertan, D. J. Duff, and G. Unal, "Single image depth estimation: An overview," *CoRR*, vol. abs/2104.06456, 2021. [Online]. Available: https://arxiv.org/abs/2104.06456

[6] A. Saxena, M. Sun, and A. Y. Ng, "Make3d: Learning 3d scene structure from a single still image." [Online]. Available: http://www.cs.cornell.edu/~asaxena/reconstruction3d/saxena_make3d_learning3dstructure.pdf

[7] S. Gur and L. Wolf, "Single image depth estimation trained via depth from defocus cues," *CoRR*, vol. abs/2001.05036, 2020. [Online]. Available: https://arxiv.org/abs/2001.05036

[8] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *ECCV*, 2012.

[9] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger, "Sparsity invariant cnns," in *International Conference on 3D Vision (3DV)*, 2017.

[10] A. Levin, R. Fergus, F. Durand, and B. Freeman, "Image and depth from a conventional camera with a coded aperture." [Online]. Available: https://groups.csail.mit.edu/graphics/CodedAperture/

[11] H. Jung, Y. Kim, D. Min, C. Oh, and K. Sohn, "Depth prediction from a single image with conditional adversarial networks," in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 1717–1721.