


Article

Nested DWT-Based CNN Architecture for Monocular Depth Estimation

Sandip Paul ^{1,2,*} , Deepak Mishra ¹ and Senthil Kumar Marimuthu ²¹ Indian Institute of Space Science and Technology, Trivandrum 695547, Kerala, India² Space Applications Centre, Ahmedabad 380016, Gujarat, India

* Correspondence: san@sac.isro.gov.in; Tel.: +91-9427553384

Abstract: Applications such as medical diagnosis, navigation, robotics, etc., require 3D images. Recently, deep learning networks have been extensively applied to estimate depth. Depth prediction from 2D images poses a problem that is both ill-posed and non-linear. Such networks are computationally and time-wise expensive as they have dense configurations. Further, the network performance depends on the trained model configuration, the loss functions used, and the dataset applied for training. We propose a moderately dense encoder-decoder network based on discrete wavelet decomposition and trainable coefficients (LL, LH, HL, HH). Our Nested Wavelet-Net (NDWTN) preserves the high-frequency information that is otherwise lost during the downsampling process in the encoder. Furthermore, we study the effect of activation functions, batch normalization, convolution layers, skip, etc., in our models. The network is trained with NYU datasets. Our network trains faster with good results.

Keywords: depth-map; discrete wavelets; nested wavelet net; loss function; training; evaluation



Citation: Paul, S.; Mishra, D.; Marimuthu, S.K. Nested DWT-Based CNN Architecture for Monocular Depth Estimation. *Sensors* **2023**, *23*, 3066. <https://doi.org/10.3390/s23063066>

Academic Editors: Simone Bianco, Marco Buzzelli and Jean Baptiste Thomas

Received: 2 February 2023

Revised: 8 March 2023

Accepted: 10 March 2023

Published: 13 March 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Information on the depth is useful for applications related to satellite remote sensing, navigating robots, autonomous landing, animal gesture identification, creation of 3D models, etc. Active 3D imaging systems, such as LIDAR, RADAR, SONAR, etc., rely on high-power sources and reflected echoes to build depth maps. Modern mobile platforms prefer minimal resources, and, hence, there is an opportunity for simple 2D visible/infrared cameras here. These cameras are simple, readily available, and use low power. Depth estimation using single 2D images has attracted many research scholars and a trend exists for this method.

Earlier methods of determining monocular depth were variations of detectors, such as focus [1], defocus [2], and apertures [3]. These methods avoided correspondence issues as with stereo methods but required a stack of images and had to address contrast (aperture variation) or magnification (focus variation) issues. Others have recovered depth from a single image using blur cue [4]. These methods have poor performance on homogeneous surfaces and zones with poor contrast. Recently, CNN-based methods were successful in training models to estimate high-resolution depth images from a single image [5]. These networks solve an ill-posed problem after training. The network model architecture consists of customized building blocks, such as convolution layers, pooling functions, activation layers, and expansion layers [6–8]. The state-of-the-art CNN use supervised or self-supervised training strategies [6,9–13]. Supervised training [5] requires a labeled depth image for the network to converge. Self-supervision uses stereo image pairs [14,15] monocular video and exploits 3D geometry with image reconstruction or camera pose estimates to estimate depth without labeled data. Recent researchers focused on transformers and attention mechanisms to preserve details of depth image [16]. Authors have implemented innovative loss functions with left-right disparity consistency loss [15], photo-metric loss [17], and symmetry loss [18]. Modern researchers also fused light fields information with photo-metric

stereo [19] or used a pair of surface orientation maps (surface normals) from photo-metric stereo [20–23] to derive accurate depth images.

Image information consists of global features, such as structure, texture, semantic information, and local features, such as edges, noise, etc. After transforming an image into the frequency domain, low frequencies represent the global features, while high frequencies represent the local features. Edges are high-frequency components that represent contours, local regions, and local features in an image [24]. Edges are created by neighboring pixels with significantly different intensities. Networks apply averaging function during pooling or down-sampling, reducing these high-frequency components and thus leading to blurred and jagged edges. This degrades object definitions and merges features with the background which results in errors during depth estimation [25].

Previous works have utilized edge-enhancement methods to improve the depth map accuracy [26–28]. Alternately, Discrete Wavelet Transforms (DWT) convert spatial and contrast domain images into the frequency domain and separate low-frequency and high-frequency components. This happens along with down-sampling and avoids artifacts [29–32]. As DWT provides three different high-frequency coefficients, training can enhance edges while noise can be reduced by learning the required coefficients. The high-quality image can be reconstructed with Inverse DWT (IWT) up-sampling after training.

Most networks for depth estimates are based on DenseNet, ResNet, VGG, etc., which are time and computation intensive. Lately, UNet-like architectures have been used for depth estimation [33,34] for faster learning and implementation in less computationally intensive systems. The skip connections here inherently lead to boundary preservation of depth maps. Earlier these were used for medical analysis and Semantic applications. Ref. [35] used DWT-based down-sampling and up-sampling blocks in UNet-like models.

Our literature survey indicated that monocular depth estimation using lightweight UNet-like networks is under-explored. As the inclusion of wavelets in a network is useful, we propose a moderately dense network using DWT for depth estimates. The application of DWT preserves the detailed features compared to the present UNet and UNet++. This paper discusses our proposed network architecture, variants of this network, performance with the public datasets, ablation studies, and results of experiments carried out. Our main contributions are

- A nested DWT-based CNN architecture is proposed for monocular depth estimation,
- Dense Skip functions are implemented with attention function so as to improve the learning of local features,
- Dense convolution blocks are incorporated for higher feature extraction and learning.

2. Wavelets

Wavelets are orthogonal and rapidly decaying functions of the Wavelet Transform family and provide frequency and location information at lower scales. This 2D DWT disintegrates an image into four components, a low-frequency coefficient map (LL) and three high-frequency coefficient maps (LH, HL, and HH). These four coefficient maps have half the resolution of an input image. For the image Y , the DWT is:

$$LL, LH, HL, HH = DWT(Y) \quad (1)$$

DWT provides robustness to noise and improves accuracy in deep learning [36]. Another advantage of using DWT is the readily available high-frequency representation of edges. Further, we can learn the coefficients of LH, HL, or HH components to improve the features. The LL map can further be decomposed iteratively to get multi-scale coefficient map sets as shown in Figure 1. In our experimentation, we use a scale of 1. Higher scales are possible, however, we are replacing the pooling operation which reduces the feature map size by two.

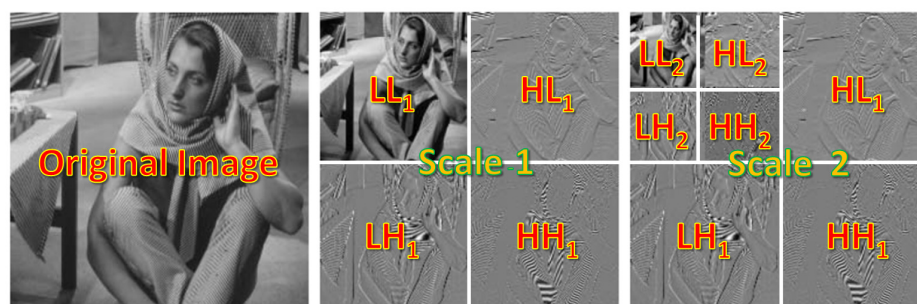


Figure 1. DWT decomposes the image into low-resolution coefficient maps. Here, the scale is 2. We use a scale of 1 to replace the down-sampling operation.

The DWT is an invertible transformer. The inverse DWT is IWT which converts the four frequency components back to the original image (twice the resolution of the coefficient maps) without any loss of feature definitions. Both Haar and Daubechies wavelets have been used by researchers. We use the Daubechies wavelet with four vanishing moments (db4) for all models. We also experiment with a Haar wavelet on the best model. Haar wavelet has the simplest basis. We follow the DWT implementation from [37].

3. Nested DWT Net Architecture

Deep neural networks train models in a systematic way to extract required image features. Most networks for depth estimates are based on DenseNet, ResNet, VGG, etc., and are time and computationally intensive. Researchers are studying less complex networks for small systems with faster learning. The simplest fully convolutional network is UNet [38] developed primarily for abnormality localization in biomedical images. Here, in this encoder–decoder model, the encoder has multiple blocks. Each block has a stack of convolution operators with the last of the stack feeding a max pooling operator (down-sampling). This sub-block extracts the image context and reduces the image resolution by half. The decoder has an equal number of blocks. Each block has an up-sampling operation that expands the size of feature maps by two. This passes through convolution operations. Skip connections, taken from the corresponding encoder block, are fed to the decoder block to enhance the output predictions. The decoder adds localization information to the input context information. The final output has a similar resolution to the image taken for prediction. UNet provides detailed segmentation maps using limited training datasets.

UNet was further improved by [39] by adding soft attention. Using attention, the network learns to focus on relevant zones with low computational complexity. Attention to the skip connections reduces redundant image features and improves prediction accuracy. Residual UNet was developed [40] to overcome accuracy degradation and ease the training of networks. Refs. [31,41–45] proposed wavelet transform to extract sharp edge and strong object boundaries for image dehazing, enhancing the low-light image, reducing artifacts, image segmentation, and depth map estimation. Here, they replace the down-sampling layer with a DWT and the up-sampling layer with an IWT. Ref. [30] used multiple wavelet-based transforms for down-sampling. Recently, Refs. [31,41,46] studied wavelet-based loss function to improve structural details while [32] proposed learning sparse wavelet coefficients to estimate depth maps. Researchers have also replaced the UNET encoder backbone with dense pretrained image networks, such as ResNet, DenseNet [7], etc., and tuned the decoder accordingly to estimate depth. However, these deep networks need computationally intensive resources. UNET++ was defined by [47] and uses (a) convolution operations on skip paths, (b) a series of nested skip paths (dense), and (c) deep supervision. The prediction of this model was improved by reducing the gaps in encoder–decoder feature maps at the sub-block levels. This also made learning easier. The published performance is better compared to UNet and wide UNet. To date, UNET++ has been mostly used for medical analysis and Semantic applications [48]. We realize that this is a

less researched area for estimating depth maps. We study UNet and UNet++ to propose a moderately dense network using Discrete wavelets to preserve depth map details.

Our network architecture, Nested Discrete Waveform Transform Net or NDWTN for depth estimation uses an encoder, multi-scale decoders, and skip paths. We replace the down-sampling and up-sampling layers with wavelet transforms. All the coefficients of the wavelet transform (LL, LH, HL, HH) are trainable so as to preserve the details. We also apply nested dense skip paths and convolution, such as UNet++. We also evaluate variants of NDWTN by implementing (a) attention in the skip path, (b) residual blocks in place of convolution blocks, (c) batch normalization layers, and (d) different activation layers. Our network architecture is shown in Figure 2. The structure is similar to UNet++ and has a single encoder path and multiple decoder paths of different scales, all connected through dense skip connections. These skip connections enable nested networks which reduces the semantic gap and provides deep supervision of the output. NDWTN has four scales having a UNet structure. The networks and scales are indicated with yellow, blue, green, and pink colors. Each decoder has independent outputs which are connected to the final output through skip connections.

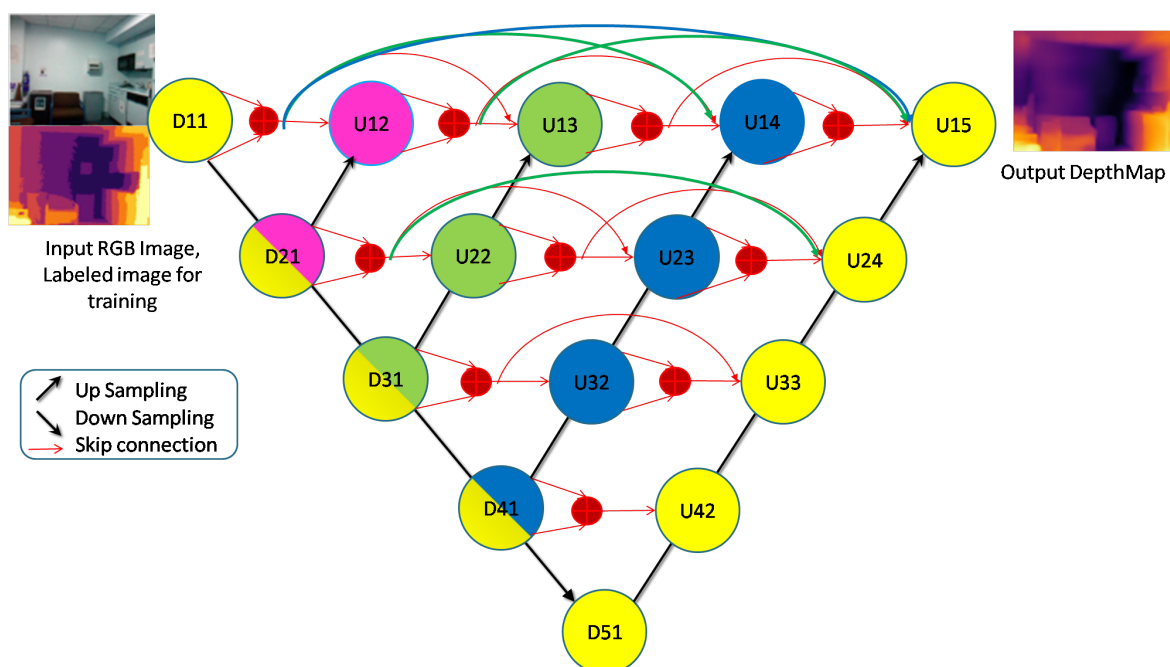


Figure 2. Our network architecture (NDWTN).

An encoder down-sampling block is shown in Figure 3 and consists of a learnable convolution block and a DWT layer. This block can have residual convolution as indicated by '+' sign. DWT layer downsamples the input. This block provides input to the skip layer and the succeeding block. A typical convolution block includes convolution, activation, and batch normalization layers. The activation function can be either ReLU (Rectified Linear Unit) or Leaky ReLU. The number of each layer can be increased to improve network density Figure 4. Convolution blocks can be replaced with residual convolution blocks. Residual blocks have additional skip paths. These skip paths are taken from the output of the first convolution operator in the block and then added to the final output of the block. A typical residual block is shown in Figure 5. Each down-sampling block feeds the next block and the skip paths. There are five down-sampling blocks.

A decoder up-sampling block is shown in Figure 6 and consists of an IWT layer for up-sampling and a learnable convolution block. The convolution block is similar to the encoder and can be replaced with a residual block. Two inputs are received (a) from the preceding block and (b) from the skip function. The input from the skip layer is concatenated. The number of convolutions, batch normalization, and activation layers in the convolution

block is variable. A typical full decoder block is shown in Figure 7. Each up-sampling block feeds the next block. There are five up-sampling blocks.

The supervised training attempts to predict pixel-wise depth from models by minimizing the regression loss with the help of ground truth images. In the network, the down-sampling and convolution layers of the encoder reduce the information details of these input images. The feature maps arising from initial encoders have higher resolution but lack global information when compared with the final encoders. Our encoder-to-decoder skip connections improve these detail preservation by concatenating high-resolution local features from initial encoders with the decoders global contextual features. Each skip connection responds to the encoder features energy using a gating signal to generate an attention grid that preserves the higher local features.

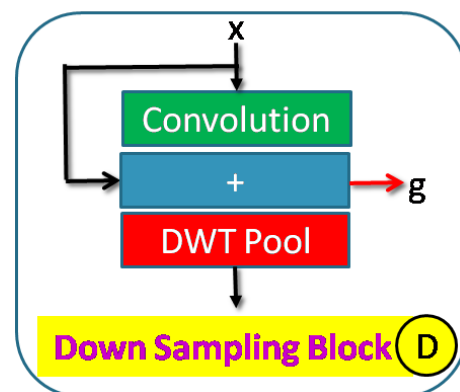


Figure 3. Structure of down-sampling block.

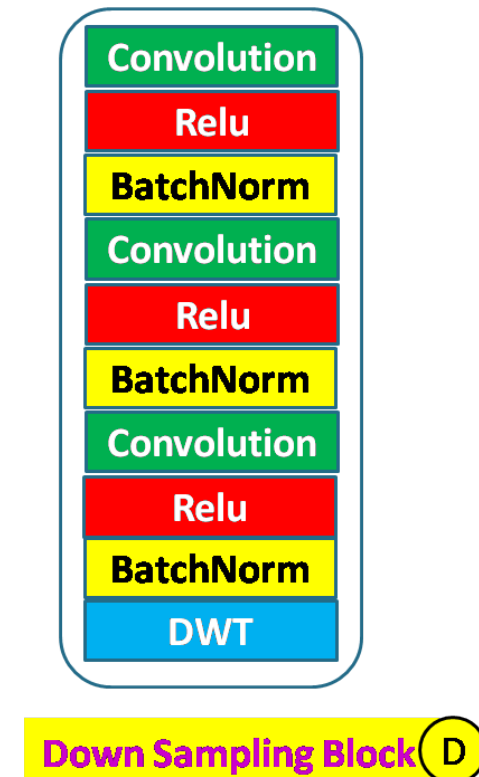


Figure 4. Down-sampling block details: The stack of convolution operators and the sequence of Batch-Norm and activation layers are customized.

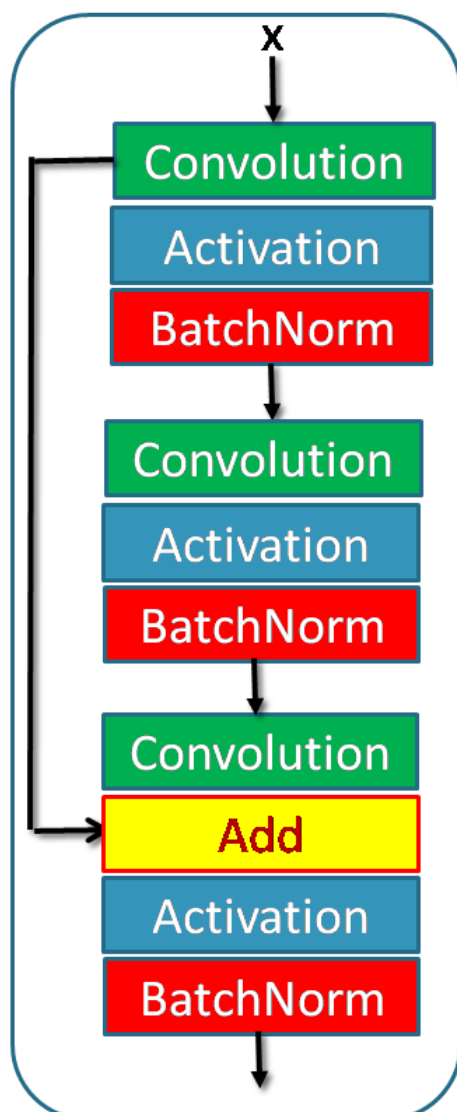


Figure 5. Residual convolution block. The stack of convolution operators and the sequence of Batch–Norm and activation layers are customized.

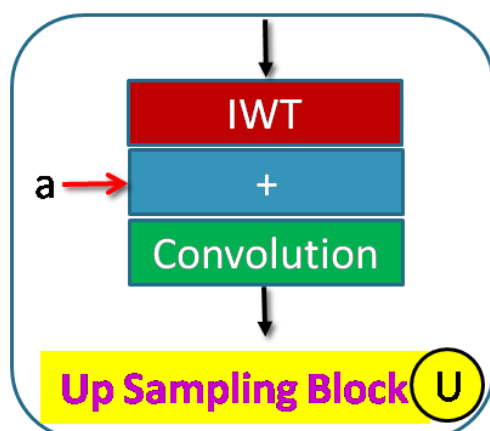


Figure 6. The up–sampling block provides IWT and convolution operations. Information from the skip path ‘a’ is also concatenated.

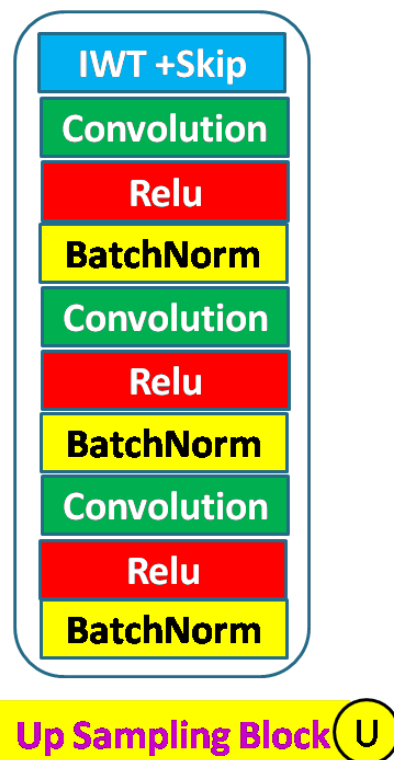


Figure 7. Up-sampling block details: The stack of convolution operators and the sequence of Batch-Norm and activation layers are customized. The convolution stack can be replaced with a residual block.

Typically, the skip layer feeds the encoder output to the decoder. The skip function improves the feature details while up-sampling in decoders. The skip paths go to every decoder horizontally (same scale). This provides nested dense skip functions totaling 10. We also add an attention gate inside the skip layer. This skip layer takes two inputs, one from the preceding encoder block (higher scale) and one from the horizontal encoder block (same scale), and provides input to the attention gate. A strided-convolution operation on the lower input lets us match the feature dimensions. Summing these two vectors results in higher aligned weights and small unaligned weights. The resultant vector is fed to a ReLU function followed by a convolution to reduce the feature dimensions to 1. This vector is finally scaled within $[0, 1]$ using the Sigmoid activation function. This stage gives the attention coefficients. The attention coefficients are upsampled to the dimensions of the lower scale and multiplied with the lower scale input. This input is given to the decoder block. Figure 8 represents the attention block.

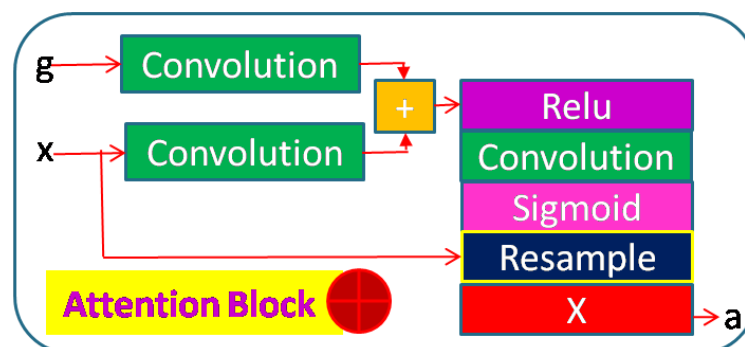


Figure 8. Skip layer with attention. This layer takes two inputs from encoder blocks of different scales, 'g' from the higher scale or input to the encoder and 'x' from the lower scale or output of the encoder, and feeds the decoder block with attention vectors 'a'.

Finally, the decoder output layer is represented by Figure 9. The block is the same for all four decoder outputs. The block has the Sigmoid function as the activation layer to provide the final prediction statistics. Though we can use the scaled four outputs independently, we feed three outputs to the final fourth output via nested skip layers to have the best prediction accuracy. We made variants of our network based on convolution blocks, residual convolution blocks, and attention features. These architecture variants are designated as:

- NDWT: Basic NDWTN;
- NADWT: NDWTN with attention on skip paths;
- NRDWT: NDWTN with residual blocks;
- NARDWT: NRDWT with attention on skip paths.

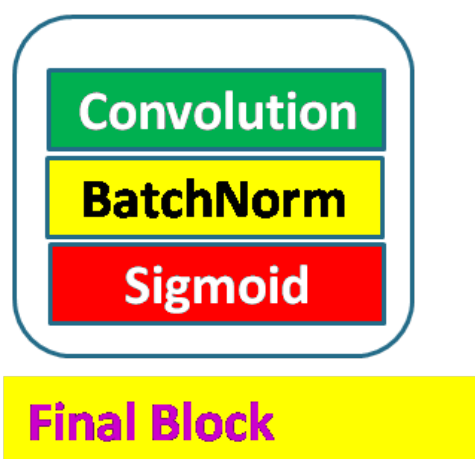


Figure 9. Output block with Sigmoid activation layer.

Table 1 compares our model with published models. The trainable parameters in our models are higher than most UNet types. The non-trainable parameters are less than DenseNet backbone UNet [7].

Table 1. Parameters and models (in millions).

Parms	DWT	ADWT	UNET++	DenseNet [7]	AdaBins [16]	NDWTN
Total	13.39	14.88	13.23	53.99	78.0	42.82
Trainable	13.39	14.87	13.22	53.97	–	42.66

The input RGB image resolution used is $240 \times 320 \times 3$ pixels and the labeled data resolution is 240×320 pixels. The estimated depth map resolution is also 240×320 pixels. These dimensions are arbitrarily chosen.

4. Loss Function

A loss function reduces the training loss and estimates depths comparable to the reference labeled data by converging the model, through gradient descent, during training. Hence, an optimum formulation is crucial for good performance and faster training. The convergence is helped by error functions such as the Mean Absolute Error (MAE) or the L1 loss function. This loss is robust to outliers as it has a large and constant gradient. The orange curve in Figure 10 shows the simulated loss. The pixel-wise error for MAE is given as:

$$L_{pix} = \frac{\sum_{i=1}^n |Y_i - Y_{pred_i}|}{N} \quad (2)$$

where the ground truth data (labeled data) pixel is Y_i , the estimated depth pixel is Y_{pred_i} , and the number of pixels in the depth map totals to N .

MAE loss is linear and offers equal weight for both lower and higher residual values. This reduces overall prediction and hence, MAE is not good for learning. The reversed Huber loss function (BerHu Loss) [6,9] is a better alternative. This provides higher weight to pixels with higher residuals (mean square error) and also to pixels with smaller residuals similar to MAE (Figure 10, dark green curve). The transition between mean square error (MSE) and MAE is defined by a threshold c . In a training batch, c is 20% of the maximum error. The BerHu Loss is given as:

$$L_{pix} = \begin{cases} |Y_{pred_i} - Y_i| & |Y_{pred_i} - Y_i| \leq c, \\ ((Y_{pred_i} - Y_i)^2 + c^2)/2c & |Y_{pred_i} - Y_i| > c \\ c = 0.2 \max_i(|Y_{pred_i} - Y_i|) \end{cases} \quad (3)$$

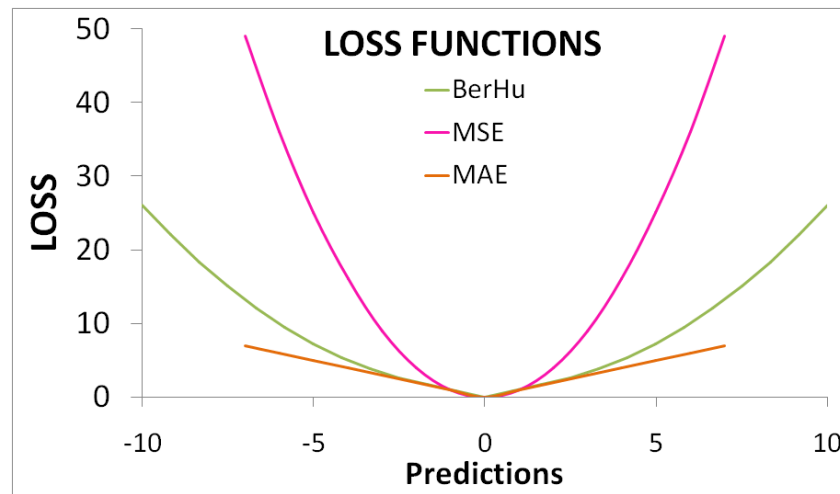


Figure 10. A comparison between MAE, MSE, and BerHu functions.

The depth map needs parameters to define the image structure and object features other than pixel-wise errors. This information comes from interdependent neighboring pixels [49]. The image structure is constructed by the Structural Similarity Index (SSIM). The perceptual difference is indicated by SSIM loss between images. Identical images score the lowest. The loss function is:

$$L_{SSIM} = 1 - SSIM(Y_i, Y_{pred_i}) \quad (4)$$

when SSIM loss is 0, the two images have the same structure. Usually, a weight of 0.5 is applied to this loss to weaken the penalization.

The features and edges in an image are represented by high-frequency structure components. Edge functions make the depth map sharper and more detailed. Gradient edge loss functions are based on the maximum of the derivatives. However, this function sometime leads to double edges as in lines. The derivative will give one positive and one negative peak in such cases. The intensity gradient of an image Y is given in the horizontal and vertical directions as:

$$\begin{aligned} \frac{\partial Y}{\partial h} &= Y(h+1, v) - Y(h-1, v), \\ \frac{\partial Y}{\partial v} &= Y(h, v+1) - Y(h, v-1), \\ L_{edges} &= \text{mean}(|\frac{\partial Y_{pred}}{\partial v} - \frac{\partial Y}{\partial v}| + |\frac{\partial Y_{pred}}{\partial h} - \frac{\partial Y}{\partial h}|) \end{aligned} \quad (5)$$

We used a comprehensive loss function formulated on the above loss functions. These function outputs are further weighted with hyper-parameters to control the penalization.

Our loss function comprises pixel-wise loss (MAE or BerHu), structural loss (SSIM), and edge loss (gradient). The total loss is:

$$L_{total}(Y_i, Y_{pred_i}) = \lambda_1 L_{pix}(Y_i, Y_{pred_i}) + \lambda_2 L_{SSIM}(Y_i, Y_{pred_i}) + \lambda_3 L_{edges}(Y_i, Y_{pred_i}) \quad (6)$$

5. Datasets

The NYU dataset [50] provides registered RGB–depth image pairs of 640×480 pixels resolution. The dataset contains diverse 464 indoor scenes. The depth images are obtained from a Kinect 3D camera which is processed (in painted) to fill in the missing information. Hence, the pixel-level depth information is semi-synthetic. These semi-synthetic depth maps have a maximum depth range of 10 m. The dataset is popular for semantic and depth-related studies and is, hence, suitable for comparing performance. This dataset is divided into three parts and used for training models, validating loss, and finally evaluating our models.

6. Standard Performance Metrics

Performance evaluation and comparison of our trained models are based on several prior works [5,7,51,52]. These are Root Mean Squared Error (RMS):

$$RMS = \sqrt{\frac{1}{N} \sum_{i \in N} (Y_i - Y_{pred_i})^2} \quad (7)$$

Average relative error (REL):

$$REL = \frac{1}{N} \sum_{i \in N} \frac{|Y_i - Y_{pred_i}|}{Y} \quad (8)$$

Logarithm error (\log_{10}):

$$\log_{10} = \frac{1}{N} \sum_{i \in N} |\log_{10}(Y_i) - \log_{10}(Y_{pred_i})| \quad (9)$$

The pixel-level percentage having a relative error below the defined threshold (1.25 , 1.25^2 , 1.25^3) is defined as threshold accuracy (δ_i). It is based on the maximum ratio of labeled data pixels and predicted pixels [51]. This is represented as $Y_i\%$ s.t. $\max(\frac{Y_{pred_i}}{Y_i}, \frac{Y_i}{Y_{pred_i}}) = \delta < th$ for $th = 1.25, 1.25^2, 1.25^3$; and Y is the average value of pixels in labeled data.

Smaller values of RMS, REL, \log_{10} error are the goals here while higher values of δ below the defined threshold are a good indicator.

7. Experiments and Ablation Studies

We train the model on Google Co Laboratory Pro. This gave us a faster GPU (T4, P100/V100) with 25 GB GPU memory. The training was stopped after 10 epochs so as to compare performances. Batch sizes 4 and 8 were used to meet the allocated memory limits. The learning rate was 0.0001. The learning rate exponentially decayed for successive epochs. The filter weights were randomly initialized. The loss optimizer was ADAM. The Batch Normalization layer in our network reduces internal Covariate Shift, speeds up training, and reduces overfitting. This layer has two learnable parameters (β and γ) and two non-learnable parameters (Mean and Variance Moving Averages). The original paper [53] proposes this layer before the non-linear activation function. However, many researchers advocated better results when this layer is placed after the activation function.

Hence, as part of the ablation study, this aspect will be verified. Convolutional layers extract features from input images through learnable filter parameters and remove redundant information by weight sharing. Higher convolution layers lead to an exclusive compressed feature map of the image which ultimately provides informative decisions. This layer also consumes most of the training time. Optimal use of convolution layers is thereby necessary. We experiment with the density of convolution layers in our network architecture and study the performance. The activation layer filters the information (neuron) transmitted to the succeeding layer by a non-linear function. These layers activate the selected neurons by increasing their weight. ReLU (Rectified Linear Unit) is computationally efficient for positive values and allows backpropagation. However, negative input values prevent backpropagation and learning. The Leaky ReLU (LR) activation overcomes this problem by having a small positive slope in the negative zone. ELU (Exponential Linear Units) are better than these two activation functions, but are computationally intensive and, hence, not used. In practice, there is no evidence that Leaky ReLU is always better than ReLU. Hence, we experiment to compare the performance between ReLU and LR. We developed many models by changing the blocks of the architecture of our network to study the impact of the various blocks on the overall performance of (a) the activation layer ReLU and Leaky ReLU; (b) the batch normalization density; (c) the sequence of batch normalization and activation layer: before or after; and (d) the convolution layers in the stack. We also experimented with different loss functions for training. The model implementation and training are in the following combinations:

1. NDWT (3C, 3R, 3Bs) + Bs
2. NADWT (3C, 3LR, 1Bs)
3. NADWT (3C, 3LR, 1Bs) + Bs
4. NADWT (3C, 3Bs, 3R) + Bs
5. NADWT (3C, 3R, 3Bs) + Bs
6. NRDWT (3C, 3R, 3Bs) + Bs
7. NRDWT (3C, 3Bs, 3R) + Bs
8. NARDWT(3C, 3LR, 3Bs) + Bs
9. NARDWT (3C, 3R, 3Bs) + Bs
10. NARDWT (3C, 3Bs, 3LR) + Bs
11. NARDWT (3C, 3Bs, 3LR)
12. NARDWT (3C, 3LR)
13. NARDWT (4C, 4Bs, 4LR) + 1Bs

Where, C: Convolution LAYER, R: ReLU, LR: Leaky ReLU, Bs: Batch Normalization, and NUMBER: Number of LAYERS implemented.

Here, NDWT is our basic model, which implements multistage networks with skip layers. There are five down-sampling blocks (block D in Figure 4) with one input and two outputs. These outputs cater to the lower D block and skip layer (Figure 2). The sequence of the activation layer and bath normalization layer is interchangeable in our studies. The DWT layer replaces the Maxpool layer in this network. These blocks make the encoder. The upsampler block (U as in Figure 7) is similar to the D block with the exception of the DWT layer. An IWT layer at the output upscales the estimated image. There are 10 such blocks which for four decoder chains. The last decoder U15 takes all outputs of decoders via skip paths to provide the final estimate. NADWT augments our base with attention gates (Figure 8) in all the skip functions, which makes the training focus on image zones of higher energy. In the NRDWT model, we replace the convolution with residual convolution as in Figure 5 for encoders and decoders. This model augmented with attention gates gives NARDWT model.

The NYU dataset is used by most researchers and hence we used this for benchmarking. The trained model was evaluated with performance metrics as given by [5,7]. This gives an easy and error-free comparison method. The evaluation dataset (NYU-test) is used. We experiment with the hyper-parameters λ_1 to λ_3 and empirically find that the

optimum weights are $\lambda_1 = 0.5$, $\lambda_2 = 1$ and $\lambda_3 = 0.1$. We also modified the loss function and replaced the MAE with BerHu.

The trained models were tested on some complex indoor images having good depths and variation in contrasts. The performances of our models are plotted in Figures 11–13. The training loss and training accuracy are given in Figures 14 and 15. The validation loss and validation accuracy are given in Figures 16 and 17.

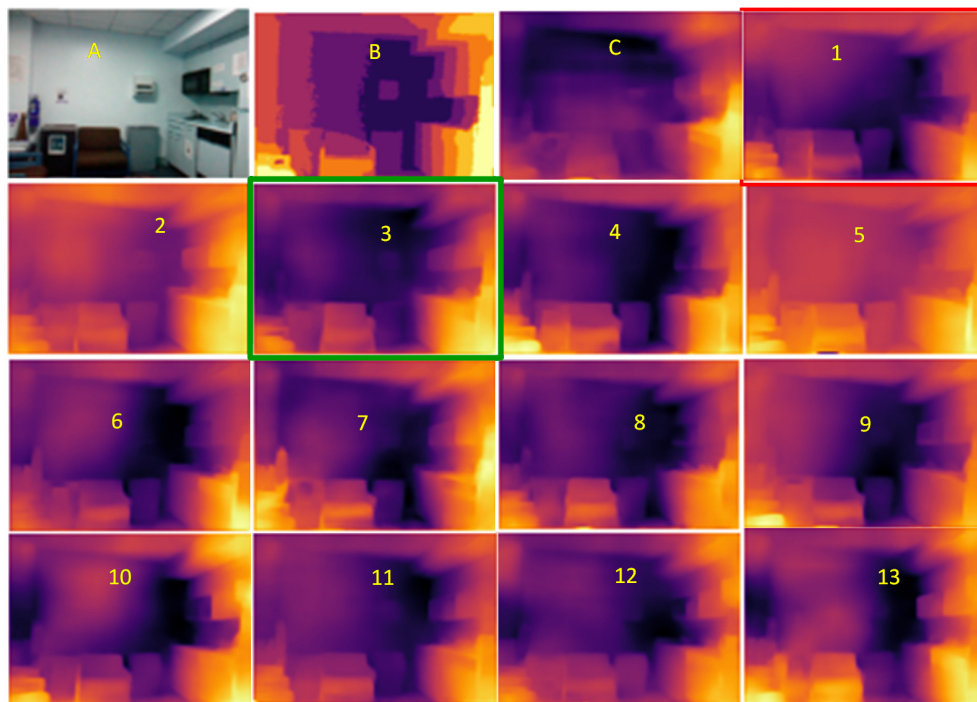


Figure 11. Depth map prediction after training, a visual comparison. **A:** Input image, **B:** Ground Truth, **C:** UNETP, **1:** NDWT (3C, 3R, 3Bs) + Bs, **2:** NADWT (3C, 3LR, 1Bs), **3:** NADWT (3C, 3LR, 1Bs) + Bs, **4:** NADWT (3C, 3Bs, 3R) + Bs, **5:** NADWT (3C, 3R, 3Bs) + Bs, **6:** NRDWT (3C, 3R, 3Bs) + Bs, **7:** NRDWT (3C, 3Bs, 3R) + Bs, **8:** NARDWT(3C, 3LR, 3Bs) + Bs, **9:** NARDWT (3C, 3R, 3Bs) + Bs, **10:** NARDWT (3C, 3Bs, 3LR) + Bs, **11:** NARDWT (3C, 3Bs, 3LR), **12:** NARDWT (3C, 3LR), **13:** NARDWT (4C, 4Bs, 4LR) + 1Bs).

In Figure 11 we take UNET++ as the base for work (Figure 11C). The details of feature depths are barely visible. Our basic NDWT (Figure 11(1)) has a configuration of three convolution layers, three ReLU layers, and post-batch normalization. This model provides better feature detailing showing that edges and high-frequency features are propagated from early encoder features to the estimated features. This model is augmented with attention gate as NADWT (3C, 3LR, 1Bs) NADWT (3C, 3LR, 1Bs) + Bs, NADWT (3C, 3Bs, 3R) + Bs, and NADWT (3C, 3R, 3Bs) + Bs (Figure 11(2–5)). An attention grid generated in these models improves the areas of relevance with higher weight and brings out the object boundaries (Figure 11(2)). LR activation with batch normalization layers only at the final stages improved the depth dynamic range (Figure 11(3)). This model gives the best performance. The same effect is seen with batch normalization layers before each activation layer (Figure 11(4)). Batch normalization after each activation layer reduces the depth range (Figure 11(5)) as the activation function has pruned the lower value neurons. NDWT with residual convolution further improves the object details (Figure 11(6,7) left corner objects) but blurs the edges lightly. Here, again, batch normalization before the activation layer is better. An attention gate is also added to NRDWT models (Figure 11(8–13)). Here, a model with R activation layers makes the estimation better when compared with ground truth. A reduction in the batch norm before the final output shows a slight loss of detail of the sofa arm (Figure 11(11)). Removing all batch norm layers leads to the degradation of definitions in the estimated image (Figure 11(12)). We increased the convolution layers in

one model (Figure 11(13)). The near objects are stronger but definitions at the end of the room are lost. Figure 12 plots the performance of training loss of each of the 13 models. We also show the performance of UNet with DWT here for comparison. NADWT (3C, 3LR, 1Bs) + Bs has the lowest loss and correlates with the depth image in Figure 11(3). The model evaluation accuracy performance (Figure 13) also supports this. Figures 14–17 show the model training loss performance, accuracy performances, and validation accuracy performances. The curves are close, indicating that the models are well-trained. The jagged lines are indications of over-fit or under-fit. The training is fast and reaches near saturation within 10 epochs.

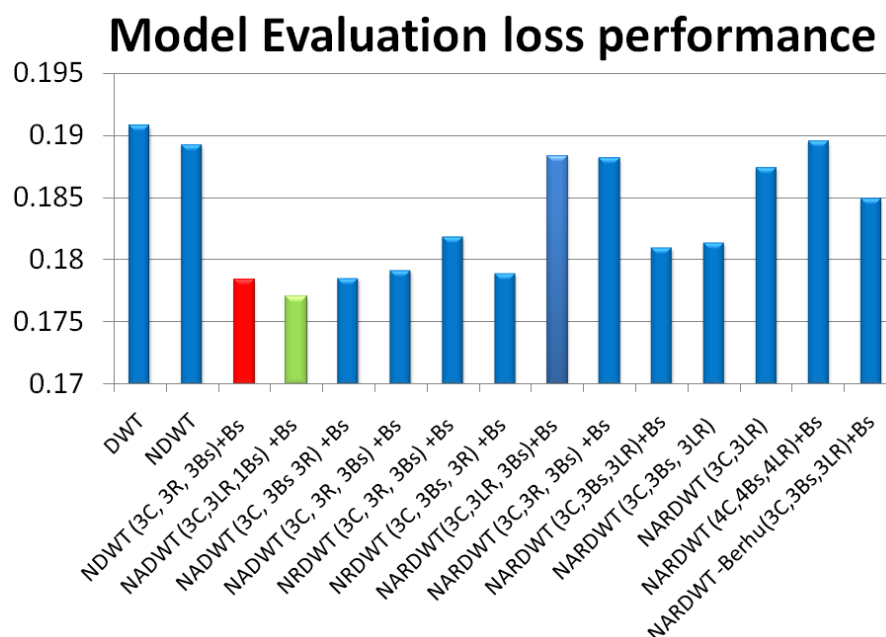


Figure 12. Model loss performance. The best is DWT + Attention followed by Residual + Attention architecture.

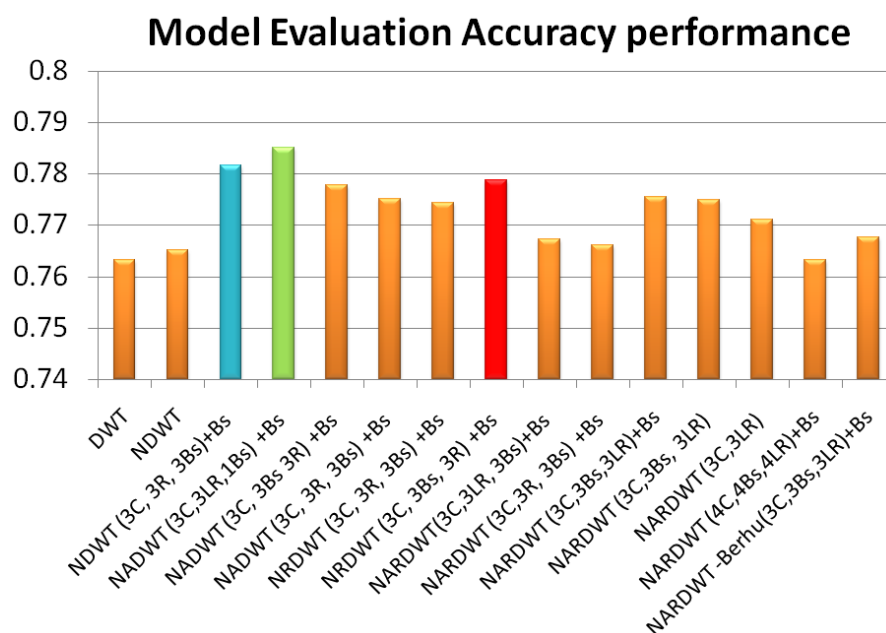


Figure 13. Model evaluation accuracy performance.

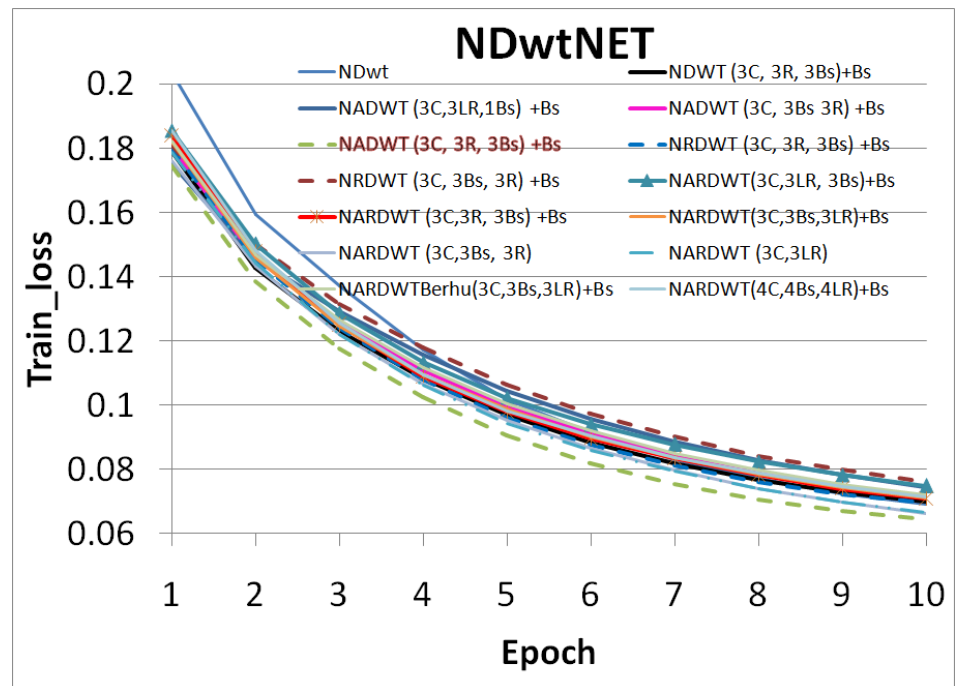


Figure 14. Model training loss performance.

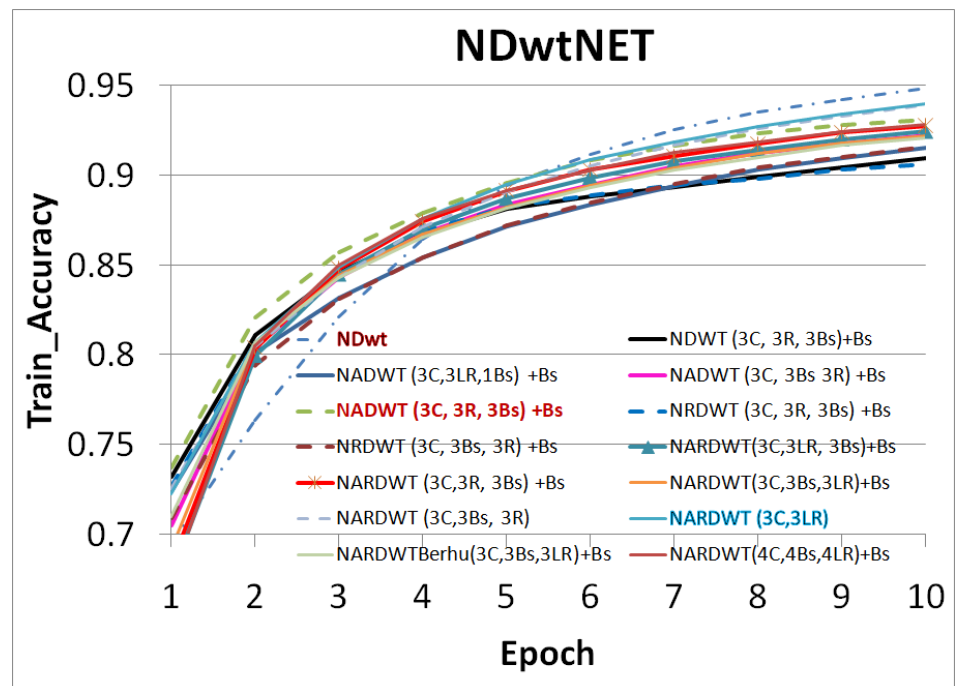


Figure 15. Model training accuracy performance.

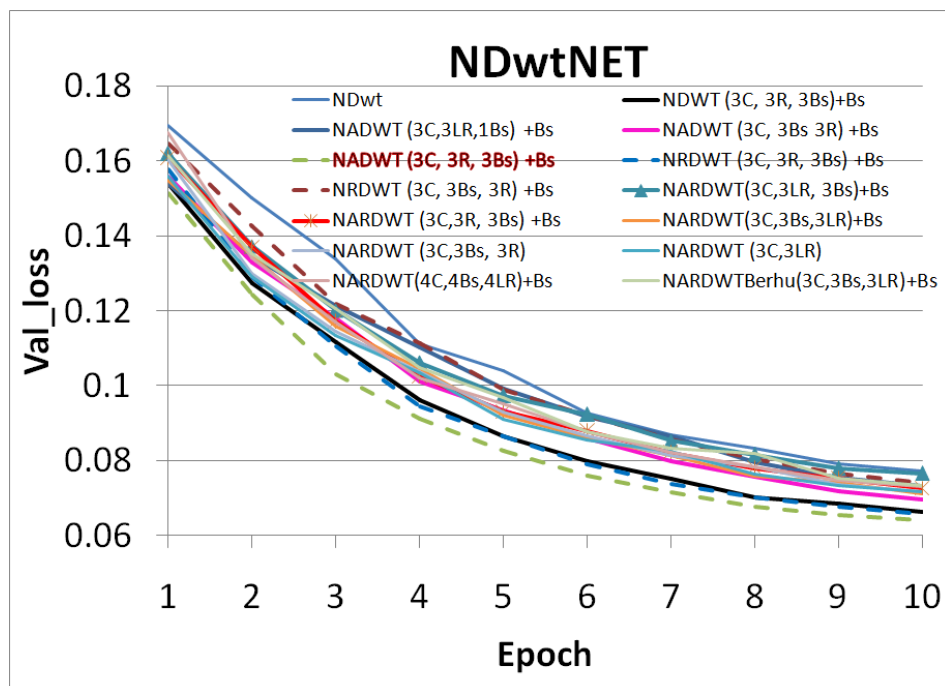


Figure 16. Model validation loss performance.

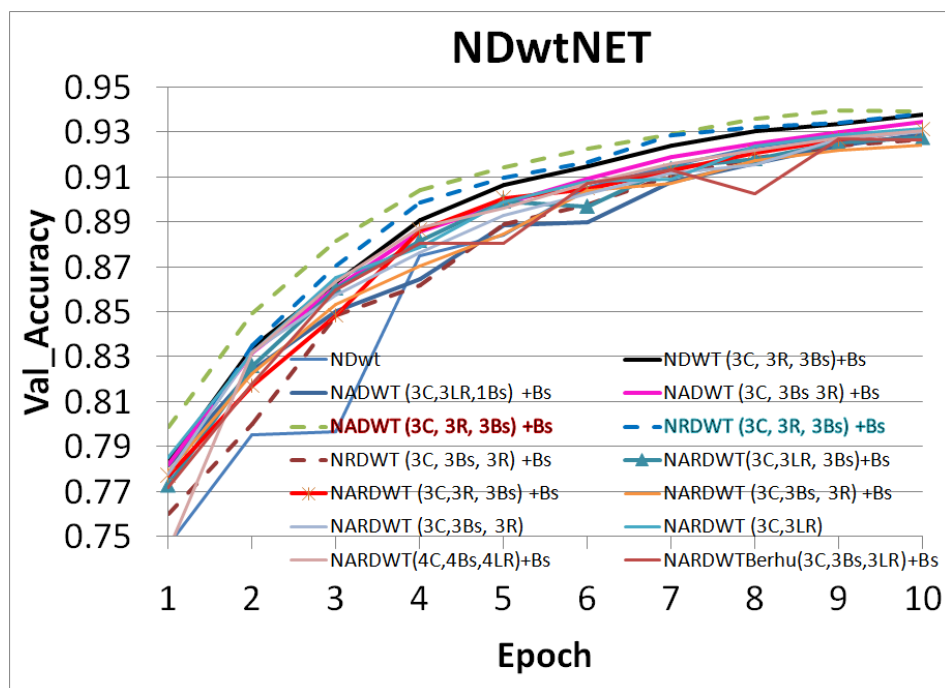


Figure 17. Model validation accuracy performance.

8. Results and Observation

Figure 11 shows the visual quality of models after training. We summarize our observations below:

- Batch normalization: improves the depth range and loss. Batch normalization after the activation layer degrades loss. Additional computations and trainable parameters.
- Activation: among activation layers, the LR activation function offered higher performance. Training and validation performance is better with ReLU.
- Attention: gives higher training, validation, and evaluation scores.

- Residual: gives lower training and validation accuracy but the evaluation score is moderately better. Requires more training.
- Convolution: higher convolution layers do not improve performance, but visually give better representation.
- Loss function: replacing MAE loss with BerHu loss did not show improvement.

The overall best loss performance is from the NDWT + Attention (NADWT) network followed by the NDWT model and third the Residual (NRDWT) model (Figures 12 and 13). It is also observed that the NARDWT models need more training iterations. The best training accuracy is from the NDWT model followed by the NARDWT model. The validation accuracy performance plots (Figure 17) show that NARDWT models tend to saturate faster. We verified the performance of our model–3 with Haar wavelets instead of db4 wavelets. There seems to be a minor improvement in performance as tabulated in Table 2.

The performance parameters are tabulated in Table 3. Our models are superior to published DWT-type models and UNET++ for depth prediction. Our network has better scores for all six types of performance metrics. All our model variants performed better as seen in Figures 12 and 13. The primary improvement is due to higher convolution layers. We experimented with low convolution layer density NDWT having two layers and NDWT with three layers. The performance was better with three layers. This was the optimal number as blocks with four convolution layers had a lower performance. The performance further improved with the inclusion of the attention function. Attention enables the learning of finer structures leading to performance improvements. Regularization with bath normalization before the activation layer corrects the co-variance shift of learned weights. This adds to the improvement. We also compared our results with a UNET based on a DenseNet backbone encoder. This model performed higher as (1) it used pretrained models and weights (2) it was additionally trained on a more extensive set of improved image sets (50,000) and (3) The trainable parameters are very large. We trained our model from scratch on less than 400 image sets from NYU.

The training time increases with convolution density, batch normalization layers and density, attention feature, and residual convolution blocks. The NARDWT models took approximately 17 mins to train with the NYU dataset per epoch using A100, 40 GB GPU (premium GPU). The training increased to about 97 mins for T4 GPU (standard GPU). This was the same with the Haar wavelet-based model. Our light network trains faster compared to UNET with a DenseNet backbone which, in our experimentation, took about 150 min per epoch.

Table 2. Performance of model with different wavelets.

Models	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	REL \downarrow	RMSE \downarrow	log10 \downarrow
db4	0.33	0.61	0.81	0.39	0.16	0.18
Harr	0.34	0.62	0.82	0.39	0.15	0.17

Table 3. Comparison of model performances.

Models	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	REL \downarrow	RMSE \downarrow	log10 \downarrow	Year
DWT	0.27	0.52	0.73	0.54	1.76	0.21	2023 *
ADWT	0.27	0.51	0.70	0.80	1.57	0.23	2023 *
UNET++	0.29	0.55	0.75	0.66	1.69	0.21	2023 *
DenseNet [7]	0.85	0.97	0.99	0.12	0.52	0.05	2018
DORN [54]	0.83	0.97	0.99	0.12	0.51	0.05	2018
P3Depth [55]	0.898	0.98	0.996	0.1	0.36	0.04	2022
NewCRFs [56]	0.92	0.99	0.998	0.095	0.33	0.04	2022
ZoeD-M12-N [57]	0.96	0.995	0.999	0.075	0.27	0.03	2023
NADWT(3)	0.33	0.61	0.81	0.39	0.16	0.18	2023

* Trained with NYU dataset.

9. Conclusions

We developed a DWT-based dense network model that successfully predicts depth from an image. Our network learns to estimate the wavelet coefficients through loss functions consisting of MAE, SSIM, and gradient functions. We experiment with various variants of our network and demonstrate that the performance is better than UNet and UNet++. The network can train fast with NYU datasets, and the average accuracy reaches more than 92% within 10 epochs. The training time of 17 min per epoch is faster than other models based on dense networks. We completed ablation studies with batch normalization and activation types and infer that evaluation performance is best with Leaky ReLU activation and dense batch normalization. The activation layer before batch normalization provided the best-trained models. We studied the density of convolution layers in our models. More convolution layers in a block increase the trainable feature map density and hence higher performance. Higher-density convolution layers yielded better visual results also. The speed of training is an advantage of our model. This speed is primarily due to lower trainable feature maps compared to dense networks like DenseNet, RESNET, etc. The lower feature maps have the disadvantage of lower accuracy. It is observed that estimations are poor for smooth surfaces at the far end of the scene. These aspects require more analysis and study. Further, the network performance for the outdoors will be studied using the KITTI dataset in our subsequent versions of this work. The scope for future studies is increased blocks in the network with higher trainable parameters and pruning of the non-performing weights in these feature maps. This will be a trade-off study for speed and performance.

Author Contributions: S.P.: Conceptualization, Methodology, Investigation, Software, Validation, Writing—Original draft preparation. S.K.M.: Supervision. Writing—Reviewing and Editing. D.M.: Project administration, Supervision, Investigation, Resources, Writing—Reviewing and Editing. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

2D	Two dimensional
3D	Three dimensional
ADAM	Adaptive Moment Estimation
BerHu	Reversed Huber loss
CNN	Convolution Neural Network
DWT	Discrete wavelet transforms
GPU	Graphic processing unit
IWT	Inverse DWT
LIDAR	LIght Detection and Ranging
LR	Leaky ReLU
MAE	Mean Absolute Error
MSE	Mean square error
mins	Minutes
NDWTN	Nested Discrete Waveform Transform Net
NYU	New York University
RADAR	Radio Detection and Ranging
ReLU	Rectified Linear Unit

RGB	Red, Green and Blue
RMSE	Root mean square error
SONAR	Sound Navigation and Ranging
SSIM	Structural Similarity Index

References

1. Ens, J.; Lawrence, P. An investigation of methods for determining depth from focus. *IEEE Trans. Pattern Anal. Mach. Intell.* **1993**, *15*, 97–108. [\[CrossRef\]](#)
2. Xian, T.; Subbarao, M. Performance evaluation of different depth from defocus (DFD) techniques. *Proc. SPIE* **2005**, *6000*, 87–99. [\[CrossRef\]](#)
3. Lee, S.; Hayes, M.H.; Paik, J. Distance estimation using a single computational camera with dual off-axis color filtered apertures. *Opt. Express* **2013**, *21*, 23116–23129. [\[CrossRef\]](#)
4. Mather, G. The Use of Image Blur as a Depth Cue. *Perception* **1997**, *26*, 1147–1158.
5. Eigen, D.; Puhres, C.; Fergus, R. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. *arXiv* **2014**, arXiv:1406.2283.
6. Harsányi, K.; Kiss, A.; Majdik, A.; Sziranyi, T. A Hybrid CNN Approach for Single Image Depth Estimation: A Case Study. In Proceedings of the Multimedia and Network Information Systems (MISSI 2018), Wroclaw, Poland, 12–14 September 2018; Choroś, K., Kopel, M., Kukla, E., Siemiński, A., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 372–381.
7. Alhashim, I.; Wonka, P. High Quality Monocular Depth Estimation via Transfer Learning. *arXiv* **2018**, arXiv:1812.11941.
8. Shivakumar, S.S.; Nguyen, T.; Miller, I.D.; Chen, S.W.; Kumar, V.; Taylor, C.J. DFuseNet: Deep Fusion of RGB and Sparse Depth Information for Image Guided Dense Depth Completion. *arXiv* **2019**, arXiv:1902.00761.
9. Laina, I.; Rupprecht, C.; Belagiannis, V.; Tombari, F.; Navab, N. Deeper Depth Prediction with Fully Convolutional Residual Networks. *arXiv* **2016**, arXiv:1606.00373.
10. Zhao, C.; Sun, Q.; Zhang, C.; Tang, Y.; Qian, F. Monocular depth estimation based on deep learning: An overview. *Sci. China Technol. Sci.* **2020**, *63*, 1612–1627. [\[CrossRef\]](#)
11. He, L.; Wang, G.; Hu, Z. Learning Depth From Single Images With Deep Neural Network Embedding Focal Length. *IEEE Trans. Image Process.* **2018**, *27*, 4676–4689. [\[CrossRef\]](#)
12. Chi, J.; Gao, J.; Qi, L.; Zhang, S.; Dong, J.; Yu, H. Depth estimation of a single RGB image with semi-supervised two-stage regression. In Proceedings of the 5th International Conference on Communication and Information Processing, Chongqing, China, 15–17 November 2019; pp. 97–102. [\[CrossRef\]](#)
13. Masoumian, A.; Rashwan, H.A.; Cristiano, J.; Asif, M.S.; Puig, D. Monocular Depth Estimation Using Deep Learning: A Review. *Sensors* **2022**, *22*, 5353. [\[CrossRef\]](#)
14. Zhu, J.; Liu, L.; Liu, Y.; Li, W.; Wen, F.; Zhang, H. FG-Depth: Flow-Guided Unsupervised Monocular Depth Estimation. *arXiv* **2023**, arXiv:2301.08414.
15. Godard, C.; Mac Aodha, O.; Brostow, G.J. Unsupervised Monocular Depth Estimation with Left-Right Consistency. *arXiv* **2016**, arXiv:1609.03677. <https://doi.org/10.48550/ARXIV.1609.03677>.
16. Bhat, S.F.; Alhashim, I.; Wonka, P. AdaBins: Depth Estimation Using Adaptive Bins. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 19–25 June 2021. [\[CrossRef\]](#)
17. Li, B.; Zhang, H.; Wang, Z.; Liu, C.; Yan, H.; Hu, L. Unsupervised monocular depth estimation with aggregating image features and wavelet SSIM (Structural SIMilarity) loss. *Intell. Robot.* **2021**, *1*, 84–98. [\[CrossRef\]](#)
18. Zhao, S.; Fu, H.; Gong, M.; Tao, D. Geometry-Aware Symmetric Domain Adaptation for Monocular Depth Estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019.
19. Antensteiner, D.; Štolc, S.; Huber-Mörk, R. Depth Estimation with Light Field and Photometric Stereo Data Using Energy Minimization. In Proceedings of the Progress in Pattern Recognition, Image Analysis, Computer Vision and Applications (CIARP 2016), Lima, Peru, 8–11 November 2016; Beltrán-Castañón, C., Nyström, I., Famili, F., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 175–183.
20. Woodham, R.J. Photometric Method For Determining Surface Orientation From Multiple Images. *Opt. Eng.* **1980**, *19*, 191139. [\[CrossRef\]](#)
21. Chen, G.; Han, K.; Wong, K.Y.K. PS-FCN: A Flexible Learning Framework for Photometric Stereo. *arXiv* **2018**, arXiv:1807.08696. <https://doi.org/10.48550/ARXIV.1807.08696>.
22. Chen, G.; Han, K.; Shi, B.; Matsushita, Y.; Wong, K.Y.K. Deep Photometric Stereo for Non-Lambertian Surfaces. *arXiv* **2020**, arXiv:2007.13145. <https://doi.org/10.48550/ARXIV.2007.13145>.
23. Ju, Y.; Jian, M.; Guo, S.; Wang, Y.; Zhou, H.; Dong, J. Incorporating Lambertian Priors Into Surface Normals Measurement. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–13. [\[CrossRef\]](#)
24. Van Dijk, T.; de Croon, G.C.H.E. How do neural networks see depth in single images? *arXiv* **2019**, arXiv:1905.07005.
25. Yue, H.; Zhang, J.; Wu, X.; Wang, J.; Chen, W. Edge Enhancement in Monocular Depth Prediction. In Proceedings of the 2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA), Kristiansand, Norway, 9–13 November 2020; pp. 1594–1599. [\[CrossRef\]](#)

26. Xie, J.; Feris, R.S.; Sun, M.T. Edge-Guided Single Depth Image Super Resolution. *IEEE Trans. Image Process.* **2016**, *25*, 428–438. [\[CrossRef\]](#)
27. Zhang, C.; Tian, Y. Edge Enhanced Depth Motion Map for Dynamic Hand Gesture Recognition. In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops, Portland, OR, USA, 23–28 June 2013; pp. 500–505. [\[CrossRef\]](#)
28. Paul, S.; Jhamb, B.; Mishra, D.; Kumar, M.S. Edge loss functions for deep-learning depth-map. *Mach. Learn. Appl.* **2022**, *7*, 100218. [\[CrossRef\]](#)
29. Wolter, M.; Garcke, J. Adaptive wavelet pooling for convolutional neural networks. *Proc. Mach. Learn. Res.* **2021**, *130*, 1936–1944.
30. Ferrà, A.; Aguilar, E.; Radeva, P. Multiple Wavelet Pooling for CNNs. In Proceedings of the Computer Vision–ECCV 2018 Workshops, Munich, Germany, 8–14 September 2018; Leal-Taixé, L., Roth, S., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 671–675.
31. Yang, H.H.; Yang, C.H.H.; James Tsai, Y.C. Y-Net: Multi-Scale Feature Aggregation Network With Wavelet Structure Similarity Loss Function For Single Image Dehazing. In Proceedings of the ICASSP 2020—2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 4–8 May 2020; pp. 2628–2632. [\[CrossRef\]](#)
32. Ramamonjisoa, M.; Firman, M.; Watson, J.; Lepetit, V.; Turmukhambetov, D. Single Image Depth Estimation using Wavelet Decomposition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021.
33. Yu, B.; Wu, J.; Islam, M.J. UDepth: Fast Monocular Depth Estimation for Visually-guided Underwater Robots. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), London, UK, 29 May–2 June 2023.
34. Zioulis, N.; Albanis, G.; Drakoulis, P.; Alvarez, F.; Zarpalas, D.; Daras, P. Hybrid Skip: A Biologically Inspired Skip Connection for the UNet Architecture. *IEEE Access* **2022**, *10*, 53928–53939. [\[CrossRef\]](#)
35. Luo, C.; Li, Y.; Lin, K.; Chen, G.; Lee, S.J.; Choi, J.; Yoo, Y.F.; Polley, M.O. Wavelet Synthesis Net for Disparity Estimation to Synthesize DSLR Calibre Bokeh Effect on Smartphones. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020; pp. 2404–2412. [\[CrossRef\]](#)
36. Li, Q.; Shen, L.; Guo, S.; Lai, Z. Wavelet Integrated CNNs for Noise-Robust Image Classification. *arXiv* **2020**, arXiv:2005.03337. <https://doi.org/10.48550/ARXIV.2005.03337>.
37. Liu, P.; Zhang, H.; Lian, W.; Zuo, W. Multi-level Wavelet Convolutional Neural Networks. *IEEE Access* **2019**, *7*, 74973–74985. [\[CrossRef\]](#)
38. Olaf Ronneberger, P.F.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In Proceedings of the Medical Image Computing and Computer-Assisted Intervention, MICCAI 2015, Munich, Germany, 5–9 October 2015; Springer International Publishing: Cham, Switzerland, 2015; pp. 234–241. [\[CrossRef\]](#)
39. Oktay, O.; Schlemper, J.; Folgoc, L.L.; Lee, M.; Heinrich, M.; Misawa, K.; Mori, K.; McDonagh, S.; Hammerla, N.Y.; Kainz, B.; et al. Attention U-Net: Learning Where to Look for the Pancreas. *arXiv* **2018**, arXiv:1804.03999.
40. Zhang, Z.; Liu, Q.; Wang, Y. Road Extraction by Deep Residual U-Net. *IEEE Geosci. Remote Sens. Lett.* **2018**, *15*, 749–753. [\[CrossRef\]](#)
41. Yang, H.H.; Fu, Y. Wavelet U-Net and the Chromatic Adaptation Transform for Single Image Dehazing. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; pp. 2736–2740. [\[CrossRef\]](#)
42. Wang, Y.; Zhu, X.; Zhao, Y.; Wang, P.; Ma, J. Enhancement of Low-Light Image Based on Wavelet U-Net. *J. Phys. Conf. Ser.* **2019**, *1345*, 022030. [\[CrossRef\]](#)
43. Li, Y.; Wang, Y.; Leng, T.; Zhijie, W. Wavelet U-Net for Medical Image Segmentation. In Proceedings of the Artificial Neural Networks and Machine Learning—ICANN 2020: 29th International Conference on Artificial Neural Networks, Bratislava, Slovakia, 15–18 September 2020; Part I; Springer: Berlin/Heidelberg, Germany, 2020; pp. 800–810. [\[CrossRef\]](#)
44. Chuter, J.L.; Boullanger, G.B.; Saez, M.N. U-Net. In *1 T: A U-Net exploration, in Depth*, 2018. Stanford University CS229 Projects, Fall 2018 Edition. Available online: <https://cs229.stanford.edu/proj2018/report/34.pdf> (accessed on 9 March 2023).
45. Sharma, M.; Sharma, A.; Tushar, K.R.; Panneer, A. A Novel 3D-Unet Deep Learning Framework Based on High-Dimensional Bilateral Grid for Edge Consistent Single Image Depth Estimation. In Proceedings of the 2020 International Conference on 3D Immersion (IC3D), Brussels, Belgium, 15 December 2020; pp. 1–8. [\[CrossRef\]](#)
46. Liu, P.; Zhang, Z.; Meng, Z.; Gao, N. Monocular Depth Estimation with Joint Attention Feature Distillation and Wavelet-Based Loss Function. *Sensors* **2021**, *21*, 54. [\[CrossRef\]](#)
47. Zhou, Z.; Siddiquee, M.M.R.; Tajbakhsh, N.; Liang, J. UNet++: A Nested U-Net Architecture for Medical Image Segmentation. *arXiv* **2018**, arXiv:1807.10165.
48. Peng, D.; Zhang, Y.; Guan, H. End-to-End Change Detection for High Resolution Satellite Images Using Improved UNet++. *Remote. Sens.* **2019**, *11*, 1382. [\[CrossRef\]](#)
49. Gur, S.; Wolf, L. Single Image Depth Estimation Trained via Depth from Defocus Cues. *arXiv* **2020**, arXiv:2001.05036.
50. Silberman, N.; Hoiem, D.; Kohli, P.; Fergus, R. Indoor Segmentation and Support Inference from RGBD Images. In Proceedings of the Computer Vision–ECCV 2012, Florence, Italy, 7–13 October 2012; Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; pp. 746–760.

51. Lubor Ladicky, J.S.; Pollefeys, M. Pulling Things out of Perspective. In Proceedings of the CVPR '14: 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; IEEE Computer Society: Washington, DC, USA, 2014; pp. 89–96. [[CrossRef](#)]
52. Wang, Y. MobileDepth: Efficient Monocular Depth Prediction on Mobile Devices. *arXiv* **2020**, arXiv:2011.10189.
53. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv* **2015**, arXiv:1502.03167.
54. Fu, H.; Gong, M.; Wang, C.; Batmanghelich, K.; Tao, D. Deep Ordinal Regression Network for Monocular Depth Estimation. *arXiv* **2018**, arXiv:1806.02446. <https://doi.org/10.48550/ARXIV.1806.02446>.
55. Patil, V.; Sakaridis, C.; Liniger, A.; Van Gool, L. P3Depth: Monocular Depth Estimation with a Piecewise Planarity Prior. *arXiv* **2022**, arXiv:2204.02091. <https://doi.org/10.48550/ARXIV.2204.02091>.
56. Yuan, W.; Gu, X.; Dai, Z.; Zhu, S.; Tan, P. NeW CRFs: Neural Window Fully-connected CRFs for Monocular Depth Estimation. *arXiv* **2022**, arXiv:2203.01502. <https://doi.org/10.48550/ARXIV.2203.01502>.
57. Bhat, S.F.; Birkel, R.; Wofk, D.; Wonka, P.; Müller, M. ZoeDepth: Zero-shot Transfer by Combining Relative and Metric Depth. *arXiv* **2023**, arXiv:2302.12288. <https://doi.org/10.48550/ARXIV.2302.12288>.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.