# Patch-Wise Attention Network for Monocular Depth Estimation

**Sihaeng Lee,** [1] **Janghyeon Lee,** [2] **Byungju Kim,** [2,3] **Eojindl Yi,** [2] **Junmo Kim** [1,2]
[1] Division of Future Vehicle, KAIST, Daejeon, South Korea
[2] School of Electrical Engineering, KAIST, Daejeon, South Korea
[3] Mathpresso Inc, Seoul, South Korea
{haeng, wkdgus9305, feidfoe, djwld93, junmo.kim}@kaist.ac.kr

## Abstract

In computer vision, monocular depth estimation is the problem of obtaining a high-quality depth map from a two-dimensional image. This map provides information on three-dimensional scene geometry, which is necessary for various applications in academia and industry, such as robotics and autonomous driving. Recent studies based on convolutional neural networks achieved impressive results for this task. However, most previous studies did not consider the relationships between the neighboring pixels in a local area of the scene. To overcome the drawbacks of existing methods, we propose a patch-wise attention method for focusing on each local area. After extracting patches from an input feature map, our module generates attention maps for each local patch, using two attention modules for each patch along the channel and spatial dimensions. Subsequently, the attention maps return to their initial positions and merge into one attention feature. Our method is straightforward but effective. The experimental results on two challenging datasets, KITTI and NYU Depth V2, demonstrate that the proposed method achieves significant performance. Furthermore, our method outperforms other state-of-the-art methods on the KITTI depth estimation benchmark.

## Introduction

Monocular depth estimation is the task of generating a dense depth map from a single RGB image. It has been studied extensively as a fundamental problem in computer vision, which is relevant in various applications such as robotics and autonomous vehicles. An accurate dense depth map provided for a corresponding RGB image is particularly useful for understanding the three-dimensional (3D) geometric information of a scene for solving various computer vision tasks. Therefore, an estimated high-quality depth map could be used as prior information for processing an RGB image, and it is practical in both academia and industry. For this reason, the depth estimation task is critical and requires a solution. However, the estimation of a depth map from a monocular image is ambiguous and ill-posed since a two-dimensional (2D) scene may be projected from an infinite number of real-world 3D scenes.

Early researchers used statistically useful hand-crafted features based on texture, color information, and perspective geometry to overcome this ambiguity (Saxena, Chung, and Ng 2006; Saxena, Sun, and Ng 2008; Malik and Choi 2008; Karsch, Liu, and Kang 2014). In recent years, approaches based on convolutional neural networks (CNNs) have made breakthroughs with excellent performance in numerous computer vision tasks. Based on the success of CNNs in other tasks, many previous studies (Fu et al. 2018; Lee et al. 2019; Yin et al. 2019; Diaz and Marathe 2019; Zhang et al. 2019; Kim et al. 2020) on monocular depth estimation employed CNNs to extract features that are more significant than hand-crafted ones, by using a large amount of labeled data.

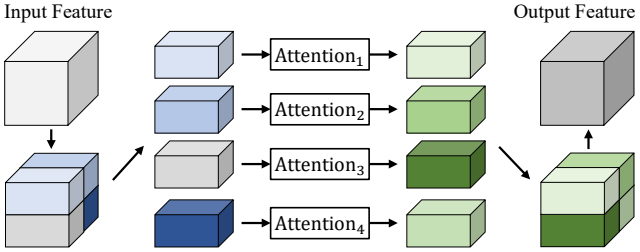


Figure 1: Example of proposed patch-wise attention. Each local patch extracted from an input feature has a corresponding attention module. Attention maps that pass through each attention module return to their initial position and merge into one output feature.

However, these methods do not include a component to consider the relationships between the neighboring pixels in a local area. Intuitively, it can be considered that neighboring depth values of the same object should be close (Gan et al. 2018). That is, the relationship between the neighboring pixels representing the same object in a local area should be close, whereas those of different objects should be far. Few studies have considered the relationships between neighboring pixels. Saxena, Chung, and Ng (2006) introduced the first learning-based method for monocular depth estimation and employed a Markov random field (MRF) to learn the relationships between different points or parts of the image (Saxena, Sun, and Ng 2008). Gan et al. (2018) proposed the affinity layer to extract relative features, which represents the correlation of the values with the surrounding pixels. This layer can be integrated into a fully end-to-end

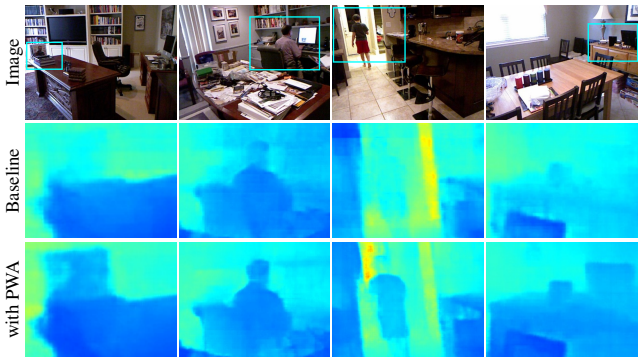


Figure 2: Effectiveness of proposed method. These results demonstrate that our patch-wise attention effectively provides objects reconstruction and cleaner boundaries in the local area.

CNN architecture. Zhang et al. (2018) proposed a regularization method using the gradient information of the object boundary in the local area.

Moreover, depth estimation in autonomous driving has additional challenges. The light detection and ranging (LiDAR) value is inevitably noisy owing to its characteristics. To create a wide field of view, a mechanical LiDAR uses a rotating assembly. When the mechanical LiDAR is rotating, if the LiDAR sensor is moving by itself or if the surrounding environment changes, the sensor value may differ even for the same object. For example, the laser scanner, which is a component of the KITTI dataset (Geiger et al. 2013) recording system, spins at 10 frames per second. This means that the laser scanner sees the same direction once again after 0.1 s. As the KITTI dataset was obtained from a driving car, we can reasonably assume that the data are noisy. Therefore, considering the relationships among the neighboring pixels in the local area is important for improving the performance of the depth estimation task.

To address the abovementioned problems, we propose the patch-wise attention (PWA) method that is straightforward but effective. Unlike most attention methods that are applied to the whole scene, the proposed method is designed to focus on the relationships among the neighboring pixels in the local areas of a scene. To achieve this, we first combine the input features, including the local context information and the global context features, which already represent information on each local area. Subsequently, we feed them to the convolutional layer for converging the global and local context information and extract patches of a predefined patch size. Next, each local patch sequentially passes through the corresponding channel and spatial attention modules to generate an attention map and find the important features. Because each local patch has the corresponding attention module, it can focus on its position and learn the relationship among the neighboring pixels. The attention maps return to their initial position and merge into one attention map. Finally, the attention map is multiplied by the input feature to produce a refined feature.

We conduct comprehensive evaluations on two challenging datasets, KITTI Eigen split (Geiger et al. 2013) and NYU Depth V2 (Silberman et al. 2012). The results demonstrate that the proposed method achieves significant improvements compared to state-of-the-art methods, both qualitatively and quantitatively. Furthermore, our method outperforms other state-of-the-art methods on the online KITTI depth estimation benchmark (Uhrig et al. 2017).

## Related Work

### Supervised Monocular Depth Estimation

Monocular depth estimation has been extensively studied. Saxena, Chung, and Ng (2006) used a discriminatively-trained MRF to predict the depth with local and global image features. Using a similar methodology, they incorporated monocular and stereo cues by applying an MRF (Saxena et al. 2007). Liu, Gould, and Koller (2010) performed two phases, semantic segmentation and depth reconstruction, to leverage context information for depth estimation. Liu, Salzmann, and He (2014) defined depth prediction as a discrete-continuous optimization problem and suggested particle belief propagation to solve this task.

In recent years, with the success of CNNs, numerous depth estimation studies have been published. Eigen, Puhrsch, and Fergus (2014) suggested a multiscale deep network with two component stacks: the coarse-scale network and fine-scale network. Furthermore, they proposed a scale-invariant error to measure the relationships between pixels in the image, regardless of the global scale. Li et al. (2015) employed a CNN model to learn the mapping from the image to the depth or surface normal and refined it to the pixel level using conditional random fields. Fu et al. (2018) and Diaz and Marathe (2019) addressed the problem by adopting a spacing-increasing discretization strategy and reformulated the training as ordinal regression. Yin et al. (2019) used not only 2D space information, but also geometric constraints in 3D space. They reconstructed a 3D point cloud from the estimated depth and exploited 3D geometry to enhance the performance. Lee et al. (2019) applied local planar guidance layers in the decoder to recover the original resolution effectively. Kim et al. (2020) suggested two new modules to refine the multiscale encoder features and obtain rich contextual features. Lee and Kim (2020) proposed a new algorithm for adaptively initializing and balancing the weights of multiple losses that were used in training monocular depth estimators. Wang et al. (2020) introduced additional tasks such as depth-based scene classification and depth reconstruction, to generate hierarchical embeddings. A loss function based on the embeddings and a newly proposed fusion network was adopted to enhance accuracy and generalization performance.

### Attention Mechanism

Attention methods, which have already shown remarkable performance in natural language processing, are also emerging as a standard for improving the performance of various computer vision tasks. Attention mechanisms are favored because they are highly cost-effective. They improve

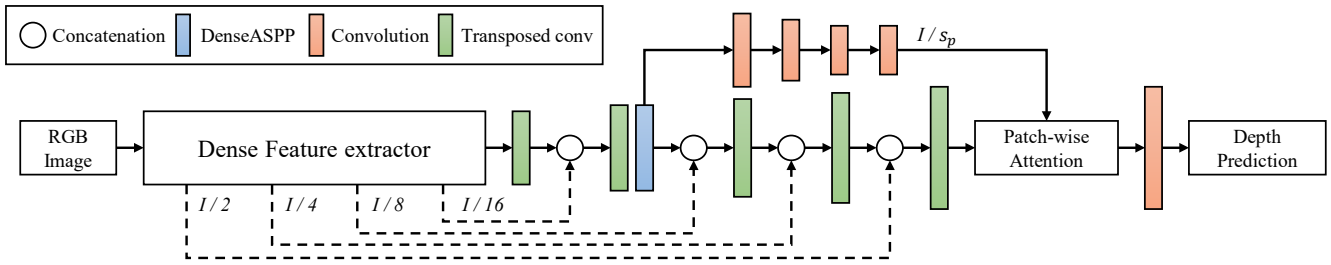


Figure 3: Overall architecture. The proposed architecture is an end-to-end framework based on the encoder-decoder scheme (Ronneberger, Fischer, and Brox 2015). The proposed patch-wise attention module is located after the last upsampling stage. $s_p$ is the patch size.

the overall performance with reasonable additional parameters. Among them, few methods exhibit high compatibility, and can be readily integrated into existing CNN architectures. Hu, Shen, and Sun (2018) introduced a squeeze-and-excitation block that adaptively recalibrated features via channel-wise attention generated from global average-pooled features. Woo et al. (2018) extended the aforementioned module by applying both max-pooling and average pooling when generating attention maps and also arranged the pathways in a sequential manner. Fu et al. (2019) proposed a dual attention network for scene segmentation. They opted for a position attention module instead of spatial attention, which captured the contextual relationships between each and every feature. Wang et al. (2018) presented a general purpose non-local block that captures long-range dependencies by computing the response at a position as a weighted sum of features over all positions. Huynh et al. (2020) devised an attention mechanism called depth-attention volume that was tailored for enhancing monocular depth estimation. It enhances learning by capturing a coplanar relation between non-local points.

# Method

## Overall Architecture

The overall architecture is shown in Figure 3. We employ the encoder-decoder method proposed by Ronneberger, Fischer, and Brox (2015). The transposed convolutional layers in the decoder have $3 \times 3$ kernels with a stride of 2. Following the second upsampling stage, DenseASPP (Yang et al. 2018) is implemented to capture multiscale information from both the encoder and decoder features. At the end of the decoder, our patch-wise attention (PWA) module uses the feature from the last upsampling layer and the global context features from DenseASPP. Next, we feed PWA module's output to the last convolutional layer. Subsequently, the output is input into the sigmoid function. Finally, the desired depth map is generated by scaling to a predefined maximum depth value.

## Patch-Wise Attention Module

We propose a patch-wise attention (PWA) to consider the relationships among the neighboring pixels in the local area. Unlike most attention methods focus on the whole scene, PWA focuses on each local patch extracted from the input feature. Figure 4 shows the proposed PWA module. We apply patch-wise channel attention and then another step of patch-wise spatial attention sequentially. The PWA module takes two inputs, a local context feature $F \in \mathbb{R}^{C\times H\times W}$ and a global context feature $F_G \in \mathbb{R}^{C\times H/s_p\times W/s_p}$, where $s_p$ denotes the predefined patch size. Each pixel of $F_G$ corresponds to each patch, which has the size of $s_p \times s_p$, extracted from $F$. Therefore, the one pixel of $F_G$ and the one patch of $F$ represent the same area of the input feature. The number of patches is set to $H/s_p \times W/s_p$.

**Patch-Wise Channel Attention.** We first operate the max-pooing and average-pooling using a size of $s_p \times s_p$ to generate two different feature maps $F^c_{max}$ and $F^c_{avg}$, respectively aggregating spatial information, where $\{F^c_{max}, F^c_{avg}\} \subset \mathbb{R}^{C\times H/s_p\times W/s_p}$. Subsequently, $F_G$, $F^c_{max}$, and $F^c_{avg}$ are concatenated and passed through a $3 \times 3$ convolutional layer $Conv_c$ to generate a feature map $F_c \in \mathbb{R}^{C\times H/s_p\times W/s_p}$, which is a feature that combines the global and local context information.

$$\begin{aligned} F_c &= Conv_c([F_G; MaxPool_s(F); AvgPool_s(F)]) \\ &= Conv_c([F_G; F^c_{max}; F^c_{avg}]) \end{aligned} \tag{1}$$

Next, we extract patches $F^c_i \in \mathbb{R}^{C\times 1\times 1}$ from $F_c$, where $i$ denotes the patch of $i^{th}$ position in $F_c$. Each local patch $F^c_i$ is forwarded to a corresponding multilayer perceptron ($MLP_i$) that comprises one hidden layer and the ReLU activation function. We set the number of hidden layer channels to $C/8$ to reduce network parameters. Then, channel attention vectors $E^c_i \in \mathbb{R}^{C\times 1\times 1}$ are generated with a sigmoid function $\sigma$.

$$E^c_i = \sigma(MLP_i(F^c_i)) \tag{2}$$

Each $E^c_i$ returns to an initial location and appears as an attention map $E_c \in \mathbb{R}^{C\times H/s_p\times W/s_p}$. Finally, we obtain a patch-wise channel refined feature $F' \in \mathbb{R}^{C\times H\times W}$ by multiplying the interpolated $E_c$ using scale factor $s_p$ and the local context feature $F$.

**Patch-Wise Spatial Attention.** Next, we discuss the generation of a patch-wise spatial attention, which is similar to the generation of the patch-wise channel attention. To generate the spatial attention map, we first concatenate the channel refined feature $F'$ and interpolated global context feature $F^I_G \in \mathbb{R}^{C\times H\times W}$ by scale factor $s_p$. Second, we obtain

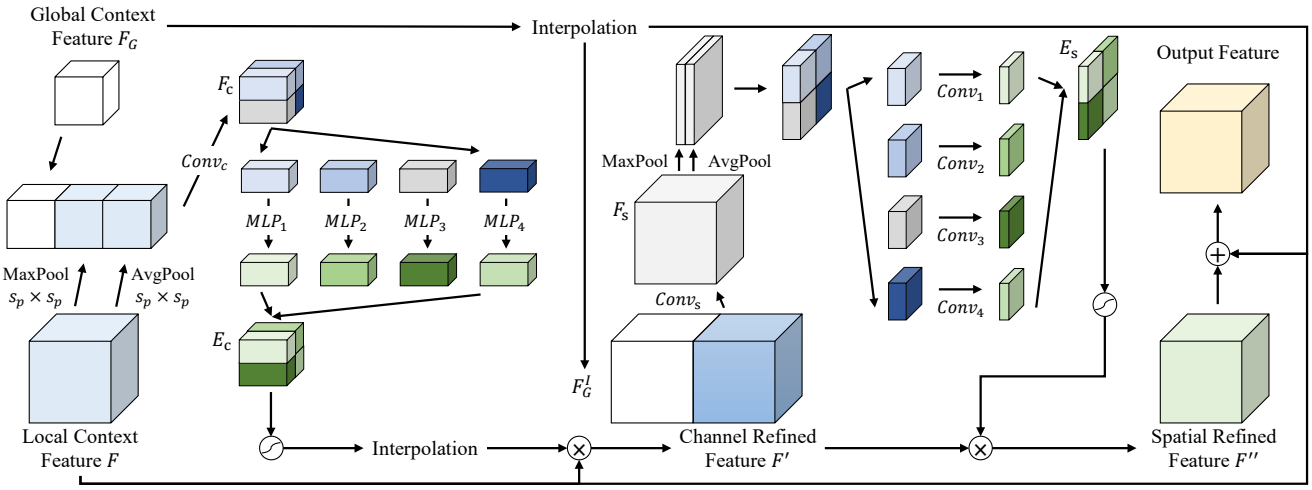


Figure 4: Patch-Wise Attention Method. Our method refines an input feature with attention weights generated by focusing on the local area. Each local patch has the corresponding attention modules. $s_p$ is the patch size.

$F_s \in \mathbb{R}^{C\times H\times W}$ when the concatenated feature is passed through $3 \times 3$ convolutional layer $Conv_s$.

$$F_s = Conv_s([F'; F_G^I]) \tag{3}$$

Next, we extract patches $F_j^s \in \mathbb{R}^{2\times s_p\times s_p}$ from a concatenation of max-pooled and average-pooled features, $F_{max}^s$ and $F_{avg}^s$, over channel dimensions of $F_s$, where $\{F_{max}^s, F_{avg}^s\} \subset \mathbb{R}^{1\times H\times W}$, and $j$ denotes the patch of $j^{th}$ position in $F_s$. Notably, $F_j^s$ and $F_i^c$ represent the same area of the scene if $j = i$. We feed each local patch to a corresponding convolutional layer $Conv_j$ with a kernel size of $7 \times 7$. Each $Conv_j$ aggregates the spatial information with the sigmoid activation to generate a spatial attention $E_j^s \in \mathbb{R}^{1\times s_p\times s_p}$ to the corresponding patch.

$$\begin{aligned} E_j^s &= \sigma(Conv_j([MaxPool_c(F_{s,j}); AvgPool_c(F_{s,j})])) \\ &= \sigma(Conv_j([F_{max,j}^s; F_{avg,j}^s])) \\ &= \sigma(Conv_j(F_j^s)) \end{aligned} \tag{4}$$

Each $E_j^s$ returns to an initial location and appears as an attention map $E_s \in \mathbb{R}^{1\times H\times W}$. After expanding $E_s$ over the channel dimension, we obtain a patch-wise spatial refined feature $F'' \in \mathbb{R}^{C\times H\times W}$ by multiplying the channel refined feature $F'$.

Finally, we generate the desired final output using a skip connection to add $F''$, the original local context feature $F$ and the interpolated global context feature $F_G^I$. Therefore, patch-wise channel attention and spatial attention can learn the relationship between neighboring pixels by focusing on the local area in the scene.

## Training Loss Function

To train the network, we adopt a well-known training loss for depth prediction proposed by Eigen, Puhrsch, and Fergus (2014). Specifically, the loss is a mixture of element-wise $l_2$ and the scale-invariant error. If we denote a network prediction and the corresponding ground truth by $y$ and $y^*$, respectively, and set

$$d_i = \log y_i - \log y_i^* \tag{5}$$

to be the difference between the prediction and ground truth at pixel $i$, then the loss function $L$ is defined as

$$L = \sqrt{\frac{1}{n}\sum_i d_i^2 - \frac{\lambda}{n^2}\left(\sum_i d_i\right)^2}, \tag{6}$$

where $n$ is the number of valid pixels of the ground truth. We set $\lambda$ to 0.5 as in Eigen, Puhrsch, and Fergus (2014).

# Experiments

We conducted extensive experiments to compare our method with state-of-the-art approaches on two datasets: NYU Depth V2 (Silberman et al. 2012) and KITTI (Geiger et al. 2013). Moreover, we demonstrated the evaluation results on the KITTI online benchmark server (Uhrig et al. 2017). All experiments were implemented on PyTorch (Paszke et al. 2019).

## Datasets and Evaluation Metrics

**NYU Depth V2 dataset.** The NYU Depth V2 dataset (Silberman et al. 2012) contains 464 indoor scenes, which contain 120K images and paired depth maps with a resolution of 640×480. As in the previous studies, we divided the dataset into 249 scenes for training and 215 scenes (654 images) for testing. All images used for the experiments on NYU Depth V2 were collected from Lee et al. (2019). In the evaluation, we applied a center crop from Eigen, Puhrsch, and Fergus (2014) used in Fu et al. (2018); Lee et al. (2019); Yin et al. (2019).

**KITTI dataset.** The KITTI dataset (Geiger et al. 2013) contains 61 outdoor scenes captured by multiple sensors mounted on a driving car. The dataset contains stereo images with a resolution of approximately $1241 \times 375$ and the

| Variant | | $\delta_1 \uparrow$ | $\delta_2 \uparrow$ | $\delta_3 \uparrow$ | AbsRel↓ | SqRel↓ | RMSE↓ | RMSE$_{log}$ ↓ | SILog↓ |
|---|---|---|---|---|---|---|---|---|---|
| NYU Depth V2 | | | | | | | | | |
| Baseline | | 0.783 | 0.956 | 0.990 | 0.154 | 0.109 | 0.512 | 0.192 | 16.131 |
| Baseline + CBAM (Woo et al. 2018) | | 0.814 | 0.966 | 0.992 | 0.137 | 0.092 | 0.481 | 0.175 | 14.323 |
| Baseline + DualAttention (Fu et al. 2019) | | 0.808 | 0.963 | 0.993 | 0.143 | 0.096 | 0.485 | 0.177 | 14.542 |
| Baseline + Patch-Wise Attention | $s_p = 8$ | 0.808 | 0.962 | 0.992 | 0.141 | 0.094 | 0.483 | 0.179 | 15.004 |
| | $s_p = 16$ | 0.814 | 0.964 | <u>0.993</u> | 0.140 | 0.093 | 0.481 | 0.178 | 14.668 |
| | $s_p = 32$ | 0.824 | 0.967 | **0.994** | 0.135 | 0.086 | 0.461 | 0.170 | 14.062 |
| | $s_p = 64$ | <u>0.833</u> | **0.971** | **0.994** | <u>0.131</u> | **0.083** | **0.454** | **0.165** | **13.478** |
| | $s_p = 128$ | **0.834** | <u>0.969</u> | **0.994** | **0.130** | <u>0.084</u> | **0.454** | <u>0.166</u> | 13.698 |
| | $s_p = 256$ | <u>0.833</u> | <u>0.969</u> | <u>0.993</u> | <u>0.131</u> | 0.087 | 0.464 | 0.167 | <u>13.667</u> |
| | $s_p =$ img size | 0.803 | 0.960 | 0.992 | 0.145 | 0.100 | 0.492 | 0.182 | 15.198 |
| Baseline + Patch-Wise Attention + DenseASPP (Yang et al. 2018) | $s_p = 32$ | 0.852 | **0.976** | **0.995** | 0.121 | 0.074 | 0.430 | 0.155 | 12.776 |
| | $s_p = 64$ | **0.860** | **0.976** | **0.995** | **0.119** | **0.072** | **0.424** | **0.152** | **12.541** |
| | $s_p = 128$ | 0.858 | 0.975 | 0.994 | 0.122 | 0.076 | 0.428 | 0.155 | 12.641 |
| KITTI Eigen split | | | | | | | | | |
| Baseline | | 0.937 | 0.991 | 0.998 | 0.072 | 0.299 | 3.040 | 0.112 | 10.365 |
| Baseline + CBAM (Woo et al. 2018) | | 0.939 | 0.991 | 0.998 | 0.072 | 0.294 | 3.004 | 0.111 | 10.177 |
| Baseline + DualAttention (Fu et al. 2019) | | 0.929 | 0.989 | 0.997 | 0.079 | 0.329 | 3.225 | 0.119 | 11.124 |
| Baseline + Patch-Wise Attention | $s_p = 8$ | 0.940 | <u>0.991</u> | **0.998** | 0.071 | 0.290 | 3.018 | 0.110 | 10.126 |
| | $s_p = 16$ | 0.942 | <u>0.991</u> | **0.998** | 0.070 | 0.286 | 2.965 | 0.109 | 10.020 |
| | $s_p = 32$ | <u>0.944</u> | **0.992** | **0.998** | <u>0.069</u> | **0.279** | **2.961** | <u>0.108</u> | <u>9.857</u> |
| | $s_p = 64$ | **0.945** | **0.992** | **0.998** | **0.068** | <u>0.280</u> | <u>2.964</u> | **0.107** | **9.746** |
| | $s_p = 128$ | 0.942 | **0.992** | **0.998** | 0.070 | 0.288 | 3.038 | 0.109 | 9.984 |
| | $s_p = 256$ | 0.941 | <u>0.991</u> | **0.998** | 0.071 | 0.293 | 3.020 | 0.109 | 10.150 |
| | $s_p =$ img size | 0.940 | <u>0.991</u> | **0.998** | 0.071 | 0.286 | 2.985 | 0.109 | 10.082 |
| Baseline + Patch-Wise Attention + DenseASPP (Yang et al. 2018) | $s_p = 32$ | 0.946 | **0.992** | **0.998** | **0.069** | 0.283 | 2.982 | 0.106 | 9.713 |
| | $s_p = 64$ | **0.947** | **0.992** | **0.998** | **0.069** | **0.277** | **2.927** | **0.105** | **9.493** |
| | $s_p = 128$ | 0.944 | **0.992** | **0.998** | **0.069** | 0.280 | 2.944 | 0.107 | 9.715 |

Table 1: Ablation study on both NYU Depth V2 and KITTI Eigen split datasets. Here, $\delta_i$ denotes $\delta < 1.25^i$. The baseline is the encoder-decoder network that uses MobileNetV2 (Sandler et al. 2018) as a backbone network. $s_p$ is the patch size. Best results are in bold and second results are underlined in each category.

corresponding LiDAR point clouds. We used a recently released official dataset from KITTI, which includes RGB images and the post-processed depth maps from the projected LiDAR point clouds as the ground truth. The ground truth is produced by combining LiDAR scans. In the experiments on the KITTI Eigen split, we followed the common data split proposed by Eigen, Puhrsch, and Fergus (2014) for comparison with previous studies. In the evaluation, we used the center crop proposed by Garg et al. (2016) as in Guo et al. (2018); Lee et al. (2019). For the online KITTI depth prediction, we used the official benchmark split (Uhrig et al. 2017). The KITTI benchmark dataset contains 85,898 training and 1,000 selected validation data, respectively, as well as 500 test data without the ground truth. The test data was cropped to a size of 1216×352.

**Evaluation Metrics.** For the NYU Depth V2 and KITTI Eigen split datasets, we compared our results with previous studies using the following commonly used evaluation metrics by Eigen, Puhrsch, and Fergus (2014): the accuracy under the threshold ($\delta_i < 1.25^i, i = 1, 2, 3$), mean absolute relative error (AbsRel), mean squared relative error (SqRel), root mean squared error (RMSE), root mean squared log error (RMSE$_{log}$), and mean $\log_{10}$ error (log10). We used the following metrics for the online KITTI depth prediction benchmark (Uhrig et al. 2017): scale invariant logarithmic error (SILog), percentage of AbsRel and SqRel (absErrorRel, sqErrorRel), and root mean squared error of the inverse depth (iRMSE).

## Implementation Details

We used ResNeXt101 (Xie et al. 2017), DenseNet161 (Huang et al. 2017), and MobileNetV2 (Sandler et al. 2018), which were pretrained on image classification using the ImageNet-1K dataset (Russakovsky et al. 2015), as the backbone networks. The output channels of transposed convolutional layers were [512, 256, 128, 64, and 32] on ResNeXt101 and DenseNet161, and [320, 96, 32, 24, and 32] on MobilinetV2, sequentially. For the network training, we used a mini-batch size of 8 and 16 on NYU Depth V2 and KITTI dataset. We adopted the ADAM optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. The learning rate started from 0.0005 on MobileNetV2 and 0.0001 on ResNeXt101 and DenseNet161. We used a polynomial decay method with power $p = 0.9$ to schedule the learning

| Method | Backbone | cap | $\delta_1 \uparrow$ | $\delta_2 \uparrow$ | $\delta_3 \uparrow$ | AbsRel↓ | SqRel↓ | RMSE↓ | $\text{RMSE}_{log} \downarrow$ |
|---|---|---|---|---|---|---|---|---|---|
| Saxena, Sun, and Ng (2008) | - | 0-80m | 0.601 | 0.820 | 0.926 | 0.280 | 3.012 | 8.734 | 0.361 |
| Eigen, Puhrsch, and Fergus (2014) | - | | 0.692 | 0.899 | 0.967 | 0.190 | 1.515 | 7.156 | 0.270 |
| Gan et al. (2018) | ResNet50 | | 0.890 | 0.964 | 0.985 | 0.098 | 0.666 | 3.933 | 0.173 |
| Guo et al. (2018) | VGG-16 | | 0.892 | 0.967 | 0.986 | 0.096 | 0.641 | 4.095 | 0.168 |
| Fu et al. (2018) | ResNet101 | | 0.932 | 0.984 | 0.994 | 0.072 | 0.307 | 2.727 | 0.120 |
| Yin et al. (2019) | ResNeXt101† | | 0.938 | 0.990 | 0.998 | 0.072 | - | 3.258 | 0.117 |
| Lee et al. (2019) | DenseNet161 | | 0.956 | 0.993 | 0.998 | **0.059** | 0.241 | 2.756 | 0.096 |
| Kim et al. (2020) | ResNeXt101 | | **0.958** | 0.993 | **0.999** | 0.060 | 0.231 | 2.650 | 0.094 |
| Ours | MobileNetV2 | 0-80m | 0.947 | 0.992 | 0.998 | 0.069 | 0.277 | 2.928 | 0.105 |
| | DenseNet161 | | 0.956 | **0.994** | **0.999** | 0.062 | 0.240 | 2.708 | 0.096 |
| | ResNeXt101 | | **0.958** | **0.994** | **0.999** | 0.060 | **0.221** | **2.604** | **0.093** |
| Gan et al. (2018) | ResNet50 | 0-50m | 0.898 | 0.967 | 0.986 | 0.094 | 0.552 | 3.133 | 0.165 |
| Guo et al. (2018) | VGG-16 | | 0.901 | 0.971 | 0.988 | 0.092 | 0.515 | 3.163 | 0.159 |
| Fu et al. (2018) | ResNet101 | | 0.936 | 0.985 | 0.995 | 0.071 | 0.268 | 2.271 | 0.116 |
| Lee et al. (2019) | DenseNet161 | | 0.964 | 0.994 | **0.999** | **0.056** | 0.169 | 1.925 | **0.087** |
| Kim et al. (2020) | ResNeXt101 | | 0.964 | **0.995** | **0.999** | 0.058 | 0.163 | 1.893 | 0.088 |
| Ours | MobileNetV2 | 0-50m | 0.956 | 0.994 | 0.998 | 0.066 | 0.204 | 2.124 | 0.098 |
| | DenseNet161 | | 0.963 | **0.995** | **0.999** | 0.059 | 0.175 | 1.952 | 0.089 |
| | ResNeXt101 | | **0.965** | **0.995** | **0.999** | 0.057 | **0.161** | **1.872** | **0.087** |

Table 2: Performance on KITTI Eigen split. All methods are evaluated on the split by Eigen, Puhrsch, and Fergus (2014). Our approach achieves state-of-the-art results. Here, $\delta_i$ denotes $\delta < 1.25^i$, and † denotes the method using ResNeXt101 (32x4d).

rate, and we set the numbers of epochs for training our networks to 10. We augmented data as follows. Randomly horizontal flipping was applied in all experiments. Thereafter, a random rotation of the inputs was applied in ranges of [-5, 5] and [-1, 1] for the NYU Depth V2 and KITTI datasets, respectively. Furthermore, we randomly changed the brightness, contrast, and saturation of the input images in the range of [0.8, 1.2], and the hue in the range of [0.9, 1.1]. All data augmentations were performed with a 50% probability. For the KITTI dataset, we trained our networks $1216 \times 352$ using a bottom-center crop. For the NYU Depth V2, we train our networks on full-size images. If the width and height of an input image are not divided by the patch size, we applied the zero-padding operation to the input image.

## Ablation Study

We performed extensive ablation studies to verify the effectiveness of our proposed methods. Both the KITTI and NYU Depth V2 datasets were used in these experiments. The results are displayed in Table 1. Moreover, examples of the qualitative results on the NYU Depth V2 dataset are presented in Figure 2. In this study, the baseline network included the encoder and decoder and used MobileNetV2 as a backbone.

The proposed PWA method could outperform global attention method counterparts, such as CBAM (Woo et al. 2018) and DualAttention (Fu et al. 2019), for a proper patch size of $s_p$. If $s_p$ is too small, then each patch represents an area that is extremely small; therefore, it was difficult to produce a meaningful attention map despite the increased number of parameters, resulting in poor performance. However, if $s_p$ is too large, the PWA lost its advantage, and became similar to global attention methods. Therefore, moderate value of $s_p$ would work effectively; experimentally, the value was established as approximately on both NYU Depth V2 and KITTI Eigen split datasets. $s_p = 64$ also worked well with DenseASPP (Yang et al. 2018) on both datasets, with significant performance gains. Therefore, 64 was a robust optimal value of $s_p$ in various situations. We used this value for further experiments.

## Comparison with State-of-the-art

For comparison, we set $s_p$ to 64 and used MobileNetV2, DenseNet161, and ResNeXt101 (32x8d) as backbone networks.

**KITTI Eigen split.** We compared the proposed networks with state-of-the-art methods on the KITTI Eigen split dataset. In the experiments, we set the maximum depth value to 80 m. As indicated in Table 2, our ResNeXt-based model achieved a state-of-the-art performance in the ranges of both 0 to 50 m and 0 to 80 m, except for AbsRel. Especially, our MobileNet-based model outperformed the methods from Fu et al. (2018); Yin et al. (2019) that used ResNet101 and ResNeXt101 (32x4d) as backbone networks, respectively.

**KITTI Benchmark.** We also evaluated the proposed method on the online KITTI depth prediction benchmark server[1]. In the experiments on the KITTI benchmark, we set the maximum depth value to 90 m. The results are presented in Table 3. Compared with the other methods, our ResNeXt-based model achieved state-of-the-art performance in terms of SILog, which is the main metric of the benchmark. The qualitative results are presented in Figure 5.

[1]http://www.cvlibs.net/datasets/kitti/eval_depth.php?benchmark=depth_prediction


Figure 5: Qualitative comparison with other state-of-the-art methods on the KITTI benchmark dataset. Our method provides cleaner boundaries and more effective object reconstruction than the other methods.

| Method | SILog | sqErrRel | absErrRel | iRMSE |
|---|---|---|---|---|
| PAP (Zhang et al. 2019) | 13.08 | 2.72 | 10.27 | 13.95 |
| VNL (Yin et al. 2019) | 12.65 | 2.46 | 10.15 | 13.02 |
| SORD (Diaz and Marathe 2019) | 12.39 | 2.49 | 10.10 | 13.48 |
| DORN (Fu et al. 2018) | 11.77 | 2.23 | **8.78** | 12.98 |
| BTS (Lee et al. 2019) | 11.67 | **2.21** | 9.04 | **12.23** |
| Ours | **11.45** | 2.30 | 9.05 | 12.32 |

Table 3: Comparison with state-of-the-art methods on online KITTI depth prediction benchmark.

**NYU Depth V2.** To validate our method's efficiency in indoor scenes, we evaluated our networks on the NYU Depth V2 dataset. Table 4 indicates that our method outperforms the other methods on all metrics. The proposed DenseNet-based model reduces AbsRel and RMSE by approximately 4% compared to that of Lee et al. (2019) using the same backbone network. Moreover, our ResNeXt-based model outperformed the methods from Kim et al. (2020) using the same backbone network. Furthermore, our MobileNet-based model showed the best performance among the methods using a lightweight backbone network, as shown in Table 5.

| Method | higher is better | | | lower is better | | |
|---|---|---|---|---|---|---|
| | $\delta_1$ | $\delta_2$ | $\delta_3$ | AbsRel | log10 | RMSE |
| Eigen and Fergus (2015) | 0.769 | 0.950 | 0.988 | 0.158 | - | 0.641 |
| Fu et al. (2018) | 0.828 | 0.965 | 0.992 | 0.115 | 0.051 | 0.509 |
| Zhang et al. (2019) | 0.846 | 0.968 | 0.994 | 0.121 | - | 0.497 |
| Yin et al. (2019) | 0.875 | 0.976 | 0.994 | 0.108 | 0.048 | 0.416 |
| Kim et al. (2020) | 0.878 | 0.981 | 0.995 | 0.111 | 0.047 | 0.388 |
| Huynh et al. (2020) | 0.882 | 0.980 | 0.996 | 0.108 | - | 0.412 |
| Lee et al. (2019) | 0.885 | 0.978 | 0.994 | 0.110 | 0.047 | 0.392 |
| Ours (ResNeXt101) | **0.892** | 0.984 | **0.997** | **0.105** | **0.045** | 0.376 |
| Ours (DenseNet161) | **0.892** | **0.985** | **0.997** | **0.105** | **0.045** | **0.374** |

Table 4: Performance on NYU Depth V2. Our method outperforms the state-of-the-art methods on all evaluation metrics. Here, $\delta_i$ denotes $\delta < 1.25^i$.

| Method | higher is better | | | lower is better | | |
|---|---|---|---|---|---|---|
| | $\delta_1$ | $\delta_2$ | $\delta_3$ | AbsRel | RMSE | $\text{RMSE}_{log}$ |
| Nekrasov et al. (2019) | 0.790 | 0.955 | 0.990 | 0.149 | 0.565 | 0.205 |
| Yin et al. (2019) | 0.829 | 0.956 | 0.980 | 0.134 | 0.485 | 0.185 |
| Lee et al. (2019) | **0.860** | 0.974 | 0.993 | 0.121 | 0.431 | 0.156 |
| Ours | **0.860** | **0.976** | **0.995** | **0.119** | **0.424** | **0.152** |

Table 5: Results on NYU Depth V2 with a lightweight backbone (MobileNetV2). Here, $\delta_i$ denotes $\delta < 1.25^i$.

## Conclusion

In this paper, we proposed a patch-wise attention method to consider the relationships among the neighboring pixels in the local area, which is straightforward but effective. As opposed to the previous attention modules, which were applied to the whole scene, our method applies a separate attention module to each local patch extracted from the input feature. Because each local patch has the corresponding attention module, our method can focus on each local area and learn the relationships between neighboring pixels. The results of extensive experiments using various backbone networks on two challenging datasets, NYU Depth V2 and KITTI, demonstrated the effectiveness of the proposed method. Our method outperformed other state-of-the-art methods. As part of our future work, we plan to investigate the design of a specific layer to learn more complicated relationships among the neighboring depth values in the local area.

## Acknowledgments

This paper was result of the research project supported by SK Hynix Inc.

## References


Diaz, R.; and Marathe, A. 2019. Soft labels for ordinal regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4738–4747.

Eigen, D.; and Fergus, R. 2015. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2650–2658.

Eigen, D.; Puhrsch, C.; and Fergus, R. 2014. Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems (NIPS)*, 2366–2374.

Fu, H.; Gong, M.; Wang, C.; Batmanghelich, K.; and Tao, D. 2018. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2002–2011.

Fu, J.; Liu, J.; Tian, H.; Li, Y.; Bao, Y.; Fang, Z.; and Lu, H. 2019. Dual attention network for scene segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3146–3154.

Gan, Y.; Xu, X.; Sun, W.; and Lin, L. 2018. Monocular depth estimation with affinity, vertical pooling, and label enhancement. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 224–239. Springer.

Garg, R.; BG, V. K.; Carneiro, G.; and Reid, I. 2016. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 740–756. Springer.

Geiger, A.; Lenz, P.; Stiller, C.; and Urtasun, R. 2013. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research* 32(11): 1231–1237.

Guo, X.; Li, H.; Yi, S.; Ren, J.; and Wang, X. 2018. Learning monocular depth by distilling cross-domain stereo networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 484–500. Springer.

Hu, J.; Shen, L.; and Sun, G. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 7132–7141.

Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4700–4708.

Huynh, L.; Nguyen-Ha, P.; Matas, J.; Rahtu, E.; and Heikkila, J. 2020. Guiding Monocular Depth Estimation Using Depth-Attention Volume. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 581–597. Springer.

Karsch, K.; Liu, C.; and Kang, S. B. 2014. Depth transfer: Depth extraction from video using non-parametric sampling. *IEEE transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 36(11): 2144–2158.

Kim, D.; Lee, S.; Lee, J.; and Kim, J. 2020. Leveraging Contextual Information for Monocular Depth Estimation. *IEEE Access* 8: 147808–147817.

Lee, J. H.; Han, M.-K.; Ko, D. W.; and Suh, I. H. 2019. From big to small: Multi-scale local planar guidance for monocular depth estimation. *arXiv preprint arXiv:1907.10326* .

Lee, J.-H.; and Kim, C.-S. 2020. Multi-Loss Rebalancing Algorithm for Monocular Depth Estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 23–28. Springer.

Li, B.; Shen, C.; Dai, Y.; Van Den Hengel, A.; and He, M. 2015. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1119–1127.

Liu, B.; Gould, S.; and Koller, D. 2010. Single image depth estimation from predicted semantic labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1253–1260.

Liu, M.; Salzmann, M.; and He, X. 2014. Discrete-continuous depth estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 716–723.

Malik, A. S.; and Choi, T.-S. 2008. A novel algorithm for estimation of depth map using image focus for 3D shape recovery in the presence of noise. *Pattern Recognition* 41(7): 2200–2225.

Nekrasov, V.; Dharmasiri, T.; Spek, A.; Drummond, T.; Shen, C.; and Reid, I. 2019. Real-time joint semantic segmentation and depth estimation using asymmetric annotations. In *International Conference on Robotics and Automation (ICRA)*, 7101–7107. IEEE.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NIPS)*, 8024–8035.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 234–241. Springer.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)* 115(3): 211–252.

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4510–4520.

Saxena, A.; Chung, S. H.; and Ng, A. Y. 2006. Learning depth from single monocular images. In *Advances in Neural Information Processing Systems (NIPS)*, 1161–1168.

Saxena, A.; Schulte, J.; Ng, A. Y.; et al. 2007. Depth Estimation Using Monocular and Stereo Cues. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, volume 7, 2197–2203.

Saxena, A.; Sun, M.; and Ng, A. Y. 2008. Make3d: Learning 3d scene structure from a single still image. *IEEE transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 31(5): 824–840.

Silberman, N.; Hoiem, D.; Kohli, P.; and Fergus, R. 2012. Indoor segmentation and support inference from rgbd images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 746–760. Springer.

Uhrig, J.; Schneider, N.; Schneider, L.; Franke, U.; Brox, T.; and Geiger, A. 2017. Sparsity Invariant CNNs. In *International Conference on 3D Vision (3DV)*, 11–20. IEEE.

Wang, L.; Zhang, J.; Wang, Y.; Lu, H.; and Ruan, X. 2020. CLIFFNet for Monocular Depth Estimation with Hierarchical Embedding Loss. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 316–331. Springer.

Wang, X.; Girshick, R.; Gupta, A.; and He, K. 2018. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 7794–7803.

Woo, S.; Park, J.; Lee, J.-Y.; and So Kweon, I. 2018. Cbam: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 3–19. Springer.

Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; and He, K. 2017. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1492–1500.

Yang, M.; Yu, K.; Zhang, C.; Li, Z.; and Yang, K. 2018. Denseaspp for semantic segmentation in street scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3684–3692.

Yin, W.; Liu, Y.; Shen, C.; and Yan, Y. 2019. Enforcing geometric constraints of virtual normal for depth prediction. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 5684–5693.

Zhang, Z.; Cui, Z.; Xu, C.; Yan, Y.; Sebe, N.; and Yang, J. 2019. Pattern-Affinitive Propagation across Depth, Surface Normal and Semantic Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4106–4115.

Zhang, Z.; Xu, C.; Yang, J.; Tai, Y.; and Chen, L. 2018. Deep hierarchical guidance and regularization learning for end-to-end depth estimation. *Pattern Recognition* 83: 430–442.