IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 28, 2021, accepted January 20, 2022, date of publication January 25, 2022, date of current version February 14, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3145969

FastMDE: A Fast CNN Architecture for Monocular Depth Estimation at High Resolution

THIEN-THANH DAO[1], QUOC-VIET PHAM[2], (Member, IEEE), AND WON-JOO HWANG[3], (Senior Member, IEEE)

[1]Department of Computer Engineering, Pusan National University, Yangsan-si 50612, South Korea
[2]Korean Southeast Center for the 4th Industrial Revolution Leader Education, Pusan National University, Busan 46241, South Korea
[3]Department of Biomedical Convergence Engineering, Pusan National University, Yangsan-si 50612, South Korea

Corresponding author: Won-Joo Hwang (wjhwang@pusan.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) Grant, which was funded by the Korean Government through the Ministry of Science and ICT (MSIT) under Grant NRF-2019R1C1C1006143 and Grant NRF-2019R1I1A3A01060518; in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) Grant, which was funded by the Korean Government (MSIT) under Grant 2020-0-01450; and in part by the Artificial Intelligence Convergence Research Center through Pusan National University.

ABSTRACT A depth map helps robots and autonomous vehicles (AVs) visualize the three-dimensional world to navigate and localize neighboring obstacles. However, it is difficult to develop a deep learning model that can estimate the depth map from a single image in real-time. This study proposes a fast monocular depth estimation model named *FastMDE* by optimizing the deep convolutional neural network according to the encoder-decoder architecture. The decoder needs to obtain partial and semantic feature maps from the encoding phase to improve the depth estimation accuracy. Therefore, we designed FastMDE with two effective strategies. The first one involved redesigning the skip connection with the features of the squeeze-excitation module to obtain partial and semantic feature maps of the encoding phase. The second strategy involved redesigning the decoder by using the fusion dense block to permit the usage of high-resolution features that were learned earlier in the network before upsampling. The proposed FastMDE model utilizes only 4.1 M parameters, which is much lesser than the parameters utilized by state-of-art models. Thus, FastDME has a higher accuracy and lower latency than previous models. This study also demonstrates that MDE can leverage deep neural networks in real-time (i.e., 30 fps) with the Linux embedded board Nvidia Jetson Xavier NX. The model can facilitate the development and applications with superior performances and easy deployment on an embedded platform.

INDEX TERMS Efficient CNN, deep neural network, depth map, supervised learning, self-supervised learning.

I. INTRODUCTION

Depth map prediction from a single image is a fundamental aspect of several applications that involve three-dimensional (3D) visualizations of the real world. It can be deployed in multiple applications, such as robotics, autonomous vehicles, and drones [1], [2]. It assists robots in building good simultaneous localization and mapping (SLAM) for autonomous obstacle avoidance [3], [4]. However, existing depth sensors, such as Light Detection and Ranging (LiDAR), structured-light sensors, etc., are typically bulky, heavy, and consume a lot of power. This makes them unsuitable for small robotic platforms. Thus, the development of a depth estimation technique by using a monocular camera is being explored due to its compact size, low cost, and low power consumption.

Researchers have recently been tried to address this problem by performing monocular depth estimation (MDE) through deep learning. Studies [5]–[7] used the encoder-decoder as the backbone of the architecture. The encoder is commonly used in complex networks that were designed for object detection and recognition problems, such as VGG-16 [8], ResNet [9], and DenseNet [10], due to their high expressive power and accuracy. For instance, Alhashim and Wonka [11] used transfer learning and DenseNet-169 for high quality depth estimation. Lee *et al.* [12] used DenseNet-161 with local planar guidance to extract dense encoding features.

The associate editor coordinating the review of this manuscript and approving it for publication was Xinyu Du.

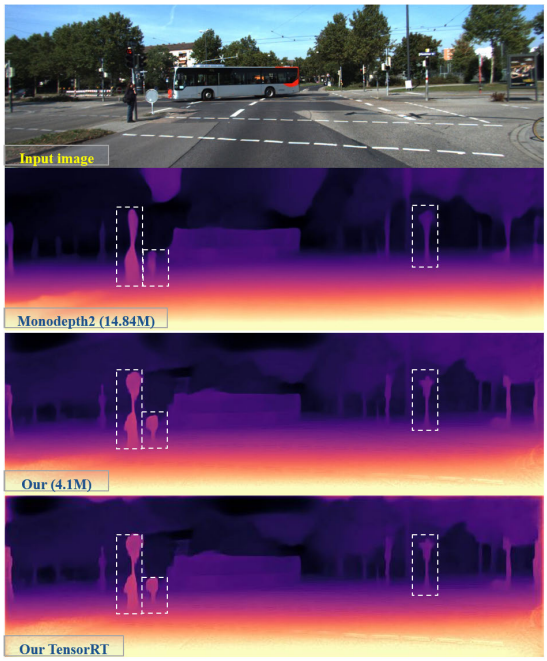


FIGURE 1. MDE on KITTI datasets. The results of the current method were compared with those of Monodepth2 [5]. The current method can produce images with higher quality and sharpness than those of Monodepth2 [5], despite using 3.5 times fewer parameters than the latter. The image resolution in the current study is 1024 × 320.

Godard *et al.* [5] adopted ResNet-18 for the encoder network architecture. However, such complex and deep convolutional neural network (DCNN) architectures, which require over 14 M parameters, lead to high complexity and high latency. Therefore, it is very difficult to deploy these models in real-time industrial applications because they have limited hardware resources on embedded platforms. It is essential to develop an efficient convolutional neural network (CNN) model that can run in real-time on embedded devices.

Previous studies on real-time MDE have adopted network pruning techniques to reduce the size of the model. Wofk *et al.* [13] used MobileNet [14] for the encoding phase and network pruning processes in their model. They obtained fast depth estimation by using the Nvidia Jetson TX2 at 178 fps on a GPU and 27 fps on a CPU with an input image resolution of 224 × 224. However, the model proposed in [13] has low accuracy and supports low-resolution input images. Thus, researchers have adopted reinforcement learning to design high-efficiency model architecture. Shaw *et al.* [15] proposed a fast-semantic segmentation model named SqueezeNAS. This study utilized reinforcement learning to develop the most optimum network architecture for segmentation tasks. They only utilized 1.8 M parameters and achieved high accuracy. Several studies [16]–[19] reported high performances by optimizing the hyper-parameters of the neural network architecture. These studies revealed that the computer could adopt the neural architecture search technique to design neural network architectures automatically. However, this approach requires a lot of GPU computational resources and hundreds to thousands of computational days to obtain the optimal neural network. Thus, this study demonstrates a high-efficiency neural network architecture by accounting for properties that improve the depth prediction results. Since the depth estimation problem requires pixel-based information, the model needs the semantic features and spatial information of the object to predict its boundaries at a high resolution. Semantic feature maps are appropriate for semantic segmentation tasks and can be used to produce boundaries between different objects. Spatial information can help visualize the boundaries of objects, thereby enhancing the depth map estimation.

An analysis to infer high-quality depth estimation is conducted to develop a lightweight model for MDE with high accuracy.

- The deep layers in the encoder of the DCNN contain additional channels to extract high-level features from the input image, thereby ensuring that the semantic features are relatively detailed. Therefore, a network that can capture the high-level features in the deep layers from the encoder and merge them into the features within the decoder is needed. This helps the decoder layers construct a highly detailed semantic features output.
- A bilinear interpolation or interpolates the nearest neighbor at a scale factor of 2 is usually used in upsampling of the decoder. These two methods generate smooth edges or pixelated, interpolated images with stair-step artifact [20]. Therefore, upsampling an image with blurred or noisy edges makes it difficult for the model to estimate the edge information. Thus, the model must predict a sharp edge at a high resolution before it undergoes upsampling.''

Thus, the DCNN architecture is redesigned in this study to achieve high accuracy and low latency for MDE. Our main contributions are summarized below.

- A FastDME architecture that can perform better than state-of-the-art methods while using 3.5 times fewer weights than Monodepth2 [5] and HR-depth [7] is proposed in this study. A comparison between the depth map estimation of the current method and Monodepth2 is shown in Fig. 1. The results demonstrate that the FastDME architecture can predict the depth map at a higher quality with sharper edges than the state-of-art methods.
- The skip connection is redesigned, and the squeeze-excitation (SE) block is used to extract important features of the encoding phase and is referred to as eSE. Thus, the decoder contains more details on the spatial and semantic feature maps.
- The dense connection in the decoder part is redesigned, namely fDense, to easily learn the high-resolution features that are obtained from the skip connection and encoder features to produce highly-detailed edge information before the upsampling process. This allows the model to predict sharper edges at higher accuracy.''

- The TensorRT engine model, which is easily deployed in the Nvidia Linux embedded board (Tested with Nvidia Xavier NX), is then proposed.

The remaining sections of this manuscript are organized as follows. Section II discusses studies related to the proposed model. Section III introduces the proposed FastDME, which includes the novel skip connection with an SE block and a unique fusion dense module. The training specifications are also described in this Section. Section IV compares the results of the FastMDE technique with those of a few state-of-art methods. Section V discusses the ablation studies to analyze the performance improvement realized by the FastDME model. An ablative analysis is performed on different architectural components introduced with the baseline model Monodepth2 [5]. Section VI concludes this study.

II. RELATED STUDIES

This section describes studies related to the deep learning methods used for MDE, such as supervised learning and unsupervised learning. This is followed by a discussion on the existing lightweight MDE networks.

A. THE MONOCULAR DEPTH ESTIMATION APPROACH

1) SUPERVISED LEARNING

MDE is an auto-encoder problem that feeds an input image and permits it to be transformed according to multiple reasonable depths. Earlier studies were based on supervised learning, wherein the model was trained by applying a ground truth depth for the calculation loss. The first supervised learning method [21] was trained on the RGB-D dataset. The network designed in [21] can predict the depth map in two states. The first state has a globally coarse depth, whereas the second state has a locally fine depth to generate pixel-by-pixel depth values. This local prediction assists the model in generating the depth map in great detail. Researchers later developed new network architecture to improve the model's accuracy such that the model was comparable to a depth sensor. Zhang *et al.* [22] proposed a hard-mining network that adopted a similar approach to that of [21]. The study used intra-scale and inter-scale refinement sub-networks to accurately localize and refine the hard-mining regions. This assisted the model in improving the performance of MDE in hard-regions where it is difficult to predict the depth value. Lee *et al.* [12] used atrous spatial pyramid pooling to extract contextual information and local planar guidance for every scale of the decoding phase. Bhat *et al.* [23] is the current state-of-art method for depth estimated based on supervised learning. This study introduces an adaptive bin-width estimator block that divides the depth into bins whose center value is estimated adaptively per image. It then calculates the depth by using a linear combination of the bin center values. However, all supervised learning methods required the ground truth depth value, as discussed above. It is difficult and expensive to generate this value. Further, the readings are sparse and flawed, despite using an expensive depth sensor (e.g., the inability of the sensor to capture the depth information of a moving object). The lack of a dataset of ground truth values results in a poor generalization performance, which leads to biased predictions. As a result, researchers began to explore self-supervised learning (i.e., unsupervised learning) for MDE.

2) UNSUPERVISED LEARNING

The unsupervised learning models for MDE can be trained by multiple monocular images. The input can be a sequence of monocular images or stereo image pairs. Unsupervised learning uses the photometric reprojection error between the corresponding pixels of multiple input images to generate an output depth image. Stereo image pairs were initially used for training unsupervised learning models [6], [24]–[26]. These models used the left and right images of the same scene to compute the disparity map by calculating the displacement between the corresponding pixels of these two images. Several studies [5], [27]–[32] proposed unsupervised learning models that used sequences of images. The depth map prediction was based on the output of two DCNNs, namely the pose estimation network (PoseNet) and the depth estimation network (DepthNet). PoseNet can regress the transformation between adjacent frames, which is used for the reconstruction of the target image. DepthNet predicts the depth map according to the output of PoseNet (additional details are included in Section III). Their studies allow the model to be trained entirely with monocular image sequences. Although the pose network and depth network are used simultaneously during the training process, they can operate separately during the testing process. Monodepth2 [5] introduced a loss function to calculate the minimum photometric error instead of calculating the average error, which was a technique used in a previous study [33]. This improves the sharpness of the occlusion boundary, which significantly increases the accuracy of the model. As a result, Monodepth2 became a widely used baseline. Guizilini *et al.* [32] aimed to further improve the MDE performance by developing a pre-trained network with semantic image segmentation tasks to guide the network learning process. [34] proposed a 3D packing and unpacking network to preserve the spatial information in images and low-level features. This study demonstrates that the standard max pooling and bilinear upsampling techniques are not good enough to preserve semantic and spatial information for depth estimation in detail. In addition, the model uses pack and unpack blocks with 3D convolution. As a result, the number of parameters is greatly increased, thereby making it impossible for it to be deployed in embedded devices to operate in real-time. These two studies also demonstrate that abundant semantic and spatial information is important to obtain sharp images and improve the accuracy of depth estimation. Jiang *et al.* [35] predicted the ego-motion through optical flow, which permitted large, open sources, such as YouTube videos, to be leveraged without labels. This study thus demonstrates the application of unsupervised learning to predict depth from raw videos. The pre-trained model

with semantic segmentation tasks is also applied for depth estimation.

The aforementioned studies use very expensive DCNN architecture to generate spatial and semantic feature maps for estimating the depth with high accuracy. Our design obtains these features with a lightweight DCNN architecture and successfully estimates the depth map with sharp edges.

B. LIGHTWEIGHT MONOCULAR DEPTH ESTIMATION NETWORKS

The proposed model must be optimized to carry out a real-time estimation of the depth in embedded systems and make MDE viable for industrial applications with limited computational resources. Wofk *et al.* [13] proposed a lightweight architecture that is based on the Jetson TX2 board in real-time. The total number of parameters used by the network is equal to 1.34 M after pruning. Elkerdawy *et al.* [36] reported that a lightweight monocular depth model could be developed from a complex pre-trained model by using pruning methods. Their baseline model was trained according to Monodepth2 [5]. Although such network architectures have few parameters, their performances are also relatively poor. Lyu *et al.* [7] developed a lightweight MDE by teaching the lightweight network with a high-performance network. However, this lightweight network also reported a lower accuracy than that of the original network. Despite using lightweight model architectures, the depth estimation networks proposed in [7], [13], [36] have low accuracy. This study addresses these problems by developing a lightweight DCNN with high accuracy for depth map prediction.

III. METHODOLOGY

This section describes the proposed CNN architecture for fast MDE with high resolution, followed by the training method and implementation of the model.

A. MODEL ARCHITECTURE

Zhang *et al.* [37] reported that high-level feature maps of the encoder contain more semantic features than low-level feature maps. However, the upsampling procedure followed by the bilinear interpolation in the decoder causes the model to generate a low-resolution dense output with large gradient regions. Therefore, the following strategies have been adopted to extract a high-resolution performance. The skip connection is developed, and the important features of the encoder are identified and merged with those of the decoder. Second, the decoding phase is redesigned to preserve features of the encoder in maximum detail before upsampling.

1) ENCODING PHASE

Several CNN architectures, such as SqueezeNet [38], MobileNet [14], MobiletNetV2 [39], and MobileNetV3 [40], were analyzed in this study. These networks were designed for object classification and did not require complex computations. Therefore, they can be easily deployed in embedded systems and edge devices in real-time. The depth estimation problem depends on the pixel information, so the encoder phase needs to extract semantic features and spatial information of the object as much as possible. Moreover, MobileNetV2 is a more effective feature extractor for pixel-based tasks than other networks. Thus, the lightweight MobileNetV2 [39] is selected for the design of the encoder. We also used the transfer learning technique with MobileNetV2. Since it is difficult to get a model trained from scratch to converge, our model was trained with initial weights that were pre-trained with large datasets obtained from ImageNet [41]. Therefore, the channels of the encoder are similar to those of the original MobileNetV2 model. Thus, we had 16, 24, 32, 64, and 160 channels at 1/2, 1/4, 1/8, 1/16, at 1/32 scales, respectively.

2) DECODING PHASE

The semantic features from the encoder are leveraged during the decoding phase to infer a high-quality depth map. We focus on the design of the skip connection to utilize additional semantic information from the encoding phase. The decoder is redesigned according to the following modules.

a: eSE MODULE

The representation of semantic layers in deep neural networks increases with the increasing depth of the layers. Let X^e_{i+1} denote high-level encoder feature maps at scales of $\frac{1}{2^{(n+1)}}$ from the original resolution, X^e_i represents the encoder feature maps at scales of $\frac{1}{2^n}$ from the original resolution, where n is a scale factor such that $n \in [1, 4]$. The skip connection is uniquely designed by combining high-level encoder features X^e_{i+1} and encoder features X^e_i with the SE block, which is called the eSE module, as shown in Fig. 2, to improve the accuracy of the predicted depth map. Based on the methodologies adopted by previous studies [7], [42], [43], we used the SE module for channel attention, which is essentially a detector response map of the corresponding filter. The proposed eSE module squeezes the high-level encoder features X^e_{i+1} and encoder features X^e_i according to global average pooling to generate channel information. The fully connected neural network is then used to determine and activate the important channels with relatively high weights. We also used the 1×1 convolution to fuse the feature maps. The SE block was used to estimate the important features of the encoding phase to concatenate them with the features of the skip connection and merge them with the features within the decoder.

Our proposed eSE has two advantages. The first one involved reducing the semantic and resolution gap of encoder and decoder features. The second advantage involved fusing the channels to obtain high-quality textural information of the image. Notably, our eSE module takes two continuous layers from the encoder, and focuses on the fusion the channel of these layers. Therefore, our eSE module can help FastMDE reduce the number of parameters from 4.2M to 4.1M, when compared with that using the SE module, and the accuracy of the model also be improved. The proposed model architecture

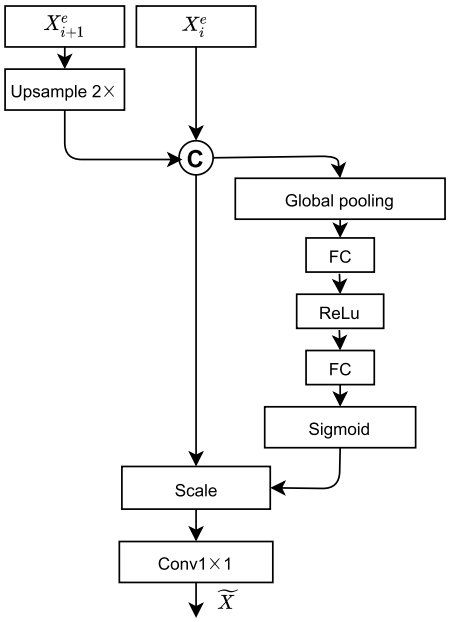


FIGURE 2. The eSE module used to extract encoder features. The module concatenates © two layers of the encoder and estimates the importance channels of the input layers by using a fully connected (FC), followed by the fusion of the feature maps by using a convolution of 1 × 1.

is shown in Fig. 3. The output of the encoder node X_i^e is equal to x_i^e, the output of the decoder node X_i^d is equal to x_i^d, and x_i denotes the output of the central node X_i. Consider a single image I_{input} that is supplied as an input to the network. The stack of the feature maps is calculated as shown below.

$$x_i^e = \begin{cases} E(I_{input}), & \text{if } i = 1, \\ E(x_{i-1}^e), & \text{if } i > 1. \end{cases} \tag{1}$$

$$x_i = eSE(x_i^e, U(x_{i+1}^e)), \quad \forall i \in [1, 4]. \tag{2}$$

$$x_i^d = \begin{cases} D(x_i, x_i^e, U(x_{i+1}^e)), & \text{if } i = 4, \\ D(x_i, x_i^e, U(x_{i+1}^d)), & \text{if } 0 < i < 4. \end{cases} \tag{3}$$

$E(\cdot)$ represents a feature extraction block, which is similar to the MobileNetV2 [39] block at every half scale of input. $eSE(\cdot)$ is the squeeze-excitation block for both features from the encoder x_i and x_{i+1}. $U(\cdot)$ represents an upsampling block that uses the nearest neighbor interpolation operation with double scaling of the input features.

b: FUSION DENSE BLOCK (FDENSE)

The deep convolutional layers are concatenated with the previous subsequent layers channel-wise in the dense block. Thus, the layer $l^{(i)}$ contains the feature maps of the previous layers (i.e., $l^0, l^1, \ldots, l^{(i-1)}$). Therefore, each layer in the dense block gains additional feature maps due to the ''collective knowledge'' gained by the previous layers. The dense block operation is not used to summate the output feature maps and incoming feature maps; instead, they are concatenated. The equation is then rewritten as follows.

$$x_l = F_l([x_0, x_1, \ldots, x_{l-1}]). \tag{4}$$

Herein, $l = 4$ layers with a growth rate of $k = 32$ for each layer. Dense block can alleviate the vanishing-gradient problem, strengthen feature propagation, and encourage feature reuse [10]. Since the feature maps are concatenated, the output channel dimension of the dense block increases k times for every layer. Therefore, the output channel of a dense block with an input channel of k_0 can be calculated by $k_l = k_0 + k \times (l - 1)$. The number of parameters is further reduced by redesigning the dense block through the addition of the convolutional with the 1×1 kernel to fuse the channels with the output channel of the dense block remain as input channel k_0. The fusion channels are used to obtain high-quality features and also to reduce the number of parameters of the network. We used the fDense block on the basis of the techniques adopted by [44] and [45]. This allows the higher resolution features learned previously in the network to be used before upsampling, as shown in Fig. 3.

c: dSE MODULE

The SE block is also used in the decoding phase to improve the accuracy and is referred to as dSE to simplify the process. The input channel of the disparity convolution block needs to be re-weighted before it predicts the depth map. Thus, the dSE is also applied before the disparity convolution block. [42] demonstrated that the SE blocks provide significant performance improvements in existing models, such as ResNet [9]. Since the SE module uses an inexpensive channel-wise scaling operation, the model requires relatively few additional computational resources. However, the accuracy of the SE-ResNet-50 model is similar to that of the deep ResNet-101 network. Thus, we utilize the SE block to enhance the accuracy of the monocular depth prediction.

B. PROBLEM FORMULATION

The approach used in this study was based on a self-supervised learning technique with a sequence of input images. The main idea involved estimation of the appearance of a target image from another image's viewpoint. Therefore, a sequence of images was needed as input for the training network. Further, an additional network to estimate the camera pose from a sequence of images, along with a depth prediction network, is required, as described in [6], [28]. Thus, the model architecture includes two separate networks, which are PoseNet and DepthNet. The proposed approach calculates the loss by trying to minimize the photometric reprojection error of the image sequences and predict the depth map for the target image D_t. Let consider the sequences of images I_t and I_{t-1}. The PoseNet and DepthNet are trained to predict $T_{t\to t-1}$ and D_t, respectively. These are used to establish the $I_{t'}$ projection relationship between the sequences of the images I_t and I_{t-1}. The loss is equal to the difference between the real image I_t and the reconstructed image $I_{t'}$. The per-pixel minimum photometric loss procedure is followed to handle occlusion, as shown below.

$$L_{pe} = \min_{t'} pe\left(I_t, I_{t'}\left\langle proj(D_t, T_{t\to t-1}, K)\right\rangle\right). \tag{5}$$

pe represents the photometric reconstruction error, and K is the camera intrinsic matrix. According to [5], [6], [46] pe can

TABLE 1. Summary of our FastMDE network architecture for self-supervised monocular depth estimation. $U(\cdot)$ refers to upsampling, and © refers to the concatenated feature maps. We use the disparity convolution block at each node dSE, the input channel of this convolution block is the output channel of the dSE, which is one channel. The disparity convolution block includes a convolution of 3 × 3 with the sigmoid activation function.

	Layer Description	channels	input dimension	output dimension	note
#0	Input RGB image	3	$H \times W$	-	
Encoder Layers: MobileNetV2					
#1	Feature layer 1	3/16	$H \times W$	$H/2 \times W/2$	
#2	Feature layer 2	16/24	$H/2 \times W/2$	$H/4 \times W/4$	
#3	Feature layer 3	24/32	$H/4 \times W/4$	$H/8 \times W/8$	
#4	Feature layer 4	32/64	$H/8 \times W/8$	$H/16 \times W/16$	
#5	Feature layer 5	96/160	$H/16 \times W/16$	$H/32 \times W/32$	
Decoder Layers					
#6	eSE layer 4	256/160	$H/32 \times W/32$©$H/16 \times W/16$	$H/16 \times W/16$	$U(\#5)$©#4
#7	eSE layer 3	128/64	$H/16 \times W/16$©$H/8 \times W/8$	$H/8 \times W/8$	$U(\#4)$©#3
#8	eSE layer 2	56/32	$H/8 \times W/8$©$H/4 \times H/4$	$H/4 \times W/4$	$U(\#3)$©#2
#9	eSE layer 1	40/24	$H/4 \times W/4$©$H/2 \times H/2$	$H/2 \times W/2$	$U(\#2)$©#1
#10	fDense block 4 + Upsampling 4	160/160	$H/32 \times W/32$	$H/16 \times W/16$	Layer #5
#11	Conv3 × 3 + dSE + fDense block 4	384/160	$H/16 \times W/16$	$H/16 \times W/16$	#10©#6©#4
#12	Upsampling 3	160/160	$H/16 \times W/16$	$H/8 \times W/8$	Upsampling #11
#13	Conv3 × 3 + dSE + fDense block 3	256/64	$H/8 \times W/8$	$H/8 \times W/8$	#12©#7©#3
#14	Upsampling 2	64/64	$H/8 \times W/8$	$H/4 \times W/4$	Upsampling #13
#15	Conv3 × 3 + dSE + fDense block 2	120/32	$H/4 \times W/4$	$H/4 \times W/4$	#14©#8©#2
#16	Upsampling 1	32/32	$H/4 \times W/4$	$H/2 \times W/2$	Upsampling #15
#17	fDense block + Upsampling + Conv3 × 3 + dSE 1	80/24	$H/2 \times W/2$	$H \times W$	#16©#9©#1

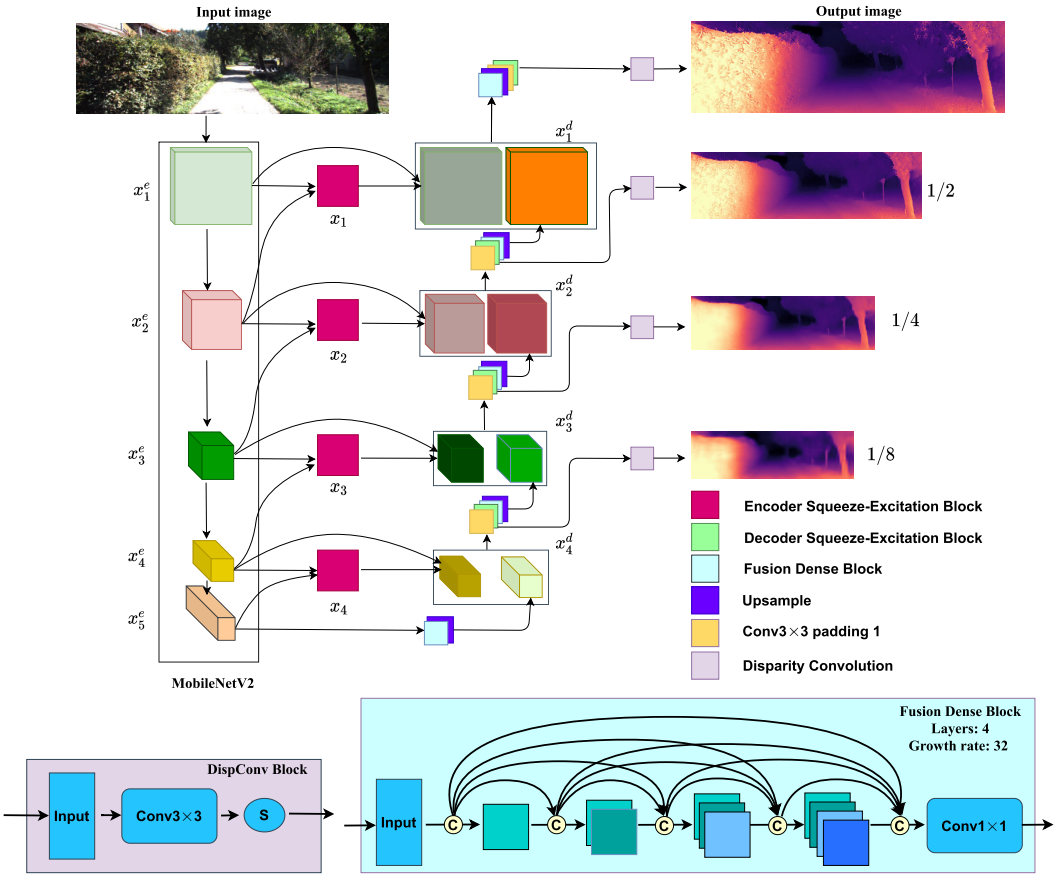


FIGURE 3. The FastDME model used for monocular depth estimation. At every half scale of input has six main components, namely eSE block, dSE block, fusion dense block (fDense), upsample block, convolution 3 × 3 block, and disparity convolutional block. The architecture can be expressed as equations (1), (2), (3), where x_i^e is the output of encoder node X_i^e, x_i^d is the output of encoder node X_i^d, and x_i is the output of the central node X_i.

be calculated by using the structural similarity index measure (*SSIM*) [47], with over a 3 × 3 pixel window and addition $L1$ regulation:

$$pe(I_t, I_{t'}) = \frac{\alpha}{2}(1 - SSIM\,(I_t, I_{t'})) + (1-\alpha)\parallel I_t - I_{t'} \parallel_1 . \tag{6}$$

The disparities between a texture-less edge and the low gradient regions of the image are difficult to detect. This was overcome by using the term edge-aware smoothness [6].

$$L_{smooth} = \left|\frac{\partial D_t}{\partial x}\right| \times e^{-\left|\frac{\partial I_t}{\partial x}\right|} + \left|\frac{\partial D_t}{\partial y}\right| \times e^{-\left|\frac{\partial I_t}{\partial y}\right|}, \tag{7}$$

where I_t denotes the target image and D_t represents the depth map estimation from the depth network. The final training loss function is a combination of the photometric reprojection error and smooth loss, as shown below.

$$L_{total} = \frac{1}{s}\sum_{i=1}^{s}\left(L_{pe}^{i} + \lambda \times L_{smooth}^{i}\right), \tag{8}$$

where $s = 4$ represents the number of scales of the decoding phase, and $\lambda = 10^{-3}$ is the weight of the edge-aware smoothness error term.

C. IMPLEMENTATION DETAILS

This section contains additional details on network architectures, training specifications, and the datasets that were used to train the FastMDE model.

1) PoseNet

The architecture of the PoseNet model is described in [5]. It is built on ResNet-18. Since the network only has two input frames, the channel at the first-level convolution changes from 3 to 6. The PoseNet network can estimate the six degrees of freedom (DoF) of the camera pose relative to a scene. The first three dimensions represent the translation vectors, and the last three dimensions represent the Euler angles.

2) DepthNet

The DepthNet model uses the standard MobileNetV2 [39] as its encoder. The details of the architecture proposed in this study are listed in Table 1. The disparity map output of each scale is used to calculate the loss during training. The output scale is similar to that of the input image for comparison purposes.

3) DATASET

The KITTI dataset [48], which is most widely used for depth evaluation, is used in this study. The data split was based on [49], and the static frame removal procedure was performed according to [28]. The model was trained, validated, and evaluated with 39810, 4424, and 697 images, respectively. All images have the same intrinsic properties. The principal point of the camera is fixed at the image center. The focal length is equal to the average of the focal lengths in the KITTI dataset. The transformation between the two stereo frames is treated as a purely horizontal translation of fixed length for stereo training.

4) IMPLEMENTATION

An open-source PyTorch library is used to train the models in this study. The detail hyper parameter is set at 20 epochs with a learning rate of 10^{-4} for the first 15 epochs, followed by a reduction to 10^{-5} for the remaining 5 epochs. The Adam optimizer [50] is used for training with exponential decay rates of $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The batch size is set to 8. Input images with a resolution of 640 × 192 are trained on a single GTX 2080 GPU. The training process required 24 hours for completion. However, due to the limited memory of the GPU, images with a resolution of 1024×320 are trained on three GPUs for 10 hours. Data parallelism techniques are used to decompose the datasets into subsets. These subsets are used in batches on different GPUs through the same model.

5) TensorRT

This is built on the open-source TensorRT engine, which was developed by Nvidia to accelerate the deep learning performance of NVIDIA GPUs. It can be easily deployed on Linux embedded systems that support TensorRT. The open neural network Exchange (ONNX) [51] is used to convert the PyTorch model into the TensorRT engine. The inference time of the TensorRT engine is approximately 30 milliseconds. This study also makes it possible for users to utilize TensorRT to easily develop applications.

IV. RESULTS

A. EVALUATION KITTI DATASET

An analysis of the MDE performance based on the KITTI dataset was performed by using the evaluation metrics described in [21]. These error metrics are defined as:

- Absolute relative difference (abs rel): $\frac{1}{n}\sum_p^n \frac{|y_p - \hat{y_p}|}{y}$;
- Squared relative difference (sq rel): $\frac{1}{n}\sum_p^n \frac{(y_p - \hat{y_p})^2}{y}$;
- RMSE: $\sqrt{\frac{1}{n}\sum_p^n \left(y_p - \hat{y_p}\right)^2}$;
- RMSE log: $\sqrt{\frac{1}{n}\sum_p^n \left(\log y_p - \log \hat{y_p}\right)^2}$;
- Threshold accuracy (δ_i): % of y_p s.t. $max(\frac{y_p}{\hat{y_p}}, \frac{\hat{y_p}}{y_p}) = \delta < threshold$ for $threshold = 1.25, 1.25^2, 1.25^3$;

where y_p is a pixel of ground trust depth image y, $\hat{y_p}$ is a pixel of predicted depth image, n is the total number of pixels of depth image.

The results of different variants of the model were compared. The variants were trained according to different self-supervision techniques, namely monocular video only (M) and monocular and stereo (MS). The results obtained from our models are compared with those of state-of-art models and other unsupervised learning methods. The evaluation results are shown in Table 2. The results demonstrate that the most lightweight model is the PyD-Net [52] model with 1.9 M parameters. Similarly, the second-most lightweight model is the Lite-HR-Depth model [7]. However,

TABLE 2. Comparison between the results of state-of-art techniques and the proposed technique, based on self-supervised learning methods and the KITTI dataset. Relatively low values of metric evaluation indices, such as absolute relative difference (Abs Rel), squared relative difference (Sq Rel), linear root-mean-square error (RMSE), log root-mean-square error(RMSE log), are desirable. However, accuracy evaluating indices, such as threshold $\delta < 1.25$, $\delta < 1.25^2$, $\delta < 1.25^3$, must be as high as possible.

Method	Supervision	Resolution	Abs Rel	Sq Rel	RMSE	RMSE log	δ<1.25	δ<1.25^2	δ<1.25^3	Parameters(M)
Zhou, Tinghui et al. [28]	M	416×128	0.208	1.768	6.856	0.283	0.678	0.885	0.957	34.2[6]
PyD-Net [52]	M	512×256	0.182	1.481	6.501	0.267	0.725	0.906	0.963	1.9[1]
Lite-HR-Depth [7]	M	640×192	0.116	0.845	4.841	0.190	0.866	0.957	0.982	3.1[2]
Monodepth2 [5]	M	640×192	0.115	0.903	4.863	0.193	0.877	0.959	0.981	14.84[5]
PackNet-SfM [34]	M	640×192	0.111	**0.785**	**4.601**	0.189	0.878	0.960	0.982	128.29[7]
HR-Depth [7]	M	640×192	0.109	0.792	4.632	**0.185**	**0.884**	**0.962**	**0.983**	14.62[4]
FastDME(Our)	M	640×192	**0.109**	0.789	4.829	0.192	0.878	0.959	0.981	4.1[3]
Monodepth2 [5]	MS	640×192	0.106	0.818	4.750	0.196	0.874	0.957	0.979	14.84[3]
HR-Depth [7]	MS	640×192	0.107	**0.785**	**4.612**	**0.185**	**0.887**	**0.962**	**0.982**	14.62[2]
FastDME(Our)	MS	640×192	**0.107**	0.816	4.718	0.194	0.874	0.959	0.980	4.1[1]
Monodepth2 [6]	M	1024×320	0.115	0.882	4.701	0.190	0.879	0.961	0.982	14.84[3]
HR-Depth [7]	M	1024×320	0.106	**0.755**	**4.472**	**0.181**	0.892	**0.966**	**0.984**	14.62[2]
FastDME(Our)	M	1024×320	**0.104**	0.768	4.538	0.182	**0.894**	0.964	0.983	4.1[1]
Monodepth2 [6]	MS	1024×320	0.106	0.818	4.750	0.196	0.874	0.957	0.979	14.84[3]
HR-Depth [7]	MS	1024×320	0.101	**0.716**	**4.395**	**0.179**	**0.899**	**0.966**	**0.983**	14.62[2]
FastDME(Our)	MS	1024×320	**0.101**	0.740	4.512	0.186	0.885	0.962	0.982	4.1[1]

these networks have poor accuracies. The accuracy of the proposed network architecture is similar to that of the state-of-art HR-depth model and outperforms a recent model [34] with 128.29 M parameters. In addition, our network uses 3.5 times fewer parameters than the HR-depth model (Number of parameters used by the HR-depth model and the current model are 14.62 M and 4.1 M, respectively).

Fig. 4 shows a qualitative comparison between the MDE performances of the proposed FastMDE model and the other state-of-art methods. The current model can predict edges with higher quality and more sharpness than those predicted by the Monodepth2 model [5]. The current model's performance is comparable to that of the recently developed state-of-art HR-depth [7] model architecture while utilizing fewer parameters than the latter. Although semantic and spatial feature maps cannot be captured easily through small DCNN architectures, these two feature maps are well-captured well with the proposed model.

B. EVALUATION Make3D DATASET

Table 3 provides the reliable estimation results of our proposed model on the Make3D dataset [53]. We evaluate our model on the 134 images (collected using 3D scanner) of Make3D with a center crop of 2×1 ratio. Therefore, we crop the original images of the Make3D dataset to 1704×852 and then resize them to 640×192, and finally pass them through the network. From the table, our proposed FastMDE model outperforms all the compared methods that use monocular supervision. such as Monodepth and Monodepth2. Moreover, our estimated depth results are shown in Fig. 5. It is shown that our method reliably shows depth maps with clear image in complicated boundaries of various objects.

C. EVALUATION ON REAL IMAGES

To evaluate that our model architecture, FastMDE, can achieve good stability and generalization, we test our model with images captured by a hand-held phone camera. The size of an original image captured by a phone camera is 2048×1536. We crop the captured images to 2048×640, then resize it to 1024×320, and make no image enhancement. Our estimated depth results are shown in Fig. 6. The results show that our model achieves a strong generalization on the real scenes captured by mobile phones.

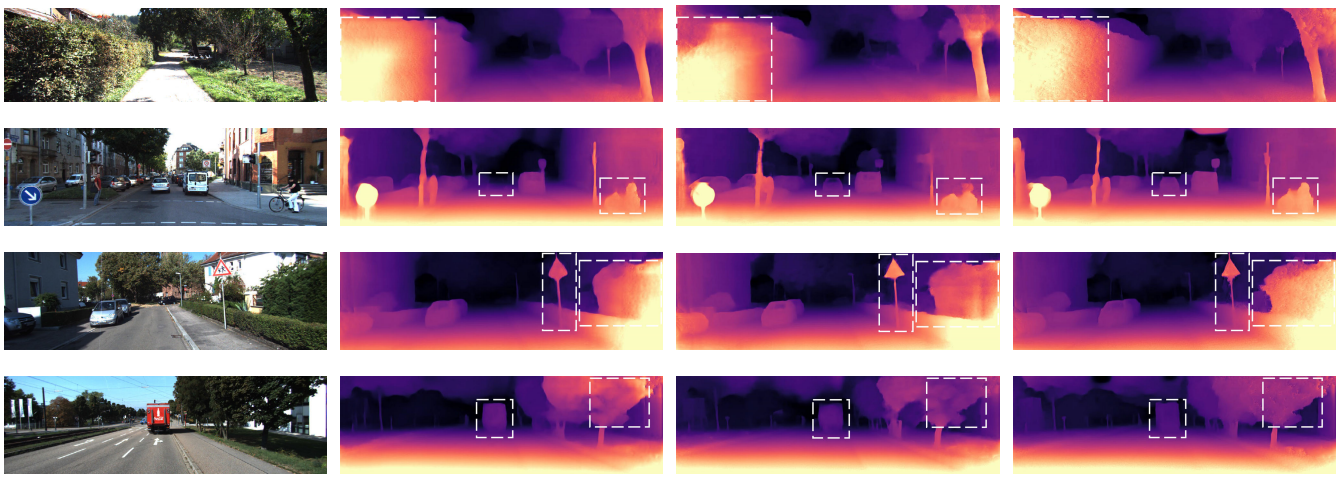
TABLE 3. Quantitative results of depth estimation on Make3D dataset. All methods were trained on KITTI using monocular supervision then evaluate on Make3D dataset.

Method	Abs Rel	Sq Rel	RMSE	RMSE log
Monodepth [6]	0.544	10.94	11.760	0.193
Zhou [28]	0.383	5.321	10.470	0.487
DDVO [54]	0.387	4.720	8.090	0.204
Monodepth2 [5]	0.322	3.589	7.417	0.163
FastMDE (Our)	**0.309**	**2.752**	**6.890**	**0.161**

V. ABLATION STUDIES

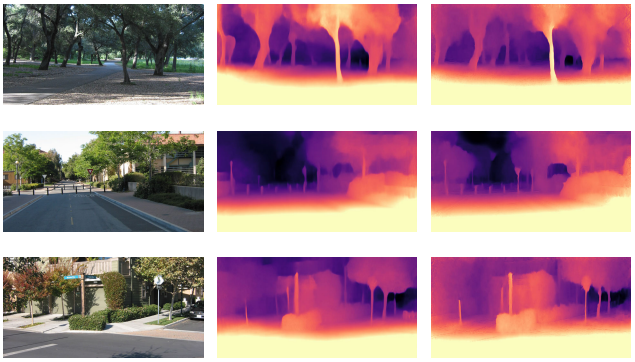
A. ADVANCE OF SKIP CONNECTION WITH eSE BLOCK AND dSE BLOCK

The eSE module and skip connection are redesigned to capture the maximum number of important features from the encoding. The selected features belong to two output layers X_i^e, X_{i+1}^e from the encoder part. The eSE re-weighs the important channels and then alleviates the channels by using the standard convolution 1×1 kernel, thus the eSE module helps not only reduce the number of parameters but also improve the performance of the network, which is even better than that of the SE block as shown in the Table 4. Moreover, it is difficult to fuse the features with large semantic gap between the encoder and decode. Therefore, the eSE skip connection is used to generate more intermediate semantic features from encoder to effectively reduce the semantic gap, as shown in the Fig. V-A. With more semantic information, we can significantly improve the depth map estimation. The



(a) Input image (b) Monodepth2 (**14.84M**) (c) HR-depth (**14.62M**) (d) FastMDE (**4.1M**)

FIGURE 4. Comparisons between the results of the proposed technique and state-of-art methods based on the KITTI Eigen Split. The HR-depth method used an input image resolution of 1280×384, whereas the Monodepth2 and the proposed model used an input image resolution of 1024×320.



(a) Input image (b) Monodepth2 (c) FastMDE

FIGURE 5. Comparisons between the results of the proposed technique and Monodepth2 method based on the Make3D dataset. The input resolution uses 640 × 192 for both of models.

TABLE 4. eSE and SE studies. Results for different model variants with eSE module and SE module. We trained two models with fDense block and the hyper-parameters were set the same of two models.

Method	Abs Rel	Sq Rel	δ<1.25	Para (M)
SE	0.113	0.828	0.879	4.2
eSE	0.109	0.789	0.878	4.1

dSE block is applied in the decoder section to re-weigh the important features before generating the depth prediction map. The influence of the eSE + dSE blocks with the skip connection is shown in Table 6. The model accuracy improved from 0.114 to 0.109 times the absolute relative difference evaluation metrics by using the eSE + dSE blocks.

B. ADVANCE OF fDense BLOCK CONNECTION

The redesigned dense block connection allows the model to capture additional semantic information. The fDense block connection reuses the feature maps of the previous layers. The features learned in the previous layers are passed forward and removed. The need to learn redundant features encourages the learning of a different set of features. This permits the application of the high-resolution features that were learned earlier in the network in the upsampling process. Therefore, it improves the accuracy of the model and helps predict sharp edges. The model can produce high-resolution feature maps, as shown in Fig. 7. The model with a fDense connection, as shown in the right panel, can easily capture an image's textural information, despite it having a low resolution. To evaluate our fDense block, we apply the original dense block to eSE + dSE block and compare it with our fDense block. As shown in the Table 5, our fDense module can reduce the number of parameters from 4.8 M to 4.1 M. Interestingly, the accuracies (i.e., abs rel and sq rel) achieved by fDense are higher than those of the original dense block. The results of the ablation studies are shown in Table 6. Although the application of the fDense connection in the decoder increases the number of parameters from 3.3 M to 4.1 M, the fDense connection significantly improves the accuracy from 0.114 to 0.109 times the absolute relative difference evaluation metric.

TABLE 5. fDense and Dense studies. Results for different model variants with fDense module and Dense module. We trained two models with eSE block and the hyper-parameters were set the same of two models.

Method	Abs Rel	Sq Rel	δ<1.25	Para (M)
Dense	0.111	0.790	0.878	4.8
fDense	0.109	0.789	0.878	4.1

TABLE 6. Ablation studies. Results of different model variants obtained through monocular training on KITTI datasets at a low-resolution input of (640 × 192) on the Eigen split. Monodepth2 is used as the baseline model in this comparison. The influence of the fDense connection in the decoding phase without the encoder SE and decoder SE modules and the influence of the encoder SE and decoder SE modules without a fDense connection (eSE + dSE) are compared.

Method	Abs Rel	Sq Rel	δ<1.25	Para(M)
Baseline	0.115	0.903	0.877	14.84
fDense	0.114	0.781	0.871	4.2
eSE+dSE	0.114	0.849	0.875	3.3
eSE+fDense+dSE	**0.109**	**0.789**	**0.878**	4.1

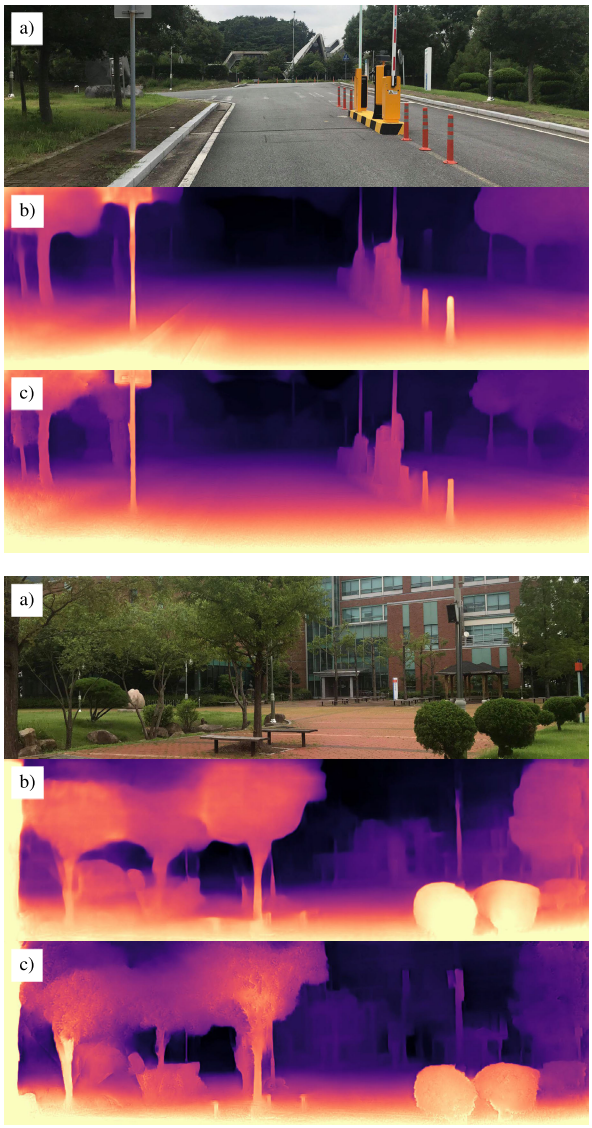


FIGURE 6. Our model achieves good qualitative results on real images. a) the input image taken by iPhone 7 plus, b) the depth estimation of Monodepth2, c) the depth estimation of FastMDE (images were captured at Pusan National University, Yangsan campus, Republic of Korea).

Despite the application of a single module in a system based on two effective modules, the proposed model outperforms the baseline Monodepth2 while using fewer parameters than the latter, as shown in Table 6.

C. TensorRT

TensorRT is a software development kit (SDK) developed by Nvidia that is used to formulate a high-performance deep learning model. We used the TensorRT SDK to build the model engine in this study. It can be easily deployed in Nvidia Linux development kits (tested with Nvidia Xavier NX). The model is converted from the Pytorch model to ONNX and then built with the TensorRT engine. Since the TensorRT engine does not support reflection padding from the ONNX model, we changed it to zero padding. The weights from the reflection pad model are used, followed by fine tuning at a learning rate of 10^{-5} for five epochs to obtain the accurate weights of the zeros padding model. The result is reduced due to the influence of zero padding, as shown in Table 7.

To show our model architecture, FastMDE, can achieve higher speed than state-of-arts models, we build the TensorRT model engine for various depth estimation algorithms. Then, we compare the speed of various model architectures by

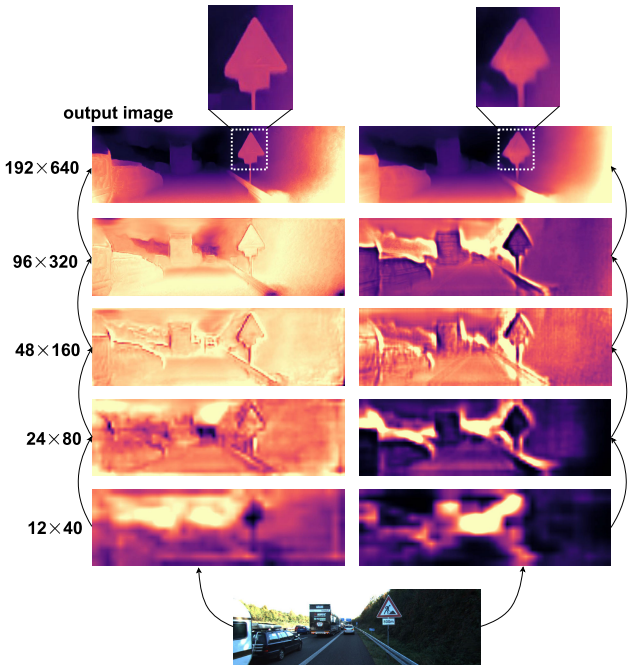


FIGURE 7. Illustration of the advance of the fDense block connection for monocular depth estimation. The left panel depicts the first channel map with the fDense connection, and the right panel depicts the first channel map without the fDense connection. The fDense connection ensures that the sharp edges are detected with high resolution. Further, the textural information of the image is well captured, even at low resolutions 12 × 40.

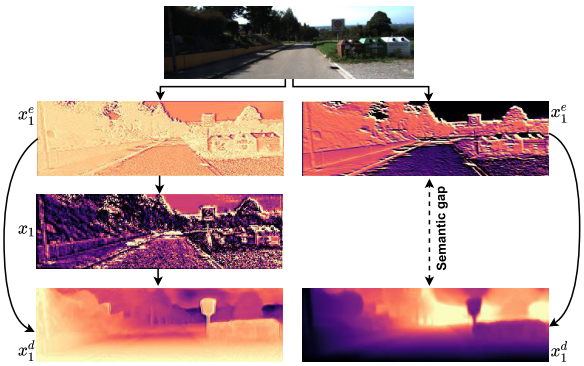


FIGURE 8. Illustration of the advance of the skip connection with eSE block for monocular depth estimation. Left side depicts feature maps in FastMDE and right side depicts feature maps in Monodepth2.

TABLE 7. TensorRT studies. Results for different model variants with reflection padding and zero padding. The reflection padding model was based on the ONNX runtime, while the zeros padding model used TensorRT engine.

Method	Abs Rel	Sq Rel	δ<1.25	Run time (s)
RF pad	0.109	0.789	0.878	15
Zeros pad (TensorRT)	0.112	0.808	0.878	0.03

TABLE 8. The runtime on an NVIDIA Jetson Xavier NX GPU for various depth estimation algorithms. Runtimes are measured by using TensorRT engine.

Method	Engine size (MB)	Runtime (s)
HR-depth	30.3	0.075
Monodepth2	29.9	0.072
FastMDE	13.9	0.03

using CUDA event function in Pytorch package, as shown in Table 8. The results show that the speed of our model outperform all the other methods.

VI. CONCLUSION

A lightweight convolutional network architecture named FastMDE was developed in this study by applying a novel skip connection with features of the eSE module, dSE module and the fDense block. The network utilized self-supervised learning for fast MDE at high-resolution. The eSE block was redesigned according to an analysis of the properties of the encoder and decoder sections to enhance the quality of depth map prediction, assist the model in identifying the important features of the encoder and decoder, and utilize the fDense connection to capture those features in great detail. Our eSE and fDense modules can totally reduce the parameters of FastMDE from 5.4M to 4.1M, when compared to the original SE and dense modules. The number of trainable parameters required for the proposed network architecture is 3.5 times lesser than that of the HR-depth model, despite maintaining a similar accuracy. We have also provided the TensorRT engine to easily deploy our model in Nvidia Linux embedded boards that support the TensorRT engine. Notably, our model can run in real-time $\sim$ 33 fps with an input image resolution of 640 $\times$ 192. Deploying the proposed model into the autonomous drone is our goal for future work.

REFERENCES

[1] H. Gao, B. Cheng, J. Wang, K. Li, J. Zhao, and D. Li, "Object classification using CNN-based fusion of vision and LIDAR in autonomous vehicle environment," *IEEE Trans. Ind. Informat.*, vol. 14, no. 9, pp. 4224–4231, Sep. 2018, doi: 10.1109/TII.2018.2822828.

[2] J. Zhang, M. S. Ramanagopal, R. Vasudevan, and M. Johnson-Roberson, "LiStereo: Generate dense depth maps from LIDAR and stereo imagery," 2019, *arXiv:1905.02744*.

[3] H.-W. Chae, J.-H. Choi, and J.-B. Song, "Robust and autonomous stereo visual-inertial navigation for non-holonomic mobile robots," *IEEE Trans. Veh. Technol.*, vol. 69, no. 9, pp. 9613–9623, Sep. 2020, doi: 10.1109/TVT.2020.3004163.

[4] Y. Dang, P. Chen, R. Liang, C. Huang, Y. Tang, T. Yu, X. Yang, and K.-T. Cheng, "Real-time semantic plane reconstruction on a monocular drone using sparse fusion," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7383–7391, Aug. 2019, doi: 10.1109/TVT.2019.2923676.

[5] C. Godard, O. M. Aodha, M. Firman, and G. Brostow, "Digging into self-supervised monocular depth estimation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3828–3838.

[6] C. Godard, O. M. Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, Jul. 2017, pp. 6602–6611, doi: 10.1109/CVPR.2017.699.

[7] X. Lyu, L. Liu, M. Wang, X. Kong, L. Liu, Y. Liu, X. Chen, and Y. Yuan, "HR-Depth: High resolution self-supervised monocular depth estimation," in *Proc. AAAI Conf. Artif. Intell.*, May 2021, vol. 35, no. 3, pp. 2294–2301. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/16329

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, 2015, pp. 1–14.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[10] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.

[11] I. Alhashim and P. Wonka, "High quality monocular depth estimation via transfer learning," 2018, *arXiv:1812.11941*.

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

[13] D. Wofk, F. Ma, T.-J. Yang, S. Karaman, and V. Sze, "FastDepth: Fast monocular depth estimation on embedded systems," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 6101–6108, doi: 10.1109/ICRA.2019.8794182.

[14] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[15] A. Shaw, D. Hunter, F. Landola, and S. Sidhu, "SqueezeNAS: Fast neural architecture search for faster semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*. Los Alamitos, CA, USA: IEEE Computer Society, Oct. 2019, pp. 2014–2024, doi: 10.1109/ICCVW.2019.00251.

[16] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 4780–4789. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/4405

[17] R. Miikkulainen, J. Liang, E. Meyerson, A. Rawal, D. Fink, O. Francon, B. Raju, H. Shahrzad, A. Navruzyan, N. Duffy, and B. Hodjat, "Evolving deep neural networks," in *Artificial Intelligence in the Age of Neural Networks and Brain Computing*. Amsterdam, The Netherlands: Elsevier, 2019, pp. 293–312.

[18] B. Baker, O. Gupta, N. Naik, and R. Raskar, "Designing neural network architectures using reinforcement learning," 2016, *arXiv:1611.02167*.

[19] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8697–8710, doi: 10.1109/CVPR.2018.00907.

[20] Y. J. Lee and J. Yoon, "Nonlinear image upsampling method based on radial basis function interpolation," *IEEE Trans. Image Process.*, vol. 19, no. 10, pp. 2682–2692, Oct. 2010, doi: 10.1109/TIP.2010.2050108.

[21] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 2366–2374.

[22] Z. Zhang, C. Xu, J. Yang, J. Gao, and Z. Cui, "Progressive hard-mining network for monocular depth estimation," *IEEE Trans. Image Process.*, vol. 27, no. 8, pp. 3691–3702, Aug. 2018, doi: 10.1109/TIP.2018.2821979.

[23] S. F. Bhat, I. Alhashim, and P. Wonka, "AdaBins: Depth estimation using adaptive bins," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 4009–4018.

[24] C. Zhou, H. Zhang, X. Shen, and J. Jia, "Unsupervised learning of stereo matching," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1576–1584, doi: 10.1109/ICCV.2017.174.

[25] K. A. Skinner, J. Zhang, E. A. Olson, and M. Johnson-Roberson, "UWStereoNet: Unsupervised learning for depth estimation and color correction of underwater stereo imagery," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 7947–7954, doi: 10.1109/ICRA.2019.8794272.

[26] N. Smolyanskiy, A. Kamenev, and S. Birchfield, "On the importance of stereo for accurate depth estimation: An efficient semi-supervised deep neural network approach," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 1007–1015, doi: 10.1109/CVPRW.2018.00147.

[27] Y. Zhang, S. Xu, B. Wu, J. Shi, W. Meng, and X. Zhang, "Unsupervised multi-view constrained convolutional network for accurate depth estimation," *IEEE Trans. Image Process.*, vol. 29, pp. 7019–7031, 2020, doi: 10.1109/TIP.2020.2997247.

[28] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6612–6619, doi: 10.1109/CVPR.2017.700.

[29] Z. Yang, P. Wang, W. Xu, L. Zhao, and R. Nevatia, "Unsupervised learning of geometry from videos with edge-aware depth-normal consistency," in *Proc. 32nd AAAI Conf. Artif. Intell.*, Apr. 2018, vol. 32, no. 1, pp. 1–8.
[30] J.-W. Bian, Z. Li, N. Wang, H. Zhan, C. Shen, M.-M. Cheng, and I. Reid, "Unsupervised scale-consistent depth and ego-motion learning from monocular video," 2019, *arXiv:1908.10553*.
[31] A. Ranjan, V. Jampani, L. Balles, K. Kim, D. Sun, J. Wulff, and M. J. Black, "Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, Jun. 2019, pp. 12232–12241, doi: 10.1109/CVPR.2019.01252.
[32] V. Guizilini, R. Hou, J. Li, R. Ambrus, and A. Gaidon, "Semantically-guided representation learning for self-supervised monocular depth," 2020, *arXiv:2002.12319*.
[33] S. Vijayanarasimhan, S. Ricco, C. Schmid, R. Sukthankar, and K. Fragkiadaki, "SfM-Net: Learning of structure and motion from video," 2017, *arXiv:1704.07804*.
[34] V. Guizilini, R. Ambrus, S. Pillai, A. Raventos, and A. Gaidon, "3D packing for self-supervised monocular depth estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, Jun. 2020, pp. 2482–2491.
[35] H. Jiang, G. Larsson, M. M. G. Shakhnarovich, and E. Learned-Miller, "Self-supervised relative depth learning for urban scene understanding," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 19–35.
[36] S. Elkerdawy, H. Zhang, and N. Ray, "Lightweight monocular depth estimation model by joint end-to-end filter pruning," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Taipei, Taiwan, Sep. 2019, pp. 4290–4294, doi: 10.1109/ICIP.2019.8803544.
[37] Z. Zhenli, X. Zhang, C. Peng, X. Xue, and J. Sun, "ExFuse: Enhancing feature fusion for semantic segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 269–284.
[38] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," 2016, *arXiv:1602.07360*.
[39] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.
[40] A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1314–1324.
[41] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and Li Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[42] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.
[43] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T.-S. Chua, "SCA-CNN: Spatial and channel-wise attention in convolutional networks for image captioning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6298–6306, doi: 10.1109/CVPR.2017.667.
[44] S. Guan, A. A. Khan, S. Sikdar, and P. V. Chitnis, "Fully dense UNet for 2-D sparse photoacoustic tomography artifact removal," *IEEE J. Biomed. Health Informat.*, vol. 24, no. 2, pp. 568–576, Feb. 2020.
[45] S. Cai, Y. Tian, H. Lui, H. Zeng, Y. Wu, and G. Chen, "Dense-UNet: A novel multiphoton *in vivo* cellular image segmentation model based on a convolutional neural network," *Quant. Imag. Med. Surg.*, vol. 10, no. 6, p. 1275, 2020.
[46] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," *IEEE Trans. Comput. Imag.*, vol. 3, no. 1, pp. 47–57, Mar. 2017, doi: 10.1109/TCI.2016.2644865.
[47] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[48] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361, doi: 10.1109/CVPR.2012.6248074.
[49] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," *CoRR*, vol. abs/1411.4734, pp. 1–9, Nov. 2014.
[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[51] J. Bai, F. Lu, and K. Zhang. (2019). *ONNX: Open Neural Network Exchange*. [Online]. Available: https://github.com/onnx/onnx
[52] M. Poggi, F. Aleotti, F. Tosi, and S. Mattoccia, "Towards real-time unsupervised monocular depth estimation on CPU," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 5848–5854, doi: 10.1109/IROS.2018.8593814.
[53] A. Saxena, M. Sun, and A. Y. Ng, "Make3D: Learning 3D scene structure from a single still image," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 5, pp. 824–840, May 2009, doi: 10.1109/TPAMI.2008.132.
[54] C. Wang, J. M. Buenaposada, R. Zhu, and S. Lucey, "Learning depth from monocular videos using direct methods," 2017, *arXiv:1712.00175*.

THIEN-THANH DAO received the B.S. degree in mechatronic engineering from the Hanoi University of Science and Technology, Vietnam, in 2016, and the M.S. degree in mechanical engineering from Inje University, South Korea, in 2019. He is currently pursuing the Ph.D. degree in computer engineering with Pusan National University, South Korea. His research interests include mathematics, machine learning, deep learning, computer vision, and intelligent navigation systems.

QUOC-VIET PHAM (Member, IEEE) received the B.S. degree in electronics and telecommunications from the Hanoi University of Science and Technology, Vietnam, in 2013, and the M.S. and Ph.D. degrees in telecommunications engineering from Inje University, South Korea, in 2015 and 2017, respectively. He is currently with Pusan National University, South Korea. He has been granted the Korea NRF Funding for Outstanding Young Researchers for the term (2019–2023). His research interests include convex optimization, game theory, and machine learning to analyze and optimize edge computing systems, resource allocation, and future wireless networks. He received the best Ph.D. dissertation award in engineering from Inje University, in 2017. He received the Top Reviewer Award from the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, in 2020. He is an Editor of *Journal of Network and Computer Applications* (Elsevier) and a Lead Guest Editor of *IEEE Internet of Things Journal*.

WON-JOO HWANG (Senior Member, IEEE) received the B.S. and M.S. degrees in computer engineering from Pusan National University, Busan, South Korea, in 1998 and 2000, respectively, and the Ph.D. degree in information systems engineering from Osaka University, Osaka, Japan, in 2002. From 2002 to 2019, he was a Full Professor with Inje University, Gimhae, South Korea. He is currently a Full Professor with the Department of Biomedical Convergence Engineering, Pusan National University. His research interests include optimization theory, game theory, machine learning, and data science for wireless communications and networking.