IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 18, 2021, accepted August 12, 2021, date of publication August 16, 2021, date of current version August 24, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3105136

Cocktail Glass Network: Fast Depth Estimation Using Channel to Space Unrolling

JUNG-JAE YU[1,2], JONG-GOOK KO[1], AND JUNMO KIM[2], (Member, IEEE)
[1]Electronics and Telecommunications Research Institute (ETRI), Daejeon 34129, Republic of Korea
[2]Korea Advanced Institute of Science and Technology (KAIST), Daejeon 34141, Republic of Korea

Corresponding author: Junmo Kim (junmo.kim@kaist.ac.kr)

This work was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP) grant funded by Korea Government, Ministry of Science and ICT (MSIT) (No. 2018-0-00198), Object Information Extraction and Real-to-Virtual Mapping Based AR Technology.

ABSTRACT Depth-estimation from a single input image can be used in applications such as robotics and autonomous driving. Recently, depth-estimation networks with UNet encoder/decoder structures have been widely used. In these decoders, operations are repeated to gradually increase the image resolution, while decreasing the channel size. If the upsampling operation at a high magnification can be processed at once, the amount of computation in the decoder can be dramatically reduced. To achieve this, we propose a new network structure, i.e., a cocktail glass network. In this network, convolution layers in the decoder are reduced, and a novel fast upsampling method is used that is known as channel-to-space unrolling, which converts thick channel data into high-resolution data. The proposed method can be easily implemented using simple reshaping operations; therefore, it is suitable for reducing the depth-estimation network. Considering the experimental results based on the NYU V2 and KITTI datasets, we demonstrate that the proposed method reduces the amount of computation in the decoder by half, while maintaining the same level of accuracy; it can be used in both lightweight and large-model-capacity networks.

INDEX TERMS Artificial neural networks, image processing, depth estimation, monocular image.

I. INTRODUCTION

Monocular depth-estimation (MDE) is a method of estimating the depth of an input image. This is an ill-posed problem because several three-dimensional (3D) scenes can be projected onto the same two-dimensional (2D) scene. Recently, significant progress has been achieved in solving this problem, using convolutional neural networks and large-scale learning data. This study aims to reduce the amount of computation, while maintaining accuracy, in a lightweight MDE network. The supervised depth-estimation network is mostly composed of an encoder/decoder in a UNet network. First, in the encoder, a backbone network trained on a large dataset, such as ImageNet, is recommended. Typically, ResNet [1] or DenseNet [2] is used as a backbone in heavy-weight networks to achieve the best accuracy, and MobileNet [3] is primarily used in lightweight networks for mobile environments. Various other methods, such as GhostNet [4], FbNet [5], and EfficientNet [6], have recently been proposed for efficient computation and they can be used as an encoder in a UNet-styled MDE network. In contrast, there is considerable scope for improving the network structure for weight reduction in the decoder. The main function of the decoder of the depth-estimation network is to gradually reduce the channel, increase the resolution of the deep features delivered from the encoder, and finally, output a channel depth map with the same resolution as the input image. Thus, upsampling ($2\times$) layers, such as up-convolution [7], are repeatedly used in the decoder, and the resolution is generally increased by a factor of two in each layer. If the resolution is increased and the channel is reduced by a large magnification in one layer, the number of layers in the decoder can be reduced. This indicates that the amount of computation and network model size can be reduced. However, we encounter the following problems. If the channel is reduced by a high magnification and the resolution is increased sequentially, although the size of the final feature data is the same as the input data, a significant amount of information is lost, because the data size decreases at a large rate in the intermediate stage. Alternatively, if the resolution is first increased by a

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

high magnification and the channel is subsequently reduced, the data size increases in the intermediate stage, and the amount of computation and memory increases. Therefore, to avoid such problems, it is necessary to simultaneously reduce the channel and increase the resolution. Considering this task, [8] introduced sub-pixel convolution (SPC); nevertheless, it is impossible for high-magnification conversion to reduce the amount of computation, as indicated in the experimental results in Section 4.

The main contribution of this study is the proposal of a new method for reducing the amount of computation in the decoder part of a lightweight depth-estimation network, while maintaining a similar level of accuracy of the estimated depth. First, we propose a new depth-estimation network structure that is known as the cocktail glass network (CGN), having reduced convolution layers in the decoder. Second, we propose a novel method for implementing a fast upsampling block in this network structure, which is known as channel-to-space unrolling (CSR), directly converting low-resolution data with thick channels into high-resolution data with thin channels. The CSR method increases the resolution as if the thick channel data are spread in the horizontal and vertical axes separately, and the two results are fused. This indicates that the CSR outputs $C_{out} \times kH \times kW$ data from $C_{in} \times H \times W$ input data in a layer ($k > 2, C_{in} \approx k \times C_{out} < k^2 \times C_{out}$). The contributions of this study are summarized as follows.

- A new depth-estimation network structure (CGN) to reduce the number of convolution layers in the decoder part of a MDE network has been proposed.
- A new upsampling method (CSR) to generates high-resolution data with thin channels from low-resolution data with thick channels has been proposed.

II. RELATED WORK

The existing studies in supervised MDE field are summarized below. In this field of study, ground truth depth data, which are measured using various depth sensors such as LiDAR or RGB-D cameras, are used. Saxena *et al.* [9] proposed a learning-based method for estimating depth from visual cues using a Markov random field. Eigen and Fergus [10] proposed a multiscale convolutional architecture, where the initial depth was first estimated, and then, progressively refined. Their method was the first learning-based method to estimate depth without any handcrafted image features. Fu *et al.* [11] considered the regression problem of continuous depth-estimation as a decision problem for quantized depth candidates. Recently, Ranftl proposed MiDaS [12], which is a method of training the depth-estimation networks by mixing various types of training data and has demonstrated good performance.

However, this study aims to increase the computational speed, while maintaining a level of accuracy similar to that of existing technology. Therefore, we focus on increasing the computational efficiency, rather than trying to maximize the accuracy of depth-estimation. Regarding the problem of

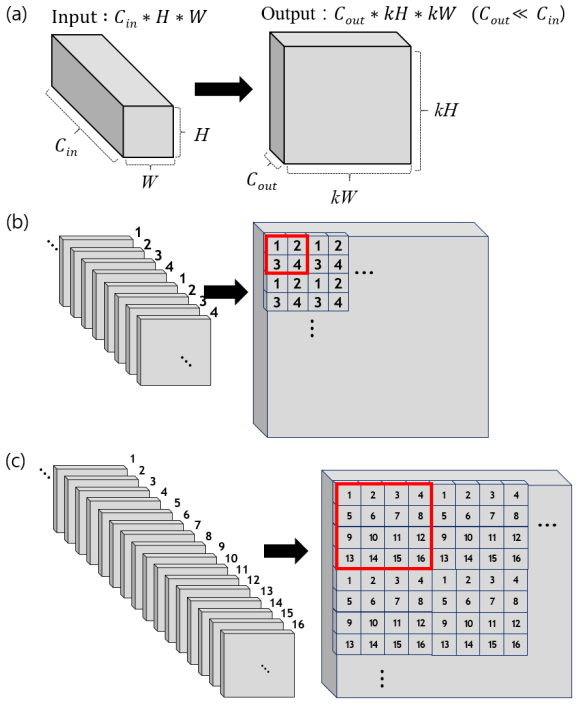

FIGURE 1. Aim of fast upsampling and a previous method, (a) aim of fast upsampling, (b) pixel-shuffling operation in SPC [13] for 2× magnification, and (c) for 4× magnification.

increasing the efficiency of the network, there have been recent studies, such as GhostNet [4], FbNet [5], and EfficientNet [6]. Although these studies do not limit the structure of the network, they do not consider the characteristics of the decoder in a UNet-style network that expands the resolution, while reducing the channel's thickness from the extracted deep features. Therefore, even if the network operation efficiency is sufficiently high in the encoder, there is still room for improvement in the process of expanding the thick channel and low-resolution deep features to the output image size in the decoder.

Wofk *et al.* [14] proposed a very fast depth-estimation network for real-time inference on an embedded system. They used a general UNet-style encoder/decoder network structure and NetAdapt [15] pruning with a compiling method specialized for the TX2 board. MobileNet V1 [3] was used as the encoder. The decoder part of their network comprised repetitively connected layers that doubled the resolution using the nearest neighbor interpolation after performing a 2D convolution comprising depthwise and pointwise convolutions [3]. The network they proposed is not superior to that of existing studies in terms of accuracy; nonetheless, it is suitable for use in a mobile environment because it exhibits a faster speed. Therefore, we used their network as a basis for our proposed method. Hereinafter, we refer to the network structure by Wofk *et al.* [14] as FastDepth.

To reduce the number of layers in the decoder of the FastDepth, we must convert the low-resolution data with a thick channel into high-resolution data with a thin channel in

a layer, without degrading the accuracy of the final output. In this study, a layer refers to the combined operations of depthwise and pointwise convolutions, normalization operation, and nonlinear activation function. We refer to this task (Fig. 1-(a)) as a fast upsampling problem, and the SPC that was proposed by Shi *et al.* [8] in the super-resolution (SR) field can be used to solve this problem. The process of SPC first expands the channel using 2D convolution, and moves each element at a certain interval in the channel axis to a fixed position in the image plane (pixel-shuffling operation, Fig. 1-(b), (c)). The SPC is similar to the CSR proposed in this study because it converts the thick channel data into high-resolution data. However, if we apply the SPC to high-magnification tasks, as shown in Fig. 1-(c), the input channel of this fast upsampling module should be $k^2 \times C_{out}$. In the deep layer of a typical UNet-style network, the channel width is approximately proportional to ratio, k, at which the resolution decreases, and is rarely proportional to k^2. Therefore, to use these types of fast upsampling, a convolution is required to increase the channel to approximately $k^2 \times C_{out}$ before the channel-to-space-mapping operation. This increases the amount of computation and memory. In contrast, the CSR requires a convolution whose output channel size is $k \times C_{out}$ before the unrolling operation; hence, the computational amount is reduced.

Recently, Guizilini *et al.* [13] proposed 3D packing/unpacking methods that used an additional dimension in the feature data and 3D convolution for data conversion between the channel and space (resolution). Unpacking is similar to the SPC [8] in terms of relocations between channels and image planes; nevertheless, it generates high-resolution information using 3D convolution. Contrary to the SPC, unpacking can be used for high magnification (channel reduction and resolution increase) in one operation. However, as confirmed by the experimental results in Section 4, the proposed CSR is superior to unpacking in terms of accuracy, with fewer computations.

III. METHODOLOGY

In this section, we first describe the structure of the proposed depth-estimation network using a fast upsampling block in the decoder. Subsequently, we explain the details of the proposed CSR for implementing a fast upsampling block.

A. NETWORK STRUCTURE

Wofk *et al.* [14] used a UNet-style encoder/decoder structure with skip-connections to implement a depth-estimation network. Fig. 2-(a) shows an example of this network structure. The yellow and blue boxes represent the convolution layers in the encoder and decoder, respectively. This structure is generally used in depth-estimation networks. Because its shape resembles an hourglass, these types of networks are known as hourglass networks, and are mainly used in the field of human pose estimation [16]. Fig. 2-(b) shows the network structure wherein the convolution layers in the decoder are

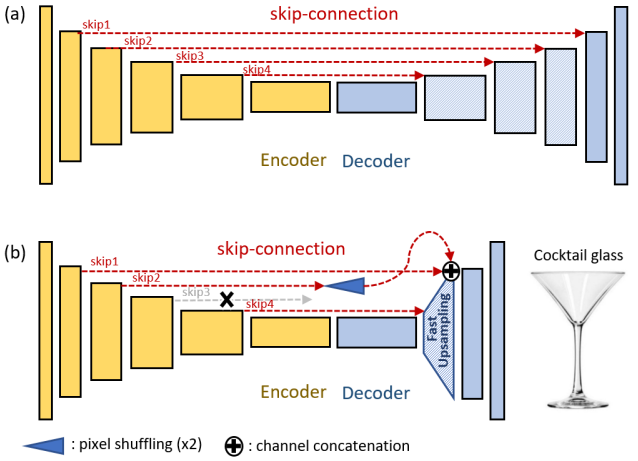

FIGURE 2. Depth-estimation network with/without a fast upsampling block: (a) Previous depth-estimation network with a UNet encoder/decoder structure; (b) proposed network structure with a fast upsampling block to reduce the number of convolution layers in the decoder.

reduced, because the fast upsampling block replaces several convolution layers. The fast upsampling block must simultaneously perform a high-magnification increase in resolution and channel reduction, and the accuracy of the final output depth-map-estimation should not be reduced. As a method for implementing a fast upsampling block that satisfies these conditions, we propose the CSR method. Its specific process will be described in the Section 3-B. The network that is shown in Fig. 2-(b) is characterized by the connection of the convolution layers in the decoder being considerably shorter than those in the encoder, and the resolution of the data increases rapidly in the decoder. In such a network structure, it is necessary to additionally consider how to utilize the skip-connection (Skips 2 and 3 in Fig. 2-(a)) that is connected to the intermediate layers in the existing network structure. Considering Skip 2, which is close to the output resolution of the fast upsampling module, the channel width is reduced to one quarter, and the resolution is doubled via the pixel-shuffling operation, as demonstrated in Fig. 1-(b). Regarding Skip 3, a similar method as in Fig. 1-(c) can be applied; however, we discarded it to reduce the amount of computation. Modified Skip 2 is concatenated to the output of fast upsampling with Skip 1, and transmitted to the next layer. Because the shape of the proposed network in Fig. 2-(b) is similar to the appearance of a cocktail glass lying on its side, we call this network a CGN.

The belief that the accuracy of the final output will not deteriorate significantly, even if the layer is shortened in the decoder, is based on recent research achievements in the SR field. SR is a technology that generates a high-resolution image using complex inference processing from a low-resolution input image. In the configuration of the SR network, for the first time, Dong *et al.* [17] showed that increasing the resolution in the last step, after sufficiently repeating the convolution in the low-resolution

image, reduced the amount of computation, and obtained excellent results. Various SR networks—such as Zhang's RCAN(Residual Channel Attention Networks) [18], which has shown excellent performance thus far and Wang's ESRGAN(Enhanced Super-Resolution Generative Adversarial Networks) [19], which uses adversarial loss in SR—follow this framework. Therefore, in the depth-estimation network, if an efficient fast upsampling block is configured and skip-connection data of intermediate resolution are properly connected, an output of similar accuracy will be achievable, even if the number of layers in the decoder is drastically reduced.

B. CSR

Fig. 3 illustrates the proposed fast upsampling method. We term the feature data that have channel, row, and column axes as tensors, $\mathbf{x} \in \mathbb{R}^{C_{in} \times H \times W}$. In the first step, two convolutions, conv_h and conv_v are applied to the input tensor, $\mathbf{x}$. To prepare the channel-to-row and channel-to-column unrolling operations, these two convolutions change the information on the channel axis, and output $\mathbf{x}^h, \mathbf{x}^v \in \mathbb{R}^{kC_{out} \times H \times W}$. From these two tensors, the channel axis is partially converted to the horizontal or vertical axis using the unrolling operation, and $\mathbf{x}^{uh} \in \mathbb{R}^{C_{out} \times H \times kW}$ and $\mathbf{x}^{uv} \in \mathbb{R}^{C_{out} \times kH \times W}$ are generated. The unrolling operations, which are shown in Figs. 3, can be implemented using simple reshaping and permutation operations. These two tensors, $\mathbf{x}^{uh}$ and $\mathbf{x}^{uv}$, are extended again using linear interpolation (nearest neighbor) in the remaining axis, and $\mathbf{x}^{lv,uh}, \mathbf{x}^{lh,uv} \in \mathbb{R}^{C_{out} \times kH \times kW}$ are generated. To obtain the final result rationally unrolled in both the horizontal and vertical directions, we must multiply $\mathbf{x}^{lv,uh}, \mathbf{x}^{lh,uv}$. However, to enable back-propagation with deep neural network libraries, such as PyTorch or TensorFlow, the results of the branched network must not be multiplied by one another. We assume that the current tensors, $\mathbf{x}^{lv,uh}, \mathbf{x}^{lh,uv}$, are the feature data of the log value of the estimated depth. Therefore, $\mathbf{x}^{sum}$ is computed by adding $\mathbf{x}^{lv,uh}, \mathbf{x}^{lh,uv}$ first; then, a thresholded exponential function is applied to $\mathbf{x}^{sum}$ (g is a threshold function for stabilizing the CSR output). Consequently, the final tensor, $\mathbf{x}^{CSR}$, is equal to the product of the two tensors comprising depth features calculated by the horizontal and vertical unrolling and additional interpolation. Through this process, the original channel, kC_{out}, is reduced to C_{out}; simultaneously, the spatial resolution is increased by k in the row and column axes.

The CSR process can be described using the following equations. Equation (1) describes the process of unrolling operations at the single-element level. The channel information is partially rearranged into a row or column axis. In (1), i, r, c are the coordinates on the channel, row, and column axes, respectively, and k is the magnification ratio. $x^h_{i,r,c}$ and $x^v_{i,r,c}$ are the elements of the output tensor of conv_h and conv_v, respectively, in Fig. 3. $x^{uh}_{i,r,c}$ and $x^{uv}_{i,r,c}$ are the elements in the enlarged tensor after channel-to-row and channel-to-column unrolling operations. In (2), f_{lin_v} is a linear interpolation that inserts (k-1) intermediate values between $x^{uh}_{i,r,c}$ and $x^{uh}_{i,r+1,c}$, and f_{lin_h} inserts values between $x^{uv}_{i,r,c}$ and $x^{uv}_{i,r,c+1}$. In (3), $g(x)$ is the threshold function used to limit the output value from the exponential function, and (4) shows that the modified ReLU(Rectified Linear Unit) operation can be used to implement this function. $\mathbf{x}^{CSR}$ in (3) is the final result of the CSR block.

$$\begin{aligned} x^{uh}_{i,r,kc+j} &= x^h_{ki+j,r,c} \\ x^{uv}_{i,kr+j,c} &= x^v_{ki+j,r,c} \\ (j &= 0, 1, \ldots, k-1) \end{aligned} \tag{1}$$

$$\begin{aligned} \mathbf{x}^{sum} &= \mathbf{x}^{lv,uh} + \mathbf{x}^{lh,uv} \\ &= f_{lin_v}\left(\mathbf{x}^{uh}\right) + f_{lin_h}\left(\mathbf{x}^{uv}\right) \end{aligned} \tag{2}$$

$$\mathbf{x}^{CSR} = \mathrm{e}^{g(\mathbf{x}^{sum})} \tag{3}$$

$$g(\mathbf{x}) = 1 - \text{ReLU}(1 - \mathbf{x}) \tag{4}$$

As explained above, tensor $\mathbf{x}$ is considered as the log value of the target feature, $\mathbf{y}$, to calculate the depth; the final $\mathbf{x}^{CSR}$ can be interpreted as

$$\begin{aligned} x_{i,r,c} &= \log\left(y_{i,r,c}\right) \\ \mathbf{x}^{sum} &= \log\left(\mathbf{y}^{lv,uh}\right) + \log\left(\mathbf{y}^{lh,uv}\right) \\ \mathbf{x}^{CSR} &= \mathrm{e}^{g\left(\log\left(\mathbf{y}^{lv,uh}\mathbf{y}^{lh,uv}\right)\right)} \\ &\approx \mathbf{y}^{lv,uh}\mathbf{y}^{lh,uv} \end{aligned} \tag{5}$$

SPC [8] and unpacking [13] are partially similar to the CSR because they also use the conversion between the channel and spatial plane. Although both are based on the fixed one-dimensional (1D) to 2D relocating rule (Fig. 1-(b) and (c)), CSR generates extended data in a 2D plane by separately converting the channel information to the horizontal and vertical axes, and conceptually multiplying them. The effect of this principal difference can be confirmed by the difference in accuracy, as discussed in Section 4.

Considering the amount of computation and parameters, we compare CSR to the previous network structure comprising multiple layers with progressively increasing resolution. Table 1 shows the theoretical number of parameters and computations in the CSR module of the proposed network and the corresponding layers of the previous network. Table 1 does not show all the layers in the decoder part; only the CSR layer (module) in the proposed method and the corresponding layers in the previous network are represented. The actual decoder includes more layers. The amount of computations and parameters when MobileNet V2 is selected as the encoder are as follows. The decoder part uses the inverse bottleneck structure, and the hyperparameters in Table 1 are selected as (B1, B2, B3, B4) = (96, 32, 24, 16), (C2, C3, C4) = (512, 256, 128), $C_{CSR} = 256$. In this case, the number of parameters is 107,392, and the amount of computation is 1,150*HW with the existing method (Layers 2, 3, and 4). With the CSR, the number of parameters is 127,488, and the amount of computation is 498*HW. Notably, although

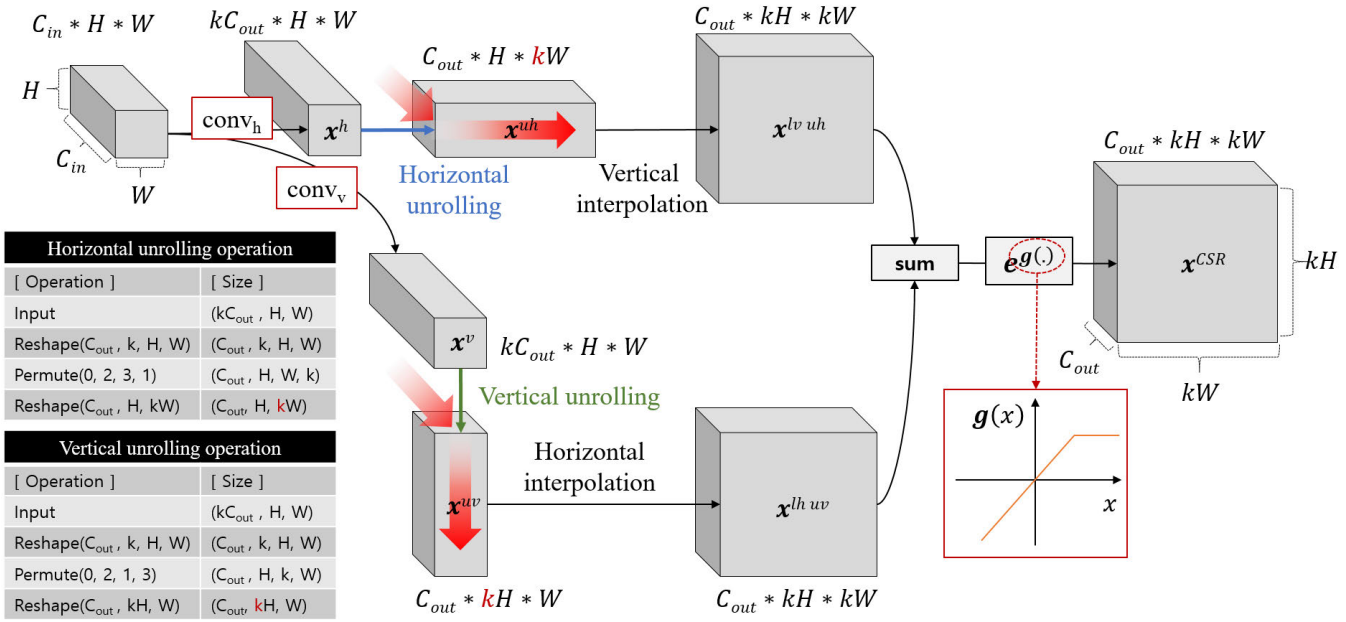

FIGURE 3. CSR process: A novel method for implementing the fast upsampling block in Fig. 2-(b). Input is a tensor, $\mathbf{x} \in \mathbb{R}^{C_{in} \times H \times W}$ and $conv_h$ and $conv_v$ are convolution operations which are pre-processings for horizontal, vertical unrolling operations. $\mathbf{x}^{CSR} \in \mathbb{R}^{C_{out} \times kH \times kW}$ is the output tensor of this CSR process.

TABLE 1. Parameters and computations included in the CSR module of Proposed2, and the corresponding layers (Layers 2, 3, and 4) of FastDepth2. C_i represents the channel of convolution of the i-th layer, and B_i indicates the output channel (bottleneck width of MobileNet V2) of the same layer. pw is the pointwise convolution, and dw is the depthwise convolution.

Layer Name	Operations	Parameters	Computations
CSR	2*pw(B_1, C_{CSR}) 2 × dw($C_{CSR}, 5$) 2*pw(C_{CSR}, B_2)	$2 * (B_1 C_{CSR} + 5^2 * C_{CSR} + C_{CSR} B_2)$	$\frac{2*H*W}{2^8} * (B_1 C_{CSR} + 5^2 * C_{CSR} + C_{CSR} B_2)$
Layer 2	pw(B_1, C_2) dw($C_2, 5$) pw(C_2, B_2)	$B_1 C_2 + 5^2 * C_2 + C_2 B_2$	$\frac{H*W}{2^8} * (B_1 C_2 + 5^2 * C_2 + C_2 B_2)$
Layer 3	pw(B_2, C_3) dw($C_3, 5$) pw(C_3, B_3)	$B_2 C_3 + 5^2 * C_3 + C_3 B_3$	$\frac{H*W}{2^6} * (B_2 C_3 + 5^2 * C_3 + C_3 B_3)$
Layer 4	pw(B_3, C_4) dw($C_4, 5$) pw(C_4, B_4)	$B_3 C_4 + 5^2 * C_4 + C_4 B_4$	$\frac{H*W}{2^4} * (B_3 C_4 + 5^2 * C_4 + C_4 B_4)$

the number of parameters increased when the proposed CSR was used, the computational amount decreased by less than half. This phenomenon occurs because, as shown in Fig. 3, the CSR performs convolution only in the thick channel and low-resolution stages; therefore, the number of computations per parameter is lower than that of the previous method. In Section 4, the theoretical prediction of the amount of computation is confirmed by measuring the flops and parameters in the experimental results.

IV. EXPERIMENTS

A. IMPLEMENTATIONS DETAILS

We implemented the proposed method using PyTorch [23]. Regarding the training, we used SGD as an optimizer, with a momentum of 0.9 and a weight decay of 0.0001. The learning rate was scheduled via exponential decay from the initial value of 0.1, with a decay factor of $p = 0.2$ for every 5 epochs. We tested MobileNet V1 [3] and MobileNet V2 [20] as lightweight models as well as ResNet50, ResNet152 [1], and DenseNet161 [2] as heavy-weight models in the encoder part. This confirms that the proposed method is not significantly affected by the capacity of the backbone network model. As a dataset for learning and evaluation, the NYU V2 dataset was used as the indoor image set, and KITTI was used as the outdoor image set. Regarding the loss, the proposed methods and other networks implemented for comparison were trained to minimize the combined loss of L1 and SSIM(Structural Similarity Index Map) [24] loss (6). We used NVIDIA Titan V

TABLE 2. Experimental results obtained using MobileNet V1 [3] and MobileNet V2 [20] as backbone (encoder) networks on the NYU Depth V2 dataset. The flops and parameters represent the amount measured only in the decoder. A higher value is better for $\delta < 1.25$, $\delta < 1.25^2$, and $\delta < 1.25^3$, whereas a lower value is better for AbsRel and RMSE.

Method	Backbone	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	AbsRel	RMSE	Flops (M)	Parameters (M)
FastDepth1	MobileNet V1	**0.779**	**0.946**	0.983	0.565	0.162	173.7	0.754
Proposed1	MobileNet V1	0.770	0.944	**0.985**	0.569	**0.160**	102.0	0.734
FastDepth2	MobileNet V2	0.778	0.943	0.983	0.568	0.164	132.7	**0.274**
Proposed2	MobileNet V2	0.776	0.944	0.984	**0.559**	0.163	**73.5**	0.385

TABLE 3. Experimental results on the NYU Depth V2 dataset. Proposed3, 4, and 5 are CGNs with CSR using various heavy models (ResNet50, ResNet152 [1], and DenseNet161 [2]) as backbone (encoder) networks, where the flops and parameters represent the amount measured only in the decoder.

Method	Backbone	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	AbsRel	RMSE	Flops (M)	Parameters (M)
Laina et al. [7]	ResNet50	0.811	0.953	0.988	**0.127**	0.573	.	.
Li et al. [21]	ResNet50	0.788	0.958	**0.991**	0.143	0.635	.	.
Qi et al. [22]	ResNet50	**0.834**	**0.960**	0.990	0.128	0.569	.	.
UNetResNet50	ResNet50	0.809	0.957	0.988	0.145	**0.520**	452	2.88
Proposed3	ResNet50	0.811	0.957	0.988	0.146	**0.520**	**287**	**2.84**
UNetResNet152	ResNet152 [1]	0.822	0.959	0.988	0.141	0.507	452	2.88
Proposed4	ResNet152	**0.831**	**0.964**	**0.991**	**0.137**	**0.492**	**287**	**2.84**
Proposed5	DenseNet161	0.827	0.964	0.990	0.133	0.499	121	1.07

TABLE 4. Experimental results on the KITTI dataset. Proposed2, 3, and 4 are CGNs with CSR using MobileNet V1 [3], ResNet50, and ResNet152 [1] as backbone (encoder) networks. The flops and parameters represent the amount measured only in the decoder.

Method	Backbone	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	AbsRel	RMSE	Flops (M)	Parameters (M)
FastDepth2	MobileNet V2	0.860	0.968	0.992	0.111	4.426	325	**0.274**
Proposed2	MobileNet V2	**0.868**	**0.970**	**0.993**	**0.107**	**4.259**	**180**	0.385
UNetResNet50	ResNet50	0.878	0.975	0.994	0.104	4.051	1,105	2.88
Proposed3	ResNet50	**0.889**	**0.977**	0.994	**0.098**	**3.977**	**703**	**2.84**
UNetResNet152	ResNet152	0.886	0.976	0.995	0.101	4.007	1,105	2.88
Proposed4	ResNet152	**0.901**	**0.980**	0.995	**0.093**	**3.850**	**703**	**2.84**

for the training and testing, and the precision was maintained at a 32-bit floating point.

$$loss = loss_{L1} + 0.4 \times loss_{SSIM} \tag{6}$$

B. NYU DEPTH V2 DATASET

We trained our network and evaluated its accuracy on the NYU V2 dataset [25] with an official train/test split. For the training, the total number of epochs was set to 30, with a batch size of 128. First, we used the pre-trained models of MobileNet V1 [3] and V2 [20] as encoders, and compared the depth-estimation results with the proposed and FastDepth networks (skip-add version before pruning) when the input image was resized to 224 × 224. Table 2 lists the quantitative errors, computation amounts, and parameters. "Proposed1" and "Proposed2" in Table 2 are the depth-estimation networks, whose structures are the same as those in Fig. 2-(b). MobileNet V1 and V2 were used as encoders, and CSR was used as a fast upsampling block. The number of flops and parameters indicate the amount measured only in the decoder. Considering the accuracy, Proposed1 is similar to FastDepth1; however, Proposed2 is better than FastDepth2. Comparing the flops in the decoder parts, Proposed1 and Proposed2 were only 59% and 55% of FastDepth1 and FastDepth2, respectively. Considering the parameters, Proposed1 and Proposed2 was 97% and 141% of FastDepth1 and FastDepth2, respectively. When using the CSR, the flops are reduced by half; nevertheless, the parameter is similar (Proposed1) or even increased (Proposed2). This experimental result is consistent with the prediction in III-B above, demonstrating that the flops can be decreased even with more parameters in the proposed method.

Fig. 4 shows the depth-estimation resulting images of the networks using the MobileNet V2 encoder. The red-dotted ellipse indicates a scenario where the depth-estimation error occurs partially in FastDepth2, and Proposed2 exhibits better results. In conclusion, the proposed method reduced the amount of computation (flops) in the decoder by more than 40%, maintaining a similar or better accuracy in a lightweight depth-estimation network.

In our network, we also used ResNet50, ResNet152 [1], and DenseNet161 [2] as encoders, instead of MobileNet. This experiment aims to determine whether the proposed CGN structure and CSR operate normally in a scenario where a large-capacity model is used as a backbone model. Therefore, we connected the decoder in Proposed1 to ResNet50, ResNet152, and DenseNet161 with minimal modifications, although our decoder aimed for a scenario where a lightweight model, such as MobileNet, was used as an encoder. Table 3 presents the results of these experiments. "Proposed3" ("Proposed4" and "Proposed5")

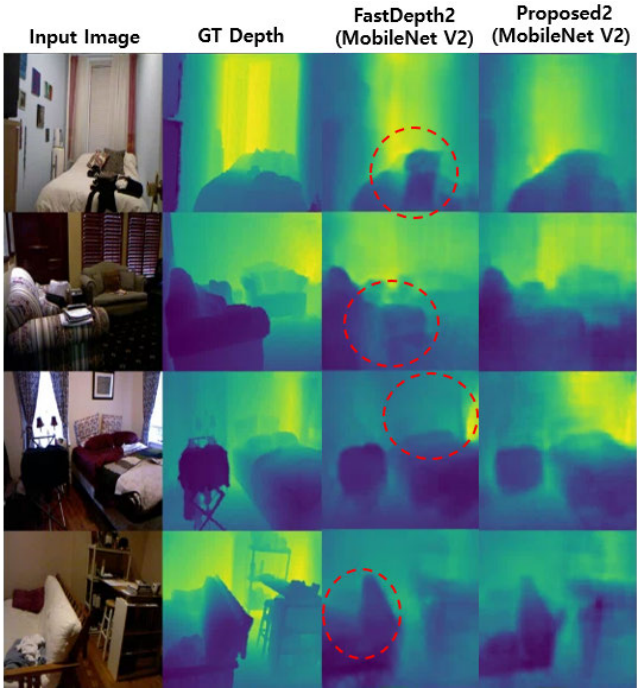

FIGURE 4. Depth-estimation results of FastDepth [14] (baseline) and proposed methods using MobileNetV2 encoders on the NYU Depth V2 dataset. Each method corresponds to the same name in Table 2. The color values of each result are individually scaled.

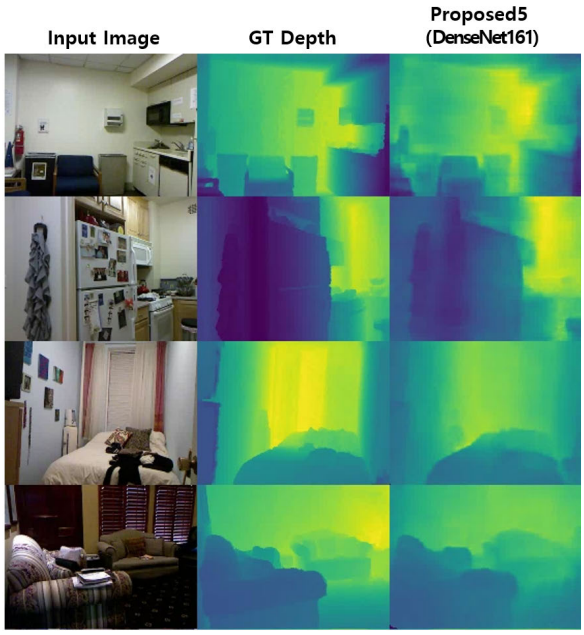

FIGURE 5. Depth-estimation results of Proposed5 with DenseNet161 encoder (backbone). This is the result of the experiment of attaching the decoder part of the lightweight network to the DenseNet161 encoder. The input images are 640 × 480, and the color values of each result are scaled individually.

in Table 3 was a network similar to Proposed1 (or Proposed2) in Table 2, with only the encoder being replaced with ResNet50 (ResNet152 and DenseNet161) and the data size of each layer being adjusted. ''UNetResNet50'' (''UNetResNet152'') was a network created by combining the ResNet50 (ResNet152) encoder and the partially modified version of the FastDepth decoder for a direct comparison with the proposed method. When the CGN structure with the CSR was used (Proposed3 and Proposed4), accuracy was maintained at a level similar (or even better) to that of the general UNet structure (UNetResNet50 and UNetResNet152); meanwhile, the computational amount was reduced to approximately 60%. Finally, the DenseNet161 [2] encoder was used in Proposed5 (input images are resized to 640 × 480), and the flops and parameters in the decoder were lower than those in Proposed3 and 4; meanwhile, the accuracy was similar to each other. Fig. 5 shows the depth-estimation resulting images of Proposed5, and confirms that the depth map is clearly estimated.

C. KITTI DATASET

We provide the experimental results of the proposed method on the KITTI dataset. This experiment aims to confirm that the proposed CGN structure and CSR function normally, even for outdoor images such as the KITTI dataset. For the training, the total number of epochs was set to 20, and the batch size was set to 24. The input image was resized to 192 × 640 pixels. Table. 4 shows the quantitative performance of the proposed networks on the KITTI dataset. FastDepth2 and Proposed2 are same networks as those in Table. 2, and UNetResNet50, UNetResNet152, Proposed3, and Proposed4 are same networks as those in Table. 3. Regardless of whether the backbone network (encoder) was a lightweight or heavy-weight model, when the proposed method was used, the computational amount (flops) in the decoder was reduced to approximately 60%, and the accuracy was found to be better than that of the UNet structure network (FastDepth2, UNetResNet50, and UNetResNet152), which were implemented for comparisons. Fig. 6 shows the results of the proposed (Proposed2 and Proposed4) and previous methods (FastDepth2 and UneResNet152). We show only the resulting images with MobileNet V2 and ResNet152 encoders because our purpose is to show a consistent effect of the proposed method with a lightweight or heavy-weight network encoder. In Fig. 6, the upper area of the image, without the GT information, shows that the previous methods tend to generate black holes; however, the proposed method generates a relatively better result in this area. From this, It seems that the inference ability learned in the region where GT exists works better in the region without GT (the upper area of the images) when the proposed CSR is used. From this, we can guess that the function of CSR to directly derive high-resolution data from deep layer data improves the spatial consistency of depth inference.

D. COMPARISON WITH PREVIOUS UPSAMPLING

Table 5 shows the results of using various methods in the fast upsampling block in the network structure of Fig. 2-(b). We used no inner-skip-connection in each fast upsampling

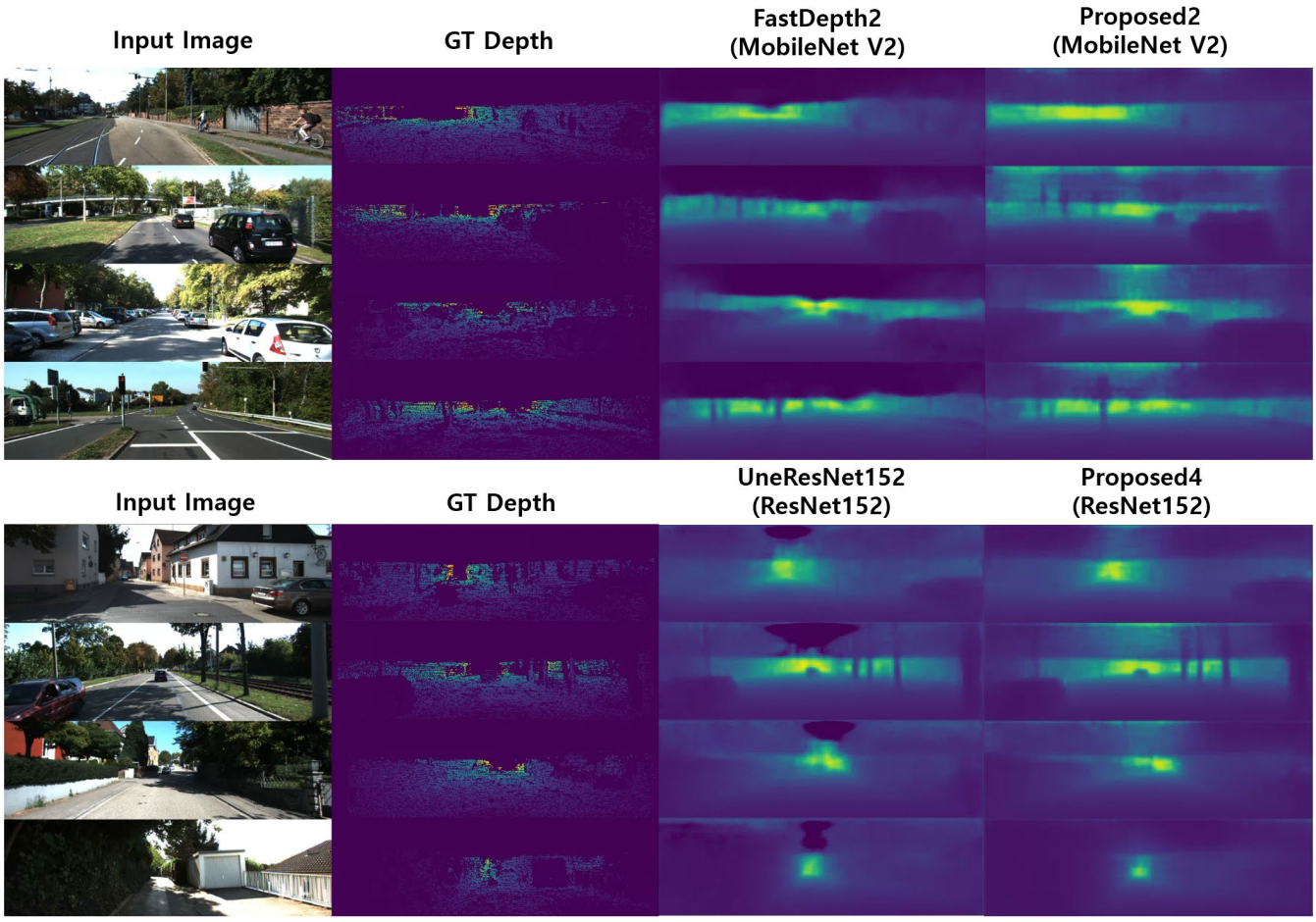

FIGURE 6. Depth-estimation results on the KITTI dataset using MobileNet V2 [1] and ResNet152 [1] as backbone (encoder) networks.

TABLE 5. Experimental results of various fast upsampling method (no-skip-connection in each upsampling process), on the NYU Depth V2 dataset using MobileNet V2 [20] as a backbone network. The flops and parameters represent the amount measured only in the decoder. Bilinear: bilinear interpolation, SPC (Seq): SPC [8] with sequential 2× operations, SPC (OneShot): SPC with 8× operation, Unpack (Seq): Unpacking operation [13] with sequential 2× operations, Unpack (OneShot): Unpacking operation with 8× operation.

Method	Backbone	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	AbsRel	RMSE	Flops(M)	Parameters(M)
Bilinear	MobileNet V2	0.777	0.945	0.984	0.561	**0.159**	**58.4**	**0.166**
SPC (Seq)	MobileNet V2	0.774	0.944	0.984	0.569	0.162	1,820	1.403
SPC (OneShot)	MobileNet V2	0.778	**0.946**	**0.985**	**0.557**	0.161	330.2	1.552
Unpack (Seq)	MobileNet V2	0.777	0.945	**0.985**	0.561	0.161	157.6	0.449
Unpack (OneShot)	MobileNet V2	0.773	0.945	0.984	0.563	0.164	113.2	0.417
CSR	MobileNet V2	**0.780**	**0.946**	**0.985**	0.558	0.160	80.5	0.279

block to compare only the effect of each fast upsampling method. Fig. 7 shows the resulting images of each method in Table 5. ''SPC(Seq)'' was the method used by Shi *et al.* [8] in their GitHub code, and expanded the resolution eight times by repeating the 2× expansion operation three times, while maintaining the number of channels before and after the calculation. ''SPC(OneShot)'' was a method that enlarged eight times at a time, and reduced the number of channels to 1/64. SPC(OneShot) and ''Unpack(OneShot)'' are methods that were previously used for expanding the resolution eight times in the spatial plane, reducing the channel in a layer. These methods were similar to the concept that is shown in Fig. 1-(c). Regarding the accuracy, SPC(Seq) was the most accurate; however, the computation amount was significant. ''Bilinear'' had the best computational and parameter quantities, and its accuracy was not poor. Nonetheless, some serious errors were observed (red-dotted circles in Fig. 7-Bilinear) when the resulting images were compared using the naked eye. The CSR was excellent in terms of both accuracy and amount of computation; it had the highest accuracy in terms of $\delta < 1.25$, $\delta < 1.25^2$, and $\delta < 1.25^3$ and the second lowest flops and parameters after bilinear.

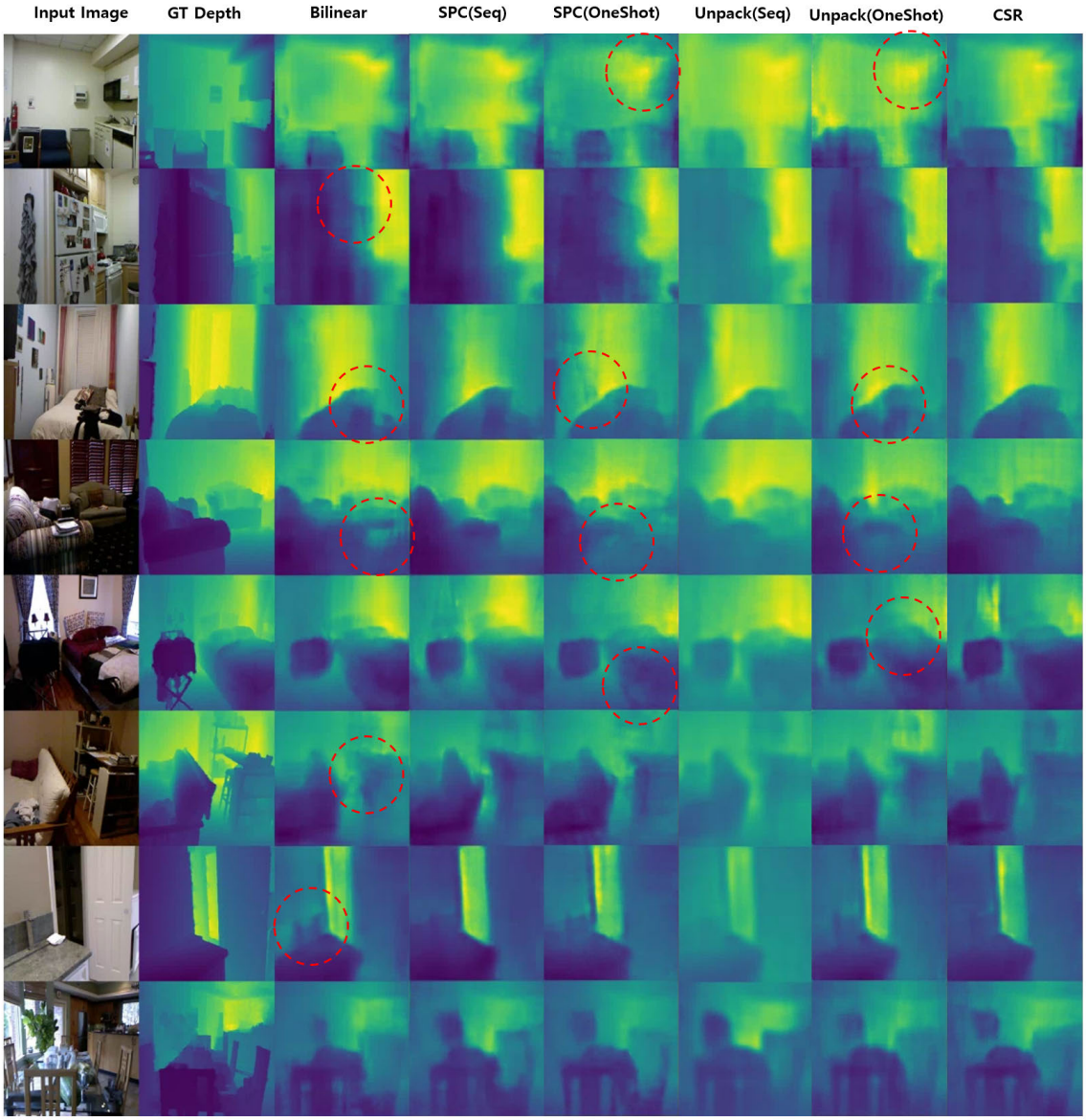

FIGURE 7. Depth-estimation results on the NYU V2 dataset using various fast upsampling methods and no-skip-connection in each fast upsampling method. The color values of each result are individually scaled.

V. CONCLUSION

In this study, we propose a modified UNet-style depth-estimation network, which is known as CGN, and a novel fast upsampling method, which is referred to as CSR. The structural modification in CGN focuses on reducing the number of convolution layers in the decoder. CGN uses a fast upsampling block which outputs high-resolution data with a thin channel from spatially low-resolution data with a thick channel. We also propose a novel fast upsampling block, CSR which unrolls the information in the thick channel in the spatially horizontal and vertical axes separately and fuses the two results. When the proposed CSR is used as the fast upsampling block in CGN, the amount of computation in the decoder part is reduced by half while maintaining the inference accuracy similarly, compared to the case using the existing methods (SPC [8] and unpacking [13]). In this study, the proposed CGN structure and CSR are applied to only the depth-estimation problem; however, it can be applied to other problems such as segmentation problems, where similar network structures are used.

ACKNOWLEDGMENT

The authors would like to thank Seungjae Lee for his advice and help on the KITTI dataset experiment.

REFERENCES

[1] 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.

[2] 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.
[3] 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*. [Online]. Available: http://arxiv.org/abs/1704.04861
[4] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "GhostNet: More features from cheap operations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1580–1589.
[5] B. Wu, K. Keutzer, X. Dai, P. Zhang, Y. Wang, F. Sun, Y. Wu, Y. Tian, P. Vajda, and Y. Jia, "FBNet: Hardware-aware efficient ConvNet design via differentiable neural architecture search," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10734–10742.
[6] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.
[7] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *Proc. 4th Int. Conf. 3D Vis. (3DV)*, Oct. 2016, pp. 239–248. [Online]. Available: https://ieeexplore.ieee.org/document/7785097/
[8] W. Shi, J. Caballero, F. Huszar, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1874–1883.
[9] A. Saxena, S. H. Chung, and A. Y. Ng, "Learning depth from single monocular images," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 1161–1168.
[10] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 2650–2658.
[11] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2002–2011.
[12] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," 2019, *arXiv:1907.01341*. [Online]. Available: http://arxiv.org/abs/1907.01341
[13] 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)*, Jun. 2020, pp. 2485–2494.
[14] D. Wofk, F. Ma, T. Yang, S. Karaman, and V. Sze, "FastDepth: Fast monocular depth estimation on embedded systems," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 6101–6108. [Online]. Available: https://ieeexplore.ieee.org/document/8794182/
[15] T.-J. Yang, A. Howard, B. Chen, X. Zhang, A. Go, M. Sandler, V. Sze, and H. Adam, "NetAdapt: Platform-aware neural network adaptation for mobile applications," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 285–300.
[16] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2016, pp. 483–499.
[17] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2016, pp. 391–407.
[18] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 286–301.
[19] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. C. Loy, "ESRGAN: Enhanced super-resolution generative adversarial networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV) Workshops*, Sep. 2018, pp. 63–79.
[20] 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.
[21] J. Li, R. Klein, and A. Yao, "A two-streamed network for estimating fine-scaled depth maps from single RGB images," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3372–3380.
[22] X. Qi, R. Liao, Z. Liu, R. Urtasun, and J. Jia, "GeoNet: Geometric neural network for joint depth and surface normal estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 283–291.
[23] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, and A. Desmaison, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8026–8037.
[24] A. Hore and D. Ziou, "Image quality metrics: PSNR vs. SSIM," in *Proc. Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 2366–2369. [Online]. Available: http://ieeexplore.ieee.org/document/5596999/
[25] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2012, pp. 746–760.

JUNG-JAE YU received the B.S. degree in electronic and electrical engineering from Yonsei University, Seoul, South Korea, in 2003, and the M.S. degree in electronic and electrical engineering from KAIST, Daejeon, South Korea, in 2005, where he is currently pursuing the Ph.D. degree. He joined the Electronics and Telecommunications Research Institute (ETRI), in 2005, where he is currently a Principal Researcher. His research interests include image-based 3D reconstruction, pattern recognition, and image generation using deep neural networks.

JONG-GOOK KO received the M.S. degree in electronic and electrical engineering from GIST, Gwangju, South Korea, in 2000. Since 2000, he has been a Research Scientist with the Electronics and Telecommunications Research Institute (ETRI). His research interests include deep learning, pattern recognition, and computer vision.

JUNMO KIM (Member, IEEE) received the B.S. degree from Seoul National University, South Korea, in 1998, and the M.S. and Ph.D. degrees from Massachusetts Institute of Technology, in 2000 and 2005, respectively. From 2005 to 2009, he was with the Samsung Advanced Institute of Technology, South Korea. In 2009, he joined Korea Advanced Institute of Science and Technology, as a Faculty Member, where he is currently a Tenured Associate Professor in electrical engineering. His current research interests include image processing, computer vision, statistical signal processing, machine learning, and information theory.